

Qu'est-ce que le HTML et le CSS ?

Tout d'abord, quelques courtes définitions :

Un site internet

Un site internet est en fait un ensemble de pages WEB (de fichiers .html) reliés ensemble par des liens hypertextes.

Le HTML

Le HTML est le format de fichier avec lequel vous structurez vos pages : c'est le contenu. Il est formé de blocs qui servent à délimiter, et décrire la signification de chaque information... Un bon site sépare **le contenu et la forme** : c'est ce que nous allons faire.

Le CSS

Le CSS est un (ou plusieurs) fichiers séparés qui définit la mise en page des pages HTML : la forme. C'est là est seulement là que vous ajouterez des couleurs, des images, mettez plusieurs colonnes... Vos fichiers HTML ne doivent contenir que les informations : pas de mise en gras, d'ajustement de la police, etc...

Quels sont leur bénéfices ?

Ce tutoriel se sépare en deux parties importantes : tout d'abord, vous apprendrez uniquement les balises nécessaires et non désuètes pour représenter le contenu de vos pages. Ce n'est que dans la seconde partie que vous apprendrez à mettre en forme ces pages.

Utiliser le HTML Strict et le CSS permet de séparer le contenu et la forme.

Cela permet :

D'avoir des pages HTML organisées et structurées, lisibles autant par des humains que par des robots (moteurs de recherche), et même par des personnes handicapées ;

De ne définir la présentation qu'une seule fois, dans un seul fichier. Et de pouvoir la changer très simplement, une seule fois, que ce soit juste un petit changement ou le changement de tout le style du site ;

D'avoir des pages HTML très légères à télécharger (et oui : les modems 56k existent encore !). De plus la présentation étant dans un seul fichier, celui-ci n'a pas à être re-téléchargé à chaque page que les visiteurs téléchargent ;

D'avoir des fichiers HTML très clairs à éditer : il n'y a que le minimum de code, sans fioritures et sans s'occuper de la présentation ;

Note : Nous travaillerons directement sur du XHTML Strict (sans les balises de transition, et reconnu par tous les navigateurs) ! Avec les tournures du style "Il est interdit de", nous nous référerons à ce format strict.

Il est parfois possible de passer outre ces interdictions (ou plutôt recommandations) mais vous perdrez la compatibilité de vos pages avec les standards du WEB.

Le but de justement travailler avec les standards : être quasi sûr que ça passe partout... et d'être reconnu non seulement par tous les navigateurs, mais aussi par les moteurs de recherches, etc...

Note : Toutes les balises, propriétés, etc... sont données en minuscules et chaque attribut de balise XHTML est entourée par des guillemets : ces recommandations sont obligatoires pour respecter les standards XHTML.

Organisation de votre site

Lorsque vous faites un site, pensez dès le début à une organisation : vous pouvez soit mettre toutes les pages HTML dans le dossier racine, les fichiers dans un sous dossier images... Ou alors, si votre site est composé de plusieurs rubriques, faire un sous dossier par rubrique et placer tous les fichiers concernant une rubrique dans son dossier (même le ou les fichiers HTML).

Lorsque vous nommez vos pages, fichiers et dossiers, veillez à leur donner un nom court (mais descriptif) sans majuscules, espaces ou caractères accentués. En effet, certains serveurs font la distinction entre majuscules et minuscules. De même, vos visiteurs retiendront mieux des adresses courtes dont tous les caractères sont en minuscules. Il est recommandé d'utiliser les lettres majuscules, les chiffres et les caractères `;` `-' et `_' . Les autres caractères sont à éviter.

En ce qui concerne le format des images, il est fortement recommandé de les compresser en .gif pour les graphismes de peu de nuances, et en .jpg pour les photos. Le format .png est meilleur de que .gif (il est aussi sans perte ou limitation de couleurs !). Je vous recommande de l'utiliser pour une meilleur qualité. Cependant, Internet Explorer rend mal les .png avec plus d'un seul niveau de transparence : il vaudrait mieux donc se limiter à un seule couleur de transparence au maximum (de toute façon le format .gif ne permet pas non plus d'aller plus haut).

Testez votre site

Même si les formats XHTML Strict et du CSS sont des standards du WEB et donc reconnus par tous les navigateurs, il est fortement conseillé de tester vos pages WEB (surtout les CSS) dans au moins deux navigateurs :

- IE,
- Mozilla Firefox, Mozilla,
- Konqueror, Safari,
- Lynx...

Validez de votre site

Valider permet non seulement de s'assurer que votre site est conforme aux normes du W3C, et donc passe sur tous les navigateurs le supportant, mais aussi pour corriger vos erreurs : vous verrez, après quelques validations, vous aurez progressé et le validateur, par la suite, ne vous sortira plus beaucoup d'erreurs, sauf si vous avez faite une erreur PAS FREQUENTE.

Le HTML

Nous allons commencer notre apprentissage de la création de sites WEB par le plus important : le contenu ! Pour cela nous utiliserons le HTML Strict et ne nous occuperons dans un premier temps que de l'information, sa hiérarchie et sa signification. Les pages créées ne seront pas très agréables à voir mais elles seront lisibles par tous : autant par les navigateurs graphiques ou textes / oraux / brailles, que des moteurs de recherches et autre robots indexateurs...

Structure en balises

HTML signifie HyperText Markup Language. En clair, c'est un langage (format de fichier) qui permet d'écrire des pages internet, et cela au moyen de balises (Markup, en anglais).

Qu'est-ce que des balises ? C'est une façon de hiérarchiser ses données. Exemple : pour indiquer un titre, on utilise la balise H1 ; on dit que ce texte est contenu dans la balise. En fait, il est entouré par deux morceaux de textes : `<h1>Mon titre</h1>`.

Les balises peuvent s'imbriquer les unes dans les autres pour former une hiérarchie. Exemple courant : une liste est décrite par une balise UL, et chaque ligne est délimitée par une autre balise, qui est comprise à l'intérieur de la balise de liste.

Ensuite, peu importe le nombre d'espaces (espaces, ou tabulations, ou même retour à la ligne que vous mettez dans votre source (le fichier que vous éditez), le navigateur remplacera tous ceux qui se suivent par un et un seul espace (sauf en début de ligne : là il n'en mettra aucun).

Nous avons maintenant dit au navigateur que nous avons bien créé une page HTML et lui avons donné son titre à afficher dans la barre des tâches.

Mais rien n'y est encore affiché. Nous allons donc remplir le corps (`<body>`) avec du contenu.

Globalement, notre contenu est composé de plusieurs types de données : des titres plus ou moins importants, des paragraphes, des listes, des liens, des images ou des tableaux : il n'y a pas grand chose d'autre, si ce n'est que certains passages ont besoin d'être insistés, d'autres sont plus importants...

Notez que je n'ai pas parlé de passages en gras ou soulignés : le HTML s'intéresse à la signification du texte, pas à son rendu à l'écran ou au papier, car une page HTML peut aussi être lue par un logiciel de synthèse vocale, ou de bien d'autres façons : dans ces cas il est aberrant de parler de texte en rouge, par exemple (dans ce cas, il faut s'interroger sur le pourquoi de ce rouge : est-ce important, est-ce dangereux... les balises HTML seront là pour donner la signification, la forme sera fonction du style CSS pour les écrans où de l'intonation pour les synthétiseurs vocaux).

En HTML il existe deux sortes de balises :

Les blocs : On passe à la ligne avant et après le bloc. Il s'agit des paragraphes.

Les blocs peuvent contenir d'autres blocs, mais aussi des passages en ligne,

Les passages en ligne : pour donner la signification d'un mot ou groupe de mots à l'intérieur d'un bloc ou paragraphe, ou d'un autre passage en ligne : on ne va évidemment pas à la ligne.

Ceux-ci ne peuvent contenir que des balises elles-mêmes en ligne.

La page

Un fichier HTML débute toujours par indiquer au navigateur que notre page est faite en HTML, et indique aussi quelle version de ce format est utilisé.

C'est ce qu'on appelle le DOCTYPE. Ne vous effrayez pas : il sera toujours le même et vous pouvez vous contenter de le copier / coller dans toutes vos pages sans le comprendre (il n'y a d'ailleurs pas grand chose à comprendre). Pour le HTML 4.01 Strict que nous allons utiliser, voici le DOCTYPE à inclure :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">
```

Maintenant que le navigateur sait cela, nous rentrons dans la structure balisée du HTML : on commence par ouvrir une balise `<html>`, et on la referme à la fin du fichier avec `</html>` : après cette balise de fin il est interdit.

À l'intérieur (entre la balise ouvrant et fermante), on trouve :

Un entête (<head> en anglais ; idem : il faut le refermer, donc </head>) qui contient des descriptions et informations annexes qui ne sont pas affichées : se sont les meta-données,

Puis, un corps (<body> et on referme aussi </body>) : c'est ce qui sera affiché ou lu par les navigateurs.

Dans notre entête, se trouvent des meta-données ; pour commencer on ne verra que la plus importante : <title>Ceci est le titre de la page</title>

On récapitule donc, voici le squelette (mince, pour l'instant) de toute page HTML :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Insérer ici le titre de la page</title>
  </head>
  <body>
    C'est ici que nous allons mettre le contenu de la page !
  </body>
</html>
```

Note : dans la suite des exemples, je ne donnerais plus tout le fichier HTML. Je tiendrais pour acquis que tout fichier HTML doit contenir les balises <html>, <head> et <body>. Les exemples viendront alors s'intercaler entre les balises <body> et </body>.

Les paragraphes

L'élément fondamental dans les pages internet c'est le texte (hormis les sites de jeux flash et autres divertissements, bien sûr). Tout texte dans une page HTML doit toujours être contenu dans une balise bloc : un paragraphe, une liste (nous verrons cela bientôt), etc... mais jamais directement dans le <body> ! Voyons donc les balises principales :

<p></p> (Paragraphe) : la plupart des textes de vos pages s'y trouvent,

<pre></pre> (Paragraphe préformaté) : cette balise est presque identique au paragraphe ci-dessus, à la différence près que les espaces et autres retours à la ligne sont gardés tel quels. En pratique vous ne l'utiliserez presque jamais, sauf pour inclure des exemples de code, si bien sûr votre site est destiné à des programmeurs,

 (Saut de ligne) : c'est une balise seule (pas besoin de la fermer avec un éventuel </br> par exemple). Parfois vous aurez besoin d'aller à la ligne dans vos paragraphes (ou dans une ligne de liste...)

Attention : il n'est pas recommandé d'utiliser deux
 à la suite : en y réfléchissant, vous voudrez sûrement créer un nouveau paragraphe ; dans ce cas fermez le paragraphe courant et ouvrez en un autre.

Voyons donc un petit exemple, en regardant le code HTML et le résultat produit :

```
<p>Ceci est un paragraphe normal...<br />... de deux lignes :
```

```
Remarquez que les retours à la ligne s'effectuent avec la balise br
```

```
et que tous les autres espacements et retours à la ligne sont remplacés par un seul espace par le navigateur.</p>
```

```
<p>Deux paragraphes sont espacés de manière à les distinguer.</p>
```

```
<pre>Voici maintenant un paragraphe préformaté.
```

```
  Dans celui-ci, les espaces supplémentaires, tabulations et retours
à la ligne sont pris en compte par le navigateur.
```

```
  Veuillez l'utiliser avec parcimonie, lorsque cela est vraiment nécessaire,
```

```
comme les exemples de code source (que nous verrons plus tard).</pre>
```

Deux paragraphes sont espacés de manière à les distinguer.

Les titres

C'est bien beau d'avoir des pages avec du texte, mais ce n'est pas très lisible. Pour les séparer et les identifier, on utilise généralement des titres.

Voici donc les balises à utiliser :

`<h1></h1>` (Titre de page) : une seule fois en tout début de page, comme son nom l'indique,
`<h2></h2>` à `<h6></h6>` (Sous-titres) : ils sont hiérarchisés du plus important (h2) au plus précis (h6),
 Quelques précisions : chacune de vos pages ne doit contenir qu'un et un seul titre de niveau 1, chaque paragraphe sera introduit par un titre de niveau 2 lorsque nécessaire, et les sous-titres seront des titres de niveaux 3 jusque 6. Ainsi de suite...

Nous allons créer votre première page.

Reprenez le squelette et copiez/collez-le dans un nouveau fichier, nommé page.html par exemple,
 Modifiez la balise title avec le titre de votre page,
 Puis dans le corps (`<body>`) commencez par une balise `<h1>` et rappelez le titre (celui-là sera affichée par le navigateur),
 Ajoutez éventuellement un court paragraphe d'introduction (`<p>Ma superbe introduction</p>`),
 Puis, pour chaque grand partie de la page :
 Ajoutez un titre `<h1>`,
 Mettez son contenu dans un paragraphe `<p>` et utilisez `
` si vous devez passer à la ligne sans changer de paragraphe (ne pas trop en abuser),
 Éventuellement, ajoutez un sous-titre avec `<h2>` puis le paragraphe correspondant...
 Et voilà ! Vous venez de faire votre première page internet ; voici à quoi votre code devrait ressembler :

```
<html>
<head>
  <title>Tutoriel HTML et CSS</title>
</head>
<body>
  <h1>Tutoriel HTML et CSS</h1>
  <h2>Utilité</h2>
  <p>Ces langages sont les standards du WEB et sont reconnus par tous les navigateurs<br />
  Apprenons-les donc !</p>
  <h2>Apprentissage</h2>
  <h3>Le HTML</h3>
  <p>Et vous ne le connaissez pas déjà ? Tant pis, ce n'est qu'un exemple !</p>
  <h3>Le CSS</h3>
  <p>Pareil : nous le verrons plus tard</p>
</body>
</html>
```

OK : ce n'est pas très beau ; mais ne vous inquiétez-pas : nous verrons comment allécher le visiteur lorsque nous verrons le CSS : ce sont deux choses différentes et nous nous concentrons pour l'instant sur le contenu. Comme vous le voyez c'est très simple et on arrive déjà à quelque chose de potable et d'utilisable. Si vous n'avez qu'une seule page à mettre en ligne et sans liens ou images (juste pour expliquer un sujet) c'est déjà efficace (sauf s'il vous faut des listes).

Les listes

Vous avez trois sortes de listes : les listes non ordonnées (ou "à puces"), les liens ordonnées (numérotées), et les listes de définitions (c'est un peu particulier et on n'en a pas souvent besoin).

Voici pour commencer les deux premiers styles de listes : on les délimite par ul ou ol selon leur nature, et chaque ligne est entourée par li :

```
<ul></ul> (Liste non ordonnée) : liste à puces (souvent rondes, carrées...),
<ol></ol> (Liste ordonnée : liste numérotée, 1. pour le premier élément, 2. pour le second, etc...),
<li></li> (Ligne de liste) : impérativement à l'intérieur d'un <ul> ou d'un <ol> ; tout texte à l'intérieur d'une
liste (ul ou ol) doit être contenu dans une <li></li>.
```

Note : la balise ol peut prendre un argument start : `<ol start="1">`, afin de déterminer le numéro du premier item. C'est attribut désigné 'dépreciated' par le standard HTML mais il n'existe pas de substitue ; or il peut parfois être intéressant (mais ne pas en abuser).

Voici un exemple des listes qui, en plus, vous montre comment imbriquer les listes les unes dans les autres,

en ouvrant une seconde liste `` ou `` à l'intérieur d'une ligne de la liste parente :

```
<ul>
<li>Première ligne de la liste non ordonnée (puces)</li>
<li>Seconde ligne de la liste non ordonnée (puces)</li>
<li>Troisième ligne, mais on ne referme pas la balise : on ouvre une autre balise ul ou ol :
  <ul>
    <li>Première ligne de la liste ordonnée, qui est imbriquée dans la première liste</li>
    <li>Ensuite, ne pas oublier de refermer la ligne de la première liste après avoir fermée la seconde liste</li>
  </ul>
</li>
<li>Si vous suivez mon raisonnement... :-)</li>
</ul>
<p>Et voici une liste ordonnée :</p>
<ol>
<li>Commencer par porter l'eau à ébullition</li>
<li>Puis, placez les œufs dans la casserole</li>
<li>Et là, veuillez faire ces étapes dans l'ordre :
  <ol>
    <li>Prenez une petite assiette</li>
    <li>Placez-y la bave de crapaud (bon d'accord, je manque d'imagination pour cet exemple)</li>
    <li>Et puis débrouillez-vous...</li>
  </ol>
</li>
<li>Bon appétit, bien sûr !</li>
</ol>
```

Les listes de définitions

Parfois, vous aurez à définir une liste de termes ou, plus généralement, à avoir une liste dont chaque ligne est formée d'un objet suivi d'une description.

Ces listes étant peu utilisées, vous pouvez passer ce paragraphe, mais il est intéressant de connaître leur existence !

Voici donc les balises utilisées pour cette troisième forme de liste :

```
<dl></dl> (Liste de définitions) : englobe la liste,
<dt></dt> (Item de Terme) : quelques mots à définir,
<dd></dd> (Item de Définition) : définition du ou des terme(s) le(s) plus proche(s) au dessus.
Vous pouvez très bien définir un terme (dt) sans lui associer de définition (dd).
Vous pouvez aussi associer plusieurs définitions (dd) à un seul terme (dt).
```

Voici donc quelques exemples :

```
<dl>
<dt>Éléments de niveau bloc</dt>
<dd>
  Éléments qui peuvent généralement contenir des blocs à leur tour, ou du contenu en ligne.
</dd>
<dt>Éléments de niveau texte</dt>
<dt>Éléments en ligne</dt>
<dd>
  Éléments qui ne peuvent contenir que des éléments en ligne.
</dd>
</dl>
```

Comme pour les autres balises, nous verrons dans la partie CSS comment rendre plus agréable ces éléments : comment différencier mieux les termes des définitions, comment les placer sur une ligne...

Enfin, ce genre de liste, comme dit précédemment, peut être utilisée pour autre chose que de simples définitions.

Ainsi, un calendrier peut utiliser cette structure : c'est une liste d'items, composée d'un court en-tête, la date, et d'un texte qui en dépend, le ou les événements. Comme nous le verrons plus tard, il est alors judicieux d'attribuer une classe différente à ces différentes utilisations de la liste de définitions afin de pouvoir les styler en conséquence.

Une pièce de théâtre est peut aussi être rendue à l'aide d'une liste de définition : les termes sont alors les personnages, et les "définitions" sont les répliques !

Voici un exemple de calendrier créé à l'aide d'une liste de définition :

```
<dl class="calendrier">
  <dt>18 février</dt>
  <dd>
    Journée du livre à Trifouilli les Oies.
  </dd>
  <dt>1 mars</dt>
  <dd>
    Anniversaire d'une grande personnalité.
  </dd>
</dl>
```

Les adresses

Avant de voir les liens et les images, il est nécessaire de s'arrêter quelques instants sur la façon de désigner ces fichiers qui sont externes à la page.

Il existe deux façons de décrire une adresse (un chemin d'accès au fichier, que ce soit une autre page ou une image) :

Les adresses absolues : on indique l'adresse complète de la page ou de l'image.

Par exemple : <http://www.kde.org/index.html> ou encore `C:\Site\index.html` .

Il ne faut les utiliser que pour des liens vers des autres sites. Certains logiciels insèrent des liens comme `C:/Site/index.html` : il est évident que sur les machines de vos visiteurs, ce lien ne marchera pas,

Les adresses relatives : à utiliser pour lier une page ou une image du même site. Le fichier sera recherché relativement à la page courante :

Peu importe l'endroit où se trouve la page (en local ou sur un serveur WEB) si l'image se trouve dans le même dossier on met que le nom de l'image (exemple : "image.png" si cette image se trouve dans le même dossier que la page WEB actuelle).

Si il se trouve dans un sous dossier, par exemple le dossier "fichiers", on indique simplement "fichiers/l_image.jpg". Tout simplement.

Si un fichier se trouve dans le dossier parent, on y accède grâce à "..". Par exemple, si la page WEB se trouve dans "monsieur.fr/tutorial/index.html" et l'image se trouve sur "monsieur.fr/image.png", il faut remonter un dossier : on accède à l'image par "../image.png".

Les liens

Évidemment, tout site qui se respecte contient des liens, que ce soit vers d'autres pages du site, vers d'autres sites ou vers des fichiers variés que le visiteur pourra télécharger.

C'est un peu spécial, car il n'y a qu'une seule balise pour deux concepts un peu différents :

`Texte du lien` : Lien vers adresseCible (adresse relative ou absolu : Hypertext REFERENCE) ; le texte qui sera visible (souligné) dans le navigateur est celui entre les deux balises ouvrante et fermante.

`` (ancre) : ceci place une ancre dans la page, qui pourra être atteinte grâce à un lien. Oui, c'est la même balise que pour un lien et on ne met rien entre la balise entrante et fermante : c'est pas très intuitif mais c'est comme ça ! Veillez bien à refermer la balise.

Petites explications sur les ancres, tout de même : si vous avez une page assez conséquente, il est assez judicieux d'afficher un sommaire en début de page, avec des liens qui mèneront aux paragraphes les plus importants. Un bon exemple est cette page du tutoriel : elle est assez grande et en début de page il y a une liste des paragraphes. Chaque paragraphe contient une ancre.

Attention : votre ancre doit être placée à l'intérieur d'un paragraphe, d'un titre, d'une ligne, etc... Mais pas

directement dans le corps de la page HTML.

Exemple de bon code : `<h2>Finalité</h2>`

Il faut juste remarquer que l'attribut href est utilisé pour un lien, et que l'attribut name pour une ancre. Et quand on veut l'utiliser pour faire un lien vers elle on a recours au caractère '#'.

Exemple : `Création du site` (le href contient un lien relatif vers le fichier liens.html du même dossier que celui qui contient le index.html) : lors du clic sur ce lien, liens.html sera chargée par le navigateur et celui-ci ira tout de suite à l'ancre dont le nom est donné après le '#' : cela évite au visiteur de devoir scroller pour trouver le paragraphe recherché dans la page,

Particularité : quand on écrit `Création du site` il s'agit d'une ancre qui est sur la page courante : pratique si l'on a de longues pages : il n'y a qu'à faire une liste en début de page en guise d'index, avec des liens vers les passages concernés pour une navigation plus rapide,

Autre particularité : `Haut de page` permet de revenir en haut de la page courante.

Exemple :

```
<ul>
  <li><a href="#premier_paragraphe">Aller au premier paragraphe</a></li>
  <li><a href="#second_paragraphe">Aller au second paragraphe</a></li>
</ul>
<p>Voici un texte contenant un <a href="une_page.html">lien vers une autre page du site</a>,
puis un autre <a href="http://www.kde.org/">lien vers un site externe</a>,
ainsi qu'un lien pour télécharger <a href="monprog3000_pro.zip">MonProgramme3000 Pro Edition</a>.</p>
<h2><a name="premier_paragraphe"></a>Premier paragraphe</h2>
<p>Avec de l'imagination, on voit ici un paragraphe très long mais très intéressant.</p>
<h2><a name="second_paragraphe"></a>Second paragraphe</h2>
<p>Encore un peu d'imagination...</p>
```

Les images

Afin de rendre les pages plus attractives, l'insertion d'images est un bon moyen simple et efficace (attention cependant à ne pas abuser des GIFs animés, vite fatigant et pouvant être assimilés à de la publicité).

Note importante : nous parlons ici de HTML, donc de contenu (oui, je radote !). Il n'est donc pas question ici de parler d'images de fond, ou de bordures et autre objets faisant partie du design du site : ces aspects seront traités par les feuilles de style CSS. Nous parlons ici d'images illustrant un article, des schémas, les photo de vacances, celles d'un produit, le logo du site, etc...

C'est ici assez simple, il n'y a qu'une balise :

`` (Image) : pas de balise fermante car il n'y a rien à mettre dedans (comme le br, et contrairement aux autres balises) : avec le fichier image dans le src (ça marche de la même façon que le href du lien) et un texte explication (alternatif) dans le alt (essayer de toujours en fournir un).

Note : La hauteur et largeur dans height et width sont là pour aider le navigateur à mettre en page alors que les images ne sont pas encore chargées (ça évite des pages qui "sautent" à chaque fois que le navigateur a chargé une image car il n'a pas fait assez de place pour celle-ci), c'est facultatif, mais je suis sûr que vous avez déjà visité une page avec un long texte et des images entre deux : vous commencez à lire et le texte se retrouve dix lignes plus bas d'un seul coup à cause d'une image qui s'est chargée au dessus.

Note 2 : le texte alternatif est obligatoire. Il sera affiché pendant le chargement de l'image, ou si elle n'a pas été trouvée. Mais aussi pour les navigateurs textuels ou brailles, histoire que vos visiteurs non-voyants puissent profiter des images aussi. S'il s'agit d'icônes, d'un logo ou d'une image qui n'a pas d'intérêt à être "lu" plutôt que "vu", spécifiez un texte alternatif vide, comme ceci : ``. Certains navigateurs affichent aussi ce texte dans une info-bulle (tooltip). Si vous souhaitez que le texte apparaisse en info-bulle dans tous les navigateurs, utilisez l'attribut title.

Note 3 : les images doivent toujours être positionnées dans un bloc, généralement un paragraphe.

Les balises de sémantique en ligne

Après avoir vu les balises basiques pour tout document (titres et paragraphes), en voici quelques autres qui permettent de donner une signification au texte ! Elles doivent être utilisées au lieu de balises pour mettre en gras, italique, etc...

`` : Exposant, exemples : Nous sommes le 1^{er} du mois, et $24 = 16$

`` : Indice, exemples : H₂O est la formule chimique de l'eau et C₂₁H₂₇NO celle du méthadone,

`` : Emphase (EMphasized, en anglais) : pour être plus clair, c'est un morceau de texte (celui entre la balise ouvrante et fermante) que l'on souhaite mettre en valeur : en italique par défaut (mais, comme le ``, ce sera modifiable par CSS : mettre en gras, couleur, souligner, transformer un texte en majuscules...).

`` : Emphase forte pour insister sur quelques mots importants (par exemple, tous les mots en gras dans ce tutoriel),

`<abbr title="Signification">Abbr.</abbr>` : abréviation : la signification doit être dans le titre, et l'abréviation doit être entre la balise ouvrante et fermante : cela donne des informations supplémentaires afin que le visiteur ne soit pas perdu dans les abréviations inconnues,

`<acronym title="Signification">S.I.G.L.E.</acronym>` : La même chose mais il s'agit là d'un Sigle (initiales),

`<dfn></dfn>` : Définition d'un terme, pour expliquer aux lecteurs la signification d'un mot ou d'un groupe de mot (par exemple dans un document technique qui se baserait sur cette connaissance pour le reste des pages),

`<cite></cite>` : Citation d'ouvrage, telle qu'un titre de magazine ou de journal, nom de vaisseau, référence à d'autres sources, ou citations d'auteurs (souvent rendue en italique),

`<q cite="Personne originale"></q>` : Citation exacte, ie. citation telle quelle, sans ajouter ou retirer du contenu à la phrase de l'auteur original. Doit être entouré par des guillemets par les navigateurs (vu que ce n'est pas le cas pour tous les navigateurs, nous verrons comment remédier à cela avec CSS).

`<del cite="http://www.site.org/changes.html#1.3" datetime="2004-02-18T00:00:00-05:00" title="Machin est obsolète">En utilisant Machin, vous serez à la pointe de la technologie !` : Partie supprimée (ce peut être un bloc ou un contenu en ligne). Les attributs sont facultatifs : `cite` indique une page expliquant la raison de la suppression ; `title` en donnant une courte explication (typiquement rendue dans une info bulle) et `datetime` indique la date et l'heure de suppression. Une feuille de style peut rendre cet item avec une couleur grise par exemple, le barrer, ou même le cacher,

`<ins cite="http://www.site.org/changes.html#1.4" datetime="2004-02-18T00:00:00-05:00" title="Truc est meilleur">En utilisant Truc, vous produirez mieux que Machin !</ins>` : Partie insérée : mêmes commentaires que pour ``. Peut être rendu en italique, dans une couleur différente (vert ?) ou une intonation différente,

Attention : ce n'est pas parce que par défaut le `` produit un effet gras et que le `` produit un texte en italique que vous pouvez les utiliser simplement pour mettre un texte en italique ou en gras : rappelez-vous que vous devez choisir ces balises pour sa signification et non pour son aspect visuel. Rappelez-vous qu'un lecteur en braille, par exemple, ne possède pas la notion de gras ou d'italique.

C'est volontairement que je ne vous donne pas les balises pour changer la fonte ou le formatage des caractères : ce n'est pas conseillé (même si ces balises existent) et cela sera fait en CSS : vous évitez ainsi d'apprendre comment faire des choses avec deux systèmes différents sachant que le CSS est meilleur (à mon point de vue). C'est aussi pour cela que vous n'avez pas beaucoup de balises HTML à connaître...

Les balises de code

Il se peut que vous fassiez un site sur les mathématiques, ou sur la programmation informatique (portail d'aide en PHP, ou tout simplement un tutoriel HTML comme celui-ci). Il est alors intéressant de marquer certaines parties de texte en tant qu'exemples de code source, de résultat obtenu, etc...

Voyons-voir comment faire :

`<code></code>` : Code informatique comme un extrait de source C++, ou même HTML (tel que dans ce tutoriel). Peut servir pour de petits morceaux à l'intérieur d'un paragraphe, ou pour encadrer un code source complet,

`<kbd></kbd>` : Texte à rentrer par l'utilisateur, à utiliser pour donner une chaîne que l'utilisateur doit taper, ou un raccourci clavier.. ,

`<samp></samp>` : Exemple de sortie pour délimiter un listing ou une sortie d'un programme, qu'il soit en ligne de commande ou dans une fenêtre graphique,

`<var></var>` : Variable mathématique ou variable dans un langage informatique (typiquement rendu en

italique mais les feuilles de styles peuvent permettre de le rendre en monospace par exemple), Il est à noter que ces balises sont toutes en ligne, c'est à dire qu'elle doivent être intégrée dans un bloc : un paragraphe (préformaté ou non), une ligne de liste...

Très souvent, les espaces dans un code source doivent être gardés tel quels. Dans ce cas, placez le code dans un paragraphe préformaté (bloc `<pre>`) : les espaces et retours à la ligne seront gardés sans avoir recours à des lourdes balises HTML : c'est l'une des rares exceptions de l'utilisation de `<pre>` (dans tous les cas, `<code>` doit être placé dans un paragraphe ou un autre bloc).

Les balises de sémantique bloc

Voici maintenant de nouvelles balises bloc pour structurer et donner une signification du contenu de vos pages. Certaines balises sont les pendants bloc de celles que nous venons de voir :

`<hr />` (Ligne horizontale) : sans balise fermante, elle permet de séparer deux blocs de texte par un filet horizontal... Nous apprendrons plus tard comment lui donner un aspect plus convivial,
`<blockquote cite="adresse"></blockquote>` : Bloc de citation pour citer tout un paragraphe (la réponse précédente dans un forum, par exemple). Ce n'est qu'un conteneur : vous devez y placer des paragraphes ou d'autres blocs à l'intérieur,
`<address></address>` : Informations de contact pour y placer les informations relatives à une personnes : nom, adresse, lien vers son site internet...
`` et `<ins></ins>` sont aussi des blocs en ligne (c.f. plus haut),
 Notez bien que `<cite>` est conçu pour des citations courtes, comme le nom d'un livre, et `<blockquote>` pour des... blocs, comme son nom l'indique, comme des citations lors de réponses dans un forum ou d'un e-mail...

Quelques exemples d'utilisation réelles, surtout pour `blockquote` et `address`, qui peuvent paraître étranges :

```
<blockquote cite="voir.php?message=precedent">
  Oui, moi je dit que le noir c'est mieux<br />
  Et je m'y connais.
</blockquote>
<p>Moi, par contre, je suis convaincu que le blanc c'est mieux.</p>
<hr />
<address>
  Si vous avez des questions sur l'enregistrement, contactez-nous à
  <a href="mailto:webmaster@site.org">webmaster@site.org</a>, ou téléphonez-nous au 09 76 34 67 12.
</address>
```

Les tableaux

Il vous faut parfois afficher une liste de résultats, nombre, et autres données tabulaires... Ceci est généralement fait dans un tableau.

Attention : utilisez un tableau si vous avez plusieurs colonnes (et de préférence un titre à mettre à ces colonnes) ou autres >>données tabulaires>>. Dans le cas contraire des puces feront peut-être mieux l'affaire.

Attention : je n'est pas dit que les tableaux c'est mal : mais beaucoup de gens s'en servent pour faire une mise en page (les cellules en bord de tableau accueillent des images décoratives qui forment un cadre...) ou en ayant juste en tête une idée de présentation et la réalisent avec de lourds tableaux. Il est conseillé de peser le pour et le contre avant d'utiliser des tableaux : gardez à l'esprit que pour l'instant nous faisons du HTML et que nous nous intéressons au contenu (donc, ne pas raisonner avec "Je veux les mots à gauche et la définition alignée à droite : j'utilise un tableau de deux colonnes").

Un tableau est entouré par les balises `<table summary="Texte"></table>`, où `summary` (facultatif) décrit le contenu du tableau (si non présent, veuillez à expliquer la raison du tableau dans un paragraphe),

Il est conseillé de donner une courte légende au tableau, pour cela on utilise la balise `<caption></caption>` : elle apparaîtra en haut du tableau,

Ensuite, on fourni un en-tête de tableau : `<thead></thead>`,

On indique ensuite le corps du tableau avec `<tbody></tbody>` juste après un `thead` : c'est lui qui contiendra les données (les lignes / cellules utiles) du tableau. Notez que vous pouvez avoir plusieurs corps dans votre tableau : cela peut servir à grouper certaines lignes entre elles (ça ne se verra pas dans le navigateur, mais nous pourrons modifier se comportement plus tard (par exemple encadrer les différents corps),

Et, éventuellement, on peut mettre un pied de tableau avec `<tfoot></tfoot>`.

Une fois la structure du tableau en place, voici comment créer des lignes et des cellules :

À l'intérieur de chacun des blocs `tbody`, on définit une ligne de tableau avec `<tr></tr>` (Table Row en anglais) (plusieurs lignes pour un `tbody`, et (idéalement) une seule pour les `thead` ou un `tfoot`),

Dans chaque lignes, on place des cellules avec `<td></td>` (Table Data en anglais) : les données sont contenues à l'intérieur (enfin !),

Par contre, à l'intérieur des blocs `thead` et `tfoot`, on définit également une ligne avec `<tr></tr>`, mais les cellules sont définies à l'aide des balises `<th></th>` (Pour Head).

Après cette flopée de détails, voici ce que cela donne, c'est long, donc vous pourrez le copier / coller directement :

```
<table summary="Cette table liste les unités SI (Système International),
          en donnant le nom et le symbole de ces unités.">
  <caption>Unités SI (partiel)</caption>
  <thead>
    <tr>
      <th>Nom</th>
      <th>Symbole</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>mètre</td>
      <td>m</td>
    </tr>
    <tr>
      <td>kilogramme</td>
      <td>kg</td>
    </tr>
  </tbody>
</table>
```

Parfois, il peut être intéressant de regrouper des cellules. Il faut alors utiliser l'attribut `colspan`, pour répandre une cellule sur plusieurs colonnes, ou `rowspan`, pour étendre une cellule sur plusieurs lignes.

Un exemple valant mieux qu'un long discours :

```
<table summary="Exemple de colspan et rowspan">
  <caption>Exemple de colspan et rowspan</caption>
  <thead>
    <tr>
      <th colspan="2">Nom</th>
    </tr>
    <tr>
      <th>Bidule</th>
      <th>Machin</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td rowspan="2">Cellule sur deux lignes</td>
      <td>Voilà...</td>
    </tr>
    <tr>
      <td>Et la dernière ligne</td>
    </tr>
  </tbody>
</table>
```

Structurer des pages complexes

Dans les sites plus complexes qu'un enchaînement de titres et de paragraphes, il est intéressant de séparer les éléments du site en plusieurs blocs.

C'est là qu'interviennent les balises génériques sans signification particulière, mais qui pourra avoir un style différent plus tard :

`<div></div>` (bloc générique) : il a à peu près le même comportement qu'un paragraphe (retour à la ligne avant et après) mais utilisé comme un bloc pour regrouper des informations (et n'a pas de marges comme le paragraphe en a par défaut, mais nous en reparlerons dans la partie CSS).

Il est courant d'identifier plusieurs blocs : l'en-tête (affiché, avec bandeau de publicité, etc...), le menu (à droite, en longueur...), le contenu de la page (l'article...) et le pied de pages (copyrights...) par exemple. Nous verrons bientôt comment les utiliser : retenez qu'il s'agit juste d'un niveau de hiérarchie en plus : vous écrivez votre page à l'intérieur comme si ces balises n'étaient pas là.

`` (Contenu "en ligne") : on ne passe pas de ligne avant et après : rendu de la même façon que acronym, abbrev ou em mais par défaut il n'y a pas moyen de le distinguer visuellement : servira à définir ses propres sémantiques.

Pour l'instant, vous ne devez pas trop en voir l'intérêt. Nous en reparlerons dans la prochaine partie, car ces balises sont souvent utilisées afin de structurer les pages et de donner un style à cette structure grâce au CSS.

Les bases en CSS

Nous venons de voir comment créer des pages HTML qui ne contiennent que le contenu.

Ce n'est pas très beau, et c'est normal : nous allons maintenant aborder le CSS, le format qui vous permettra de définir la forme. Mais avant de s'amuser avec, nous devons d'abord voir comment on le forme : c'est la partie théorie.

Note : si la théorie ne vous dit rien, vous pouvez lire la page sur les techniques en CSS, où nous allons directement mettre en forme la ou les page(s) créées précédemment. Cependant, si vous ne comprenez pas des choses, n'hésitez pas à revenir sur cette page et à la lire !

Introduction au CSS

La structure d'un fichier CSS est simple (plus que celle d'un fichier HTML).

Pour faire simple, le CSS sert à appliquer des styles à une balise HTML, certaines balises, ou un groupe de balises,

Il se compose de règles, débutants chacune par les balises concernées (leur nom, ou une expression permettant de sélectionner les balises concernées). Ensuite, entre accolades (entre { et }), on définit les règles que l'on souhaite appliquer,

Chaque règles commence par le nom de la propriété à modifier, suivi de deux points (':'), puis des valeurs à donner à cette propriété, terminé par un point virgule (;').

C'est assez abstrait comme description : ne vous cassez pas la tête car nous allons tout de suite voir ce que cela donne concrètement, et si vous ne comprenez toujours pas, passez simplement aux exemples concrets qui vous aideront à y voir plus clair car se sont des exemples qui vous serviront (que vous pourrez recopier et adapter).

Exemples

Pour commencer, nous allons voir comment centrer le titre de niveau 1 de nos pages WEB.

Vous savez que celui-ci s'écrit `<h1>Mon titre de page</h1>` : il utilise la balise h1, donc nous devons appliquer un style à la balise h1. Nous voulons centrer le texte, il s'agit donc de la propriété `text-align` (ne vous inquiétez pas : je vous donnerai la liste des propriétés), et nous devons lui donner la valeur `center`. Cela se traduit donc par :

```
h1 {
  text-align: center;
}
```

Si maintenant nous voulons une couleur bleue pour ce même titre, nous devons appliquer une seconde règle pour la balise h1 : il s'agit de la propriété `color`, et nous lui donnons la valeur `blue` ; ce qui donne :

```
h1 {
  text-align: center;
  color: blue;
}
```

Comme vous le voyez, c'est simple. De manière générale, le code CSS balise { propriété: valeur; } servira à relooker le contenu de la balise `<balise></balise>` du fichier HTML correspondant.

Les sélecteurs

Sans le savoir, vous avez utilisé un sélecteur : h1. Qu'est-ce donc ?

C'est une petite expression qui vous permet de sélectionner à quelles portions du code HTML s'applique un ensemble de règles.

Vous avez utilisé un sélecteur pour choisir une seule balise. Voici :

Vous pouvez très bien spécifier deux balises séparées par un espace : par exemple `li ul` permet de ne sélectionner que les balises `` contenue dans des balises `` (soit, comme vu dans le tutoriel HTML, une sous-liste). Avec notre exemple :

```
<ul>
```

```

<li>
  <ul>
    <li>Cette sous-liste est concernée car elle est définie
      par une balise ul, elle même contenue dans une balise li</li>
    </ul>
  </li>
  <li>Mais pas cette liste</li>
</ul>

```

Remarque : il est bien dit que la seconde balise doit être incluse dans la première. Mais il n'est pas obligé que ce soit la balise directement fille,

Si vous tenez à ce que la règle s'applique à la balise directement fille (dans notre exemple précédent, pour les balises ul contenues directement dans une balise li), vous devez utiliser le sélecteur `>`. Par exemple :

```

p > img {
  vertical-align: middle;
}

```

Cette règle ne s'appliquera qu'aux balises img contenues directement dans un paragraphe (p) : les images contenues dans une liste, elle même contenue dans un paragraphe ne se verront pas appliquer la règle.

Astuce : vous pouvez utiliser la balise body pour n'affecter que les balises de premier niveau (non imbriquées : celles contenues directement dans le <body>). Par exemple, pour ne sélectionner que les titres h2 contenues à la racine (utile pour exclure les titres h2 contenues dans des <div>, entre autre), vous pouvez spécifier body > h2,

Lorsque deux noms de balises sont séparés par un virgule (','), alors l'ensemble de règles s'appliquera aux deux balises. Cela permet d'éviter ce genre de code dupliqué :

```

balise1 {
  background-color: white;
  color: blue;
  text-decoration: underline;
}
balise2 {
  background-color: white;
  color: blue;
  text-decoration: underline;
}

```

en le remplaçant directement par :

```

balise1, balise2 {
  background-color: white;
  color: blue;
  text-decoration: underline;
}

```

Il peut arriver que vous souhaitiez sélectionner n'importe quelle balise : dans ce cas, à la place d'un nom de balise particulier, utilisez l'étoile (*'). Pour sélectionner toutes les balises directement filles du corps mais pas les sous balises, vous en aurez besoin et écrirez ceci :

```
body > *
```

Les classes

Il y a encore un autre moyen de restreindre la sélection : et celui-ci est très puissant (et très utilisé) : il s'agit des classes.

Pour mettre en place ce moyen, il faut modifier quelque peu notre fichier HTML : par exemple, modifiez l'un de vos paragraphes (ou titre, ou ce que vous voulez) pour y inclure l'attribut class="exemple". Pour être plus précis, si vous souhaitez appliquer une classe exemple à un paragraphe, modifiez la balise ouvrante du paragraphe comme ceci : <p class="exemple">.

Vous commencez peut-être à comprendre : en ayant défini cette classe, nous allons pouvoir sélectionner plus précisément certaines balises, et pas toutes les balises de même nom. Voici comment la sélectionner dans votre code CSS:

```
.exemple {
  text-decoration: underline;
}
p.exemple {
  color: red;
}
p.exemple a {
  background-color: yellow;
}
```

Cette avalanche de CSS vous montre un bon éventail de l'utilisation des classes : la première règle stipule que toutes les balises avec l'attribut class égal à exemple doivent être soulignés ; que ce soit un titre, un paragraphe, un lien, un div, un span...

La seconde règle sélectionne précisément un paragraphe avec l'attribut class=exemple.

Le troisième exemple montre un composite entre les classes et les sélecteurs appris un peu plus haut : il s'applique à tout lien contenu dans un paragraphe d'exemple.

Je vous laisse le soin de définir une règle pour sélectionner juste les liens de classe important, contenue directement dans un paragraphe, lui même contenu dans une balise de classe contenu :-)

Les priorités et héritages

Il arrivera souvent que plusieurs règles s'applique à une balise particulière. Par exemple, pour :

```
<div>
  <p class="important">
  </p>
</div>
```

vous pouvez avoir défini trois règles :

```
p {}
.important {}
div > p {}
```

Dans ce cas, toutes les règles se superposent mais si quelques propriétés sont définies plusieurs fois, seule celle ayant la plus haute priorité est appliquée : il s'agit de la plus précise. Ceci est assez intuitif : div>p est plus précis que simplement p, idem : .important est aussi plus précis.

Comme on peut s'y attendre, si on définit une couleur de texte pour une balise, les balises filles hériteront aussi de cette propriété (sauf si une règle plus prioritaire dit le contraire). Par contre, et c'est logique, si on définit un cadre autour d'un bloc (un paragraphe, par exemple), les balises filles (des liens, des listes... par exemple) n'hériteront pas de cette propriété : cela donnera des cadres dans des cadres, et fera du plus mauvais effet. En règle général, les propriétés qui sont héritables ou pas sont assez intuitives, mais vous pouvez vous conforter à la doc. ... Bloc_VS_Text ???

Styler ses pages avec CSS

Après avoir appris le fonctionnement général des feuilles de style, passons à la pratique. Tout d'abord, nous allons voir comment styler vos pages HTML de façon simple : nous verrons que la séparation du contenu et de la forme, et plus précisément n'utilise "que" le HTML Strict, n'est pas faite au détriment de la présentation (ceux connaissant déjà le HTML avant ce tutoriel verront que toute leur présentation peut être faite en CSS : s'ils changent il gagneront de la bande passante et pourront redéfinir leur style plus aisément en modifiant un seul fichier).

Fort de ceci, nous verrons alors comment tirer pleinement parti du CSS pour définir des effets impossibles en HTML Transitionnel (autorisant encore les balises et attributs de mise en forme) : c'est la partie amusante de tout le tutoriel !.

Enfin, nous verrons comment contenter les plus exigeants, et notamment comment se passer de tableaux HTML purement présentationnels en définissant plusieurs colonnes, et d'autres structures inévitables pour les sites avancés.

Nous allons commencer doucement en apprenant comment mettre en forme des pages tel qu'on le faisait en HTML.

Si vous avez déjà fait un site mais sans CSS, avec des balises font partout, des mises en italique ou en gras ou en souligné... vous verrez pourquoi je ne vous ai pas appris ces balises :depreciated: sur la page HTML : on peut faire la même chose en CSS et ainsi, la forme et le contenu sont bien dissociés, et surtout, vous n'avez pas besoin de modifier toutes vos pages HTML au moindre changement de design : tout ce fera dans le CSS que vous définirez une seule fois.

Ensuite, dans les paragraphes suivants et une fois que vous aurez bien saisi que la mise en forme dans le HTML est inutile, nous verrons que le CSS permet bien plus : il permet de magnifiques effets que le HTML ne permettait pas. Et là vous vous réglerez, et je vous promets que vous n'aurez pas appris le HTML Strict plus le CSS pour rien.

Je vous conseille d'ouvrir un nouvel onglet dans votre navigateur, si votre navigateur le permet (cliquez du milieu sous Firefox, Maj+clic pour ouvrir une nouvelle fenêtre dans Internet Explorer) avec la page d'Aide-mémoire CSS, ainsi vous pourrez vous documenter sur les propriétés utilisées.

IDEA: Voici ce qui est possible en HTML ; version CSS. Ensuite, on verra les mêmes balises, mais stylées de façon autrement intéressantes : possible uniquement en CSS.

MIEUX: Nous allons maintenant suivre la progression du tutoriel HTML : nous allons styler chaque balise une par une, et identifier les choses qui sont possibles de faire : à vous d'appliquer les styles que vous souhaitez avoir et à les modifier pour avoir une présentation unique et originale. À la fin de cette page vous aurez un site fonctionnel !

La page

La page entière est définie par le sélecteur body

Nous allons commencer par donner une couleur de fond à la page : c'est la propriété background-color. Si vous voulez une image de fond, utilisez background-image en donnant une URL.

Si vous définissez une image de fond, il est conseillé de définir aussi une couleur de fond en même temps, grâce à la propriété background. Choisissez la couleur de fond la plus proche de celle de l'image. En effet, lors du chargement de l'image, c'est la couleur qui est utilisée (par exemple si vous utilisez une image très noire et un texte blanc, vous avez intérêt à choisir noir comme couleur de fond, sinon avant que l'image soit affichée le visiteur verra un texte blanc sur fond blanc !!

Nous pouvons aussi spécifier des marges afin que le texte ne soit pas collé contre les bordures de la fenêtre : propriété margin à 1em par exemple !

Jusque là, ce n'est pas bien compliqué, voici deux exemples de pages :

```
body {
  background-color: #BADBFF; /* Fond de page bleu clair */
  margin: 0; /* Supprimer toute marge à la page : le texte colle les bordures */
}
```

Voici une page qui définit une couleur d'arrière plan unie et remet à zéro les marges du document. Notez que

la suppression des marges nuit à la lisibilité, surtout dans une vraie page où le texte sera littéralement collé aux bordures de la fenêtre ou de l'écran. Ceci peut pourtant s'avérer utile pour des pages complexes, ou les éléments décoratifs sont censés prendre toute la page. Dans ce cas les textes (titres, paragraphes, listes...) devront se voir attribuer soit une marge soit un espacement différent de zéro. Nous verrons cela dans la prochaine section (sur les titres).

```
body {
  background: #D5DEEC url('fond.png'); /* Fond de page bleu clair, avec une image de fond (bleu clair aussi) */
  font: 14px sans-serif;           /* Style et taille de la police de la page */
  margin: 1em;                     /* Marge du document HTML */
}
```

Puis maintenant, une deuxième page avec une marge de 1em (c'est à dire une fois la taille d'un caractère dans la police du bloc) et un fond défini par une image bleue, et une couleur bleue, en attendant que l'image se charge (il est important de veiller à l'adéquation de la couleur dominante de l'image avec celle de fond : le visiteur pouvant désactiver les images, il serait dommage qu'il lise votre site en blanc sur blanc avec le simple prétexte que votre image de fond était principalement noire).

Toutes les balises (et plus particulièrement les textes) hériteront de la police de caractère (serif c'est une police avec empatements (Times New Roman sous Windows...), sans-serif c'est sans empatements (Arial sous Windows, Helvetica sous MacOS...), ou monospace pour une police à chasse fixe (Courier New...)), donc pas besoin de la redéfinir plus tard.

```
body {
  background: white url('katie.jpg') no-repeat 50% 50%; /* Fond blanc non répété et centré */
  background-attachment: fixed; /* Même si la page scroll, l'image reste au centre de l'écran.
  Vous pouvez tester sans :
  dans ce cas (et avec de grandes pages) l'image restera collée au bord/centre
  de la page et pas de l'écran */
}
```

Voici maintenant une image non répétée et centrée dans la fenêtre du navigateur.

Essayez de redimensionner la fenêtre de votre navigateur pour tester.

Il est également possible de placer l'image dans un coin (en bas à droite, au milieu et à 75% en bas...).

Je vous laisse découvrir les autres valeurs de background-position.

Et comme il faut bien avoir une hauteur suffisante pour tester l'image de fond, je rajoute du texte.

Les paragraphes

Les navigateurs n'utilisent pas tous la même police de caractères. Généralement, c'est celle définie par l'utilisateur (ou plus souvent celle par défaut). Si vous désirez changer la police de caractères, c'est font-family, et la taille font-size ! Les deux peuvent être regroupés en font. Pour l'écran, une police sans-serif (Arial or Helvetica...) est plus lisible (sur papier, une police serif (Times New Roman, par exemple) est préférable).

Enfin, pour faire plus joli, vous pouvez demander à justifier les paragraphes, via text-align: justify;

Vous pouvez même vous permettre un formatage particulier pour la première lettre des paragraphes avec le sélecteur p:first-letter. Nous n'allons pas nous gêner :-)

Tout ceci nous donne cette règle CSS (avec un résultat un peu fantésiste mais le principal y est) :

```
body {
  background: #D5DEEC url('fond.png'); /* Fond de page bleu clair, avec une image de fond (bleu clair aussi) */
  font: 14px serif;                 /* Style et taille de la police de la page */
  margin: 1em;                     /* Marge du document HTML */
}
p, li {
  text-align: justify;             /* Justifier le texte des paragraphes et des lignes de listes */
}
p:first-letter {
  font-size: x-large;             /* Utiliser une police large */
  font-weight: bold;              /* Et grasse */
  color: #0000AA;                 /* De couleur bleu foncée */
}
```

Ceci est un paragraphe montrant le style général de la page (l'image est un simple dégradé). Vous pouvez

aussi vous rendre compte de la justification du paragraphe ! Dans le cas contraire, redimensionnez la fenêtre de votre navigateur afin de faire tenir ce paragraphe sur plusieurs lignes. En espérant que cela suffira, mais ce n'est pas bien grave.

Les titres

Après la page et les paragraphes, nous voici maintenant arrivé aux titres !

C'est ici que vous pouvez être le plus fantaisiste : on va s'amuser :-) Mais on va aussi utiliser beaucoup de propriétés !

Pour personnaliser le look des titres, nous avons aussi besoin de la propriété `color` pour la couleur du texte. Vous pouvez aussi vous amuser avec `background-color` ou `border` pour entourer les titres (plus particulièrement le titre de niveau 1). `text-decoration: underline`; est aussi intéressant pour les souligner, et `text-align` pour centrer le titre de page, par exemple...

C'est parti :

```
h1 { /** gros titre */
  font-family: serif; /* On se permet un changement de police : pas plus de deux par page, et à utiliser
avec parcimonie (c'est le cas) */
  background: #0066CC none; /* Fond du titre bleu (et sans image de fond) */
  color: white; /* Donc on utilise une police blanche pour que le texte soit visible */
  border: 3px dotted #9999FF; /* On encadre le titre d'un bleu clair de 3 pixels de largeur */
  padding: 0.3em; /* Espacement intérieur non nul pour que le texte ne colle pas à la bordure du cadre */
  text-align: center; /* Le titre doit être centré ! */
  letter-spacing: 0.3em; /* On espace les caractères pour que ce soit joli :-) */
}
h2 { /* Les titres de paragraphes */
  text-decoration: underline; /* On souligne ces titres */
  font-variant : small-caps /* Et on les met en majuscules (les minuscules sont remplacées par des petites
majuscules) */
  color: #000077; /* Texte de couleur bleu très foncé, presque noir (donc lisible) */
}
h3 { /* Les sous-titres (titres de niveau 3) */
  color: #2222CC; /* Texte de couleur bleu, plus clair que le titre h2 */
  font-style: italic; /* Texte en italique */
  text-indent: 1em; /* Et indenté par rapport aux autres titres et paragraphes, pour montrer la hiérar-
chie */
}
```

Les listes

On arrive à quelque chose plus intéressant. En effet, beaucoup de personnes utilisent de vulgères images à côté d'un texte pour simuler des listes. Si vous avez suivi mes conseils, vous avez bien utilisé la balise `` pour créer des listes.

Vous voulez surement avoir autre chose qu'un simple rond noir ! Pas de problèmes :

```
li {
  list-style-image: url("dossier.png");
}
```

Ceci est une liste

Avec une image personnalisée en tant que puce

Mais bien évidemment, vous pouvez mettre une image de petite puce ronde avec texture

Ou laisser aller vos envies :-)

Et pour être un peu plus exhaustif, il suffit de coder `<li class="ouvert">` et d'associer l'image "dossier_ouvert.png" à une telle ligne (li.ouvert) pour créer une arborescence :-). Voir même associer l'image "fichier.png" à la règle `ul li` pour affecter (styler) les sous-listes (`ul li` est plus précis que `li` donc il l'emporte)...

À noter qu'il n'est pas possible d'avoir une puce d'une autre couleur que celle du texte. Il y a deux astuces pour y arriver :

Soit mettre tout le texte de chaque li dans un span et : utiliser `color: black` pour ce span et la couleur de la puce pour les li,

Soit utiliser une image qui représentera cette puce en couleur, c'est l'astuce utilisé par ce site et ça ne néces-

site pas de modifier tout le site pour rajouter les span à chaque ligne de liste (meilleure séparation du contenu et de la forme).

C'est fini : vous utilisez du vrai HTML pour faire des listes, vous ne polluez pas vos pages avec des images et celles-ci sont visibles si un visiteur désactive les images (il verra des ronds noirs, mais c'est son choix : nous le respectons très bien ici) et les lecteurs d'écran ou navigateurs braille sauront rendre le fait qu'il s'agit d'une liste. C'est pas joli ? ;-)

Les liens

Viennent ensuite la personnalisation des liens : c'est le sélecteur `a` pour les liens normaux, `a:visited` pour les liens déjà visités, `a:hover` pour le lien actuellement sous la souris et `a:active` pour le lien actif (sélectionné en utilisant les touches de tabulation).

Un exemple assez simple qui ne modifie que les couleurs :

```
a          { color: #0066CC; } /* Liens en bleu (entre clair et foncé) */
a:visited  { color: black;   } /* et en noir (comme le texte normal, mais souligné) si déjà visités */
a:hover, a:active { color: white;          /* Lors du passage de la souris ou activation, */
               background-color: #0066CC; } /* couleur blanc sur fond bleu          */
```

Voici un paragraphe avec des liens, et encore des liens, et toujours de nouveaux liens ! De quoi tester le rendu des liens en CSS, comme par hasard ;-)

Ceci n'est qu'un petit éventail des possibilités du CSS : regardez l'aide mémoire fourni en annexe et testez d'autres solutions avec d'autres propriétés... Vous pouvez mettre en gras les liens lors de leur survol, en italique ou le désouligner... Vous pouvez aussi le désouligner mais mettre en gras (pour le distinguer du texte normal)...

Les images

Vous avez sûrement remarqué que si vous avez une image à l'intérieur d'un lien (typiquement ``) les navigateurs ajouteront une bordure (souvent bleue, de deux ou trois pixels de largeur) autour de l'image.

La majorité des visiteurs, et moi le premier, ne trouve pas ça joli ! Qu'à cela ne tienne, nous allons rajouter une règle CSS pour faire disparaître cela (ceux qui connaissaient déjà le HTML avaient sûrement recours à l'attribut `border`, à chaque image : ce temps est révolu) :

Il est à noter que vous pouvez aussi spécifier la propriété `vertical-align` pour aligner verticalement vos images par rapport au texte qui les entourent.

```
img {
  border: none;          /* Ne pas entourer les images, même s'ils contiennent des liens */
  vertical-align: middle; /* Dans le cas d'une icône suivie d'un texte, par exemple */
}
```

Les tableaux

Nous voici arrivés aux tableaux, ces structures qui vous permettent de structurer des données tabulaires. Il y a beaucoup de choses stylables dans les tableaux ; voici ce qui peut être fait :

`<table></table>` : Tableau : vous pouvez faire à peu près tout ce qui est possible de faire sur un paragraphe : fond, bordures, formatage du texte...

`<caption></caption>` : Légende de tableau : idem, tout ce que vous voulez,

`<thead></thead>` : En-tête de tableau : ici c'est assez restreint, vous pouvez jouer sur le fond de chaque case qui les compose, le formatage du texte, mais pas de bordures,

`<tbody></tbody>` : corps du tableau : pareil qu'au dessus,

`<tfoot></tfoot>` : Pied de tableau : idem,

`<tr></tr>` : Ligne de tableau : encore pareil,

`<td></td>` : Cellule de tableau : là vous pouvez jouer sur les bordures des cases, les paddings et margins ! Vous pouvez aussi renseigner la propriété `vertical-align` pour aligner le contenu des cases en haut, au centre, ou en bas de celles-ci,

`<th></th>` : Cellules d'en-tête pour tableaux, pareil que pour les cases.

```

table {
  width: 50%; /* La largeur c'est pour le test, mais ça montre qu'il est aussi possible de la spécifier en CSS
*/
  border: 1px dashed gray; /* Et on met une jolie bordure originale autour du tableau */
}
caption {
  text-align: center; /* Classique : texte centré */
  font-style: italic; /* en italique */
  font-weight: bold; /* en gras */
  color: #0066CC; /* et en bleu */
}
th { /* Préférable à thead car offre plus de possibilités */
  background-color: #D3E2ED; /* Chaque case d'en-tête aura un fond bleu */
  color: gray40; /* Et un gris un peu plus foncé que la moyenne */
}
td {
  /*border: 1px inset gray;*/ /* Vous pouvez essayer de décommenter : cela produit un affichage classique
des cases */
}

```

Les filets de séparation horizontaux

Deux solutions : soit garder l'aspect filet simple et jouer sur les couleurs et la taille, soit carrément utiliser une image (et oui : pas besoin de "tricher" en utilisant une image HTML, un "vrai" filet suffit et on va juste le styler). C'est parti pour les deux solutions :

```

hr.f1 { /* Premier filet bleu clair */
  border: none; /* On supprime la bordure (inset par défaut)... */
  background-color: #66A8E7; /* ... que l'on remplace par un fond bleu uni */
  height: 1px; /* On définit la hauteur de 1 pixel pour que le fond soit visible ! */
  width: 70%; /* Et on précise que l'on désire un filet de longueur 70%... */
  align: center; /* ... et centré, par exemple */
}
hr.f2 { /* Second filet bleu foncé, en pointillés */
  border: none; /* Idem : on ne veut pas des bordures... */
  border-top: 6px dashed #000099; /* ... on en veut juste UNE en pointillés, de hauteur 6 pixels */
  height: 0px; /* Le filet est rendu grâce à la bordure haute : pas besoin du "contenu" (si on peut
dire) */
  width: 80%;
  align: center;
}
hr.f3 { /* Troisième type de filet, avec une image de fond, cette fois */
  border: none; /* Encore une fois, pas de bordures */
  background-image: url('travaux.png'); /* L'image à répéter... */
  height: 4px; /* ... sur une hauteur de 4 pixels */
  width: 90%;
  align: center;
}
hr.f4 { /* Dernier filet */
  border: none;
  background: url('hr_gradient.png') #89B5FF; /* Cette fois on veut une image */
  background-repeat: no-repeat; /* non répétée, et on poursuit avec la couleur bleu une fois l'image
terminée */
  height: 2px;
}

```

Aide-mémoire HTML

Les balises de bloc

`<h1></h1>` : Titre de page (une seule fois en tout début),
`<h2></h2>` à `<h6></h6>` : Sous-titres (hiérarchisés du plus important (h2) au plus précis (h6)),
`<p></p>` : Paragraphe (la plupart des textes de vos pages s'y trouvent),
`<div></div>` : Division (bloc générique),

Les balises en ligne

`
` : Saut de ligne,
`Texte` : Lien Texte vers l'adresse cible,
`` : Image contenue dans le fichier `adresse_image`,
`` : Ancre (pour faire un lien vers une ancre : `Texte_du_lien`).

Les balises de sémantique en ligne

`` : Exposant,
`` : Indice,
`` : Emphase (EMphasized, en anglais) (usuellement en italique).
`` : Emphase forte (usuellement en gras),
`<abbr title="Signification">Abbr.</abbr>` : Abréviation,
`<acronym title="Signification">S.I.G.L.E.</acronym>` : Sigle (ininitiales),
`<dfn></dfn>` : Définition d'un terme ,
`<cite></cite>` : Citation,
`<del cite="adresse" datetime="2004-02-18T00:00:00-05:00" title="Explication">` : Partie supprimée (bloc ou en ligne),
`<ins cite="adresse" datetime="2004-02-18T00:00:00-05:00" title="Explication"></ins>` : Partie insérée (bloc ou en ligne),
`<code></code>` : Code informatique,
`<kbd></kbd>` : Texte à rentrer par l'utilisateur (texte ou touche(s)),
`<samp></samp>` : Exemple de sortie,
`<var></var>` : Variable (mathématique ou informatique),
`` : Bloc en ligne générique (utilisé pour définir une langue ou un style différent, par exemple)
`q cite=""` (short Quotation) : court : équivalent inline de blockquote

Les balises de sémantique bloc

`<hr />` : Ligne horizontale,
`<blockquote cite="adresse"></blockquote>` : Bloc de citation,
`<address></address>` : Informations de contact,
`` et `<ins></ins>` sont aussi des blocs en ligne (c.f. plus haut),

Les listes

`` : Liste non ordonnée (lise à puces, rondes ou carrées),
`` : Liste ordonnée (numérotée, 1. pour le premier, 2., etc...),
`` : Ligne de liste (impérativement à l'intérieur d'un `` ou d'un ``),
`<dl></dl>` : Liste de définitions,
`<dt></dt>` : Item de Terme à définir,
`<dd></dd>` : Item de Définition.

Les tableaux

`<table summary="Texte"></table>` : Tableau, où `summary` (facultatif) décrit le contenu du tableau,
`<caption></caption>` : Légende de tableau (ne peut contenir que des blocs en ligne),
`<thead></thead>` : En-tête de tableau,

`<tbody></tbody>` : corps du tableau,
`<tfoot></tfoot>` : Pied de tableau,
`<tr></tr>` : Ligne de tableau (Table Row),
`<td></td>` : Cellule de tableau (Table Data),
`<th></th>` : Cellules d'en-tête pour tableaux.

Autres attributs

Ces attributs sont valables pour toutes les balises

`title="info_bulle"` : Informations complémentaires, habituellement affichées dans des info-bulles.
`lang="fr"` : Langue du contenu, si elle est différente de celle de la page : fr, en, de... ,
`dir=""` : Direction de lecture, de gauche à droite () ou de droite à gauche,
`class="nom_classe"` : Classe CSS,
`id="nom_id"` : Identifiant CSS ou Javascript,
`style="règle_style"` : Style CSS en ligne, fortement déconseillé (utiliser plutôt une classe),

Rappel sur les adresses (liens et images)

Les adresses absolues : adresse complète d'un fichier. Par exemple : `http://www.kde.org/index.html` ou encore `C:\Site\index.html` . Fortement déconseillé pour des liens à l'intérieur du site,

Les adresses relatives : adresse relative à la page courante d'un fichier. À utiliser pour lier une page ou une image du même site.

`"image.png"` : ce fichier se trouve dans le même dossier que la page en cours,

`"fichiers/image.png"` : ce fichier se trouve dans le sous dossier "fichiers/", relativement à l'emplacement de la page courante,

`"../image.png"` : ce fichier se trouve dans le dossier parent de celui qui contient la page en cours.

Squelette de tout fichier HTML

Vous pouvez très bien faire un copier / coller de cette structure pour démarrer une nouvelle page (ce qui est dans le body est juste une suggestion de structure) :

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html lang="fr">
  <head>
    <title>Titre de la page</title>
  </head>
  <body>
    <h1>Titre de la page</h1>
    <p class="intro">Introduction</p>

    <h2>Première partie</h2>
    <p>Premier paragraphe</p>
  </body>
</html>
  
```

Squelette de deux listes imbriquées

```

<ul>
  <li>Ligne</li>
  <li>Ligne qui introduit la sous-liste :
    <ul>
      <li>Ligne</li>
    </ul>
  </li>
</ul>
  
```

Aide-mémoire CSS

Propriétés de la boîte

margin : Marge : all ; haut+bas gauche+droite ; haut droit bas gauche

Si deux ou trois valeurs données, elle est prise du bord opposé.

margin-top : Marge de haut

margin-right : Marge de droite

margin-bottom : Marge de bas

margin-left : Marge de gauche

padding : Espacement : all ; haut+bas gauche+droite ; haut droit bas gauche

Si deux ou trois valeurs données, elle est prise du bord opposé.

padding-top : Espacement de haut

padding-right : Espacement de droite

padding-bottom : Espacement de bas

padding-left : Espacement de gauche

border <border-width> || <border-style> || <color> : Bordure (les quatres)

Si deux ou trois valeurs données, elle est prise du bord opposé.

border-top : Bordure de haut

border-right : Bordure de droite

border-bottom : Bordure de bas

border-left : Bordure de gauche

border-width : Taille de bordure : all ; haut+bas gauche+droite ; haut droit bas gauche

Si deux ou trois valeurs données, elle est prise du bord opposé.

border-width-top : Taille de bordure de haut

border-width-right : Taille de bordure de droite

border-width-bottom : Taille de bordure de bas

border-width-left : Taille de bordure de gauche

border-color couleur : Couleur de la bordure

Vous pouvez donner entre un et quatre paramètres : les valeurs manquantes seront reprises du bord opposé.

border-style : Style de ligne de la bordure. Vous pouvez donner entre un et quatre paramètres : les valeurs manquantes seront reprises du bord opposé.

none

dotted

dashed

solid

double

groove

ridge

inset

outset

width : Longueur

Longueur (si pourcentage : par rapport à la longueur de l'élément parent)

auto

height : Hauteur

Longueur (sauf pourcentages)

auto

float : Bloc flottant ou pas, permet de le 'sortir' du bloc actuel ; le texte (et contenu) environnant sera 'word-wrapped' autour.

none

left

right

clear : Mettre des blocs flottants à côté : Si clear est défini sur un côté (ou les deux), aucun élément float ne sera autorisé de ce côté ; et s'il y en avait déjà, le bloc est déplacé vers le bas jusqu'à satisfaire la demande.

none

left

right

both

Couleur et propriétés du fond de bloc

color couleur : Couleur de texte (et de soulignement et de bordure par défaut)

background : <background-color> || <background-image> || <background-repeat> || <background-attach-ment> || <background-position> : Fond de bloc

background-color : Couleur de fond du bloc

couleur

transparent

background-image : Image de fond du bloc

URL

none

background-repeat : Répétition de l'image de fond

repeat

repeat-x

repeat-y

no-repeat

background-attachment : Scroller ou non l'image de fond avec le canevas

scroll

fixed

background-position : Position de départ de l'image de fond

[|]{1,2} | [top | center | bottom] || [left | center | right]

Propriétés de la police de caractères

font : [<font-style> || <font-variant> || <font-weight>]? <font-size> [/ <line-height>]? <font-family>

font-family : Police(s) de caractères

serif (e.g., Times)

sans-serif (e.g., Arial or Helvetica)

cursive (e.g., Zapf-Chancery)

fantasy (e.g., Western)

monospace (e.g., Courier)

font-style : Style de police

normal

italic

oblique

font-variant : Variation de la police

normal

small-caps

font-weight : Épaisseur de la police

normal

bold

bolder

lighter

100 à 900

font-size : Taille de la police

xx-small

x-small

small

medium

large

x-large

xx-large

Propriétés du texte

word-spacing : Espacement entre les mots

normal

Longueur

letter-spacing : Espacement entre les lettres

normal

Longueur

text-decoration : Décoration du texte

- none
- underline
- overline
- line-through
- blink

vertical-align : Alignement vertical du texte

- baseline
- sub
- super
- top
- text-top
- middle
- bottom
- text-bottom

text-transform : Transformation du texte

- none

capitalize : Majuscule du premier caractère de chaque mot

- uppercase : Majuscules
- lowercase : Minuscules

text-align : Alignement horizontal du texte

- left
- right
- center
- justify

text-indent : Indentation de la première ligne du texte (peut être négatif ?)

- Longueur

line-height : Hauteur de la ligne de base

- normal
- (nombre) Multiplie ce nombre par la taille de la police
- Longueur

Propriétés spécifiques

display : Mode d'affichage des blocs

- bloc : en tant que bloc (une ligne est passée avant et après le bloc)

- inline : en ligne

- list-item : en tant que ligne de liste

- none : Ne pas afficher le bloc et ses enfants

- white-space : Rendu des espaces

- normal : les espaces, tabulations et retours à la ligne sont réunis et affichés comme un seul espaces

- pre : espacements affichés tels quels

- nowrap : ne passe jamais à la ligne même si le contenu est trop grand (dans ce cas une barre de défilement horizontale apparaît)

- list-style : <list-style-type> || <list-style-position> || <url> : Type de puce

- L'URL sert à définir la propriété list-style-image.

list-style-type : Type de puce ou de numérotation des listes (s'applique à ul, ol et li, et éléments avec

- display: list-item)

- disc : un disque plein

- circle un cercle (creu)

- square : un carré plein

- decimal : 1 2 3 4 ...

- lower-roman : i ii iii iv ...

- upper-roman : I II III IV ...

- lower-alpha : a b c d ...

- upper-alpha : A B C D ...

- none

list-style-image : Image à la place d'une puce (s'applique à ul, ol et li, et éléments avec display: list-item)

- URL

- none

list-style-position : Position de la puce (s'applique à ul, ol et li, et éléments avec display: list-item)
inside : la puce est à l'intérieur, alignée avec le texte (et donc la première ligne est décalée à droite)
outside : la puce est à l'extérieur, le texte est lui aligné

Unités

Préférez des unités relatives, car elles peuvent être adaptées au visisteur (mal voyant ou pas) et au périphérique (écran ou imprimé), et banissez les unités absolues (sauf si les relations sont directement définies pour un seul périphérique, par exemple). Ces valeurs peuvent être positives comme négatives (+/-), la valeur 0 ne requérant pas de spécifier d'unité.

Unités de longueur relatives

em : hauteur de la police de caractère (ou de celle de la balise parente, lors de définition de cette même taille)
ex : hauteur du caractère x minuscule ('x')
px : taille d'un pixel (différent à l'écran et sur papier... mais calculés pour être dans les proportions)

Unités de longueur absolues

cm : centimètres (1 cm = 10 mm)
mm : millimètres
in : inches (1 in = 2.54 cm)
pt : points (1 pt = 1/72 in)
pc : picas (1 pc = 12pt)

Le pourcentage : Pourcentage de la longueur de l'élément parent

Couleurs

Il existe déjà 16 couleurs prédéfinies : aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, et yellow.

Une couleur est donnée au format RGB (Red Green Blue : Rouge, puis Vert, puis Bleu) déclarée ainsi :

#rrggbb (par exemple, du vert moyen, #00cc00)

#rgb (même exemple : #0c0)

rgb(x,x,x) où x est un entier entre 0 et 255 inclu (par exemple, rgb(0,204,0))

rgb(y%,y%,y%) où y est un nombre à virgule compris entre 0.0 et 100.0 inclu (par exemple, rgb(0%,80%,0%))

URLs

Une URL (adresse WEB) est définie par url(l_adresse). L'adresse peut être entre guillemets simples ('') ou doubles ("").

Les parenthèses, virgules, espaces et guillemets dans l'adresse doivent être échappés avec un antislash (\, pour une virgule, \" pour un guillemet). Si une adresse relative est donnée, elle est relative à la feuille de style (le fichier .css) et non à la page WEB (.html) !