# LANGAGE HTML

# PAGES STATIQUES

AVERTISSEMENTS :

Ce document est destiné aux concepteurs de pages Web qui veulent programmer sans l'aide d'éditeurs avancés (Frontpage, Dreamweaver).

Ces éditeurs, aussi évolués soient-ils, ne nous proposent pas toutes les possibilités du langage, et ne nous permettent pas de régler tous les problèmes.

Le document décrit de manière exhaustive tous les éléments normalisés du langage. Les éléments non normalisés, spécifiques à un navigateur donné, sont d'une manière générale évoqués.

Le concepteur ou créateur de pages web peut ainsi se promener dans les coulisses du langage de description, sans aucune appréhension, et devenir un professionnel de ce langage.

Le titre de ce premier volume indique que l'on se limite à décrire les moyens statiques d'élaboration de pages. Un deuxième volume est consacré à l'étude des moyens permettant de donner un comportement dynamique aux pages Web.

# SOMMAIRE

# Chapitre 1 : Langages de Description

## 1. Langage HTML

### 1.1 Introduction

Le langage “HTML” (**HyperText Markup Language** = langage de marques hypertextes) est un langage permettant de **décrire** le comportement d'un **document** HTML ou **page Web** destiné à être visualisé par un **navigateur**.

Cette page Web peut afficher aussi bien du **texte** que des éléments **multimédias** (images, sons, etc…). Mais en réalité, le fichier écrit dans ce langage de description n’est autre qu'un fichier texte interprété par le navigateur.

Ce langage est né dans sa première version en 1992, de manière parallèlement à avec d’autres techniques telles que le protocole HTTP, l’adresse URL, et les navigateurs.

Ce langage, créé par par Berners Lee (créateur du Web), est issu d’un langage déjà existant : le langage **SGML** (Standard General Markup Language). Ce dernier permet d’effectuer des échanges de données informatiques (EDI) de manière structurée et complexe.

### 1.2 Langage SGML

Le langage SGML, produit en 1986 par l'ISO (International Standards Organization), correspond à une norme de représentation des documents. Elle explique que les documents (au sens large) peuvent être partagés en deux parties :

* celle contenant l'information et sa structure

* celle contenant la mise en forme de l'information

Le langage SGML est en fait un métalangage qui permet de créer des balises adaptées à une terminologie, un type de données "métier", par exemple : l'enseignement, l'aspect juridique,....

Le langage SGML n'est pas un système de formatage :

- le langage DSSL (Document Style Semantics and Specifications Language) est généralement utilisé pour le formatage de documents SGML.

- le document SGML sera accompagné de sa DTD (Définition de Type de Document).

En conclusion, le SGML est un langage complexe à mettre en œuvre. Il n'est donc pas adapté à la publication d'informations sur les réseaux, mais plutôt utilisé dans l'édition de documents imprimés ou électroniques.

## 1.3 Caractéristiques et Evolutions du Langage HTML

Le langage HTML a pour but de permettre au grand public (à monsieur tout le monde) de créer un site Web en toute convivialité, aussi est-ce une version volontairement simplifiée du langage SGML.

Le langage HTML est donc un langage simple devenu le standard de la publication de données non seulement sur les réseaux (Internet et Intranet), mais aussi sur d'autres supports (CD-ROM, télévision, téléphone portable, synthétiseur vocal, etc…).

Les documents HTML sont le plus souvent constitués d'un mélange d'informations et d'instructions de formatage, il est donc difficile d'en analyser le contenu sémantique.

Contrairement au langage SGML ou au langage XML, le langage HTML ne permet pas de créer ses propres balises. La liste des balises est définie strictement par des recommandations du **W3C (World Wide Web Consortium)**.

Le langage HTML est donc le format idéal de présentation de l'information mais n'est pas toujours adapté à la structuration des données.

Le langage a connu plusieurs versions. La version HTML 2.0, développée par l'IETF, fit l'objet du RFC 1866 en novembre 1995. La version la plus couramment utilisée du langage HTML reste la version 3.2 normalisée en janvier 1997.

Elle précéda de peu la version 4.0 officialisée en décembre 1997. Cette version fut corrigée à plusieurs reprises, notamment en **décembre 1999** par la **version 4.01**.

Si la version 3.2 est accessible au grand public à travers les éditeurs disponibles sur le marché, la version 4.0 s'adresse plus à des professionnels de l'informatique et donc du Web. Aussi est-ce la raison pour laquelle l'ensemble des chapitres du présent document s'appuie sur la version 3.2, la version 4.0 étant étudiée dans un chapitre complémentaire.

En dehors de la notion de version, le document de référence est le **RFC 2854** publié en **juin 2000**.

Ces normes sont consultables directement sur le site ***www.w3.org***, ainsi que sur le site des RFC : ***www.rfc-editor.org***.

# 2. Autres langages de description

Depuis, d'autres langages sont apparus.

- Le langage **DHTML** (Dynamic HMTL) n'en est pas un, car il n'est pas normalisé. Il ne fait que rassembler plusieurs techniques afin de rendre encore plus dynamique le comportement de votre page Web : version 4.0 du langage HTML, langages de scripts, feuilles de style, programmation utilisant les objets DOM (Document Object Model). Il est principalement utilisé par les sites de publicités tournantes.

- Le langage **XML** (eXtensible Markup Language) en 1998 permettant de retrouver le côté formalisateur du langage d'origine (SGML). Il permet de plus de personnaliser ses propres éléments de description (possibilité de créer ses propres balises). Il est présenté comme le nouveau format universel pour les documents structurés accessibles sur le Web. Il est complété par le langage XSL (eXtensible Stylesheet Language), un langage de formatage des documents équivalent du CSS (Cascading Style Sheets) pour le langage HTML. On utilisera le langage XML si les données (au moins une condition) représentent un volume

important, ou font l'objet d'une fréquente mise à jour, ou sont échangées avec d'autres systèmes d'exploitation.

- Le langage **XHTML** avalisé en janvier 2000 par le consortium w3c, visant d'une part à formaliser le langage HTML, et d'autre part à être considéré comme un langage intermédiaire entre HMTL et XML. Il s'agit en fait d'une reformulation de la version 4 du langage HTML en tant qu'application XML 1.0. Sa conception modulaire permet une gestion plus facile de données type graphiques vectoriels, formules mathématiques ; des navigateurs utilisés sur un nombre de plate-formes croissant (mobiles, assistants numériques personnels....).

En raison de l'apparition de ces nouveaux langages, le langage HTML ne connaît plus d'évolutions au profit du langage XML.

En conclusion, la ressemblance entre les documents SGML et XML est le signe d'un lien de parenté étroit entre les deux langages, de même que les documents HTML et XHTML.

# 3. Principes

Les navigateurs Web proposent tous une fonction "Afficher le source" (ou "source", ou "code source") permettant de visualiser le code HTML de la page en cours.

Pour apprendre le HTML, il faut prendre le temps d'analyser quelques pages.

## 3.1 Balises

Ce langage de description de pages utilise des **marques** - ou **balises** (ou **tags** en anglais) - pour spécifier la façon dont un élément doit apparaître, pour afficher des images ou définir des actions.

Ces balises sont toujours placées entre les signes **"<"** et **">",** sans espace de séparation. Elles agissent très souvent par paire.

La première spécifie le début d'application du style (ou de l'action).

La seconde, qui comporte en plus le signe "**/**" juste derrière le signe "<", marque la fin d'application du style (ou de l'action).

**Exemple :**

```
<balise>texte</balise>

<B>texte en gras</B>
```

- Ne pas oublier cette balise de fin, car le style ou l'action défini est actif tant que le navigateur ne rencontre pas la balise de fin.

  Ainsi, si une balise "**mettre en italique**" est utilisée au début de votre document et que la balise "**enlever l'italique**" est oubliée, tout le document apparaîtra en italique !

**Remarque 1 :**

L'ensemble des deux balises s'appelle un élément du langage HTML.

**Remarque 2 :**

Les balises peuvent être saisies **indifféremment en majuscules ou minuscules**.

Tous les éditeurs de documents HTML (dont Microsoft Frontpage, Netscape Composer, etc.), ainsi que les navigateurs, reconnaissent les deux syntaxes.

**Remarque 3 :**

Le navigateur MSIE, quelle que soit sa version, interprète toutes les balises, alors que le navigateur Netscape 4.x n'interprète pas celles de MSIE. La version 6.0 de Netscape est censée régler les problèmes de compatibilité.

**Remarque 4 :**

Il est possible d'oublier certaines balises de fermeture sans altérer l'interprétation du code : </p> (paragraphe), </li> (article de liste), </td> (cellule de tableau), etc…

**Remarque 5 :**

Les balises peuvent s'imbriquer les unes dans les autres. L'exemple suivant consiste à rendre le mot *contenu* en style gras (b) et italique (i) : <I><B>contenu</B></I>.

Elles peuvent même se chevaucher, comme dans l'exemple ci-dessous, alors que le langage XHTML l'interdit : <B><I>contenu</B></I>.

**Remarque 6 :**

Les navigateurs ignorent ce qu'ils ne reconnaissent pas. Ils ne provoquent pas d'erreurs. Il peut survenir les phénomènes suivants : ce qui n'est pas interprété peut ne pas être affiché, un mauvais code fait que le code lui-même peut être affiché.

## 3.2 Attributs

Un glossaire sur les balises HTML présente chacune d'entre elles disposant d'un certain nombre d'attributs paramétrables dans un **format spécifique**.

Ces attributs sont insérés juste après le nom de la balise de début de style, séparé de ce dernier par un espace.

Chaque attribut est suivi du caractère "=" et d'une valeur. Le langage HTML offre la liberté d'encadrer ou non la valeur par des guillemets (ou double quote) alors que le langage XHTML l'impose.

La dernière valeur d'attribut doit être immédiatement suivie du signe ">" de fin de balise.

**Exemple 1 :**

```
<balise attribut="valeur">texte</balise>

<FONT size="5">texte taille 5</FONT>
```

Le concepteur de pages Web emploie tout ou partie des attributs proposés (le minimum) en fonction de l'effet désiré.

Ainsi, le deuxième exemple ci-dessous permet de fixer pour le texte encadré, non seulement la taille de la police utilisée, mais aussi la couleur. Les couples attributs-valeurs successifs sont séparés par un caractère espace.

**Exemple 2 :**

```
<balise attr1="val1" attr2="val2"...>texte</balise>

<FONT size="5" color="#0000FF">texte taille couleur</FONT>
```

La première tâche d'un concepteur néophyte consiste à comprendre la structure d'une page. Celle-ci se découpe en deux parties : une affichée par le navigateur (le corps), une invisible du visiteur (l'entête).

La conception d'une page relevant d'une démarche de communication, celle-ci doit être la plus esthétique possible pour attirer l'internaute et le fidéliser.

En conséquence, dans un deuxième temps, le concepteur sera amené à effectuer une première mise en forme du corps de la page : découpage en sections et paragraphes, et ajout de couleur.

C'est le propos du prochain chapitre.

# Chapitre 2 : Outils

## 1. Navigateurs

### 1.1 Rôle des navigateurs

La présentation des navigateurs n'est pas envisagée ici, mais plutôt dans le module consacré aux services Internet. Le but de ce paragraphe est de rappeler brièvement le fonctionnement de cet outil et son rôle vis à vis de la page web.

Le navigateur est un **logiciel client** susceptible de communiquer avec **l'ensemble des serveurs Internet**. Par défaut, il reste **dédié au service d'informations Web**. C'est lui qui va lire la page web et en **afficher le résultat**.

Une fois que l'utilisateur valide une adresse URL dans la barre d'adresses du navigateur, ce dernier lance une **requête HTTP** auprès du serveur Web qui héberge le site désiré. Le serveur effectue une **réponse HTTP** en renvoyant la page demandée vers le client. Le navigateur du client **la télécharge dans son cache interne**, puis entame **l'interprétation du code source associé à la page de manière séquentielle**.

Le premier rôle attendu des **navigateurs** est donc d'**interpréter** les langages de description décrits dans le paragraphe précédent **en fonction de l'extension donnée au fichier**.

### 1.2 Extensions des fichiers

Tout fichier ayant pour extension *.htm (Windows 3.11) ou *.html (Windows 95 et plus…) sera interprété comme une page Web écrite en langage HTML.

Il en est de même pour un fichier ayant comme extension *.dhtml ou *.xhtml, ou *.xml.

Dans la famille du langage HTML, on rencontre d'autres extensions : *.shtml pour les pages incluant des appels aux scripts SSI (Server Side Includes), *.mhtml pour les pages aux multiples formats de données (type MIME), etc…

### 1.3 Navigateurs du marché

Le navigateur le plus connu du marché est Microsoft Internet Explorer (MSIE) : 90%. Deux autres navigateurs se partagent les 10% : Netscape Navigator (Netscape) et Opera de ...

Les sociétés cachées derrière les deux premiers navigateurs ont participé fortement à l'évolution du web, et de son langage premier, le langage HTML.

Elles ont développé chacune de leur côté des éléments de description (ou balises), voire des propriétés ou attributs spécifiques aux balises.

Au fur et à mesure du temps, ces balises "propriétaires" ont été soit normalisées par le **consortium W3C**, soit abandonnées.

Les navigateurs, par défaut, interprètent toutes les balises normalisées, ainsi que leurs balises spécifiques. Le navigateur « Internet Explorer », a plutôt tendance à interpréter toutes les balises. Le navigateur Netscape, en revanche, a tendance à interpréter les mêmes balises que le navigateur « Internet Explorer », mais avec une version de retard.

De plus, le rendu d'une même balise peut être différent d'un navigateur à l'autre.

Pour toutes ces raisons, il faut se soucier de la validité des éléments décrits dans sa page pour assurer une **portabilité maximale de sa page** ou de son site. En effet, le concepteur ne sait pas à l'avance de quel navigateur dispose le visiteur du site. Dans l'absolu, il doit s'assurer que les éléments décrits dans la page ne posent pas de problème d'interprétation par l'un quelconque des navigateurs cités.

En réalité, dans un premier temps, les concepteurs débutants s'attachent à travailler le meilleur rendu sous Internet Explorer en prenant la précaution de ne pas utiliser des éléments spécifiques. Dans un deuxième temps, les concepteurs avancés, connaissant les différences et les moyens de les contourner, incluent dans les pages des scripts adéquats.

# 2. Editeurs

## 2.1 Page statique ou dynamique

**Avant de décrire les différentes possibilités en matière d'édition, il faut expliciter deux définitions : le Web Statique, le Web Dynamique. Si la première nécessite seulement un éditeur, la deuxième oblige à mettre en œuvre une véritable plate-forme de développement.**

**Le Web Statique** consiste à fabriquer toutes les pages d'un site. Chaque changement impose une édition manuelle des pages à modifier. Les données sont contenues dans une collection de fichiers HTML situés sur un serveur Web. Le serveur met à la disposition du public l'ensemble des pages d'un site qu'il gère de manière indépendante. On dit qu'il héberge le site publié par tel ou tel concepteur.

Le **Web Dynamique** consiste à faire générer par des programmes les pages à la demande. L'utilisation de langages de scripts mettant en relation le programme avec des bases de données est incontournable. Une plate-forme minimale comprend successivement : un gestionnaire de bases de données (SGBD), un serveur web, un langage de script interprétable par le serveur. Les deux plate-formes les plus couramment utilisées, sont :

* MySQL/Apache/PHP

* Access/IIS/ASP

Le contenu de cette partie s'attache à décrire uniquement les éditeurs de pages statiques.

## 2.2 Editeurs de texte

Pour réaliser un document, il faut disposer d'un éditeur de texte.

Deux solutions s'offrent au concepteur :

- Ecrire les balises de sa page à la manière d'un vrai programmeur

- Ecrire sa page de manière conviviale à l'aide d'un éditeur avancé

Pour la première solution, le concepteur dispose de Bloc-Notes (NotePad) ou WordPad sous Windows, de Vi ou Emacs sous UNIX.

L'inconvénient réside dans le fait que chaque balise doit être saisie à la main. L'avantage vient du fait que tous les ordinateurs ont un éditeur de texte.

# 2.3 Editeurs HTML

Pour la deuxième solution, il dispose sous Windows du logiciel bureautique de Microsoft Word (Nouveau / onglet Page Web), du logiciel Microsoft Frontpage aussi bien dans sa version allégée (version Express livrée avec le navigateur MSIE) que dans sa version lourde (version 98 ou 2000 payante).

Sous Windows, d'autres logiciels font concurrence à ceux de Microsoft : Dreamweaver (version 4.0) de Macromédia, Adobe Golive 5.0, NamoWeb Editor, WebExpert, etc…

Ces éditeurs spécialisés bénéficient de fonctions d'édition automatisées. La structuration et le formatage des documents sont réalisés par l'intermédiaire du couple de moyens menu/icône.

L'intervention dans le code source n'a donc lieu que dans le cas où le formatage désiré n'est pas disponible dans les fonctions du logiciel. Il peut arriver aussi que des modifications de formatage soient plus aisées à travers le code source que directement dans la zone d'édition.

## Avantages

Les avantages sont doubles.

Le premier est de permettre au débutant de réaliser des pages web en se limitant à une connaissance sommaire du langage HTML. Cela est possible grâce à l'édition par menus, icônes ou assistants.

Le deuxième trouve son origine dans l'intégration d'outils de gestion de site au sein de l'éditeur. Il bénéficie à la fois, d'une vision arborescente du site de développement (voire du site publié), d'un fonction de transfert de fichiers (client FTP) pour publier le site sur le serveur web, de moyens de vérification technique du site (liens hypertextes par exemple),...

## Inconvénients

Les langages évoluent. Il faut donc mettre à jour le logiciel ou faire l'impasse sur les nouvelles possibilités. La conséquence en est un risque de dépendance vis-à-vis du logiciel.

Il faut veiller à ne pas utiliser de code et d'extensions propriétaires pour formater les documents.

**Remarque** :

Les éditeurs avancés ne sont pas présentés ici dans ce module. Ils peuvent faire l'objet d'un guide utilisateur simplifié dans un proche avenir. La description complète des fonctionnalités des éditeurs oblige le lecteur à connaître au préalable toutes les techniques du web. En

conséquence, l'assimilation, ou tout au moins le parcours des deux modules consacrés au langage HTML paraît nécessaire avant d'aller travailler sous l'éditeur de son choix.

## 2.4 Convertisseurs

Un des chapitres complémentaires du document montre qu'il est possible de convertir les fichiers élaborés sous chacun des logiciels de la suite bureautique Pack Office de Microsoft en pages Web sans aucune difficulté.

Le but est d'obtenir des pages HTML à partir de documents non-HTML.

Le code produit n'est pas optimisé, et nécessite parfois des corrections avant publication sur un site Web.

En dehors du Pack Office, il existe un certain nombre de logiciels gratuits ou partagiciels permettant de transformer ou de convertir n'importe quel format de fichier en page web : rtf2html pour le format rtf, acrobat reader 5.0 pour le format pdf,...

Pour ceux qui ne seraient pas satisfaits, le consortium W3C tient à jour une liste d'autres outils (www.w3.org/pub/WWW/Tools/Filters.html) permettant la conversion : de Framemaker..., de troff..., de LateX..., de QuarkXPress..., de PageMaker..., d'AmiPro..., etc...

# Chapitre 3 : Document HTML

## 1. Réalisation d'un document HTML

### 1.1 Structure d'un document HTML

L'élément **html** (HTML 2.0) sert à préciser que le contenu du document est écrit en **langage HTML**. Il n'a pas de réelle utilité dans le cas où le langage HTML est le seul employé dans le document. Par contre, dans un document comportant des éléments écrits dans un autre langage, un langage de script par exemple, il sert à délimiter les portions écrites en HTML.

L'élément **head** (HTML 2.0), imbriqué dans le précédent, correspond à l'entête de la page, et contient des **données invisibles** pour le navigateur. Seul l'élément **title** voit son contenu s'afficher en guise de titre de document dans la barre de titre du navigateur. Ces données circulent au sein des entêtes protocolaires **http**.

L'élément **body** (HTML 2.0) définit le **corps** du document imbriqué dans l'élément html. **Seul le contenu de cette section sera affiché par le navigateur.**

On peut rencontrer l'élément !**DOCTYPE** (HTML 2.0) tiré du langage SGML. Placé avant l'élément HTML, il indique la spécification HTML respectée par le document, et éventuellement l'adresse du « validateur » de format. Même si le tableau suivant informe sur le contenu de cet élément en fonction de la version du langage, il reste plutôt utilisé dans les documents à partir de la norme HTML v4.0. Si la présence de cet élément est optionnelle dans les documents HTML, elle est par contre indispensable dans les documents XHTML.

| Commande. | Version HTML. |
|---|---|
| **<!DOCTYPE HTML PUBLIC "-//IETF//DTD Level1//EN">** | HTML 1.0 |
| **<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">** | HTML 2.0 |
| **<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.0//EN">** | HTML 3.0 |
| **<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">** | HTML 3.2 |

**Exemple :**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
…
</head>
<body>
texte
</body>
</html>
```

**Remarque** : en fait, aucune balise n'est obligatoire sous Windows. Seul le fait de donner comme extension au fichier *.htm ou *.html suffit au navigateur pour interpréter par défaut le document comme une page HMTL et afficher le texte inséré sans balises.

## 1.2 Réalisation d'un premier document avec titre

Pour reprendre une tradition, le premier document HTML à élaborer peut comporter le texte suivant.

**Exemple 1 :** “Hello world !”

```
<html>
<body>
Hello world !
</body>
</html>
```

La réalisation de cette page s’effectue en trois étapes :

- copier ce bloc de texte dans un simple éditeur de texte style Bloc-Notes
- « enregistrer sous » le fichier en choisissant "Tous fichiers" et en forçant l’extension de type soit “htm” (version Windows 3.11) ou “html” (si le système l’autorise) : exemple hello.htm
- visualiser le résultat obtenu en ouvrant cette première page Web avec le navigateur (double cliquer sur le fichier précédemment enregistré "hello.htm" dans l'explorateur)
- si le résultat n’est pas satisfaisant, modifier dans l'éditeur le fichier "*.htm", enregistrer les modifications et actualiser l'affichage dans le navigateur (icône idoine), et ainsi de suite...

Pour modifier le code source de la page Web, il est possible de le faire depuis le navigateur. Dans MS Internet Explorer, l'action du menu « Affichage / Source » ouvre le code source de la page sous Bloc-notes.

Le résultat est le suivant :

Hello world !

L'entête permet en premier lieu de définir le **titre du document** apparaissant dans la barre de titre du navigateur à la place du nom de fichier, ceci à l'aide de l'élément **title** comme dans l’exemple ci-dessous.

**Exemple 2** : “Premier document”

```
<HTML>
<head>
<title>Mon premier document HTML</title>
</head>

<body>
```

```
Hello world !
</body>
</HTML>
```

**Résultat :**


L'absence de l'élément **title** (HTML 2.0) fait que le navigateur affiche le nom du fichier à la place du titre de la page.


Les moteurs de recherche prennent en compte cet élément. Pour représenter une page web au niveau des favoris, le navigateur choisit cet élément comme substitut à l'adresse URL de la page.

Voici un premier document HTML, rudimentaire certes mais prêt à être publié sur le Web et ainsi rendu accessible à des millions d'internautes sur toute la planète.

## 1.3 La balise de commentaire

Il est possible d'insérer des commentaires (HTML 2.0) dans le document à l'aide de la syntaxe suivante.

```
<!-- commentaires -->
```

Les commentaires ne sont pas affichés par les navigateurs. Tout ce qui est placé entre **<!--** et **-->** est donc ignoré.

# 2. Structurer le corps de la page

Un document lu est avant tout un document aéré. Pour cela, il faut diviser le contenu de votre texte en plusieurs paragraphes. Divers moyens sont à la disposition du concepteur pour diviser son document, pour séparer les différents éléments selon des interlignes différents ou par une ligne de séparation.

## 2.1 Paragraphes

Dans l'éditeur HTML, le simple fait de valider la touche <Entrée> crée un nouveau paragraphe.

L'élément **p** (HTML 2.0) définit un **paragraphe**. Le navigateur interprète cette balise en effectuant un saut de ligne et un retour en début de ligne.

La balise de fermeture **</p>** est obligatoire dans la mesure où le paragraphe court sur plus d'une ligne. Sans elle, le résultat risque de devenir imprévisible.

**Attributs**

| <p | | | |
|---|---|---|---|
| align | = | mots prédéfinis | Alignement du texte avec les valeurs : center, left, right. |
| width | = | nombre | Largeur du paragraphe par défaut à 80 colonnes. |
| >*texte du paragraphe*</p> | | | |

**Conventions :**

Dans le tableau ci-dessus, l'élément p est présenté avec l'ensemble de ses attributs ou propriétés intrinsèques.

Derrière le signe "=" est indiqué le type de valeur attendu. La troisième colonne est là non seulement pour définir cet attribut, mais aussi pour donner les valeurs prédéfinies.

Les attributs faisant appel à des notions approfondies du langage ne seront pas commentés dans les chapitres consacrés aux bases du langage HTML.

Il en sera de même pour toutes les balises du langage dans la suite du module.

**Exemple :**

```
<p>Ceci est un exemple de texte :</p>
<p>Né en 1992, le Web est aujourd'hui le service le plus utilisé d'Internet.</p>
```

**Résultat :**

Ceci est un exemple de texte :

Né en 1992, le Web est aujourd'hui le service le plus utilisé d'Internet.

## 2.2 Sections

L'élément **div** (HTML 3.2) est utilisé pour diviser un document en différentes sections comme des chapitres, des extraits et des annexes.

Une division peut englober plusieurs paragraphes, images, ou titres. Elle peut aussi encadrer des séries de tableaux.

Il est possible d'emboîter une division dans une autre pour mieux gérer l'alignement ou d'autres effets de style pour un groupe de paragraphes déterminé.

**Attributs**

| <div | | | |
|---|---|---|---|
| align | = | mots prédéfinis | Alignement du texte avec les valeurs : center, left, right, justify. |
| > | | | |

```
<p>textes des paragraphes, images, etc… </p>
</div>
```

# 2.3 Eléments de rupture

## 2.3.1 Retour de chariot

La balise **<br>** (ou "break rule", norme HTML 2.0) spécifie au navigateur un **saut de ligne** et un **retour chariot**.

L'effet rendu est différent de celui de l'élément p qui ajoute un deuxième saut de ligne. Le concepteur, qui veut jouer avec les interlignes, va alterner l'emploi de l'un et de l'autre dans la présentation du texte.

La balise de fermeture **</br>** est **interdite**.

### Attributs

| <br | | | |
|---|---|---|---|
| clear | = | mots prédéfinis | Alignement du texte à droite ou à gauche d'une image en créant un effet de flottement avec les valeurs : all, left, right, none. |
| > | | | |

L'utilisation de l'attribut **clear** est explicité dans le chapitre consacré au images.

L'élément **nobr** (no break rule) empêche le texte situé entre la balise de début et la balise de fin d'avoir un retour de chariot de la part du navigateur. Cet élément accepté par tous les navigateurs n'a pas été normalisé.

La balise **wbr** (sans balise de fermeture) est utile, au sein d'une section *nobr*, pour indiquer au navigateur l'endroit où une ligne de texte ou un mot est autorisé à recevoir un retour de chariot. Cet élément aussi n'a pas été normalisé.

**Exemple :**

```
<p> Du texte à afficher sur une seule ligne ou non selon la résolution d'écran du navigateur.
<NOBR>
Du texte à afficher sur une seule ligne, <WBR> une éventuelle deuxième ligne de texte.
</NOBR></p>
```

## 2.3.2 Ligne de séparation horizontale

Un autre moyen d'insérer une séparation est l'utilisation de la balise **<hr>** (ou "horizontal rule", norme HTML 2.0). Celle ci dessine une **ligne de séparation horizontale** et retourne à la ligne.

La balise de fermeture **</hr>** est **interdite**.

### Attributs

| <hr | | | |
|---|---|---|---|
| size | = | nombre | Spécifie l'épaisseur de la ligne horizontale en nombre de pixels. |
| width | = | nombre | Spécifie la largeur de la ligne horizontale en proportion avec la largeur de la page. La valeur exprime un pourcentage, donc un nombre compris entre 0 et 100 suivi du signe "%". |
| align | = | mots prédéfinis | Alignement avec les valeurs : all, left, right. |
| noshade | | | Permet d'avoir à la place d'une apparence bicolore, une ligne de trait opaque. |
| > | | | |

Voici un exemple avec deux paragraphes séparés par une ligne couvrant 75% de la largeur de la page, le deuxième paragraphe de deux lignes insérant un saut de ligne :

```
<HTML>
<head><title>Mes paragraphes</title></head>
<body>
<p>Hello world !</p>
<hr width=75%>
<p>Premier paragraphe<br>
Second paragraphe</p>
</body>
</HTML>
```

Notons au passage qu'il n'est pas obligatoire de retourner à la ligne pour enchaîner deux balises. La section entête est ici regroupée sur une seule ligne. La seule règle est la lisibilité.

Voici le résultat :

Hello world !

Premier paragraphe
Second paragraphe

## 2.4 Eléments de titre

Dans un document, une des premières actions du concepteur est de définir un titre de page. Il a déjà défini le titre du document (élément title) dans un paragraphe précédent. Il s'agit ici de décider du titre apparaissant en haut de la page.

Il se peut ensuite que ce document comprenne plusieurs sections et paragraphes. Le concepteur pense à utiliser des éléments de style comparables à celles rencontrés sous Word. Les éléments de titre sont là pour répondre au besoin.

La norme HTML 2.0 définit six niveaux de titrage depuis **H1** affiché comme le plus imposant jusqu'à **H6** le plus discret. Les navigateurs visuels utiliseront des polices plus ou moins grosses pour rendre compte du niveau de titre.

### Attributs

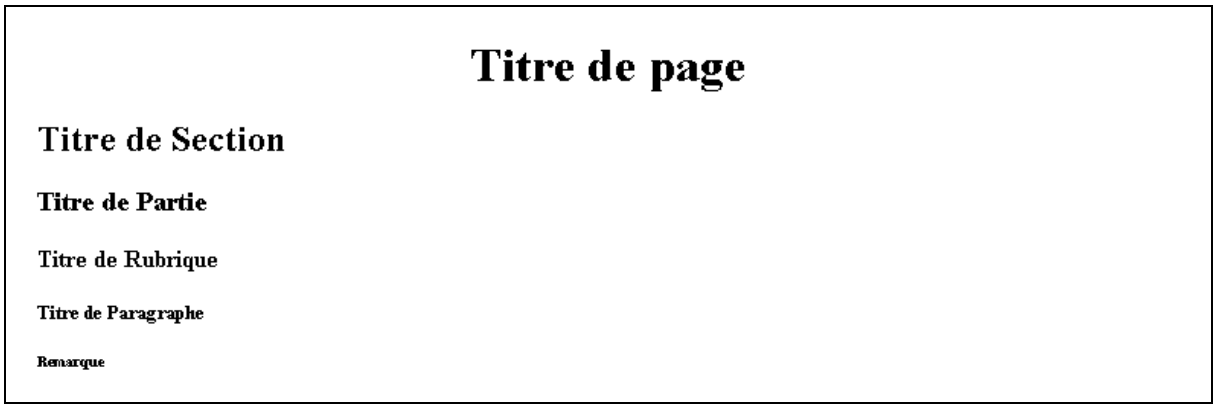
| | | | |
|---|---|---|---|
| <h1 à 6 | | | |
| align | = | mots prédéfinis | Alignement du texte avec les valeurs : center, left, right. |
| > *texte du titre* </h1 à 6> | | | |

**Exemple**

```
<h1 align="center">Titre de page</h1>
<h2>Titre de Section</h2>
<h3>Titre de Partie</h3>
<h4>Titre de Rubrique</h4>
<h5>Titre de Paragraphe</h5>
<h6>Remarque</h6>
```

Le concepteur peut se servir de ces titres pour fixer la présentation des sommaires de pages en fixant un titre et un style pour chaque niveau.

**Résultat**




# 2.5 Alignement du texte

## 2.5.1 Attribut align

Pour définir l'alignement, l'attribut **align** des éléments **p** ou **div** doit être utilisé avec l'une des trois valeurs suivantes **left, center** ou **right**.

La valeur **left** est utilisée par défaut si l'attribut **align** est inutilisé.

**Exemple** : alignement de paragraphes

```
<p>ligne de texte aligné à gauche par défaut</p>
<p align="center">ligne de texte centré</p>
<p align="right">ligne de texte aligné à droite</p>
```

**Résultat :**


Le résultat serait le même avec l'élément **div**.

**Remarque** :

Pour **centrer un bloc plus étendu** qu'un simple paragraphe, nous pouvons aussi utiliser l'élément **center** (norme HTML3.2) à la place de l'élément div. Tout élément défini entre ces balises sera centré, sauf si cet élément redéfinit l'alignement. Cet élément est déprécié avec la version HTML4.0. Néanmoins, il est toujours interprété par les navigateurs. L'emploi de cet élément peut être fort utile pour forcer le comportement du navigateur dans certains cas où il ne centre pas le texte pour des raisons inconnues.

## 2.5.2 Retrait du texte

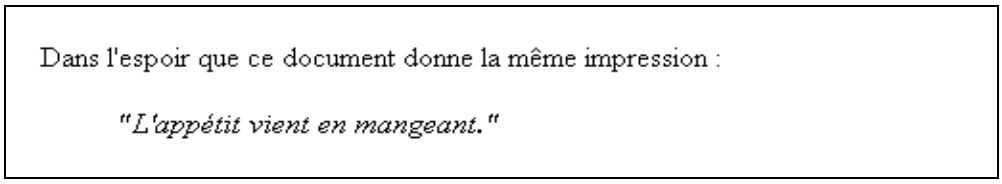
Il est possible de mettre en retrait du texte (**tabulations**) en insérant ce texte dans un élément **blockquote** (HTML 2.0). L'imbrication de plusieurs retraits est possible.

Cet élément est détourné de son but d'origine car il était prévu pour afficher des blocs de citations séparés d'un texte standard.

**Exemple :**

```
<p>Dans l'espoir que ce document donne la même impression :
<blockquote><i>"L'appétit vient en mangeant."</i></blockquote>
</p>
```

**Résultat :**




**Remarques :**

A. Le code de l'élément blockquote inclut un saut de ligne, un retour chariot, puis le décalage du texte attendu.

B. La valeur du retrait n'est pas réglable. Les effets de style (cf. chapitre "Feuilles de Styles") pallieront cette limitation.

C. Pour décaler du texte par rapport au bord de la page, il est possible de jouer avec la marge de celle-ci sous Internet Explorer (élément body).

## 2.6 Préformater le texte

L'élément **pre** (ou "preformatted", norme HTML 2.0) indique aux navigateurs que le texte inclus est "préformaté". Les navigateurs doivent traiter le texte préformaté comme suit :

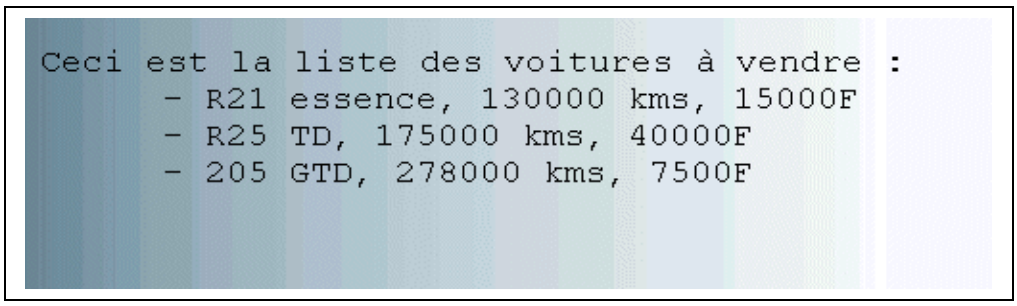
- Ils doivent laisser les espaces blancs intacts.
- Ils devront écrire le texte avec une police à espacement fixe.
- Le retour automatique à la ligne peut être désactivé.

Cet élément est utilisé pour des textes dont la mise en forme d'origine ne doit en aucune façon être altéré par l'interprétation de tel ou tel navigateur. Les caractères accentués, les espaces et les sauts de ligne, les tabulations sont respectés.

**Exemple**

```
<pre><big><big>Ceci est la liste des voitures à vendre :
        - R21 essence, 130000 kms, 15000F
        - R25 TD, 175000 kms, 40000F
        - 205 GTD, 278000 kms, 7500F
</big></big></pre>
```

**Résultat**




# 3. L'élément "corps de page"

Pour terminer une première mise en forme du document, il est nécessaire de parcourir les autres attributs de l'élément body énumérées ci-après.

## 3.1 Attributs de l'élément BODY

| <body | | | |
|---|---|---|---|
| text | = | couleur | couleur utilisée pour le texte. |
| bgcolor | = | couleur | couleur utilisée pour le fond (si “background” n’est pas utilisé). |
| background | = | URL | nom d’un fichier contenant une image et utilisé comme motif pour le fond. |

| | | | |
|---|---|---|---|
| link | = | couleur | couleur utilisée pour afficher les liens (en plus du soulignement). |
| alink | = | couleur | couleur utilisée pour les liens pendant que l'utilisateur clique un lien. |
| vlink | = | couleur | couleur utilisée pour les liens déjà visités. |
| onload, onunload | = | scripts | scripts associés au chargement ou au déchargement de la page (cf. module "HTML dynamique"). |
| >*éléments de la page*</body> | | | |

**Attributs spécifiques à Microsoft Internet Explorer**

| | | | |
|---|---|---|---|
| leftmargin | = | nombre | marge gauche de la page en pixels. |
| topmargin | = | nombre | marge supérieure de la page en pixels. |
| bgproperties | = | fixed | fixe l'image du fond pendant que le document se déroule. |

Il faut savoir que le navigateur fixe une valeur par défaut pour les marges. Pour forcer la valeur sous Internet Explorer, soit pour enlever les marges (valeur nulle), soit pour fixer la marge à une valeur fixe en fonction du décor d'arrière-plan, le concepteur emploie les attributs **leftmargin** et **topmargin**.

L'image de fond, par défaut, défile avec le texte. Pour forcer le navigateur à laisser l'image fixe, pendant que le texte se déroule, Internet Explorer propose l'attribut **bgproperties**.

Outre les attributs de marge, le concepteur de pages peut songer dès à présent à améliorer l'esthétique de sa page s'il y ajoute de la couleur, aussi bien en fond de page (arrière-plan ou background) qu'au niveau du texte, ou redéfinir pour la page donnée les couleurs des liens. Ceci est l'objet de la prochaine rubrique.

## 3.2 Couleurs d'affichage

La couleur désirée est évaluée à l'aide des trois composantes **Rouge, Vert et Bleu** pour lesquelles les valeurs sont indiquées en codage hexadécimal.

Une composante est représentée par un octet dont les valeurs varient entre 00 (0) et FF (255). Pour chacune des trois couleurs, il y a 256 nuances. La combinaison des trois couleurs peut ainsi créer plus de 16 millions de couleurs. Le langage HTML n'en accepte que 216.

Par exemple, la valeur "#FF0000" provoquera un affichage en rouge.

De même, la valeur "#00FF00" correspond à la couleur verte, "#0000FF" à la couleur bleue.

Les valeurs "#000000" et "#FFFFFF" correspondent respectivement aux couleurs noires et blanches.

La norme HTML 3.2 a défini seize couleurs reconnues par tous les navigateurs.

| | | | |
|---|---|---|---|
| Black | = #000000 | Green | = #008000 |
| Silver | = #C0C0C0 | Lime | = #00FF00 |
| Gray | = #808080 | Olive | = #808000 |
| White | = #FFFFFF | Yellow | = #FFFF00 |
| Maroon | = #800000 | Navy | = #000080 |
| Red | = #FF0000 | Blue | = #0000FF |
| Purple | = #800080 | Teal | = #008080 |
| Fuchsia | = #FF00FF | Aqua | = #00FFFF |

Il est aussi possible d'utiliser sous Netscape une palette de 140 couleurs aux noms prédéfinis. L'annexe de ce support de cours donne tous les noms de couleurs valides pour Netscape (version 3 et ultérieure) et les codes RVB correspondants.

A titre de premier exemple, la **couleur d'arrière-plan** de la page (attribut **bgcolor** ou background color) peut se définir de la manière suivante : **<BODY bgcolor**="#0000CC"**>.**

Une fois ceci réalisé, il faut la plupart du temps adapter la **couleur du texte** à la couleur du fond pour mieux le faire ressortir. Pour définir la couleur d'un texte pour l'ensemble de la page, l'attribut **text** de la balise **body** est utilisé.

Par exemple, la balise ainsi définie **<BODY BGCOLOR**="#0000CC" **TEXT**="#FFFF00"> provoquera l'affichage du texte en jaune sur fond bleu.

Ce paragraphe est aligné à gauche.

En ce qui concerne l'arrière-plan, il est possible de **substituer une image à une couleur de base**. Pour cela, l'attribut **background** de la balise **body** doit être initialisé avec l'adresse physique de l'image sur le site :

1. **<BODY background**="image.gif">, si l'image se situe au même endroit que la page qui l'appelle ;
2. **<BODY background=**"image/image.gif">, si l'image se situe dans un répertoire image de niveau inférieur à la page appelante;
3. **<BODY background="..**/image/image.gif">, si l'image se situe dans un répertoire image de même niveau où se trouve la page appelante.

**Rappels :**

Les chemins décrivant l'arborescence parcourue pour atteindre tel ou tel fichier dans le monde Internet utilisent le caractère « / » pour séparer les différents niveaux (comme sous UNIX).

Sans connaître l'arborescence, tout répertoire de niveau supérieur est désigné avec les caractères « .. ».

Pour redéfinir la **couleur d'un texte au niveau d'une section de la page** (paragraphe ou division), l'attribut **color** de l'élément **font** est utilisé. Par exemple, le code source **<font color=**"#FF0000"> change l'affichage du texte en rouge.

L'étude de l'élément **font** sera repris dans le chapitre suivant.

La normalisation HTML 4.0 rend désormais obsolète l'emploi de tous ces attributs, standards ou propriétaires, dans la mesure où le concepteur est censé utiliser les feuilles de style offrant au minimum les mêmes possibilités.

# Chapitre 4 : Mise en forme du texte

Avant de définir le style du texte lui-même, il faut évoquer le côté universel du langage qui ne prend pas en compte les caractères spécifiques de chaque pays. Si les principaux d'entre eux, correspondant aux définitions du clavier, sont acceptés par les éditeurs HTML, les autres ne sont accessibles que grâce à la consultation d'une liste spécifique.

Une fois franchi cet obstacle, le concepteur peut parcourir les différents éléments HTML aidant à la mise en place d'effets de style : styles prédéfinis ou styles particuliers.

Enfin le concepteur définit pour tout ou partie de sa page la police de caractères utilisée et ses principaux attributs : entre autres la taille et la couleur du texte.

## 1. Caractères accentués

### 1.1 Problématique

Un problème de portabilité des documents HTML réside, pour nous francophones, dans le support des **caractères accentués**. Si ces derniers sont saisis tels quels dans **un simple éditeur de texte**, les internautes risquent de voir s'afficher dans le navigateur des caractères bizarres.

Il en est de même de l'allemand, ou de l'espagnol qui utilisent des caractères accentués ou des caractères spéciaux totalement absents de l'anglais.

Par ailleurs dans le langage courant, on est parfois amené à utiliser des caractères tels que **&, @, $, ou #, ou ...**

Le langage HTML, lui-même, utilise des caractères de service qui lui sont donc réservés ( <, >, / ).

### 1.2 Jeu de caractères

Le langage HTML utilise le **codage ASCII 7 bits** (American Standard Code for International Interchange).

> Il s'agit du jeu habituel des micro-ordinateurs sous DOS ou sous UNIX. Il comporte 127 caractères accessibles au clavier. Les caractères accentués n'en font pas partie. Le langage HTML oblige à coder ces caractères appelés *entités*. Il est donc nécessaire pour rendre visible un caractère accentué, par tous sur le réseau Internet, de respecter la syntaxe qui lui est attribuée. A la réception, le navigateur décodera le caractère et l'affichera correctement à l'écran.

D'un autre côté, les claviers des pays occidentaux utilisent par défaut sous Windows le jeu **ANSI** ( American National Standards Institute ), un jeu de 256 caractères codés sur 8 bits.

> Il existe différentes version de ce jeu correspondant à des zones linguistiques. La version ANSI englobant les caractères accentués des langues occidentales s'appelle ***Latin-1*** et correspond à la norme **ISO-8859**. Les 128 premiers caractères sont ceux du

jeu ANSI. Ce jeu de caractères, utilisé par nos pages web, doit être indiqué aux navigateurs. Pour cela, l'éditeur HTML inclut sans le dire dans l'entête de la nouvelle page une balise méta qui mentionne l'emploi de ce jeu de caractères.

```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
```

## 1.3 Règle à suivre

Les éditeurs de texte simples comme Notepad ou Vi, obligent à coder les caractères accentués français. Cette tâche est désormais prise en charge par les éditeurs HTML du marché.

Les chaînes de caractères d'accentuation et caractères spéciaux principaux sont les suivants :

| | |
|---|---|
| acute | accent aigu |
| grave | accent grave |
| circ | accent circonflexe ( ^ ) |
| uml | tréma ( ¨ ) |
| tilde | tilde ( ~ ) |
| cedil | cédille de la lettre "c" |
| slash | slash de la lettre "o" (Ø) |
| lig | ligature de lettres |
|   | **un** espace |
| &amp; | & |
| &frac14; | ¼ (¾ pour ¾) |

| | |
|---|---|
| &lt; | < |
| &gt; | > |
| &copy; | © |
| &reg; | ® |
| &laquo; | « |
| &raquo; | » |
| &plusmn; | ± (plus ou moins) |
| &times; | × (multiplication) |
| &div; | ÷ (division) |
| &brvbar; | ¦ (barre vert. brisé) |
| &frac12; | ½ |

Pour accentuer un caractère donné, il faut placer devant le caractère en question le signe **&**, puis le caractère à accentuer et derrière le modificateur choisi dans la liste ci-dessus.

Si de plus ce caractère accentué n'est pas placé en fin de mot, il faut faire suivre la chaîne d'accentuation par le caractère < **;** > pour qu'il soit correctement compris par le navigateur.

**Exemple :**

Pour afficher la phrase, "Le HTML est plus aisé à lire qu'à écrire", il faut écrire :

```
Le HTML est plus ais&eacute &agrave lire qu'&agrave &eacute;crire
```

**Commentaires** :

- Insérer des espaces supplémentaires entre deux mots est possible avec le caractère accentué ** ** (non breaking space). Les éditeurs HTML lui donnent le nom d'**espaces insécables**.
- Le degré d'accentuation **lig** sert à accoler deux lettres entre elles comme "o" et "e" dans "nœud" (idem pour "a" et "e").

- Attention, si un correspondant étranger utilise le fichier, le navigateur risque de rencontrer des difficultés avec les accents utilisés.

Il reste le cas des caractères spéciaux qui ne sont pas directement accessibles au clavier : ¤, ¼ ... à moins de passer par leur code **ASCII**. Ainsi le signe **€** est obtenu au clavier par la combinaison ( ALT + 0 + 1 + 2 + 8 ), pour le coder en HTML on écrira & # 1 2 8 ; (sans espaces).
La liste non exhaustive des caractères spéciaux pour le jeu est ISO-8859-1 est disponible en annexe 2.

# 2. Styles prédéfinis

Les différents acteurs ayant contribué au développement du langage HTML ont apporté chacun des éléments de style. Nous serons exhaustifs dans notre exposé, mais nous les présenterons dans un ordre bien défini : des plus utilisés aux moins usités.

Toutes ces éléments de style doivent être utilisées avec leur balise de fermeture.

## 2.1 Mise en forme physique

Les plus courants sont les instructions de mise en forme physique qui sont plus proches du concepteur. En effet, avec les éléments ci-dessous, c'est le concepteur qui décide ce que le navigateur affiche.

| | | | |
|---|---|---|---|
| HTML 2.0 | |  texte  | affiche le **texte en gras** (bold ou boldface) |
| HTML 2.0 | |  texte  | affiche le *texte en italique* (italic) |
| HTML 2.0 | Netscape et Internet Explorer 3.0 | <s> texte </s> | affiche le ~~texte barré~~ (strikethrough) |
| HTML 2.0 | Netscape et Internet Explorer 3.0 | <u> texte </u> | affiche le <u>texte en souligné</u> (underline) |
| HTML 3.2 | Netscape 2.0 et I. Explorer 3.0 | <sub> texte </sub> | affiche le $_{\text{texte}}$ en indice (subscript) |
| HTML 3.2 | Netscape 2.0 et I. Explorer 3.0 | <sup> texte </sup> | affiche le $^{\text{texte}}$ en exposant (superscript) |

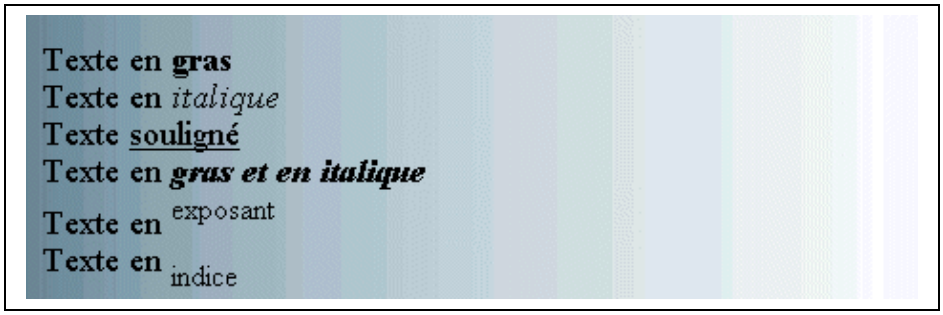
Pour appliquer un style prédéfini, il suffit d'encadrer le texte cible par les balises suivantes. Comme le montre l'exemple ci-dessous, il est possible de combiner plusieurs effets.

**Exemples**

```
<p><big>Texte en <b>gras</b><br>
Texte en <i>italique</i><br>
Texte <u>souligné</u><br>
Texte en <b><i>gras et en italique</i></b><br>
Texte en <sup>exposant</sup><br>
```

```
Texte en <sub>indice</sub><br>
</big></p>
```

**Résultat**




Il existe d'autres styles donnant le même effet.

| | | | |
|---|---|---|---|
| HTML 2.0 | |  | utilise **un renforcement** (souvent identique à <b>) |
| HTML 2.0 | |  | utilisé pour *faire ressortir un texte* (emphasize) |
| HTML 2.0 | Internet Explorer et Netscape 3.0 | <strike> | utilise ~~des caractères barrés~~ (strikeout text) |

Deux éléments sont là pour renforcer la taille de la police : **small** pour diminuer, **big** pour épaissir.

| | | |
|---|---|---|
| HTML 3.2 | <big> | affiche le texte avec une taille plus grande (par exemple taille 14 points si la taille courante était 12 points) : variante "large". |
| HTML 2.0, Netscape 2.0, I.Explorer 3.0 | <small> | affiche le texte avec une taille plus petite (par exemple taille 10 points si la taille courante était 12 points) : variante "small". |

Un dernier élément **blink**, non normalisé et non recommandé, servant au clignotement du texte, n'est reconnu que par le navigateur Netscape.

## 2.2 Mise en forme logique

Tous les styles qui suivent sont des instructions de mise en forme logique. Ils laissent le navigateur interpréter le formatage du texte. Ces éléments étaient souvent utilisés dans un environnement non graphique. Ils sont encore interprétés mais rarement utilisés par les concepteurs de pages. Ces styles assez nombreux peuvent avoir un rendu différent d'un navigateur à un autre. Deux éléments requièrent notre attention :

- l'élément **address** permettant de "signer" le document.
- l'élément **tt** permettant à la fois d'insérer des formats de données particuliers tout en leur donnant un rendu "télétype".

| | | |
|---|---|---|
| HTML 2.0 | <address> | *utilisé pour mentionner en général les coordonnées de l'auteur en fin de page, en particulier l'adresse électronique.* |
| HTML 2.0 | <cite> | utilisé pour *une citation* |
| HTML 2.0 | <code> | utilisé pour un code source |
| HTML 2.0 | <dfn> | utilisé pour *une définition* (define) |
| HTML 2.0 | <kbd> | affiche un texte **comme saisi au clavier** (keyboard) |
| HTML 2.0 | <samp> | utilise une fonte à largeur fixe pour une sortie standard de programmes, scripts, etc…(sample) |
| HTML 2.0 | <tt> | affiche le texte en mode teletype |
| HTML 2.0 | <var> | indique une instance d'une variable ou d'un paramètre d'un programme (variable). |

Tous les éléments cités dans cette partie appliquent un effet de style sans être accompagnés d'un saut de ligne. Les deux éléments ci-dessous, au contraire, incluent un saut de ligne avant et après. Ils possèdent aussi la propriété d'alignement. Toujours interprétés, ils ne sont plus mentionnés dans la norme HTML4.0. L'élément **listing** n'est plus interprété par Netscape6.

| | | |
|---|---|---|
| HTML 2.0 | <listing> | affiche le texte avec une police non proportionnelle à l'image des listing imprimés sur 132 colonnes. |
| HTML 2.0 | <xmp> | affiche le texte comme un exemple dans le document, utile lorsque des exemples de code HTML doivent être affichés (example). |

**Exemple d'élément ADDRESS**

```
<address>M. et Mme DUPONDT Jean<br>
6, place de la république<br>
35000 RENNES
</address>
```

**Résultat**


**Remarque :** les éléments accentués sont pris tels quels.

**Exemples des autres styles**


```
<tt>t&eacute;l&eacute;type</tt><br>
<code>code</code>
<listing>listing</listing>
<var>variable</var><br>
<samp>exemple</samp><br>
<kbd>clavier</kbd><br>
<cite>citation</cite>
<xmp>exemple</xmp>
<dfn>d&eacute;finition</dfn>
```

# 3. Polices de caractères

Les éléments **font** (HTML 3.2) et **basefont** (HTML 3.2.) ont pour but de fixer les caractéristiques de la police de caractères utilisée. Si le deuxième s'emploie dans l'entête de la page pour définir une fois pour toute les caractéristiques de la page, le premier permet de corriger ou de définir pour tout morceau de texte sa police et ses propriétés.

**Attributs**

| <font ou basefont | | | |
|---|---|---|---|
| size | = | nombre | modifie la taille du lettrage. La valeur par défaut est de 3. |
| face | = | polices | change la police de caractères du lettrage ou la famille de polices. Plusieurs valeurs sont possibles, alors séparées par une virgule. |
| color | = | code RVB | définit la couleur du lettrage global du document. Les valeurs acceptées sont les noms des couleurs prédéfinies (16 couleurs) ou le code hexadécimal de la couleur désirée. |
| > *texte* </font> | | | |

La balise de fermeture de l'élément **basefont** est interdite.

L'attribut size fait l'objet du paragraphe suivant.

**Exemple**

```
<p>Texte en <big>gros</big> et en <small>petit</small><br>
<font face="Arial">Texte avec une certaine police et des couleurs de base :
<font color="#FF0000">rouge</font>,
<font color="#00FF00">vert</font> et <font color="#0000FF">bleu</font>
</font></p>
```

**Résultat**


Les éditeurs HTML proposent des jeux de polices incluant deux polices et une famille de polices : face="Arial, Helvetica, sans-serif", face="Times New Roman, Times, serif", face="Courier New, Courier, mono". Les familles de polices sont indiquées en dernière valeur et en minuscules.

Le fait d'attribuer plusieurs valeurs s'explique par le fait que le visiteur ne possède pas forcément la première police indiquée. Si aucune police mentionnée n'est présente sur la machine, le navigateur applique la police par défaut notifiée dans ses paramètres.

# 4. Taille du texte

Ce paragraphe permet de définir l'expression **taille plus petite** et **taille plus grande**, et donc en conséquence de mieux comprendre comment fixer la taille des caractères.

## 4.1.Taille par défaut

Le langage HTML exprime la taille du texte avec un nombre compris entre **1** (la plus petite taille) et **7** (la plus grande). La taille de référence utilisée **par défaut** par les navigateurs correspond à la valeur **3**.

**Remarque** :

En fait, tous les navigateurs permettent de préciser cette valeur. Le choix initial est propre à l'utilisateur qui adapte ainsi la taille des caractères au confort de son moniteur ou à un éventuel handicap de lecture.

## 4.2.Valeur relative

Lorsque l'élément **big** est utilisé, celui-ci modifie la taille courante en ajoutant une unité à la taille.

Par exemple, si la taille initiale était 3, elle devient 4 jusqu'à ce que la balise **</big>** soit rencontrée. La taille initiale est alors restaurée avec sa valeur précédente : soit 3.

A l'inverse, l'emploi de l'élément **small** diminue la taille de référence.

Il est possible également de définir la taille et le style à utiliser sans recourir aux styles prédéfinis. Pour cela, le concepteur utilise l'élément **font** et notamment son attribut **size**.

Une autre façon de faire évoluer la taille de manière relative consiste à augmenter la taille des caractères à une **taille plus grande**. La syntaxe est la suivante : **<font size=+1>**.

Pour obtenir une **taille plus petite**, utilisons le code : **<font size=-1>.**

Pour obtenir une différence de taille plus prononcée, il est possible d'utiliser par exemple les valeurs "+2" ou "-2".

**Remarque** :

L'effet de la balise **<font size=+/- n>** n'est pas cumulatif. Ainsi, si la balise **<font size=+1>** est employée deux fois de suite, la taille ne sera pas augmentée de deux unités car la modification est fonction de la taille de référence et non de celle résultant de la dernière modification.

## 4.3.Valeur absolue

Il est également possible de fixer la taille des caractères à une **valeur absolue**, et non relative à la taille de référence. Pour cela, on doit préciser directement un nombre compris entre 1 et 7 comme valeur de l'attribut **size**.

Par exemple, l'emploi de la balise **<font size=5>** fixe la taille de la police à la valeur 5.

Il est possible cependant de fixer cette taille pour l'ensemble de la page. Cette modification est effectuée grâce à l'élément **basefont** qui supporte les mêmes attributs que l'élément *font* pour le navigateur MSIE, et uniquement l'attribut *size* pour le navigateur Netscape.

Ainsi, l'utilisation de la balise **<basefont size=4>** généralement dans l'entête, entraîne le fait que la taille des caractères ainsi définie devient la nouvelle taille de référence en lieu et place de celle définie au niveau du navigateur de l'internaute.

Pour choisir la taille qui convient le mieux, voici la correspondance entre les différentes valeurs de l'attribut *size* des éléments **<font>**, des tailles prédéfinies des titres **<Hn>** et les tailles de caractères exprimés en points comme sous les logiciels de bureautique.

| valeur "n" de <font size=n> | <Hn> correspondant | Taille en points |
|---|---|---|
| 0 | | 6 |
| 1 | <H6> | 8 |
| 2 | <H5> | 10 |
| 3 | <H4> | 12 |
| 4 | <H3> | 14 |
| 5 | <H2> | 18 |

| | | |
|---|---|---|
| 6<br>7 | <H1> | 24<br>36 |

- Ne pas oublier de faire correspondre une balise **</font>** à toute balise **<font ...>.**

Voici un exemple d'utilisation des éléments basefont et font :

**Exemple**

```
<HTML>
<head><title>Taille de caracteres </title>
</head>
<basefont size=3>
<body>
Ce texte est en taille 3
<font size=+1>
<p>Ce texte est en taille 3 + 1 = 4</p>
<font size=2>
<p>Ce texte est en taille 2</p>
</font>
<p>Retour en taille 4</p>
</font>
<p>Retour en taille 3 (basefont)</p>
</body>
</HTML>
```

Ce code affichera :

Ce texte est en taille 3
Ce texte est en taille 3 + 1 = 4
Ce texte est en taille 2
Retour en taille 4
Retour en taille 3 (basefont)

# 5. Eléments propriétaires

## 5.1 Elément marquee

Un autre élément **marquee**, non normalisé, reconnu depuis Internet Explorer 3.0, sert à afficher du texte défilant. Cet élément possède ses propres attributs.

| <marquee | | | |
|---|---|---|---|
| behavior | = | mots prédéfinis | Spécifie la manière dont le texte se déplace : disparaître sur les bords (SCROLL), rester visible au bord (SLIDE) ou changer de direction au bord (ALTERNATE). |
| bgcolor | = | couleur | Spécifie la couleur utilisée pour le fond. |
| direction | = | mots | Spécifie la direction vers laquelle le texte se déplace : |

| | | | |
|---|---|---|---|
| | | prédéfinis | DOWN \| LEFT \| RIGHT \| UP. |
| height width | = | nombre | Exprime en pixels la hauteur et la largeur de la bande défilante. |
| hspace, vspace | = | nombre | Spécifie en pixels la marge de gauche et de droite de la boîte de déroulement pour hspace, la marge du haut et du bas pour vspace. |
| loop | = | nombre \| infinite | Spécifie le nombre de déplacements du texte sur l'écran. Si la valeur est -1, le texte s'affiche tant que la page est ouverte. |
| scrollamount | = | nombre | Spécifie la vitesse de déplacement, en pixels, du texte sur l'écran. |
| scrolldelay | = | nombre | Spécifie le délai, en millisecondes, entre deux affichages successifs du texte. |
| > …*texte défilant*…</marquee> | | | |

La balise de fermeture est obligatoire.

Il n'existe pas d'équivalent normalisé. Le même effet peut être reproduit grâce à des éléments dynamiques : applet Java, ou bout de code en javascript (DHTML).

## 5.2 Elément MULTICOL

Cet élément, reconnu seulement par Netscape depuis sa version 3.0, permet au texte d'être affiché sur plusieurs colonnes (multicolonnage). Il possède trois attributs.

| <multicol | | | |
|---|---|---|---|
| cols | = | nombre | Cet attribut est obligatoire. Il spécifie le nombre de colonnes. Chaque colonne est délimitée par une balise <P>. |
| gutter | = | nombre | Spécifie en pixels le nombre d'espaces blancs entre chaque colonne. |
| width | = | nombre | Spécifie, en pixels ou en pourcentage, la largeur de l'espace occupé par les colonnes dans le document. |

Il n'existe pas d'équivalent normalisé. Seule la norme **CSS 2.0** relative aux feuilles de style permet cette fonctionnalité de mise en forme grâce à la propriété ***columns***.

# Chapitre 5 : Listes

Les énumérations, les sommaires imposent l'emploi de listes semblables à celles utilisées dans les logiciels de bureautique. Le langage HTML propose différents styles de listes dans les paragraphes de ce chapitre.

## 1. Listes à Puces

### 1.1 Puces Standards

#### Description

Ces listes prennent différents noms selon la littérature consultée : listes **non numérotées** (en rapport avec le nom de l'élément HTML correspondant), listes **simples**, listes **à bulles**.

Les éléments **ul** (unordered list, norme HTML2.0) définissent une liste où chaque article (ou item) est précédé d'un caractère graphique ou puce prenant souvent la forme d'une bulle.

En effet, ces puces ne décrivent aucun ordre. La hiérarchie n'est obtenue que par l'agencement des informations. Certains navigateurs offrent différentes puces selon le niveau hiérarchique de la liste.

Les items d'une liste sont toujours définis grâce à l'élément **li** (list item, norme HTML2.0). La balise de fermeture **</li>** est optionnelle.

#### Attributs

| | | | |
|---|---|---|---|
| <ul | | | |
| type | = | disc ou circle ou square | Spécifie un type de puces graphiques. |
| compact | | | Provoque le resserrement vertical des éléments d'une liste. |
| >*éléments li*</ul> | | | |

#### Exemple

```
<ul>
  <li>Item 1
    <ul>
      <li>Sous item 1
        <ul>
          <li>remarquons que la bulle change d'aspect &agrave; chaque niveau dans le
           navigateur.</li>
        </ul>
      </li>
      <li>Sous item 2</li>
    </ul>
  </li>
  <li>Item 2</li>
</ul>
```

**Résultat**

- Item 1
  - Sous item 1
    - Remarquons que la bulle change d'aspect à chaque niveau dans le navigateur.
  - Sous item 2
- Item 2

L'expression entre les balises **<li>** et **</li>** peut donc être une expression sophistiquée, y compris une nouvelle liste imbriquée, simple ou numérotée.

L'élément **li** possède l'attribut **type** qui permet, pour un article donné, de redéfinir le style de la puce.

**Exemple 2**

```
<ul type="square">
  <li>Pr&eacute;sentation
    <ol>
      <li>Historique</li>
      <li>G&eacute;n&eacute;ralit&eacute;s </li>
    </ol>
  </li>
  <li type="circle">Fonctionnement</li>
</ul>
```

**Résultat**

- Présentation
  1. Historique
  2. Généralités
- Fonctionnement

# 1.2 Puces Graphiques

## Description

Si les puces standard ne suffisent pas pour l'esthétique du site, il est toujours temps de recourir aux puces graphiques. L'opération consiste à remplacer l'élément **li** par la balise **img** associée à l'image de la puce. Cette solution est la meilleure car elle est valide quels que soient les navigateurs. Il faut néanmoins compléter la ligne d'un article par un saut de ligne (balise **br**).

**Exemple**

```
<UL>
<IMG SRC="gif/bout1.gif" BORDER="0" WIDTH="16" HEIGHT="16">Personnalisez
<BR>
<IMG SRC="gif/bout1.gif" BORDER="0" WIDTH="16" HEIGHT="16">vos puces !!
</UL>
```

**Résultat**


# 2. Listes Ordonnées

### Description

Ces listes prennent le nom de l'élément HTML correspondant, ou celui de listes numérotées.

L'élément **ol** (ordered list, norme HTML2.0) définit une liste où chaque article est précédé d'un nombre ou d'une lettre. Les items insérés permettent de créer une liste numérotée qui s'incrémente automatiquement.

### Attributs

| | | | |
|---|---|---|---|
| <ol | | | |
| type | = | A, a, I, i, 1 | Spécifie un type de numérotations : *1* par défaut. |
| start | = | nombre | Spécifie le numéro ordinal de départ du premier élément de la liste. |
| compact | | | Provoque le resserrement vertical des éléments d'une liste. |
| ><li> | | | |
| value | = | nombre | Redéfinit le numéro ordinal courant pour un article donné. |
| compact | | | Provoque le resserrement vertical du texte de l'article donné. |
| </li></ol> | | | |

L'attribut **type** permet de choisir le type de numérotation utilisé. Les valeurs possibles sont :

| | |
|---|---|
| A | pour utiliser des lettres majuscules, soit : A, B, C, …. |
| a | pour utiliser des lettres minuscules, soit : a, b, c, …. |
| I | pour utiliser des chiffres romains majuscules, soit : I, II, III, IV, … |
| i | pour utiliser des chiffres romains minuscules, soit : i, ii, iii, iv, … |
| 1 | pour utiliser des chiffres arabes, soit : 1, 2, 3, … |

### Exemple

```
<ol type="A">
  <li>Item 1
    <ol>
      <li>Sous item 1
```

```
        <ol type="i">
          <li>remarquez l'incr&eacute ;mentation</li>
          <li>automatique</li>
        </ol>
      </li>
      <li>Sous item 2</li>
    </ol>
  </li>
  <li>Item 2</li>
</ol>
```

**Résultat**

A. Item 1
    1. Sous item 1
        i. remarquez l'incrémentation
        ii. automatique
    2. Sous item 2
B. Item 2

L'attribut **start** définit la valeur initiale. Par défaut, la liste commence à la valeur un. Le type étant précisé, la valeur 5 de l'attribut start fait commencer la numérotation à "e" pour le type "a".

**Exemple 2**

```
<ol type="a" start="5">
        <li>Pr&eacute;sentation</li>
        <li>Fonctionnement</li>
</ol>
```

**Résultat**

e. Présentation
f. Fonctionnement

# 3. Listes de définitions

### Description

L'élément **dl** (definition list, norme HTML2.0) définit une liste permettant de présenter un texte décalé par rapport au bloc précédent. Cette indentation s'utilise par exemple pour définir un glossaire, donc présenter une liste des termes et expliciter leurs définitions.

Le terme à définir est spécifié grâce à l'élément **dt** (definition term, norme HTML2.0) et apparaît dans une première ligne et colonne alignée sur le bord gauche de la page. La balise de fermeture **</dt>** est optionnelle.

La définition est spécifiée grâce à l'élément **dd** (definition description, norme HTML2.0) et apparaît dans une seconde ligne et colonne décalée par rapport à la première. La balise de fermeture **</dd>** est optionnelle.

**Attributs**

| <dl | |
|---|---|
| compact | Provoque le resserrement vertical des éléments d'une liste. |
| >*éléments dt et dd*</dl> | |

**Exemple**

```
<dl>
  <dt>Terme 1</dt>
        <dd>D&eacute ;finition 1</dd>
  <dt>Terme 2</dt>
        <dd>D&eacute ;finition 2</dd>
</dl>
```

**Résultat**

Terme 1

Définition 1

Terme 2

Définition 2

# 4. Autres éléments de liste

Ces éléments HTML sont donnés à titre indicatif, dans la mesure où la norme HTML 4.0 les a déclaré obsolètes. Néanmoins, les navigateurs continuent de les interpréter.

Les attributs spécifiques aux listes à puces sont valides sous Internet Explorer 6 et Netscape 6.

## 4.1 Les listes répertoires

L'élément **dir** (directory list, norme HTML2.0) définit une liste où les items, qui ne doivent pas avoir plus de 20 caractères, sont disposés en colonnes.

Peu de navigateurs savent réaliser cet affichage en colonnes et ce style a souvent le même effet qu'une liste à bulles.

## 4.2 Les Listes Menus

L'élément **menu** (norme HTML2.0) définit une liste d'items simples tels que ceux que nous rencontrons dans un menu.

Ce type de liste a également souvent le même effet qu'une liste à puces.

**Exemple**

```
<menu>
  <li>Premier item du menu</li>
  <li>Second item du menu</li>
</menu>
```

**Résultat**

- Premier item du menu
- Second item du menu

# Chapitre 6 : Images

L'affichage d'images dans vos pages est la première étape dans la constitution de documents multimédias. Les images offrent de nombreuses possibilités.

La balise **img** permet d'appeler et d'insérer dans un document HTML (page Web) des images.

## 1. Fichiers image

Ces images doivent arborer des formats peu gourmands en taille pour pouvoir être acheminées sans encombrer le réseau. Par convention, les navigateurs graphiques reconnaissent les trois formats principaux suivants : **GIF** (Graphic Interchange Format) ou **JPEG** (Joint Photographic Experts Group) ou **PNG** (Public Network Graphic).

### 1.1 Format JPEG

Le format JPEG (ou JPG) est consacré aux photos scannées riches de milliers de couleurs car il n'est pas limité en nombre de couleurs. Afin de limiter la taille d'une photo, le fichier est toujours compressé à un certain taux. En fait, cette réduction joue sur la qualité de l'image. Il ne gère pas l'effet de transparence comme le format GIF, mais il dépasse la limite des 256 couleurs. Le taux de compression d'une image peut être déterminé entre 1 et 99 %.

**Exemples d'images compressées**

| Taux : 10% - 5.1 Ko | Taux : 30% - 3.2 Ko | Taux : 60% - 2.5 Ko | Taux : 80% - 1.9 Ko |
|---|---|---|---|
|  |  |  |  |

Plus on augmente le taux de compression, plus la qualité est mauvaise. Le meilleur taux se situe certainement dans la fourchette 10-30%. Le format JPEG gère également l'affichage progressif.

### 1.2 Format GIF

Le format GIF est certainement le format le plus utilisé sur le Web. Il est limité à 256 couleurs, donc il ne convient pas dans le cas de photographies très colorées ou avec beaucoup de nuances. Par contre, pour insérer un logo, une icône ou même une banderole, des schémas, des images animées, il est imbattable rapport qualité / taille.

Ce format propose deux caractéristiques intéressantes : la possibilité de définir une couleur dite de transparence et l'affichage progressif (ou encore entrelacé).

En conséquence, le format GIF regroupe toutes les images autres que les photos.

**Exemple de format gif avec et sans couleur de transparence**

| Avec Transparence | Sans Transparence |
|---|---|
| GIF | GIF |

Dans cet exemple, la couleur noire est la couleur de transparence.

Le nombre de couleurs joue sur le rapport taille/qualité du fichier. Voici une image (taille originale : 31 Ko - Format Tiff) enregistré sous un format gif avec différentes palettes de couleurs (256 couleurs - 64 couleurs - 32 couleurs - 16 couleurs). La différence est appréciable.

| 256 couleurs - 6.9 Ko | 64 couleurs - 5.6 Ko | 32 couleurs - 4.3 Ko | 16 couleurs - 3.2 Ko |
|---|---|---|---|
|  |  |  |  |

**Exemple de gif avec et sans couleur de transparence.**

Le cas du format GIF entrelacé se rencontre lorsqu'une image floue en début d'affichage devient progressivement nette. Il est utilisé pour charger une image de taille importante (plus de 30 Ko) sans pour autant pénaliser le travail du navigateur. Sa taille est alors très légèrement supérieure.

Pour créer une image au format GIF avec un effet de transparence, il faut choisir la version 89a du format GIF, car la version 87a ne gère pas cet effet.

**Exemple d'un gif entrelacé (256 couleurs - 6.9 ko).**

## 1.3 Format PNG

Le format PNG, encore peu présent dans les pages Web, est censé reprendre les qualités de l'un et l'autre format précité. Contrairement aux autres formats, ce n'est pas un format propriétaire. Il est recommandé par le consortium W3C pour toute utilisation. Cependant, très peu de navigateurs sont en mesure d'interpréter ce format.

Il gère la transparence (plusieurs niveaux de transparence), et il ne se limite pas à 256 couleurs. Malheureusement la taille obtenue demeure trop importante. Il se montre moins efficace que le format JPEG pour la compression d'image alors que la qualité est excellente. Le format PNG gère également l'affichage progressif.

## 1.4 Retouche d'une image

Pour insérer une image dans une page, deux cas se présentent :

- soit l'image possède le bon format, la bonne taille, etc… , alors elle peut être appelée telle quelle dans la page ;
- soit l'image ne possède pas les caractéristiques attendues, alors elle nécessite une modification à l'aide d'un logiciel de retouche photo.

Les modifications usuelles sont les suivantes :

- changement de format,
- changement de taille,
- recadrage (enlever les vides ou parties inutiles),
- changement de couleurs, effet visuel
- transparence du fond d'image.

Les logiciels du marché abondent dans ce domaine : Paint Shop Pro, Adobe Photoshop, etc…

# 2. Balise "img"

La balise img est dédiée à l'origine (norme HTML2.0) à l'insertion des images dans une page web, via une adresse URL. Depuis, les éléments EMBED (Netscape) et OBJECT permettent d'insérer des images.

**Attributs**

| <img | | | |
|---|---|---|---|
| src | = | URL | Adresse URL du fichier GIF ou JPEG (JPG) ou PNG. |
| align | = | mots réservés | Alignement de l'image par rapport au texte situé sur la même ligne. valeurs possibles : texttop, top, absmiddle, middle, absbottom, bottom, baseline, left ou right. |

| | | | |
|---|---|---|---|
| alt | = | texte | Texte affiché si l'affichage des images est désactivé (cf. navigateur). Texte utilisé comme info-bulle lorsqu'on survole l'image avec la souris. Par défaut, un éditeur avancé affiche le nom et la taille en octets. |
| border | = | nombre | Epaisseur de la bordure exprimée en pixels. Cet encadrement n'est visible que si un lien est posé sur l'image, la valeur par défaut étant 2. |
| height | = | nombre | Hauteur en pixels réservée pour l'image. |
| hspace | = | nombre | Marge, en pixels, insérée à gauche et à droite de l'image |
| ismap | = | texte | Identifie la carte graphique associée à l'image côté serveur web. |
| usemap | = | texte | Identifie la carte graphique associée à l'image côté navigateur. |
| vspace | = | nombre | Marge, en pixels, insérée au dessus et en dessous de l'image. |
| width | = | nombre | Largeur réservée à l'image en pixels. |
| > | | | |

La balise de fermeture est interdite.

**Remarque 1 :**

Seul l'attribut **src** est obligatoire, tous les autres sont optionnels.

**Remarque 2** :

Deux autres attributs non obligatoires mais fort utiles sont ceux précisant la taille prise par l'image dans la page. Un des intérêts se situe au niveau du navigateur. Le navigateur interprète le code source de la page au fur et à mesure du chargement de la page. Lorsqu'il rencontre la balise **img**, il réserve une place pour l'image en fonction de ses dimensions initiales, puis il charge l'image pendant qu'il continue la lecture du fichier.

Un paragraphe traitant de la taille d'une image explique à nouveau l'importance de ces attributs.

# 3. Compléments sur l'alignement

Les attributs **top**, **middle** et **bottom** permettent de spécifier la position de l'image par rapport à la ligne de texte dans laquelle cette image apparaît. La hauteur de cette ligne de texte devient donc la plus grande valeur entre la hauteur de l'image et la taille du texte.

Cela peut provoquer des effets indésirables si l'image est insérée au milieu d'un texte.

**Exemple avec l'alignement *middle*** :

```
<HTML>
<body>
<p>D&eacute ;but du texte <img src="mauvais.gif" align="middle"> suite du texte.<br>
Ligne suivante, l'effet obtenu est peu esth&eacute;tique.
```

```
</p>
</body>
</HTML>
```

**Résultat**

Début du texte  suite du texte.

Ligne suivante, l'effet obtenu est peu esthétique.

**Exemple avec l'alignement *bottom*** :

Ces attributs s'emploient plus volontiers en début ou fin de ligne dans le cas d'un texte isolé. Ils peuvent s'employer en début de paragraphe avec l'attribut **bottom**, comme dans l'exemple suivant :

```
<HTML>
<body>
<p><img src="oreille.gif" align="bottom">Avis aux m&eacute;lomanes :<br>
Pour profiter des qualit&eacute;s sonores….</p>
</body>
</HTML>
```

**Résultat**

 Avis aux mélomanes :

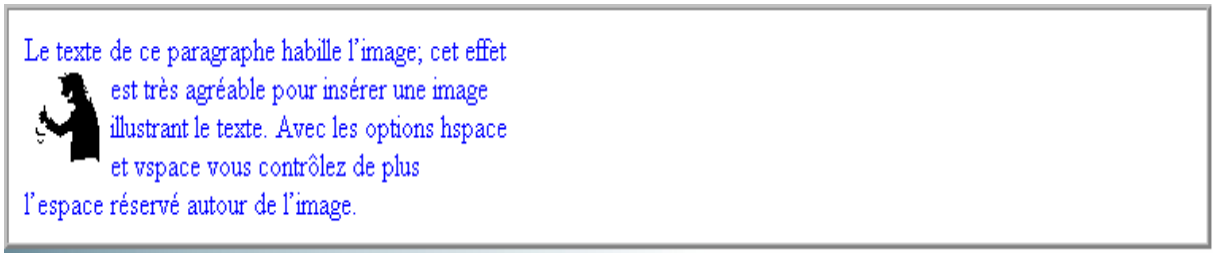
Pour profiter des qualités sonores….

**Exemple avec l'alignement *left ou right*** :

Pour insérer une image dans un paragraphe de texte, nous utiliserons préférentiellement les attributs **left** et **right**. Ces attributs placent l'image à gauche ou à droite de la page et habillent le texte autour de l'image comme dans l'exemple ci-dessous qui utilise l'attribut **left** :

```
<HTML>
<body>
<p>Le texte de ce paragraphe habille l'image ; cet effet<br>
<img src="correct.gif" align="left">
est tr&egrave;s agr&eacute;able pour ins&eacute;rer une image<br>
illustrant le texte. Avec les attributs hspace<br>
et vspace vous contr&ocirc;lez de plus<br>
l'espace r&eacute;serv&eacute; autour de l'image.</p>
</body>
</HTML>
```

**Résultat**

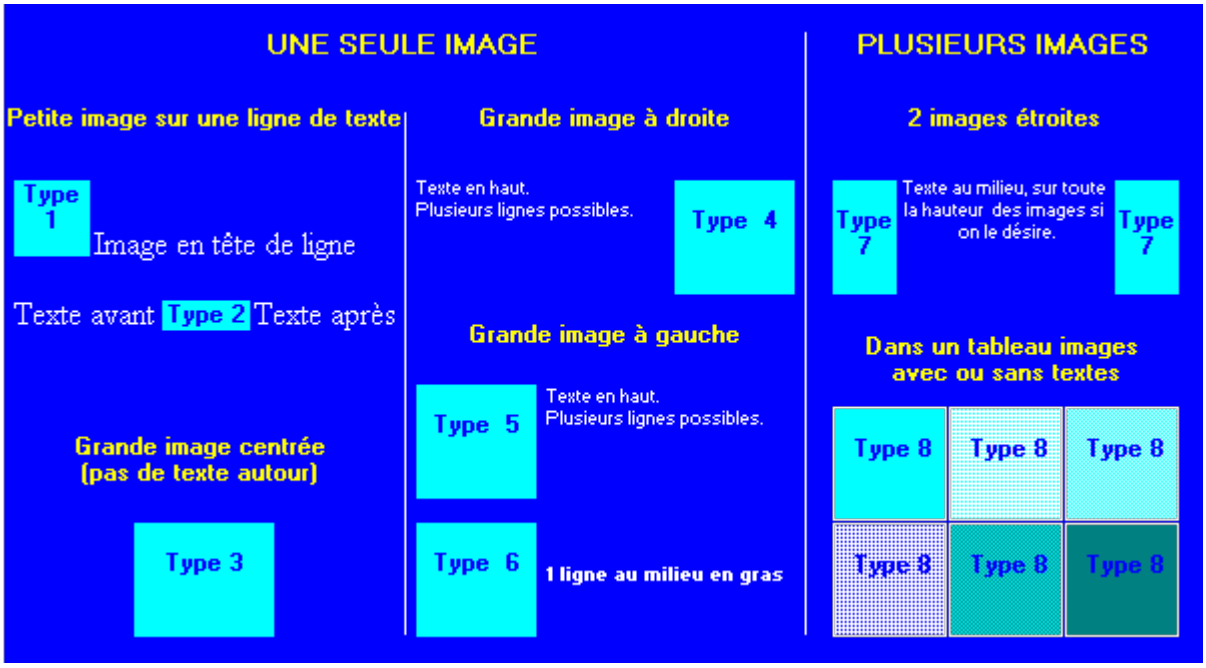



**Note** :

Les éléments ***br*** insérés ici ne servent qu'à illustrer l'exemple. Habituellement ce sont les dimensions même de la page (sa largeur) qui provoquent cet habillage du texte.

Il existe plusieurs autres façons de placer une image dans un texte. Le placement, et donc le rendu, n'imposent pas toujours l'emploi de l'attribut ***align***.

# 4. Huit types de présentations




**Type 1** : l'image est vue comme un caractère ou un mot devant le texte qui la suit.

Par exemple, cela pourrait être une lettrine en tête d'un paragraphe. Pour réaliser cette présentation, nous insérons simplement l'image en début du texte. Le source HTML correspondant est :

```
<p><img src="NomFichier"> Image en t&ecirc;te de ligne</p>
```

**Type 2** : l'image est vue comme un mot dans une ligne de texte.

Le source HTML correspondant est simplement :

```
<p>Texte avant<img src="NomFichier" height=15 hspace=10> Texte apr&egrave;s</p>
```

Ici, le fait de fixer l'attribut hspace à la valeur 10 décolle le bord gauche de l'image du texte qui la précède et son bord droit du texte qui la suit.

De plus, une image insérée dans une ligne ne doit pas être plus haute que cette ligne (voir plus haut l'effet inesthétique obtenu).

Pour assurer une bonne insertion dans un texte en fonte courante (Times New Roman 12, maigre), la hauteur de l'image est fixée à height=15.

**Type 3** : pour centrer une grande image sur la page.

```
<p><center><img src="NomFichier" height="hauteur" width="largeur" alt="texte"></center> </p>
```

L'attribut **alt** est en premier lieu utilisé pour afficher un texte *alternatif* à l'image dans le cas où le navigateur du client est limité à l'affichage de textes, dans le cas où le temps de chargement est long ou si l'internaute a désactivé l'affichage des images. Cet attribut est désormais utilisé aussi pour afficher une légende spécifique à l'image, fonction du message à communiquer.

Pour ces trois premiers types, l'attribut **align** n'était pas nécessaire. Par contre, les types 4 à 7 l'exigent.

**Type 4** : pour aligner l'image à droite, il faut utiliser l'attribut align avec la valeur right.

```
<p><img src="NomFichier" align=right> notre texte </p>
```

Le texte inséré est alors bordé à droite par l'image. Le texte est, à nouveau, aligné sur toute la largeur de la page si la balise **<br clear="right">** est insérée ou si le texte descend plus bas que l'image.

**Type 5 :** ce type est symétrique du type 4.

**(Ile de Ré)**

L'alignement de l'image est fixé à gauche ainsi que l'attribut **clear** de l'élément saut de ligne (**br**). Pour ne pas coller le texte à l'image, ajouter une valeur 10 à l'attribut **hspace**. Le code complet est donc :

```
<p><img src="Re.gif" align="left" hspace="10" alt="Ile de R&eacute" ... > notre texte </p><br clear="left">
```

**Type 6** : pour forcer l'affichage du texte à mi-hauteur de l'image, l'attribut **align** avec la valeur ***middle*** est utilisé.

La ligne de texte est suivie de "<br clear=left>" afin de libérer la largeur totale de l'écran. Le code est :

```
<p><img src="NomFichier" align="middle" ... >
<b>notre texte></b><br clear=left></p>
```

**Type 7** : pour insérer une image de chaque coté du texte, il faut les définir avant le texte.

```
<p>
<img align="left" hspace="10" src="NomFichier1" alt="texte1" height="hauteur 1"
width="largeur 1">
<img align="right" hspace="10" src="NomFichier2" alt="texte2" height="hauteur 2"
width="largeur 2">
notre texte
<br clear></p>
```

La balise **<br clear>** signifie "libérez tout" (soit à gauche et à droite). Si les deux images n'ont pas la même hauteur, il est possible de libérer un coté puis l'autre.

Le texte inséré entre les deux images peut, bien sûr, contenir n'importe quel élément y compris des images. L'attribut ***align*** avec les valeurs *left* ou *right* ne pourra cependant pas leur être appliqué.

**Type 8** : cette représentation est obtenue assez simplement en utilisant les tableaux.

Ceux-ci seront étudiés plus loin. En effet, dans les cellules d'un tableau, il est possible d'insérer aussi bien du texte que des images, et ainsi obtenir ce type de présentation.

# 5. Bordure d'image

Cet attribut **border** permet de tracer un cadre autour de l'image en précisant son épaisseur en nombre de pixels. Mais le résultat obtenu n'est pas le même avec tous les navigateurs.

Le navigateur Microsoft Internet Explorer ne dessine de bordure que si l'image cache un lien et si la valeur précisée n'est pas nulle.

Le navigateur Netscape Navigator dessine une bordure même si l'image ne cache pas un lien.

# 6. Taille de l'image

Les attributs **height** et **width,** bien qu'optionnels, sont très importants.

En effet, si ces paramètres sont absents, le navigateur réserve une place arbitraire pour l'image, et continue à afficher le texte. Puis il revient à cette image pour l'afficher. Il existe alors peu de chances pour que la place réservée corresponde à la taille réelle. En conséquence, le texte est réaffiché, ce qui provoque des effets visuels non désirables.

Si de plus l'utilisateur a choisi de ne pas afficher les images, la présentation du texte risque de ne pas correspondre à la mise en page prévue.

Pour ces raisons, il est vivement conseillé de préciser la taille de votre image avec ces attributs.

**Si l'aspect d'origine doit être préservé, il faut donner les dimensions réelles de l'image.**

Si par contre, l'image doit être réduite ou agrandie, il suffit de donner les dimensions souhaitées. Ces dimensions sont toujours données en pixels, il n'est pas possible de donner une valeur en pourcentage (relative à la taille réelle).

Les éditeurs avancés permettent de retailler celle-ci après l'avoir sélectionnée, à l'aide des poignées de redimensionnement, mais l'effet va à l'encontre de la qualité de l'image.

Si la taille de l'image importée dans le document déborde par rapport à la largeur ou à la hauteur de l'écran (ou les deux), l'affichage de la page entraîne l'apparition de barres de défilement (ou ascenseurs).

Il est vivement conseillé de redimensionner l'image avant son insertion dans la page à l'aide d'un logiciel de retouche d'images.

# 7. Affichage plus rapide

Pour améliorer la vitesse d'affichage des images, il faut améliorer leur vitesse de téléchargement. Cette vitesse est principalement fonction de leur taille en Ko. Il est conseillé de ne pas dépasser une taille de 30Ko sur une ligne classique.

A cette fin, il est conseillé de ne pas enregistrer les images en millions de couleurs si elles ne contiennent qu'un faible nombre de couleurs différentes (moins de 16 ou moins de 256).

Eventuellement, différents essais doivent être effectués avec un logiciel de dessin (ou de retouche d'images) pour trouver le meilleur compromis entre la taille du fichier et la qualité de l'image.

Le navigateur de Netscape permet également d'obtenir un affichage rapide en fournissant deux images. La première, de moindre qualité, sera affichée en premier. La seconde, de meilleure qualité, sera affichée après la première lecture du code source (texte et images en basse résolution).

L'attribut **lowsrc**, proposé par les éditeurs, est utilisé à cet effet, comme dans l'exemple ci-dessous :

```
<img src="img16Mcl.jpg" lowsrc="img16clr.gif" width="50" height="50">
```

Ceci permet un affichage rapide de l'image GIF en 16 couleurs (img16clr.gif) puis de l'image JPEG en 16 millions de couleurs (img16Mcl.jpg) une fois que le document est mis en place. Les tâches accomplies par le navigateur sont ainsi optimisées et ce dernier fournit au lecteur un document optimum à tout moment.

# 8. Images à liens (Images Map)

Une utilisation intéressante des images consiste à leur poser un ou des liens (cartes) dessus pour viser d'autres documents HTML.

L'étude de ce paragraphe impose d'avoir étudié au préalable le chapitre sur les liens hypertextes (chapitre suivant), car il s'agit en pratique de poser plusieurs liens sur une même image.

## 8.1.Image à lien unique

Envisageons d'abord le cas d'un seul lien. La valeur de l'attribut href de l'élément a (lien) désigne le document cible. L'adresse du document visé peut comprendre un chemin absolu (incluant un nom de serveur) ou un chemin relatif à l'endroit où il est stocké.

Voici un exemple de lien simple :

Pour visualiser le texte, cliquez sur la page : 🗋

Le code source HTML correspondant est :

```
<HTML>
<body>
<p>Pour visualiser le texte, cliquez sur la page :
<a href="page.htm"><img src="doc.gif" border="0"></a></p>
</body>
</HTML>
```

# 8.2.Images à liens multiples (Cartes graphiques)

## Description

Une autre façon d'utiliser une image pour viser d'autres pages web est de transformer cette image en carte graphique. Il s'agit, en général avec l'aide d'un éditeur HTML avancé, de dessiner des formes géométriques (élément area) sur l'image, puis de poser un lien sur chacune d'entre elles. L'ensemble de ces formes constitue une carte (élément map) relié à l'image par un identifiant commun. Ces "image-plan" permettent ainsi d'accéder à différents documents en cliquant sur les différentes zones sensitives de l'image.

D'autres noms sont donnés à ce genre d'images : image MAP, image sensitive, etc...

L'élément **map** (norme HTML3.2) est un bloc d'instructions qui rassemble les différentes définitions des aires (éléments **area**) dessinées sur la carte. Comme pour tout bloc, la balise de fermeture est obligatoire. Cet élément n'accepte qu'un paramètre spécifique mais obligatoire : l'attribut **name** qui permet certes de nommer cette carte comme objet HTML de la page, mais surtout de définir la carte comme cible de l'image.

L'élément map contient autant d'éléments **area** (norme HTML3.2) que de zones géométriques. Chaque zone comporte outre le système de coordonnées (attribut **coords**) de celle-ci, le type de zone géométrique (attribut **shape**), l'adresse de la page cible (attribut **href**). Il est aussi possible de rajouter l'attribut **alt** comme l'attribut **title** comme info-bulles spécifique à la zone en question.

## Attributs

| <area | | | |
|---|---|---|---|
| href | = | adresse url | Désigne le document visé par ce lien. |
| nohref | | (sans valeur) | Indique que cette zone n'est liée à aucun document. |
| target | = | texte | Spécifie dans quel cadre s'affiche le document cible du lien. Ce cadre peut servir à afficher une page de résultat. |
| alt | = | texte | Texte alternatif pour les navigateurs textuels. |
| *Avec* | | | |
| shape | = | rect *ou* rectangle | Définit une zone rectangulaire. |
| coords | = | "x1, y1, x2, y2" | Désigne les coordonnées du rectangle. |
| *Ou* | | | |
| shape | = | circ *ou* circle | Définit une zone circulaire. |
| coords | = | "centre_x, centre_y, rayon" | Désigne les coordonnées du cercle. |
| *Ou* | | | |
| shape | = | poly *ou* polygon | Définit un polygone automatiquement fermé. |

| | | | |
|---|---|---|---|
| coords | = | "x1, y1, x2, y2, x3, y3,..." | Désigne les coordonnées du polygone (au moins trois points). |
| > | | | |

**Exemple de quatre carrés de taille identique**

```
<img src="des.gif" width="100" height="100" usemap="#plan" border="0" >
<map name="plan">
   <area shape=rect coords="0,0,49,49" href="page1.htm">
   <area shape=rect coords="50,0,99,49" href="page2.htm">
   <area shape=rect coords="0,50,49,99" href="page3.htm">
   <area shape=rect coords="50,50,99,99" href="page4.htm">
</map>
```

**Résultat**

Ce plan est utilisé par l'image suivante : 

L'image et la carte ainsi définie possèdent chacun un attribut permettant de les relier. Il s'agit respectivement de l'attribut **usemap** de l'image ayant comme valeur le nom de la carte précédé du caractère "#", et l'attribut **name** de la carte.

Le caractère "#" précise que cette carte est définie dans la même page que cette balise.

**Remarque**

Une valeur **default** existe pour l'attribut shape : il concerne les points non définis précédemment.

**Remarque**

L'attribut **ismap** de la balise img est utilisé de la même façon que l'attribut usemap, au profit de la définition et du traitement des cartes graphiques. La différence fondamentale réside dans le fait que cette carte graphique n'est pas interprétée côté navigateur, comme ci-dessus, mais côté serveur web. Cet attribut permet de repérer la position de la souris sur un graphique pour exécuter des actions différentes suivant la région cliquée.

L'exemple ci-dessous donne la liste des fournisseurs d'une région cliquée de France. Pour atteindre la procédure CGI de traitement associée, il suffit d'ajouter à la balise IMG, l'attribut ISMAP, comme le montre l'exemple suivant :

```
<A HREF="/cgi-bin/fourniss.pl">
<IMG SRC="gif/france.gif" BORDER="0" ISMAP></A>
```

- *fourniss.pl* est le nom de la procédure CGI écrite en langage Perl qui va être appelée sur le serveur.

- ISMAP permet de donner les coordonnées de la souris sur l'image, l'appel sera fait sous la forme *fourniss.pl?x,y* où x et y sont les coordonnées pointées.

### Conclusion

De nos jours, l'élément **object** a tendance à remplacer l'élément image, surtout en ce qui concerne la définition de cartes graphiques.

# Chapitre 7 : Liens

## 1. Généralités

### 1.1 Définitions

Un lien permet de définir dans un document une région sensible au clic de souris et en même temps, l'adresse de la cible visée par ce clic.

Cette région sensible peut être :

- soit un texte allant du mot à la phrase complète, on définit alors un **lien hypertexte**
- soit une image (ou une zone dessinée dans cette image), on parle alors de **lien hypermédia**

Cette adresse est le plus souvent celle d'une page Web différente de celle où le lien est posé. On définit ainsi un **lien externe** entre une page appelante et une page appelée ou cible.

Cette adresse peut désigner également un endroit précis dans le même document. Il s'agit d'un **lien interne** visant une **ancre nommée** ou signet.

La juxtaposition des deux types de liens, lien externe puis lien interne, est désignée sous le vocable de **lien mixte**.

Cette adresse peut aussi bien viser des pages web que d'autres types de fichiers : fichiers multimédias (image, vidéos), fichiers exécutables (*.exe, *.cgi, *.php, *.asp, etc...).

Cette adresse peut enfin viser d'autres services que le service Web : serveur ftp, serveur de news (forum de discussion), mais aussi une boite aux lettres d'un serveur de courrier.

L'élément **a** (anchor, norme HTML2.0) est non seulement utilisé pour poser n'importe quel type de lien, mais aussi pour désigner la cible d'un lien interne.

### 1.2 Attributs

| <a | | | |
|---|---|---|---|
| name | = | nom | signet (étiquette de saut) posé dans une page visé par un lien interne ou mixte. |
| href | = | fichier cible | adresse URL d'un fichier situé sur un autre serveur Web ou chemin du fichier situé sur le même serveur. |
| target | = | nom | nom d'une fenêtre (ou cadre ou frame) dans laquelle le fichier cible s'affiche (norme HTML 4.0) |
| > *objet texte ou image sur lequel est posé le lien* </a> | | | |

Seul l'attribut **href** est obligatoire.

L'attribut **name** est explicité dans le paragraphe des liens internes.

L'attribut **target** est surtout utilisé si le concepteur de site découpe la fenêtre principale de l'écran en fenêtres secondaires ou cadres. Il désigne ainsi l'endroit où la page visée par le lien doit s'afficher : fenêtre principale du navigateur, fenêtre secondaire ou cadre ou frame, etc...

Le concepteur peut ouvrir une nouvelle instance de navigateur. La valeur réservée **_blank** (ou **_new**) est utilisée. Les autres valeurs possibles (_self, _top, _parent) sont uniquement utiles dans le cas de redistribution des pages web cibles entre les différents cadres définies dans la fenêtre principale d'un navigateur (voir chapitre sur les cadres).

**Exemple d'un lien posé sur un texte :**

```
<a href="montagne.htm">Montagne</a>
```

**Exemple d'un lien posé sur un élément multimédia :**

```
<a href="liens-1n.htm">
<img src="imag/Image846.gif" border=0 width=15 height=13>
</a>
```

**Remarques :**

Il existe d'autres attributs très peu usités analogues pour quelques uns avec ceux de l'élément d'en-tête **link** (voir chapitre sur les éléments d'en-tête) : **rel** et **rev**. Ces derniers, admettant les valeurs prédéfinies {top|contents|index|glossary|copyright|next|previous|help|search}, précisent le type de relation entre le document courant et le document visé.

La norme HTML 4.0 ajoute les attributs devenus génériques pour tous les éléments HTML. De plus, elle permet de :

- poser un lien sur une zone de carte graphique (voir chapitre sur les images) à l'aide des attributs **shape** et **coords**,
- définir une touche du clavier pour activer le lien (attribut **accesskey**), ou de définir un ordre de tabulation entre les différents liens de la page (attribut **tabindex**),
- préciser dans le cas d'un site international le jeu de caractères de la page appelée (attribut **charset**).

# 1.3.Résultats visibles dans le navigateur

Une fois posés ces liens, il faut enregistrer les modifications effectuées dans le code source du fichier (document ou page Web) et aller constater le résultat dans le navigateur.

Le texte sur lequel est posé le lien apparaît, par défaut, en bleu (lien actif non cliqué ou lien non visité). Le texte est de plus souligné.

L'image sur laquelle est posé le lien peut apparaître avec une bordure.

Dans les deux cas, le survol des objets liés (texte ou image) provoque le changement de curseur de souris en main. Deuxièmement, leur survol provoque l'apparition de l'adresse du document cible dans la barre d'état du navigateur.


En ce qui concerne la couleur des textes liés, une fois le lien cliqué (lien visité), le texte prend une autre couleur (violet par défaut).

Ces couleurs d'affichage sont définies par défaut dans les options du navigateur : onglet « Général », bouton « Couleurs ».

Le concepteur de site a le choix suivant : celui de laisser le navigateur de l'internaute choisir les couleurs, ou celui d'imposer les siennes.

Dans le deuxième cas, plusieurs solutions sont possibles. Elles peuvent être définies pour une page donnée à l'aide des attributs de l'élément BODY, comme dans l'exemple ci-dessous :

```
<BODY TEXT="#000000" LINK="#0000ff" VLINK="#800080">
```

L'attribut LINK modifie la couleur des liens activés, alors que l'attribut VLINK modifie la couleur des liens visités.

Elles peuvent être fixées pour l'ensemble des pages d'un site à l'aide des feuilles de style externes (voir le chapitre sur les feuilles de style).

# 2. Liens Internes

Les liens internes permettent de se déplacer à l'intérieur d'un fichier HTML sans que le visiteur soit obligé de faire défiler l'ascenseur.

La cible visée par un lien interne s'appelle une « ancre nommée ou signet ». L'ancre nommée est définie par l'élément **<a name="signet">***objet cible***</a>**.

Il convient de mettre en place dans le document courant la cible avant de poser le lien.

Au niveau de l'objet sur lequel est posé le lien, la valeur de l'attribut "href" est initialisée avec le nom de l'ancre définie ci-dessus précédé du signe « # », indiquant que la cible reste interne à la page.

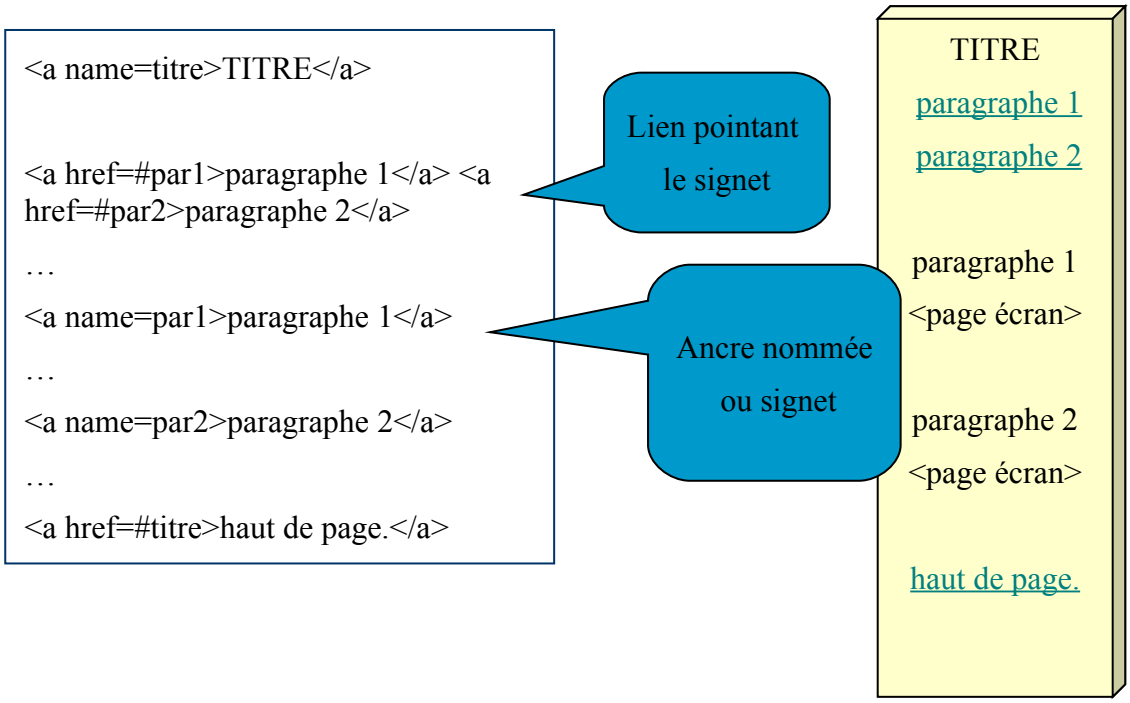
**Exemple :**

```
<p><a href="#sport">Voir le sport</a></p>
•
•  PAGE ECRAN
•
<p> <a name="sport"></a>La rubrique sport</p>
```

Il n'est pas nécessaire dans ce cas de mettre un texte ou une image, entre les balises d'ouverture et de fermeture (Le nom ne doit pas contenir d'espaces, ni d'accents, ni de caractères spéciaux).

### Exemple d'application le plus couramment utilisé :

Une page Web peut être structurée en plusieurs pages écran correspondant chacune à un paragraphe. Il est possible de définir alors en début de page un sommaire permettant d'atteindre plus aisément les différents paragraphes constitutifs de cette page.




L'exemple ci-dessus définit de plus un lien interne permettant de revenir en début de page.

Quand une page web est longue, il convient de placer en fin de page, ou même après chaque fin de paragraphe, un lien vers le haut de la page (son titre interne). Ceci permet à l'internaute de naviguer de manière autonome dans la page, sans utiliser l'ascenseur vertical.

# 3. Liens externes

Ce sont des liens visant un autre document Web situé :

1. soit dans le même dossier que la page où est défini ce lien,
2. soit à un autre endroit mais toujours sur le même serveur Web,
3. soit sur un autre serveur Web localisé n'importe où dans le monde.

## 3.1 Document cible situé dans le même dossier

```
<p><a href="sport.htm">Voir le sport</a></p>
```

## 3.2 Document cible situé dans un autre dossier mais sur le même serveur (même site Web)

```
<p><a href="sports/sport.htm">Voir le sport</a></p>
```

Dans le cas ci-dessus, la page Web se situe physiquement dans un dossier ou répertoire d'un niveau inférieur.

```
<p><a href="../sports/sport.htm">Voir le sport</a></p>
```

Dans ce cas, la page Web se situe physiquement dans un répertoire du même niveau que celui de la page appelante.

**Rappel : chemin type UNIX**

Cette manière de décrire le chemin pour atteindre la page Web désirée est celle utilisée par le système d'exploitation UNIX : séparation des différents niveaux de l'arborescence des fichiers par le caractère « / », représentation du niveau supérieur par les caractères « .. ».

**Remarque : portabilité**

Afin de rendre la page Web ainsi élaborée portable d'un serveur Web à un autre, il est vivement conseillé de ne pas utiliser au niveau des liens des chemins absolus (ou complets) faisant référence soit aux lecteurs logiques (périphériques ou disques) du serveur hébergeur du site, soit aux adresses URL du site.

Le fait de ne pas utiliser des chemins relatifs oblige le créateur du site à redéfinir tous les liens du site lors d'un transfert de celui-ci d'un serveur hébergeur à un autre.

Voici ci-après un exemple de lien à chemin absolu indiquant que le document Web se trouve sous le lecteur logique "C:". La syntaxe emploie le mot prédéfini file, suivi du caractère ":". Viennent ensuite trois "slash" au lieu de deux, le nom du lecteur, suivi du caractère ":" ou de la barre verticale (combinaison de touches "alt gr" + "6").

```
<p><a href="file:///C|/sports/sport.htm">Voir le sport</a></p>
```

Le concepteur de site, peut choisir au contraire, d'utiliser ces chemins absolus pour protéger son site, notamment des aspirateurs de site.

Les chemins relatifs décrivent le parcours à réaliser depuis la page appelante vers la page appelée.

**Notes :**

Si la cible du lien externe vise un fichier .EXE ou .ZIP, ceci revient à proposer un téléchargement de ces fichiers.

```
<A HREF="util.exe">Cliquez ici pour télécharger le fichier</A>
```

Enfin le chargement d'un fichier son ou vidéo, compressé ou non, peut être fait au moyen de cette même balise.

## 3.3 Document cible situé sur un autre site Web (autre adresse URL)

```
<p><a href="http://www.monsite.com/sports/sport.htm">Voir le sport</a></p>
```

Dans ce cas, la valeur de l'attribut « href » est initialisée avec l'adresse URL du site situé sur l'autre serveur Web.

# 4. Liens mixtes

Le lien mixte réunit les notions de liens internes et externes. C'est donc un lien vers un endroit d'un autre document.

```
<p> <a href="sport.htm#football">voir les résultats de la D1</a></p>
```

Dans ce cas, la valeur de l'attribut "href" contient aussi bien le chemin relatif nécessaire pour atteindre la page visée, que l'ancre nommée définie au préalable dans cette autre page. Les deux valeurs sont accolées.

# 5. Liens vers d'autres services

Rappelons qu'une adresse URL est toujours de la forme :

ressource ://hôte.domaine :port/chemin

où la ressource peut prendre les valeurs : file, http, ftp, news, ldap, gopher, telnet, wais ou mailto.

Nous allons donner deux exemples de liens vers les services les plus couramment utilisés du réseau Internet : ftp, mailto.

La ressource peut être une ressource locale (**file**) ou distante.

Dans le cas d'une ressource locale sous environnement Windows, un lien à un document situé sur un lecteur de disque défini par l'unité "C:" se fera par l'URL :

```
<A HREF="file:///C|fichier.htm"> fichier sur c:</A>
```

Le symbole ":" de l'adressage URL est remplacé par le symbole "|" et les symboles "//" sont remplacés par les symboles "///" pour éviter la confusion avec un nom de serveur.

Dans le cas d'une ressource distante, elle désigne le type de service visé (ou serveur) et peut prendre la valeur du protocole associé au service : http, ftp, ldap.

L'élément « hôte.domaine » représente le nom complet du serveur visé (FQDN) ou son adresse IP.

Les éléments port (numéro de port) et chemin (pathname) sont optionnels.

## 5.1 Serveur de transfert de fichiers

```
<p> <a href= "ftp://ftp.astronomie.fr ">Calculs de Trajectoire</a></p>
```

Un deuxième exemple permet à un utilisateur d'accéder à un site protégé par le couple d'identifiants (nom d'utilisateur ou login - mot de passe). Dans l'exemple, le login est toto et le mot de passe titi.

```
<A HREF="ftp://toto:titi@ftp.multimania.com/monfichier.ext"> Telecharger mon fichier
</A>
```

## 5.2 Serveur de courrier

L'argument « mailto :adresse E-Mail » permet d'envoyer un courrier électronique à l'adresse correspondante.

```
<p><a href= "mailto:dupont@aol.fr">Envoyer un message</a></p>
```

L'exemple ci-dessus permet d'envoyer un courrier électronique à Monsieur DUPONT en cliquant sur le lien « Envoyer un message ».

Il est possible de rajouter le sujet du message :

```
<p><a href="mailto:dupont@aol.fr?subject=Remarques sur le site">Envoyer un
message</a></p>
```

Il faut remarquer qu'un point d'interrogation sépare l'adresse en elle-même du paramètre *subject*. La valeur de l'attribut *href* comprend l'ensemble : adresse et couple(s) "paramètre=valeur".

Voici d'autres exemples :

- Lien avec sujet et texte dans le corps du message.

```
<A HREF="mailto:nom@provider.com?subject=renseignement&body=Votre message">
nom@provider.com</A>
```

Les différents couple(s) "paramètre=valeur" sont séparés du caractère esperluette ou "&".

- Lien avec sujet envoyé à deux destinataires (une en copie "carbone" ou CC).

```
<A HREF=mailto:nom@provider.com?cc=nom2@provider2.com&subject=renseignement">

nom@provider.com</A>
```

- Lien avec deux personnes destinataires (dont une en copie "carbone" invisible ou CCI ou BCC).

```
<A href=mailto:nom@provider.com?bcc=nom2@provider2.com&subject=renseignement">
nom@provider.com</A>
```

## 5.3 Autres cas

Voici un tableau d'exemples de liens visant d'autres services.

| **Liens** | **Exemples** |
|---|---|
| Lien vers un serveur de news | <A HREF="news:news.multimania.com"> |
| Lien vers un serveur gopher | <A HREF="gopher://gopher.site.com"> |
| Lien vers un serveur telnet | <A HREF="telnet://bbs.gateway.com"> |
| Lien vers un serveur wais | <A HREF="wais://wais.base.com"> |

# 6. Importance des liens

Il est utile de rappeler que la magie des liens permettant à l'internaute de surfer à travers le monde entier a contribué fortement au succès du Web.

En dehors des liens externes au serveur Web qui héberge le site en question, les liens permettent d'installer des éléments de navigation entre les différentes pages Web (ou documents) d'un même site :

- Retour à la page d'accueil
- Retour à la page précédente du chapitre
- Aller à la page suivante
- Haut de page dans un document

Ces éléments de navigation permettent de **jauger** un site sur son **côté fonctionnel**.

La mise en place de liens dans un site permet de définir l'organisation ou l'**architecture logique** du site. Cela constitue une des premières tâches du concepteur après que la maquette du site soit réalisée.

Les grands principes de cette organisation pourront être couchés au sein de la charte graphique du site décrivant ce dernier dans les moindres détails.

# Chapitre 8 : Tableaux

Les tableaux sont très utilisés lors de la conception de documents HTML.

Au départ, le concepteur s'en sert pour simplement transposer les tableaux de propriétés ou de résultats des logiciels de traitements de texte ou tableurs.

Une fois que le concepteur de pages a acquis quelque expérience, notamment en regardant le code source des pages, il constate que les documents HTML ne sont qu'une longue suite de tableau(x). Ceux-ci ne sont plus visibles dans le navigateur car ils ne présentent plus de bordures. Ils sont justifiés pour assurer une solide mise en page digne d'un magazine.

Leur étude montrera que les attributs de mise en forme sont nombreux et donc que les combinaisons sont très diverses.

## 1. Tableau de base

### Description

Une tableau minimal (tableau à une ligne et une colonne, donc à une cellule), présentant une bordure large d'un pixel, est défini comme suit :

```
<table border="1">
  <tr>
    <td>Item</td>
  </tr>
</table>
```


L'élément **table** (norme HTML3.2) admet obligatoirement une balise de fermeture. Il constitue un bloc d'instructions à l'intérieur duquel on définit le nombre de lignes, puis le nombre de cellules par ligne (ce qui revient au nombre de colonnes).

La ligne (rangée) est définie par l'élément **tr** (table row, norme HTML3.2). Nous trouverons donc autant d'éléments **tr** qu'il y aura de lignes. L'expression insérée entre ces balises peut être aussi complexe que désirée (cela peut être un autre tableau, etc…).

Dans chaque ligne du tableau, nous pouvons insérer un nombre quelconque de cellules (ou colonnes). Chaque cellule sera définie par l'élément **td** (table data, norme HTML3.2), la balise de fermeture étant optionnelle.

### Attributs

| <table | | | |
|---|---|---|---|
| align | = | left *ou* right | alignement horizontal du tableau par rapport à la page, l'alignement par défaut est à gauche |
| bgcolor | = | triplet (R,G,B) *ou* nom de couleur | couleur utilisée pour peindre le fond du tableau. |
| border | = | nombre | bordure, exprimée en pixels, dessinée autour du tableau. |
| cellpadding | = | nombre | espace réservé entre les bords d'une cellule et son contenu (1 pixel par défaut). |
| cellspacing | = | nombre | espacé réservé entre chaque cellule (2 pixels par défaut). |
| height | = | nombre *ou* pourcentage | hauteur du tableau exprimée soit de manière absolue en nombre de pixels ou de manière relative en pourcentage par rapport à la hauteur de la page. |
| width | = | nombre *ou* pourcentage | largeur du tableau exprimée avec les même unités que la hauteur. |
| > | | | |

**Remarques** :

- Il n'existe pas de valeurs **center** pour l'attribut **align** avec la version HTML3.2. Le tableau peut malgré tout être centré sur la page en l'insérant dans l'élément **center** ou à l'intérieur d'une division centrée : ** TABLEAU… **.
- Le tableau est redimensionné pour correspondre aux valeurs des attributs **height** et **width** si elles sont présentes.

**Exemple :** tableau à une ligne et trois colonnes, fond bleu, bordure large de 2 pixels et espace de 10 pixels entre le bord des cellules et leur contenu.

```
<table bgcolor="lightblue" border="2" cellpadding="10">
<tr>
  <td>Item 1</td>
  <td>Item 2</td>
  <td>Item 3</td>
</tr>
</table>
```

### Résultat

| Item 1 | Item 2 | Item 3 |
|---|---|---|

**Remarque** :

- Netscape ne peint que l'intérieur des cellules alors que Microsoft Explorer peint tout le tableau (cadre compris).

Comme il existe des différences de comportements entre les navigateurs pour le rendu des attributs communs, il existe également des différences au niveau des attributs acceptés par l'un ou par l'autre. Ce sont les attributs propriétaires. Voyons quels sont les attributs supplémentaires d'Internet Explorer :

| | | | |
|---|---|---|---|
| background | = | nom de fichier graphique | image utilisée pour remplir la surface du tableau (affiché sous le texte et les éventuelles images). |
| bordercolor | = | triplet (R,G,B) *ou* nom de couleur | couleur utilisée pour la bordure (à utiliser avec "border"). |
| bordercolordark | = | triplet (R,G,B) *ou* nom de couleur | couleur utilisée pour dessiner les bords sombres de la bordure vue en 3D (à utiliser avec "border"). |
| bordercolorlight | = | triplet (R,G,B) *ou* nom de couleur | couleur utilisée pour dessiner les bords clairs de la bordure vue en 3D (à utiliser avec "border"). |

Ces attributs propriétaires ont depuis été normalisés par la version HTML 4.0.

# 2. Balises de définition

## 2.1 Définition d'une ligne

L'élément HTML **tr** encadre les données formant une ligne du tableau. Elle accepte les attributs :

| | | | |
|---|---|---|---|
| align | = | left \| right \| center | Alignement du texte inséré dans les cellules. |
| bgcolor | = | couleur | Couleur d'arrière-plan des cellules d'une ligne. Valeurs : triplet Rouge, Vert, Bleu ou un nom de couleur prédéfini. |
| valign | = | mot prédéfini | Définit l'alignement vertical du texte des cellules avec l'une des valeurs suivantes :<br>• "baseline" alignement sur la ligne de base<br>• "bottom" alignement sur l'élément le plus bas<br>• "middle" positionnement à mi-hauteur<br>• "top" alignement sur la ligne supérieure |

MS Internet Explorer autorise en plus l'utilisation des attributs **background**, **bordercolor**, **bordercolordark** et **bordercolorlight** déjà mentionnés.

**Remarques :**

- les valeurs fournies au niveau de la ligne redéfinissent, le cas échéant, la valeur définie au préalable au niveau de l'élément ***table***.
- les éléments de texte ou autre insérés dans une ligne subissent la mise en forme décidée au niveau de cette ligne à moins qu'ils ne soient redéfinis au niveau de chaque cellule.

## 2.2 Définition d'une cellule

L'élément HTML **td** encadre une donnée dans une ligne du tableau. Il accepte les attributs :

| | | | |
|---|---|---|---|
| align | = | left \| right \| center | Alignement du texte inséré dans les cellules. |
| bgcolor | = | couleur | Couleur d'arrière-plan des cellules d'une ligne. Valeurs : triplet Rouge, Vert, Bleu ou un nom de couleur prédéfini. |
| valign | = | mot prédéfini | Définit l'alignement vertical avec l'une des valeurs suivantes :<br>• "baseline" alignement sur la ligne de base<br>• "bottom" alignement sur l'élément le plus bas<br>• "middle" positionnement à mi-hauteur<br>• "top" alignement sur la ligne supérieure |
| colspan | = | nombre | Définit le nombre de colonnes sur lesquelles s'exerce la fusion des cellules. |
| height | = | nombre | Définit la hauteur de la cellule en nombre de pixels (taille absolue) ou en pourcentage (taille relative). |
| nowrap | = | | Interdit le retour à la ligne automatique pour les longues séquences de mots. |
| rowspan | = | nombre | Définit le nombre de lignes sur lesquelles s'exerce la fusion des cellules. |
| width | = | nombre | Définit la hauteur de la cellule en nombre de pixels (taille absolue) ou en pourcentage (taille relative). |

MS Internet Explorer autorise en plus l'utilisation des attributs : *background*, *bordercolor*, *bordercolordark* et *bordercolorlight* qui ont la même signification et le même type de valeurs que pour l'élément ***table*** ou ***td***.

**Remarque** : les valeurs fournies redéfinissent, le cas échéant, la valeur définie à un niveau supérieur avec l'élément **tr** ou l'élément **table**.

## 2.3 Définition d'une donnée d'en-tête

### Description

L'élément **th** (table heading, norme HTML3.2) s'utilise exactement comme l'élément **td**. Il présente les même attributs et gère les données d'une cellule.

Il se différencie de l'élément **td** par le fait que le texte est centré et rehaussé généralement par un style gras.

Il est destiné aux premières lignes d'un tableau pour donner un autre style aux entêtes de colonnes. Il peut servir aux dernières lignes d'un tableau en guise de résultats ou totaux.

### Exemple

```
<TABLE BORDER="4">
<TR><TH COLSPAN="2">Titre1/TH><TH COLSPAN="2">Titre2</TH></TR>
<TR>
<TD>Élément 1</TD><TD>Élément 2</TD>
<TD>Élément 5</TD><TD>Élément 6</TD>
</TR>
<TR>
<TD>Élément 3</TD><TD>Élément 4</TD>
<TD>Élément 7</TD><TD>Élément 8</TD>
</TR>
</TABLE>
```

### Résultat

| Titre1 | | Titre2 | |
|---|---|---|---|
| Élément 1 | Élément 2 | Élément 5 | Élément 6 |
| Élément 3 | Élément 4 | Élément 7 | Élément 8 |

Les éléments TH peuvent aussi être utilisées comme tête de ligne (ou première colonne) et vice-versa.

Les éditeurs HTML graphiques ne le proposent pas.

## 2.4 Définition d'un titre

L'élément **caption** (norme HTML3.2) permet de donner un titre au tableau.

L'élément CAPTION doit venir immédiatement après la balise de début de l'élément TABLE.

| | | | |
|---|---|---|---|
| align | = | bottom \| left \| right \| top | Alignement du texte inséré dans les cellules. |

**Exemple**

```
<TABLE cols="3">
<CAPTION>Tasses de café consommées par chaque parlementaire</CAPTION>
...le reste de la table...
</TABLE>
```

Le navigateur affiche le titre la ligne précédant le tableau.

# 3. Fusionnements de cellules

Sans les attributs **colspan** et **rowspan**, toutes les lignes d'un tableau devraient définir le même nombre de colonnes - éventuellement vides - et certaines présentations seraient impossibles à obtenir.

Supposons que nous voulions créer le tableau suivant :

| Statistiques | | |
|---|---|---|
| Temperatures | Nord | Sud |
| | 17 | 23 |

Ecrivons le code suivant.

```
<table border="1">
<tr><td>Statistiques</td></tr>
<tr><td>Temperatures</td><td>Nord</td><td>Sud</td></tr>
<tr><td>17</td><td>23</td></tr>
</table>
```

Nous obtenons :

| Statistiques | | |
|---|---|---|
| Temperatures | Nord | Sud |
| 17 | 23 | |

Ceci n'est pas tout à fait ce que nous souhaitons. En effet, le titre "Statistiques" doit s'étendre sur les trois colonnes du tableau. De même, le mot "Températures" doit être centré verticalement sur les lignes 2 et 3. Corrigeons notre code.

```
<table border="1">
<tr>
        <td align="center" colspan="3">Statistiques</td>
</tr>
<tr align="center">
        <td valign="middle" rowspan="2">Temperatures</td><td>Nord</td><td>Sud</td>
</tr>
```

```
<tr align="center">
        <td width="80">17</td><td width="80">23</td>
</tr>
</table>
```

**Explications :**

- Sur la première ligne, une colonne équivaut à 3 colonnes du tableau (**colspan**=3).
- Toutes les données de la seconde ligne sont centrées car l'attribut ***align=center*** est défini au niveau de l'élément **tr**. Cette ligne contient trois données : la première est centrée verticalement (***valign=middle***) et s'étend sur deux lignes (**rowspan**=2). Elle occupera donc également la première colonne de la ligne suivante.
- La première donnée, qui s'inscrit en colonne 2, fixe la largeur de cette cellule à 80 pixels. Ce sera vrai aussi pour toute la colonne. Il en est de même pour la troisième donnée. Cette définition de largeur aurait pu être faite sur la 2ième ligne pour obtenir le même résultat.

Le résultat est alors bien meilleur.

| Statistiques | | |
|---|---|---|
| Températures | Nord | Sud |
| | 17 | 23 |

# Chapitre 9 : Cadres

Les tableaux présentent des données de façon structurée. Les cadres fonctionnent de façon tout à fait différente : il s'agit de subdiviser la fenêtre principale du navigateur en fenêtres secondaires, chaque fenêtre affichant un document différent et indépendant.

Si l'auteur de la page le désire, les utilisateurs peuvent naviguer dans les parties subdivisées de cette fenêtre principale : les cadres (frames).

Ces possibilités sont reconnues par les navigateurs Netscape 2.0 et Internet Explorer 3.0.

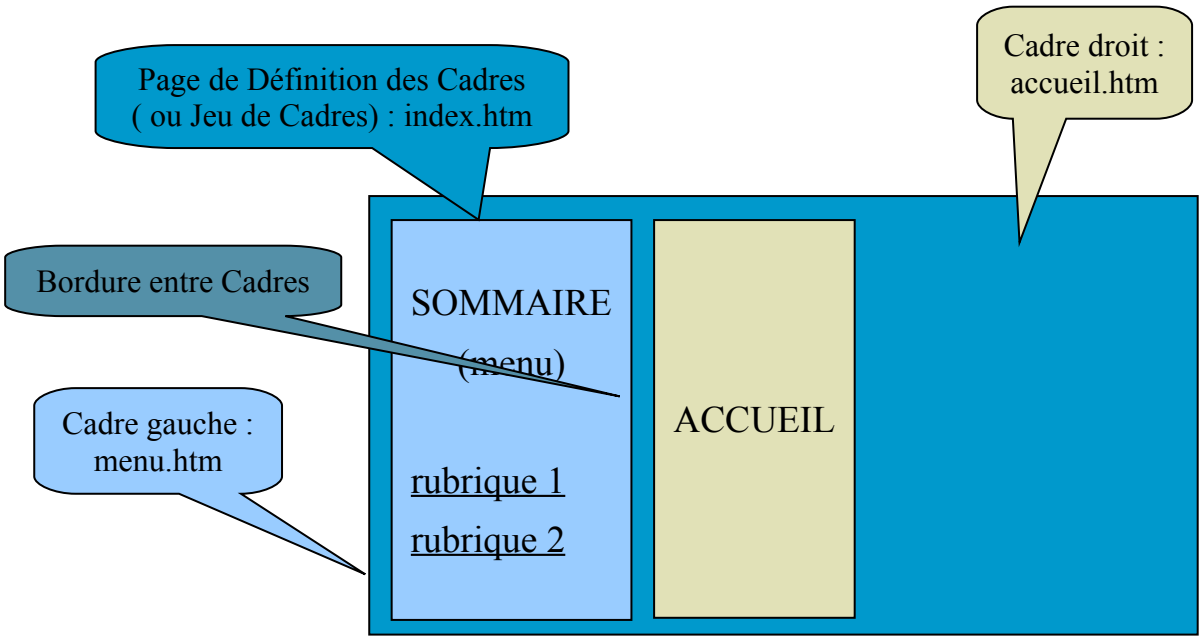
## 1. Structure générale

La fenêtre principale affiche en premier et en arrière-plan (non visible à l'écran) le document de **définition des cadres** (ou **jeu de cadres**).

Un premier élément **frameset** (ou ensemble de cadres, norme HTML4.0) définit la taille des parties issues de la division. Il débute après l'en-tête (balise </head>), et finit avec la fin de fichier (balise </html>). La première division peut-être horizontale (lignes) ou verticale (colonnes).

Cet élément constitue un bloc qui contient les définitions de chacun des cadres (balise **<frame>,** norme HTML4.0) ou parties subdivisées. Chaque cadre définit le contenu des fenêtres.

Le corps de ce document n'a pas lieu d'être présent dans la mesure où la fenêtre principale est recouverte par les fenêtres secondaires de chacun des cadres. Le concepteur peut insérer un corps de texte (balise body) au sein du bloc ***frameset*** à destination des anciens navigateurs qui n'interprètent pas les cadres. Cet élément body se trouve en fait inséré dans l'élément **noframes**. Cet élément est optionnel.



En pratique, le concepteur crée le document définissant les cadres séparément des documents appelés à s'afficher dans chaque division de la fenêtre principale.

La première application de la définition des cadres, illustrée ci-dessous, vise à laisser de manière apparente et permanente deux cadres :

- Le premier, un menu ou sommaire du site (partie latérale gauche en général) ou encore une barre d'outils (sur la partie supérieure délimitée de façon horizontale), etc....
- Le deuxième, la page d'accueil du site

Dans le schéma, volontairement et pédagogiquement, apparaît le fichier de définition des cadres dans la fenêtre principale du navigateur, normalement invisible à l'écran.

Chaque cadre peut, à l'image d'un tableau ou d'une image, être encadré par une bordure.

Dans chacun des cadres s'affiche une page web nommément désignée par la balise **frame** dans le fichier de définition.

**Remarques** :

Au niveau du navigateur, le fichier source de la page de définition des cadres est accessible par le menu, alors que les autres codes sources le sont uniquement par le menu contextuel (clic droit) à l'intérieur du cadre sélectionné.

En conséquence, seul le titre de la page de définition (balise **<title>**) apparaît dans la barre du navigateur (et non ceux des pages affichées dans les cadres), d'où l'importance de cet élément HTML à cet endroit là du site.

Voici un exemple de code source associé au document de définition des cadres :

| | |
|---|---|
| <**frameset cols**="50%,50%"> | divise la page en deux colonnes égales |
| <**frameset rows**="50%,50%"> | divise la colonne 1 en 2 rangées égales |
| <**frame src**="cellule1.html" name="cel1"> | définition de la ligne 1 de la colonne 1 |
| <**frame src**="cellule2.html" name="cel2"> | définition de la ligne 2 de la colonne 1 |
| **</frameset>** | |
| | divise la colonne 2 en 3 rangées égales |
| <**frameset rows**="33%,33%,33%"> | |
| <frame src="cellule3.html" name="cel3"> | définition de la ligne 1 de la colonne 2 |
| <frame src="cellule4.html" name="cel4"> | définition de la ligne 2 de la colonne 2 |
| <frame src="cellule5.html" name="cel5"> | définition de la ligne 3 de la colonne 2 |
| **</frameset>** | |
| <**noframes**> | début de section 'sans cadres' |
| <body> | |
| <h1 align="center">Cadres</h1> | |
| Ce document requiert des cadres. Si vous voyez ce message, votre navigateur ne les reconna&iuml ;t pas. | |
| </body> | |
| **</noframes>** | fin de section 'sans cadres' |
| **</frameset>** | fin de définition des colonnes |

L'exemple ci-dessus découpe la fenêtre principale en cinq cadres. Elle définit d'abord les colonnes, puis les rangées à l'intérieur de chaque colonne.

L'élément **frameset** est réutilisé à chaque découpage.

Une fois le découpage réalisé, le concepteur définit les caractéristiques de chaque cadre.

La balise **frame** contient un attribut obligatoire : l'attribut **src** désignant le document à afficher dans ce cadre.

Le deuxième attribut rencontré est **name**. Le nom des cadres est un attribut qui peut s'apparenter aux noms des objets en programmation objet. Il est impératif ici de donner un nom au(x) cadre(s) où s'afficheront les cibles des liens (<a href=… **target**="name">).

Revenons à l'exemple imagé du paragraphe. Le concepteur définissant son sommaire, pose des liens hypertextes sur les différentes rubriques de son site. La cible de chaque lien doit s'afficher à la place de la page d'accueil. Pour réaliser cela, le concepteur, après avoir donné un nom au cadre de la colonne de droite (**name**="droit" par exemple), doit reprendre la même valeur qu'il doit assigner à l'attribut **target**.

**Problème de compatibilité**

L'élément **noframes** introduit un contenu alternatif pour les navigateurs ne supportant pas les cadres. On peut y inviter l'utilisateur à mettre à jour son navigateur.

# 2. Jeux de cadres

## 2.1 Syntaxe

Comme le montrait le premier exemple de code source, l'élément ***frameset*** doit être utilisé pour redécouper un cadre donné en plusieurs cadres enfants.

### Attributs

| <frameset | | | |
|---|---|---|---|
| border | = | nombre | Largeur des bordures entre cadres exprimée en pixels. La valeur "0" signifie aucune bordure, la valeur par défaut est 5. |
| bordercolor | = | couleur | Utilisée pour dessiner la bordure du cadre. |
| cols | = | "n[%],m[%],*" | Définition des colonnes : la largeur de chacune d'elles est définie en pixels ou en pourcentage. |
| frameborder | = | 0 *ou* 1 | Indique si la bordure est dessinée avec un effet de relief en 3D ("1") ou en aplat ("0"). Les valeurs "yes" et "no" sont également autorisés. |
| framespacing | = | nombre | Définit un espace supplémentaire entre les cadres (en pixels). |

| | | | |
|---|---|---|---|
| rows | = | "n[%],m[%],*" | Définition des lignes, voir cols. |
| > | | | |

Toutes les attributs autres que rows ou cols sont facultatifs.

La balise de fermeture est obligatoire.

## 2.2 Découpage et taille des cadres

Un des attributs **cols** ou **rows** doit obligatoirement être présent. Les deux peuvent être simultanément employés au sein du même élément **frameset**.

La valeur absolue en pixels est à éviter, à moins de vouloir mettre un bandeau d'images fixes par exemple.

La valeur en pourcentage permet de diviser l'écran en plusieurs parties par exemple :
*<FRAMESET ROWS="10%,70%,20%">*

La valeur résultante (signe *) permet de désigner le reste de la place disponible.

- Cette valeur résultante donne la valeur à déduire. Par exemple *<FRAMESET ROWS="100,*,*">* divisera en trois lignes : 100 pixels pour une fenêtre, et deux fenêtres de hauteur identique.
- On peut entrer la syntaxe <FRAMESET ROWS="*,2*"> pour diviser l'écran en deux parties; la première faisant 1/3 de l'écran et la deuxième les 2/3 restants.

## 2.3 Bords de cadres

Orner une page Web de cadres n'est pas toujours agréable à l'oeil, sachant que l'écran se trouve divisé, limitant de ce fait de façon agressive la partie principale de la page.

Il est possible de ne plus diviser l'écran de façon physique mais uniquement de façon logique en supprimant les traits de séparation des cadres.

L'attribut **frameborder** peut être inséré dans les balises **frameset** et **frame**. La valeur par défaut est **YES**. Cela veut dire que, sans option, les bordures sont visibles avec un effet 3D.

Dans un élément **frameset**, l'attribut **frameborder=NO** ne fait que supprimer l'effet 3D et non la bordure. L'attribut **border** de l'élément **frameset** fixe l'épaisseur de la bordure. Une épaisseur à 0 supprime la bordure.

L'absence de bordures aux cadres est des plus esthétiques car il cache la présence de cadres aux internautes. Il lui donne l'impression de visualiser une seule page alors que plusieurs sont affichées par le navigateur.

Comme les bordures des cadres sont partagées, les définitions du niveau d'un cadre prennent le pas sur les définitions du niveau de l'élément **frameset**.

# 3. Propriétés d'un cadre

La balise **frame** supporte les attributs suivants :

| <frame | | | |
|---|---|---|---|
| align | = | left, center, right,top *ou* bottom | Définit l'alignement des objets, notamment du texte inclus dans la page affichée (Microsoft Explorer uniquement). |
| bordercolor | = | couleur | Utilisée pour dessiner le cadre. |
| frameborder | = | 0 *ou* 1 | Indique si une bordure est tracée en 3D ou en aplat. Redéfinit la valeur fournie avec l'élément frameset. |
| marginheight | = | nombre | Hauteur, en pixels, de la marge entre les bordures supérieure et inférieure du et son contenu. |
| marginwidth | = | nombre | Largeur, en pixels, de la marge entre les bordures droite et gauche du cadre et son contenu. |
| name | = | texte | Nom du cadre utilisé notamment par l'attribut target de la balise <a href … > (lien) ou des scripts. |
| noresize | | | Indique que les dimensions du cadre sont fixes. Si cette option est absente, l'utilisateur peut redimensionner les cadres en déplaçant les bordures avec la souris. Le passage de la souris sur une bordure transforme le curseur en double flèche horizontale. |
| scrolling | = | yes / no / auto | Indique si une barre de défilement sera affichée. La valeur auto affiche une barre uniquement si besoin est. |
| src | = | URL | Nom du document HTML à afficher dans ce cadre. |
| > | | | |

**Remarque**

Les attributs de bordures de la balise **frame** prennent le pas sur ceux de l'élément **frameset**.

# 4. Utilisation des cadres

Voici deux exemples de mise en œuvre des cadres.

## 4.1 Table des matières, titre et logo

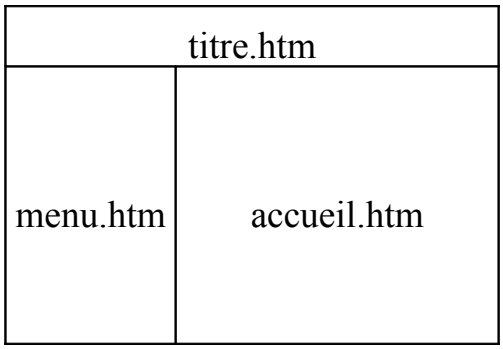
Le premier exemple simple d'utilisation des cadres est l'affichage d'une table des matières ou plan du site : un cadre étroit, généralement placé à gauche de la page, permet d'accéder aux différentes pages du site classées par rubrique. Une première variante consiste à créer un troisième cadre contenant le titre du site et son logo d'identification (partie supérieure de la fenêtre sous la forme d'une bannière par exemple).

Pour une telle présentation, la page principale a le choix entre :

- définir d'abord deux cadres verticaux (menu à gauche), puis découper celui de droite en deux cadres horizontaux (titre et logo en haut, accueil en bas)
- définir d'abord deux cadres horizontaux (titre et logo en haut), puis découper le cadre bas en deux cadres verticaux (menu à gauche, accueil en bas),

Le deuxième choix donne le code suivant pour la **page principale** :

```
<html>
<head>
<title>Page principale</title>
</head>
<frameset rows="64,*">
  <frame name="banniere" scrolling="no" noresize src="titre.htm">
  <frameset cols="158,*">
    <frame name="menu" src="menu.htm">
    <frame name="principal" src="accueil.htm" scrolling="auto">
  </frameset>
</frameset>
</html>
```




## 5.2 Fenêtres dynamiques

Si nous reprenons l'exemple précédent, nous avons défini dans le fichier “menu.htm” une liste de rubriques sur lesquelles sont posées des liens.

Pour chaque lien, il faut préciser dans quel cadre la page visée doit s'afficher.

Au niveau du lien, on utilise l’attribut **target** de la balise associée au lien.

La valeur de l'attribut target reprend la valeur de l'attribut name donné au cadre dans le code ci-dessus.

La page **menu** pourrait contenir :

```
<html>
<head>

<!-- ce titre n’est pas affiché -->
<title>menu</title>
</head>
<body>
<a href="intro.htm"   target="principal">Introduction</a><br>
```

```
<a href="page1.htm" target="principal">Page 1</a><br>
<a href="page2.htm" target="principal">Page 2</a><br>

<!-- pour une référence à un site externe, préciser "_blank" pour charger
ce site dans une nouvelle fenêtre (ou fenêtre indépendante) et non dans un des cadres -->
<a href="http://www.monSite.com/" target="_blank">monSite</a>
</body>
</html>
```

Il reste à créer les pages référencées (fichiers “page1.htm”, “page2.htm”, etc.) pour avoir un site opérationnel.

En dehors des valeurs librement choisies par le concepteur pour désigner chacun des cadres de la fenêtre principale, d'autres valeurs sont possibles pour l'attribut **target** :

| | | | |
|---|---|---|---|
| target | = | _blank | Affiche la page dans une nouvelle instance de navigateur. |
| | = | _top | Affiche la page dans la fenêtre principale du navigateur. |
| | = | _self | Affiche la page là où elle a été appelée. |
| | = | _parent | Affiche la page dans la fenêtre principale du jeu de cadres "parent". Cette possibilité ne peut s'appliquer que dans le cas où le concepteur définit des séries de cadres imbriqués, un cadre appelant une page de jeu de cadres. |

## 5.3 Des cadres emboîtés

Nous l’avons vu plus haut, il est possible d’emboîter des cadres.

Pour cela, il suffit de remplacer un élément de cadre par un deuxième élément **frameset**. Ceci est justifié si l'on veut alterner les divisions (horizontales puis verticales, ou vice versa).

Sinon, il suffit de déclarer une colonne ou rangée de plus dans la première définition.

Par exemple, pour avoir une division de la page en quatre carrés de même taille, le concepteur utilisera dans le premier procédé le code source suivant :

```
<frameset cols="50%, *">                          division verticale

                                                  définition de la première colonne, sous-division
    <frameset rows="50%, *">                      horizontale

        <frame src="source1.htm">                     définition du cadre 1 (col.1)
        <frame src="source2.htm">                     définition du cadre 2 (col.1)

    </frameset>

                                                  définition de la seconde colonne, sous-division
    <frameset rows="50%, *">                      horizontale

        <frame src="source3.htm">                 définition du cadre 1 (col.2)
        <frame src="source4.htm">                 définition du cadre 2 (col.2)

    </frameset>

</frameset>
```

En ce qui concerne le deuxième procédé, voici un exemple de code source relatif au même exemple.

```
<frameset rows="50%,*" cols="50%,*">
        <frame src="ligne1colonne1.htm">
        <frame src="ligne1colonne2.htm">
        <frame src="ligne2colonne1.htm">
        <frame src="ligne2colonne2.htm">
</frameset>
```

# Complément 1 : Animation Multimédia

## 1. Présentation des effets multimédia

### 1.1 Données multimédias

Si on s'en tient à une définition académique, le multimédia, c'est le mélange d'au moins deux types de médias (texte, image, son...) dans un même endroit. Sur Internet, on parle plutôt de multimédia dès qu'un site utilise du son, de la vidéo ou des éléments d'animation interactive.

Il existe trois types de données multimédias :

- **sonores** (MP3, Wav, MIDI...) qui ne contiennent que du son, compressé ou non ;
- **vidéo** (Real, MPEG, AVI...) qui contiennent à la fois de l'image et du son, généralement compressés ;
- **interactives** (animations Flash ou Shockwave), contiennent souvent de l'image et du son, mais permettent aussi à l'utilisateur d'agir directement sur les animations, en général des petits jeux ou des éléments d'interface bien spécifiques.

Dans l'histoire des pages web, une des premières solutions rencontrées est d'insérer des petites séquences animées composées de quelques images par seconde. Il s'agit des **images animées** présentées dans la première partie.

Mais ces effets ne sont principalement utilisés que pour insérer des puces graphiques animées ou des images de divertissement, ou des logos publicitaires. Actuellement, la mode n'est plus aux images animées. Les puces ou images de divertissement disparaissent.

Une autre solution consiste à déclencher des **vidéos**. Un inconvénient majeur réside dans le fait que cette technique monopolise la bande passante.

En dehors des vidéos au format standard, il existe plusieurs logiciels permettant de développer des animations vectorielles au format propriétaire tels que les animations Flash, ou des animations 3D au format **VRML** (introduction dans la dernière partie du chapitre).

Enfin, le langage HTML, outre les films, permet la lecture d'une **bande son**. Là aussi, des restrictions à l'emploi existent :

- Un fichier est aussi gourmand en bande passante s'il est appelé dans son format d'origine (*.wav par exemple). Cette technique est viable dans le cas de seuls fichiers au format MP3.
- Dérouler une bande son de façon répétitive peut agacer l'internaute. Aussi, est-il préconisé de dérouler la bande une seule fois ! Une musique d'introduction peut être plus volontiers appréciée.

### 1.2 La bande passante

L'inconvénient de ces animations est la place et surtout la bande passante qu'elles occupent. Si l'on n'y prend pas garde, le magnifique site risque d'être impossible à consulter : peu de

visiteurs sont en effet prêts à patienter un quart d'heure le chargement d'une page ! Voilà pourquoi il est capital d'optimiser et de réduire la taille des fichiers.

En conséquence, les fichiers audio et vidéo nécessitent une sévère compression. Il est conseillé de limiter le débit des musiques à 20 Kbps, d'interdire les vidéos de plusieurs dizaines de mégaoctets.

## 1.3 Les plug-ins

La présence de données multimédias sur le site pose un autre problème : est-ce que tous les visiteurs pourront lire les pages ? La lecture d'une vidéo ou d'autres éléments multimédias nécessite souvent la présence d'un plug-in, un petit programme qui permet à votre navigateur de lire un plus grand nombre de fichiers. En l'absence de ce plug-in, les messages d'erreur ne se feront pas attendre.

Avant d'intégrer des données multimédias dans une page, il faut se poser quelques questions : le plug-in nécessaire est-il répandu ? Faut-il prévoir une version du site ne nécessitant pas cette extension ? N'y a-t-il pas moyen d'inclure une installation automatique du plug-in si l'utilisateur ne l'a pas installé ?

Car n'oubliez jamais qu'avant d'être esthétiquement réussie, la page se doit d'être lisible !

## 1.4 Streaming ou téléchargement ?

Il existe deux façons très différentes d'intégrer des extraits sonores ou vidéo dans une page :

- **Le téléchargement**
    - Le visiteur enregistre le fichier sur son ordinateur, puis le lit à partir de son disque dur comme n'importe quel autre fichier.
    - Ceci implique que le fichier doit être intégralement rapatrié avant d'être lu. Même la lecture d'une vidéo de 2Mo risque de prendre du temps…

- **Le streaming**
    - Le visiteur lit l'extrait directement à partir du serveur, sans téléchargement du document.
    - Tant que la connexion est suffisante pour le débit imposé par la vidéo, tout se déroule parfaitement. Là encore, tout est une question de compression et de bande passante.

# 2. Images animées

## 2.1 Définition

Il est possible d'insérer des séquences d'images s'exécutant à la manière d'un film.

Ces images, exclusivement au format GIF, se nomment « images animées » ou « GIF animés ».

Ces "GIF animés" peuvent être construits par exemple avec le logiciel "MS GIF Animator" disponible gratuitement sur le site Microsoft, ou Animation Shop (Paint Shop Pro).

L'utilisation de l'attribut **lowsrc** est intéressante lorsque l'image affichée est une image GIF animée dont la taille du fichier peut être importante.

## 2.2 Création d'un gif animé

## La création d'un gif animé se passe en deux étapes :

- définir chaque image qui va constituer la source.
- inclure les différents contrôles pour réaliser l'animation finale.

**1 - Sélectionner les sources.**

Toutes les images de l'animation doivent être de même taille et avoir la même palette de couleurs. Ceci n'est pas obligatoire mais cela simplifie le travail et de plus, cela diminue et optimise la taille du fichier final.

**2 - Inclure les contrôles.**

Après avoir inséré chaque image pour réaliser le montage final, il faut définir ou ajouter :

- la hauteur et la largeur de votre animation, normalement elles s'affichent automatiquement avec les logiciels les plus courants
- l'ordre des images sélectionnées
- un bloc de contrôle, pour déterminer combien de fois l'animation sera jouée.
    - Habituellement, il se fait avec l'attribut LOOP (boucle) dont la valeur doit être fixée entre 0 et 32760
- la vitesse entre chaque image (en ms) qui permettra d'obtenir une animation plus ou moins rapide
- la transition entre chaque image
- la palette de couleurs globale de l'animation.
    - Si chacune des images a une palette de couleurs différentes, à l'aide de certains logiciels, il est possible de forcer celle-ci à utiliser une palette de couleurs unique pour l'animation.
- des blocs de commentaires, pour insérer des annotations ou un copyright.

Voici un exemple de gif animé présenté sous forme de diagramme.

| Bloc commentaire | Image 1 | Tempo | Image 2 | Tempo | Image 3 | Tempo | Image 4 | Tempo |
|---|---|---|---|---|---|---|---|---|
| All HTML - www.allhtml.com | | 50 ms | G | 50 ms | GI | 50 ms | GIF | 50 ms |

<---------- Retour vers image 1 (indéfiniment)

# 3. Séquences Vidéo

Un autre moyen d'animer le comportement de la page, est d'insérer une séquence vidéo sous la forme d'un fichier.

## 3.1 Les différents formats vidéo

Voici un tableau récapitulant les différents formats, leurs origines ainsi que les logiciels permettant de les lire.

| Extension | Origines - Observations | Logiciel |
|---|---|---|
| .ASF ou .ASX | Format Netshow (Microsoft). | W.Media.Player. |
| .AVI | Vidéo for Windows. | W.Media.Player. |
| .FLI | Animation. | W.Media.Player. |
| .MOV | Fichier Quicktime (Apple - Mac). | Quicktime. |
| .MPEG ou .MPE ou .MPG | Motion Picture Experts (MPEG-1). | W.Media.Player. |
| .RPM ou .RA ou .RM | Format RealVideo (ou RealAudio). | RealPlayer G2. |
| .SWF | Format Flash (Macromedia). | Plug-in Shockwave. |

**MOV**, **AVI** et **ASF**

- Ce sont des formats très peu compressés. Le format MOV est issu du monde Macintosh, AVI et ASF viennent de Microsoft.

**MPEG**

- Il existe différentes versions (1,2 et 4). La dernière version correspond au format adopté pour les CD-Vidéo, les DVD et la TV numérique.

**RealVideo**

- Il reste le seul format créé spécifiquement pour l'utilisation de vidéo sur le Net.

## 3.2 Intégrer une vidéo AVI

L'intégration d'un fichier AVI peut s'effectuer de différentes façons :

1. Soit avec la balise <A HREF>

2. Soit avec la balise <IMG>
3. Soit avec la balise <EMBED>
4. Soit avec la balise <OBJECT>

### 1. La balise <A HREF>

Le plus simple pour insérer une vidéo (ou un son) dans une page est d'appeler cette vidéo (ou le son) par un lien hypertexte. Il alors diffusé en mode téléchargement : le lecteur externe (RealPlayer par exemple) est alors lancé et le fichier est téléchargé complètement avant d'être joué.

```
<A HREF="other/video.avi">Ouvrez les yeux !!</A>
```

Quand l'utilisateur clique sur ce lien, deux cas se présentent :

- Aucun logiciel (ou plug-in) n'est associé à ce type de fichier. Le navigateur propose à l'utilisateur de télécharger le fichier, lequel pourra ensuite être lu hors connexion.
- Un logiciel (ou un plug-in) est associé à ce type de fichier. Le fichier est pris en charge automatiquement par le programme adéquat après téléchargement. Dans certains cas, le fichier peut également être lu en direct (***streaming***). Tout dépend du logiciel.

### 2. La balise <IMG>

Cependant, **seul MS Internet Explorer** permet d'insérer ces animations grâce à l'existence de l'attribut **dynsrc** utilisé comme dans l'exemple suivant :

```
<img src="image.gif" dynsrc="film.avi" width="50" height="50">
```

Ceci permet l'affichage de l'image GIF (image.gif) puis, lorsque le document est mis en place, la séquence vidéo (film.avi) est lancée.

L'emploi de l'attribut ***dynsrc*** s'accompagne de l'emploi d'autres attributs complémentaires : ***controls***, ***loop*** et ***start***.

L'attribut **controls** (sans paramètre) provoque l'affichage de boutons pour contrôler le déroulement de la séquence.

L'attribut **loop**, avec une valeur numérique, précise le nombre de fois où la séquence sera jouée. L'attribut **loopdelay**, avec une valeur numérique, spécifie en centièmes de seconde, le temps avant que la séquence vidéo recommence à jouer.

L'attribut **start**, avec la valeur **fileopen** ou **mouseover**, provoquera le démarrage de la séquence, dès que le fichier correspondant sera ouvert (fileopen) ou quand l'utilisateur placera la souris sur la zone réservée à cette animation (mouseover).

**Conseil :** l'attribut **src** doit toujours être utilisé avec ***dynsrc*** afin que le lecteur visualise quelque chose s'il ne dispose pas de modules permettant de jouer la séquence vidéo.

## 4. La balise <EMBED>

Pour lire un fichier multimédia dans une page Web, il est possible de forcer l'utilisation d'un **plug-in spécifique** grâce à la balise **<EMBED>** (Netscape et Internet Explorer 3.0). Cette balise, non référencée dans les normes de langage HTML, permet en général d'insérer un contrôle pour lire les fichiers multimédias de façon interactive (boutons de lecture, stop, contrôle de volume, etc.).

La balise permet donc d'inclure n'importe quel format de fichier video ou son.

Attention, elle n'est pas compatible avec certaines versions d'Internet Explorer (3.0 par exemple). Pour pallier cela, on utilise l'élément NOEMBED (ignoré par les navigateurs qui comprennent la balise <EMBED>) pour fournir un texte, ou une image de remplacement.

***Attributs***

| <embed | | | |
|---|---|---|---|
| align | = | left, right, top *ou* bottom | Définit l'alignement de l'image associée au logiciel de lecture, ou à sa console de gestion (bouton de lecture, pause, etc…). |
| border | = | nombre | Largeur de bordure. |
| width | = | nombre | Largeur de la fenêtre qui accueillera la console. |
| height | = | nombre | Hauteur de la fenêtre qui accueillera la console. |
| palette | = | background \| foreground | Spécifie si la palette doit être en avant-plan ou en arrière-plan. Par défaut en avant-plan. |
| pluginspace | = | URL | URL de la page Web où télécharger le plug-in dédié. |
| src | = | URL | Nom de document HTML à afficher dans ce cadre. |
| autostart | = | booléen | Spécifie sous Netscape si l'objet multimédia est joué automatiquement (TRUE) ou non (FALSE). Valeur par défaut : TRUE. |
| autoload | = | booléen | Spécifie si le chargement de l'objet multimédia s'effectue automatiquement (TRUE) ou non (FALSE) (défaut : TRUE). |
| loop | = | nombre ou booléen | Spécifie le nombre de fois où le fichier sera joué (si la valeur est –1, le fichier sera joué indéfiniment).<br>Dans le cas du booléen (TRUE ou FALSE), cet attribut est optionnel et permet de faire jouer le fichier audio ou vidéo à plusieurs reprises. Valide sous Netscape. |
| type | = | mots prédéfinis | Indique le type MIME, par exemple : TYPE="audio/mod" (facultatif). |
| controls | = | mots prédéfinis | Indique l'aspect de la console de contrôle :<br>▪ CONSOLE : Affiche toute la console (choix par défaut).<br>▪ SMALL CONSOLE : Affiche une console réduite. |

| | | | |
|---|---|---|---|
| | | | ▪ PLAYBUTTON : Affiche seulement le bouton lecture.<br>▪ STOPBUTTON : Affiche seulement le bouton stop.<br>▪ PAUSEBUTTON : Affiche seulement le bouton pause.<br>▪ VOLUMELEVER : Affiche seulement le bouton volume. |
| hidden | | | Cache le panneau de contrôle. |
| > | | | |

**Exemple**

```
<EMBED SRC="video.avi." WIDTH="200" HEIGHT="200" AUTOSTART="false" LOOP="1">
```

Une autre façon d'insérer des animations vidéo dans ses pages web s'effectuent à l'aide des éléments HTML object, recommandés par la norme HTM4.0 (voir module "Langage HTML : pages dynamiques").

## 3.3 Intégrer une vidéo Quicktime

Le module Quicktime (Apple) permet d'afficher des vidéos de type .MOV (mais aussi les formats AVI, MIDI, WAV ... avec Quicktime 3.0).

L'intégration de vidéo Quicktime s'effectue de manière identique au format AVI. Cependant la balise <EMBED> possède plusieurs attributs particuliers :

1. AUTOPLAY : indique si la vidéo est jouée automatiquement (TRUE) ou non (FALSE) (valeur par défaut : TRUE).
2. CONTROLLER : affiche la barre d'outils (TRUE) ou non (FALSE) (valeur par défaut : TRUE). Si elle est affichée, il faut penser à augmenter la valeur HEIGHT de 24 pixels.
3. LOOP : indique si la vidéo diffusée se fait en boucle.
4. PLAYEVERYFRAME : indique si la vidéo s'exécute au fur et à mesure du téléchargement (TRUE) ou non (FALSE). Ceci est équivalent au streaming mais l'image sera saccadée.
5. TARGET : identique pour un lien vers un cadre.
6. HREF : lien avec la vidéo.
7. PAN, TILT, FOV, NOD, et CORRECTION sont utilisés uniquement pour la diffusion de film VR (panoramique).

**Attention :**

L'installation de Quicktime sur la station ne suffit pas à le lancer automatiquement. Il faut d'abord le définir dans le type MIME de Windows sinon, c'est le contrôle ActiveMovie (ou le lecteur Windows Media d'Internet Explorer) qui s'exécutera.

**Exemple**

```
<EMBED SRC="video.mov" WIDTH="200" HEIGHT="200" CONTROLLER="false">
```

## 3.4 Intégrer une vidéo RealVideo

L'intégration de Vidéo avec RealVideo est similaire à l'intégration d'un fichier audio avec RealAudio.

Par contre, l'attribut EMBED possède certains attributs particuliers, notamment l'attribut CONTROLS qui possède des valeurs spécifiques :

- All : la fenêtre de contrôle est complète.
- ControlPanel : intègre les boutons play, pause, et stop ainsi que le curseur de position.
- InfoVolumePanel : intègre une fenêtre d'information ainsi que le curseur de volume.
- InfoPanel : n'intègre que la fenêtre d'information.
- Statusbar : intègre une barre d'état.
- Playbutton : intègre les boutons play et pause.
- Stopbutton : n'intègre que le bouton stop.

L'attribut CONSOLE, attendant un nom, permet si plusieurs fichiers sont appelés depuis une même page, de les relier par le même nom et donc de réutiliser les mêmes valeurs de l'attribut CONTROLS.

**Exemple**

```
<EMBED SRC="other/video.ra" CONTROLS="InfoPanel" WIDTH="200" HEIGHT="35"
AUTOSTART="FALSE">
```

## 2.5 Principe du streaming

Diffuser de la vidéo en streaming est un peu plus complexe qu'un téléchargement. Il faut d'abord que l'hébergeur possède des serveurs Real.

Le principe du streaming par lui-même est simple. Il s'agit d'une méthode qui permet d'obtenir un téléchargement fluide. Le fichier vidéo (ou sonore) se chargera petit à petit, ce qui sera transparent et, non pénalisant pour le visiteur.

Intégrer une video (ou un son) avec RealPlayer, en faisant appel au streaming, se passe en trois étapes.

- Créer le fichier au format RealVideo (ou RealAudio).
- Création du Metafile.

- Intégrer le fichier final dans votre page.

**1 - Créer le fichier au format RealVideo (ou RealAudio).**

Pour créer un fichier RealVideo (ou RealAudio), il faut passer par l'utilitaire **Real Producer**. Cet utilitaire, permettant de transformer un fichier .wav en **.rm** (video) ou **.ra** (audio), est très simple d'utilisation.

**2 - Création du Metafile.**

Le Metafile est juste un fichier ayant l'extension **.ram** qui contient une ligne appelant le fichier RealVideo (ou RealAudio). Par exemple, si le fichier s'appelle audio.ra, s'il a été transféré par FTP dans un répertoire se nommant son et que l'adresse du site est www.music.com :

*http://www.music.com/son/audio.ra* **ou** *pnm: /chemin/ma_video.rm*

La création de ce fichier que l'on nomme dans l'exemple *meta.ram* se fait tout simplement avec un éditeur de texte (Bloc-note par exemple). Attention à la casse ! Puis il faut publier le fichier en question sur le serveur d'hébergement aux côtés du fichier vidéo ou son.

**3 - Intégrer le fichier final dans votre page..**

Pour relier la page à ce Metafile, il suffit de créer un lien hypertexte qui appellera celui-ci et lancera RealPlayer. Par exemple, si le Metafile est transféré sur la racine du site :

***<A HREF="meta.ram">Du streaming avec RealAudio</A>***

Il est possible d'opter pour la balise <EMBED> à la place du lien.

## 2.5 Intégration au sein d'une page

Cette méthode permet d'incruster la vidéo au sein d'une page. Votre vidéo apparaîtra et sera jouée sur votre page. Différents paramètres permettent de définir l'intégration de la vidéo dans la page : vidéo lue dès l'affichage de la page, intégration d'un panneau de contrôle (boutons Lecture, Stop, etc.)... Placez pour cela le code suivant à l'endroit où vous souhaitez voir apparaître la vidéo :

**Exemple**

```
<OBJECT ID="identificateur">
<EMBED SRC="/chemin/ma_video.ram" type="audio/x-pn-realaudio-plugin" CONSOLE="identificateur" CONTROLS="ImageWindow" HEIGHT=192 WIDTH=144 AUTOSTART=true>
</OBJECT>
```

Voici quelques commentaires sur les paramètres :

- Les paramètres *identificateur* et *Clip1* servent à lier plusieurs contrôles.
  - Il s'agit d'affecter aux paramètres *ID* et *CONSOLE* des noms qui permettront par la suite de préciser qu'un bouton (Play, Stop, etc.) est associé à la vidéo.
- Le paramètre *ImageWindow* précise qu'il s'agit d'une fenêtre vidéo.

La vidéo est alors jouée dès l'affichage de la page. En l'absence d'un bouton de lecture de la vidéo, un clic droit sur la fenêtre affichera un menu contextuel permettant de jouer de nouveau la vidéo.

# 5. Sons ou Fichiers Audio

L'écoute d'une bande sonore impose à l'internaute la possession d'un lecteur adéquat selon le format des fichiers utilisés.

Les deux principaux lecteurs sous Windows restent le logiciel MediaPlayer de Microsoft ou RealPlayer de Real.

## 4.1 Les différents formats audio

Voici un tableau récapitulant les différents formats, leurs origines ainsi que les logiciels permettant de les lire.

| Extension | Origines - Observations | Logiciel |
|---|---|---|
| .AIFF ou .AIF | Format du Macintosh. | W.Media.Player |
| .AU | Format de SUN / NEXT (Unix). | W.Media.Player |
| .MID ou .MIDI | Musical Instrument Digital Interface. | W.Media.Player |
| .MP3 ou .MP2 ou .MPA | MPEG Layer 3 - 2 - 1. | Winamp |
| .RPM ou .RA ou .RM | Format RealAudio (ou RealVideo). | RealPlayer G2 |
| .WAV ou .WAVE | Format de Windows. | W.Media.Player |

**MIDI**

- Il constitue le format le plus léger. Il ne s'agit pas d'un enregistrement sonore, mais d'une reconstitution de musiques note par note à l'aide de la carte son.

**WAV**

- Il constitue le format le plus répandu. Il s'agit d'un enregistrement sonore non compressé, tel qu'on peut le trouver sur les CD audio.

**MP3** et **RealAudio**

- Ce sont des formats compressés. Le premier dispose d'une excellente qualité sonore, le second favorise la compression et la rapidité de téléchargement.

## 4.2 Comment intégrer un son MIDI, WAV, MP3 ou AU

L'intégration d'un fichier de ce type (valable aussi pour les formats .aiff) peut s'effectuer de différentes façons :

- Soit avec la balise <BGSOUND>
- Soit avec la balise <A HREF>
- Soit avec la balise <EMBED>
- Soit avec la balise <OBJECT>

**La balise BGSOUND**

Cette balise présente l'inconvénient majeur d'être spécifique à Internet Explorer (reconnu depuis la version 3.0). Elle est donc non normalisée. Elle peut s'insérer aussi bien dans l'entête que dans le corps de la page. Elle possède plusieurs attributs :

- SRC : spécifie l'URL du fichier à charger.
- LOOP : spécifie le nombre de fois où le fichier sera joué (si la valeur est –1, le fichier sera joué indéfiniment).

```
<BGSOUND SRC="dico.mid" LOOP="2">
```

**La balise <A HREF>**

Le plus simple pour insérer un son dans une page est d'appeler ce son par un lien hypertexte.

```
<A HREF="other/exson.wav">Ecoutez !!</A>
```

**La balise <EMBED>**

Il s'agit de réutiliser la balise avec les même attributs que ceux mentionnés dans le paragraphe précédent.
Quelques attributs sont spécifiques à certains plug-ins : VOLUME (volume initial en pourcentage), STARTIME (indique le temps de départ par rapport au début du fichier sous forme mm:ss), ENDTIME (indique le temps de fin par rapport au début du fichier sous forme mm:ss).

```
<EMBED SRC="other/funky2.mid" WIDTH=145 HEIGHT=60 AUTOSTART="false">
```

Désormais, l'élément object remplace tous les moyens énumérés jusqu'à présent. Des exemples sont donnés dans le chapitre consacré au langage HTML4.0.

### 4.3 Comment intégrer du son avec RealAudio

L'intégration d'un fichier RealAudio est similaire à un fichier MIDI ou WAV.

**Exemple d'un fichier RealAudio avec l'élément ControlPanel**

```
<EMBED SRC="other/exreal.ra" CONTROLS="ControlPanel" WIDTH="200"
HEIGHT="35" AUTOSTART="FALSE">
```

# 6. Introduction au 3D et au VRML

## 5.1 Le langage VRML

Le Langage VRML (Virtual Reality Modeling Language) permet de concevoir des simulations interactives en multiutilisateurs et en **trois dimensions** (cyberespace). Ce langage est né en 1994 grâce à Pesce, Kennard et Parisi. Malgré qu'il ait des difficultés à s'imposer (car trop dépendant de la vitesse de connexion), il devient un standard incontournable du Web.

## 5.2 Complément sur le VRML

Les fichiers VRML sont en fait de simples fichiers ASCII. Ceci signifie qu'un simple éditeur de texte comme Notepad par exemple, peut suffire à créer un objet ou monde VRML. Néanmoins, si le projet est important, il vaut mieux passer par un utilitaire spécialement conçu pour la création de VRML.

Les fichiers doivent être sauvegardés avec l'extension .wrl.

On peut remarquer qu'il existe deux versions :

- VRML 1.0 (monde assez statique)
- VRML 2.0 (monde dynamique, pouvant accueillir du son)

## 5.3 Insérer du VRML dans vos pages

Il existe deux façons d'insérer un fichier VRML :

- Soit à l'aide de lien hypertexte

```
<A HREF="objet.wrl">Cliquez ici</A>
```

- Soit avec la balise <EMBED> (remplacée par <OBJECT> dans HTML 4.0) déjà utilisée pour insérer du son ou de la vidéo. Il faut ici définir la hauteur (HEIGHT) et la largeur (WIDTH) de la fenêtre qui va accueillir le monde ou l'objet VRML.

```
<EMBED SRC="objet.wrl" WIDTH="300" HEIGHT="200">
```

En dehors de l'animation 3D, il existe les animations vectorielles fort en vogue actuellement. Dans ce domaine et à l'heure actuelle, Flash de Macromedia et le format **SWF** sont indétronables.

# Conclusion

Le Développeur Multimédia sera en charge d'un projet mêlant son et image, voir vidéo. La connaissance de Director, un des produits phares de Macromedia est généralement demandée. L'encodage sous Quicktime ou RealVidéo est souvent utilisé dans ce type de projet.

# Complément 2 : Balises d'en-tête de la page

## Introduction

Dès les premiers chapitres consacrés au langage HTML, une première balise d'entête a été citée : la balise **<title>** donnant un titre au document HTML une fois celui-ci affiché par le navigateur.

D'autres éléments, tous aussi intéressants, peuvent être imbriqués : BASE, BASEFONT (voir police de caractères), BGSOUND (voir composants multimédia), ISINDEX, LINK, META, SCRIPT et STYLE. L'objet de ce chapitre est de toutes les passer en revue, exceptés les deux derniers vus dans un chapitre spécifique.

## 1. Balise Base

### Description

Lorsqu'un ensemble de documents ne se trouve pas sur le serveur web d'origine (ou racine) mais sur un autre serveur (serveur secondaire ou de sauvegarde), la balise base (**HTML 2.0**) permet à la fois d'atteindre l'adresse de base, mais aussi d'employer des adresses relatives à l'intérieur de la page.

La balise BASE est employée différemment selon l'attribut qui suit. Les deux attributs ne sont jamais utilisés au sein de la même balise mais font l'objet de deux balises différentes.

### Attributs

| | | | |
|---|---|---|---|
| <base | | | |
| href | = | URL | Spécifie un lien URL comme adresse de base pour tous les liens relatifs. |
| target | = | cible | Désigne la cible de tous les liens d'un document que ce soient des cadres ou non. |
| > | | | |

La balise base est ouverte (sans balise de fermeture).

La balise suivie de l'attribut **href** permet à l'adresse URL du document courant d'être enregistrée par le navigateur. Ceci a pour but de palier aux situations dans lesquelles le document pourrait être lu hors de son contexte.

Cette technique a deux avantages pratiques:

- pour le programmeur :

- elle permet, en cas de changement de l'arborescence du disque sur lequel est chargée l'application, d'opérer rapidement les modifications en n'ayant à modifier qu'une adresse par document; pour une application lourde. Ceci constitue un gain de temps appréciable, d'autant que des éditeurs HTML avancés sont capables de modifier une chaîne de caractères sur un grand nombre de documents à la fois;

- pour l'utilisateur :

- elle permet, moyennant un minimum de modifications, de reconstituer sur le disque dur local l'arborescence de toute une application.

Lorsque l'adresse de base n'est pas spécifiée, le navigateur HTML utilise l'URL d'accès au document pour résoudre le problème des adresses relatives.

**Exemples**

L'emploi de la balise base suivie de l'attribut href indique au navigateur que les fichiers du document sont à chercher à l'adresse reconstituée : adresse de base mentionnée ici + adresse relative rencontrée dans le corps de la page.

```
<head>
<BASE HREF=http://www.pot.fr/>
</head>
```

Cet exemple mentionne comme adresse de base une adresse URL simple, sans indication de chemin derrière le nom de site. Dans ce cas, il permet de référencer les adresses de liens externes du genre *toto.html* à l'adresse :

- http://www.pot.fr/toto.html.

Le deuxième exemple qui suit mentionne une page comme adresse de référence. Cela veut dire que tous les liens référencés de cette page ne sont autre que des liens internes.

Avec les cadres (frame), il est possible de définir une fenêtre comme cible de tous les liens d'un document. Pour ce faire, il faut ajouter l'attribut **target** à la balise **base**. La valeur peut être soit un nom de cadre défini dans la page de définition des cadres, soit un nom de fenêtre conventionnel du style _blank, _top, etc…

Voici un deuxième exemple venant de la Toile :

```
<base href="http://vancouver-webpages.com/Vwbot/metatags.detail.html">
<base target="_parent">
```

# 2. Balises META

L'objet de cette section est d'étudier parmi toutes les balises d'en-tête possibles du document les balises META élaborées entre la version 2.0 et la version 4.0 du langage HTML.

Véritable "sac à malice" du concepteur de sites Web, ces mystérieuses balises META feront d'un site une gentille œuvre d'amateur ou lui donneront la signature d'un site professionnel.

# 2.1 Généralités

## Description

Conformément à la philosophie du langage HTML, les éléments relatifs au document sont dissociés du corps du document lui-même et sont donc insérés repris dans l'en-tête (c-à-d entre les balises **<HEAD>** et **</HEAD>**). Les balises META apportent une série d'informations relatives au comportement de la page Web.

Ces informations sont principalement exploitées, soit par les **moteurs de recherche** (attribut **name**) en vue du référencement de votre site, soit par le **navigateur** de l'internaute (attribut **http-equiv**), et accessoirement par des lecteurs avertis.

Ces balises sont supportées par les navigateurs Internet Explorer et Netscape depuis la version 2.0 et par le navigateur Opera 3.0.

## Syntaxe

Les balises META requièrent une balise d'ouverture mais pas de balise de fermeture.

La syntaxe de la balise suivie des deux seuls attributs importants se présente comme ci-dessous :

**<meta name**="chaîne" **content**="chaîne">

**<meta http-equiv**="chaîne" **content**="chaîne">

Les deux premiers attributs **name** et **http-equiv** précisent le type d'information à communiquer sous la forme d'un mot clé, alors que le troisième attribut **content** précise la valeur de l'information.

## Attributs

| <meta | | | |
|---|---|---|---|
| name | = | mot prédéfini | Spécifie un nom à l'élément. |
| http-equiv | = | mot prédéfini | Spécifie un lien entre le contenu de l'élément et la réponse HTTP. |
| content | = | mots | Spécifie une valeur (méta information) associée au nom ou à la réponse http qui précède. |
| > | | | |

Un quatrième attribut d'importance secondaire peut se rencontrer : l'attribut **lang**. Ce dernier spécifie la langue du texte utilisé dans les valeurs des attributs content.

Une balise META peut indiquer par exemple des mots-clés en langue française à destination des moteurs de recherche français et une autre balise en langue anglaise à destination des moteurs de recherche anglophones.

L'emploi des balises META diffère donc aussi des autres balises HTML dans la mesure où il faut utiliser autant de balises META que de mots-clés différents.

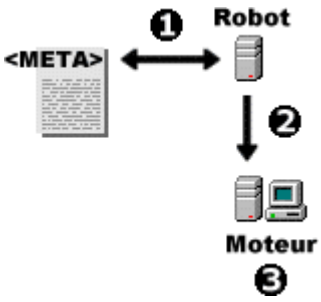
L'objet du chapitre est de passer en revue les informations principales transmises par ces deux attributs **name** et **http-equiv**.

# 2.2 L'attribut NAME

Quand on apprend que plus de la moitié des utilisateurs du Web passent par des moteurs de recherche, il importe de soigner le référencement de son site pour y apparaître de façon correcte et si possible en rang utile.

Pour ce qui est des cadres ou jeu de cadres, il convient de mettre en place les balises META dès la page de définition des cadres. Cependant certains moteurs (Hotbot) ont une fâcheuse tendance à mal indexer les sites dits à "frames".

Il parait utile de rappeler le fonctionnement du moteur de recherche et notamment de son robot.




a) Le robot scrute automatiquement et 24h/24 les balises META des pages publiées sur le réseau Internet

b) Il ramène les renseignements au moteur de recherche

c) Le moteur de recherche met à jour au fur et à mesure son index

## 2.2.1 Les Indispensables

Mots Clés ou <META NAME="Keywords" CONTENT="mot-clé1,mot-clé2,mot-clé...">

Ces mots-clefs sont repris par les moteurs de recherche et ajoutés aux mots-clefs trouvés dans le corps du document lui-même. Tous (ou presque) les moteurs de recherche l'utilisent.

Quelques commentaires cependant s'imposent :

- Les mots-clés doivent être séparés par une virgule. L'auteur préconise de ne pas mettre d'espaces à l'intérieur d'un mot-clé (même si certains webmestres confirmés l'utilisent)1 afin d'éviter toute confusion lors des recherches par mot-clé.
- La littérature sur le sujet parle d'une limitation à 1000 caractères (amplement suffisant).

- Il est conseillé d'écrire les mots clés, d'une part en minuscules pour éviter les problèmes de "casse" (différence entre majuscules et minuscules, en anglais « case sensitive » ou non), d'autre part sans accents. La raison en est que l'on ne sait pas quel est le système d'exploitation du site moteur de recherche. Dans le doute, pour assurer une prise en compte par le maximum de moteurs de recherche, sans générer des problèmes d'interprétation, il convient d'adopter ces règles.
- L'art consiste à trouver les bons mots-clés relatifs au contenu du site. Il faut se mettre à la place des lecteurs potentiels. Quels mots, quels synonymes, quelles alternatives peuvent être utilisés pour décrire le site ?
- Il est interdit de mettre des mots-clés "bidon" qui sont bien entendu très attirants mais sans rapport avec le site.

Voici un exemple tiré d'une page relative au référencement :

```
<META NAME="Keywords" LANG="fr" CONTENT="trafic, tags, abstract ,description, balise, meta, astuce, référencement">
```

La tentation est grande de répéter un certain nombre de fois un même mot-clé pour espérer un meilleur classement :

```
<META NAME="keywords" CONTENT="html,html,html,html,html>
```

Désolé, cette astuce (considérée comme du spam) est maintenant pénalisée par les moteurs de recherche.

Description ou <META NAME="Description" CONTENT="description de la page">

Une deuxième utilisation de l'attribut "**name**" consiste à renseigner le moteur de recherche sur une description sommaire de cette page. Cette description se doit d'être pertinente, attirante et brève. En effet, selon les moteurs de recherche, seuls les 150 à 240 premiers mots sont effectivement aspirés (limité à 256 caractères).

**Exemple de la FNAC :**

```
<META NAME=DESCRIPTION CONTENT="Fnac.com propose plus d'1 million de produits culturels. Vous pouvez commander et acheter en ligne vos livres, disques, jeux vidéos, cédéroms, vidéos, DVD, billets de spectacle et matériels techniques.">
```

D'autres mots-clés sont équivalents à Description : **Abstract**, **Title**.

## 2.2.2 Les Utiles

Robots ou <META NAME="Robots" CONTENT="Instructions pour les robots">

Cette balise indique aux robots si le site doit ou ne doit pas être indexé par eux. Les valeurs possibles qui sont en fait des instructions de conduite de l'indexation sont :

- All (valeur par défaut) autorise les robots à indexer les pages et à suivre les liens hypertextes d'une page à l'autre.
- None notifie aux robots de ne pas indexer les pages et de ne pas suivre les liens.
- Index autorise l'indexation des pages par les robots.
- NoIndex interdit l'indexation aux robots.
- Follow donne la permission aux robots de suivre les liens hypertextes des pages
- NoFollow pour le contraire.

**Exemple**

```
<META NAME="Robots" CONTENT="Index,Nofollow">
```

Cette balise est reconnue par tous les robots de recherche.

Fréquence de Visite du Moteur ou
<META NAME="Revisit-after" CONTENT=" Fréquence de visite du moteur">

Cette balise force le moteur de recherche à revenir sur votre site au bout de X jours.

Voici l'exemple de Paris-Match :

```
<META NAME="revisit-after" CONTENT="1 days">
```

Langage ou <META name="language" content="FR" >

Ce mot-clé permet de renseigner le moteur si la page Web aspirée est à ranger dans la catégorie web francophone ou autre.

Audience ou <META NAME="Rating" CONTENT="Destination de votre audience">

Cette balise permet de définir le contenu du site, puis de l'évaluer et de le classer. Cet indice d'audience n'a aucune influence directe sur les résultats directs d'une consultation. Il sert à mieux classer les informations recueillis. Une seule valeur est attendue. Les appréciations sont du style :

- "General" pour tout public
- "Mature" pour public adulte
- "Restricted" pour accès restreint
- "Safe For Kids" signifie "correct pour enfants"
- "14 years" pour tout public ayant au minimum 14 ans.

**Exemple**

```
<meta name="rating" content="safe for kids">
```

### Editeur ou <META NAME="Generator" CONTENT="nom de l'éditeur HTML utilisé">

Cette balise n'a aucune influence sur les moteurs de recherche, ni sur le navigateur utilisé. Cette information n'intéresse que les sites statistiques ou les sites hébergeurs ou les responsables marketing des sociétés propriétaires des éditeurs HTML pour calculer la part de marché de leur produit.

Les éditeurs avancés peuvent rajouter cette balise à l'insu du concepteur de site.

```
<META name=GENERATOR content=" Microsoft FrontPage 4.0">
```

```
<meta name="GENERATOR" content="Mozilla/4.74 [en] (Win98; U) [Netscape]">
```

### Auteur ou <META NAME="Author" CONTENT="nom de l'auteur">

Cette balise est d'une utilité discutable car rares sont les moteurs de recherche qui en tiennent compte. A priori, seul Nomade fournit l'URL du site après avoir introduit le nom de l'auteur.

Il n'empêche qu'il est d'une légitime fierté de signer son œuvre.

Il existe d'autres variantes au mot-clé Author : **Owner ou Author-Corporate**.

### Copyright ou <META NAME="Copyright" CONTENT="Copyright © date nom">

Il s'agit d'un début de protection de site qui a juste une valeur informelle et non une valeur juridique.

Il advient que des correspondants (courrier électronique) demandent la reproduction de tout ou partie d'un site. Une règle élémentaire de la Nétiquette consiste à mettre en place le copyright au sein de chaque page.

Voici deux exemples dont on peut s'inspirer.

```
<META NAME="Copyright" CONTENT=" copyright weburbia 1996, all rights reserved">
```

Ou

```
<META NAME="Copyright" CONTENT=" © TeleChoice Inc. , 1999, 2000, 2001">
```

### URL du Site ou <META NAME="Identifier-URL" CONTENT="URL du site">

Ce mot-clé indique l'adresse URL du site référencé, mot-clé dont le contenu intéresse les moteurs de recherche.

### Publieur ou <META NAME="Publisher" CONTENT="Hébergeur du site">

Ce mot-clé indique le nom de la société qui héberge le site, mot-clé dont le contenu intéresse certains moteurs de recherche à des fins statistiques.

```
<META NAME="Publisher" CONTENT="eDesign">
```

Distribution ou <META NAME="Distribution" CONTENT="Global ou Local">

Cette balise META définit la destination publique de vos documents présents sur le site Web :

- valeur **Global** pour tout site diffusé sur la Toile,
- valeur **Local** si la diffusion est restreinte,
- valeur **IU** pour une diffusion limitée à l'Intranet de l'organisme

<META NAME="Abstract" CONTENT="Description ici de votre site...">

La phrase indiquée en valeur (200 mots maximum) est prise en compte par certains moteurs de recherche comme description du site (donc non obligatoire). Cette phrase présente le site, donc soignez le sens de celle-ci.

Les mots constitutifs de cette phrase sont aussi considérés comme des mots clés par la plupart des moteurs.

Elle doit être doublée avec la balise <META NAME="Description"...> qu'elle ne remplace pas.

Il est judicieux d'établir une certaine harmonie avec la balise <TITLE> et les mots clés choisis à l'intérieur de la balise META NAME="keywords".

Catégorie ou <META NAME="Category" CONTENT="nom">

Ce mot-clé permet de spécifier la catégorie du site, utile pour les moteurs de recherche de type annuaire.

En dehors de tous les mots-clés cités dans cette section, d'autres mots-clés existent mais sont moins usités :

- Formatter (Microsoft FrontPage)
- Classification (Netscape)
- VW96.ObjectType : utilisé pour un schéma de définition de documents comme des FAQ ou HOWTO.

```
<META NAME="VW96.objecttype" CONTENT="Magazine">
```

La liste des mots-clés relatifs à l'attribut **name** n'est pas exhaustive. Il en existe d'autres encore moins usitées ou propriétaires (IBM, Microsoft, etc...), ou spécifiques à un logiciel donné (Lotus, etc...).

## 2.3 L'attribut HTTP-EQUIV

Rares mais au combien spectaculaires, certaines balises META sont directement interprétées par le navigateur pour, par exemple, charger automatiquement une page HTML après x

secondes, effectuer une transition style PowerPoint ou forcer le rechargement d'une page sur le réseau bien que déjà présente dans le cache.

L'attribut **http-equiv** force le serveur Web hébergeur à accepter la variable indiquée, et à lui faire prendre la valeur prise par l'attribut **content**.

Les serveurs HTTP utilisent la valeur spécifiée par l'attribut **http-equiv** comme nom d'un champ d'en-tête type **[RFC822]** à émettre dans l'en-tête HTTP de réponse précédant le document. Il faut consulter la spécification HTTP (**[RFC1945]** et **[RFC2068]**) pour plus de détails sur les champs d'en-tête valides.

## 2.3.1 Les Utiles

### Langage ou <META HTTP-EQUIV="Content-language" CONTENT="fr">

Cette balise déclare la langue utilisée dans le document HTML. Elle est utilisée par les navigateurs des clients. Elle est de plus en plus utile, maintenant que les moteurs de recherche anglo-saxons (et non des moindres comme Altavista et Hotbot) ont inclus la langue dans leurs critères de recherche.

**Remarque**

Cette possibilité peut se substituer à l'attribut **lang** présent au niveau de chaque balise META.

Voici quelques exemples de codification de langage :

| Langage | Valeur |
|---|---|
| Allemand | de |
| Américain | en-us |
| Anglais | en |
| Belge Flamand | nl-be |
| Espagnol | it |
| Italien | es |

### Content-Type

Le concepteur de pages Web déclare ici le format des fichiers ou des données de type MIME, suivi du jeu de caractères utilisé.

```
<META HTTP-EQUIV="Content-Type"    CONTENT="text/html; charset=iso-8859-1">
```

Ou

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

Dans les deux exemples ci-dessus, on utilise successivement les jeux de caractères reconnus au niveau de l'Europe occidentale et du continent américain, puis celui adopté au niveau international.

Si le concepteur de pages élabore un site sous Windows destiné par exemple au monde slave, il peut remplacer le jeu de caractères habituel par :

- windows-1250 pour les langues cyrilliques (russe, bulgare, etc...),
- windows-1253 pour les langues slaves non cyrilliques.

### Refresh

Une astuce classique du langage HTML utilisée depuis des années consiste, à l'aide de cette balise, de charger automatiquement la page dont l'adresse URL est spécifiée, ceci après un délai de x secondes.

```
<META HTTP-EQUIV="Refresh" CONTENT="x;URL="adresse">
```

Cette balise est fréquemment utilisée pour rediriger automatiquement un visiteur dans le cas où l'adresse du site à été modifiée. Cette possibilité de redirection se révèle d'un emploi souple et discret. Il permet à l'administrateur de site de se passer des services de l'hébergeur.

```
<META HTTP-EQUIV="refresh" CONTENT="60; URL=http://www.allhtml.com">
```

### Adresse e-mail ou <META HTTP-EQUIV=" Reply-to" CONTENT="adresse e-mail">

Cette balise permet au lecteur averti de connaître l'adresse de courrier de l'auteur de la page ou du Webmestre si elle n'est pas affichée sur la page consultée. Finalement, il est peut-être plus utile de donner son adresse électronique que son nom.

## 2.3.2 Les Eventuelles

### Expires

Cette balise META renseigne le navigateur sur la date de validité de la page en ce qui concerne la gestion de son cache. Une fois la date expirée, le navigateur supprime la page en question de son cache. Lors de la prochaine visite du site, le navigateur sera dans l'obligation de rechercher la page sur le site.

Ceci est très utile pour les pages fréquemment mises à jour notamment sur les sites d'information ou boursiers.

**Exemples :**

```
<META HTTP-EQUIV="Expires" CONTENT="Wed, 23 Feb 1999 10:49:02 GMT">
<META HTTP-EQUIV="Expires" CONTENT="0">
```

Le deuxième exemple indique que la page n'expire jamais.

En conséquence, l'emploi de cette balise peut entraîner que les robots de recherche retirent ces pages dites périmées de leur base de données.

## Pragma

Voici une autre façon de contrôler le cache du navigateur. Cette balise META demande au navigateur de l'internaute de ne pas conserver la page dans le cache.

```
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
```

On rapporte que cela fonctionne sous Netscape mais pas sous Internet Explorer...

## PICS-Label (Plateform for Internet Content Selection - Label)

Développé au profit du World Wide Web Consortium (W3C), ce sigle est devenu le standard PICS (Plateform for Internet Content Selection - Label) en ce qui concerne la sélection de contenu Internet (PICS). Le contenu du site est au préalable évalué, selon plusieurs critères, par un organisme de validation : respect de la vie privée, droits de la propriété intellectuelle, etc ... Une fois validé, le contrôle de contenu est effectué de deux façons :

- Soit les résultats sont stockées sur le serveur web du service qui a évalué le contenu du site.
- Soit le concepteur indique lui-même le résultat par l'intermédiaire d'un générateur de META TAGS.

Cette balise META permet d'insérer des mots clés contrôlés à la fois par le serveur et par le navigateur (consulter le contrôle d'accès du type RSACI dans les options de ce dernier). Il est rappelé le nom de l'organisme de validation du contrôle ainsi que des paramètres chiffrés des niveaux acceptés sur la langue, la nudité, le sexe et la violence.

Voici un premier exemple relatif au deuxième mode d'étiquetage:

```
<META HTTP-EQUIV="PICS-Label" CONTENT='(PICS-1.1
"http://www.weburbia.com/safe/ratings.htm" l r (s 0))'>
```

Voici un deuxième exemple au premier mode d'étiquetage :

```
<META http-equiv="PICS-Label" content='(PICS-1.1 "http://vancouver-
webpages.com/VWP1.0/" l gen true comment "VWP1.0" by "webmaster@allhtml.com" on
"1999.07.16T02:42-0700" for "http://www.allhtml.com" r (MC 0 Gam -1 Com 0 SF 0 Edu
-1 S 0 Can 0 V 0 Env 0 P 0 Tol 0 ))'>
```

A l'origine destiné à aider les parents et enseignants à contrôler la nature des informations que les enfants pouvaient récupérer par Internet, il pourra servir à d'autres usages d'étiquetage, dont les signatures numériques, et la gestion des droits d'auteurs et de propriété intellectuelle.

## Window-target

Cette balise force le navigateur à afficher la page dans la cible désignée, en général par exemple dans une nouvelle fenêtre (_blank).

```
<META HTTP-EQUIV=" Window-target" CONTENT="_blank">
```

## Transition

De très jolis effets de transition style PowerPoint sont possibles avec simplement une ligne de code. Mais cela ne fonctionne que **sous Microsoft Internet Explorer 4 et plus**.

Le code est :

```
<META HTTP-EQUIV="Page-Enter"
                                    CONTENT="revealTrans(Duration=1.0,Transition=23)">
<META HTTP-EQUIV="Page-Exit"
                                    CONTENT="revealTrans(Duration=1.0,Transition=23)">
```

Quelques explications :

- **Page-Enter** et **Page-Exit** indiquent respectivement que l'effet de transition se produit à l'entrée de la page et à la sortie de celle-ci.
- **Duration** détermine la durée de la transition en secondes. Elle est dans l'exemple d'une seconde. A l'usage, cette durée n'est pas d'une précision absolue.
- Transition est un nombre de 1 à 23 relatif à l'effet de transition choisi. Le chiffre 23 donne une transition aléatoire (au hasard). Les autres transitions se répartissent de 1 à 22. Ainsi, 7 ouvre la page de droite à gauche, 17 a le même effet mais en diagonale, 22 découvre la page avec un effet de lignes horizontales aléatoires, etc...

Précisons que si ces transitions ressemblent furieusement aux transitions de PowerPoint, elles fonctionnent très bien même si PowerPoint n'est pas installé sur la machine de l'internaute.

Pour terminer, les effets de cette transition ne sont perceptibles que si l'internaute accède à la page à partir d'une autre page.

## Set Cookie

Ce mot clé permet, sans aucun script, de poser un cookie dans le cache du navigateur du visiteur.

```
<META HTTP-EQUIV="Set-Cookie" CONTENT="cookievalue=xxx; expires=Saturday,
25-Sep-99 12:12:30 GMT; path=/">
```

En dehors de tous les mots-clés cités, d'autres existent mais sont moins usités :

- Content-Script-Type

```
<META HTTP-EQUIV="Content-Script-Type" CONTENT="text/javascript">
```

- Content-Style-Type

```
<META HTTP-EQUIV="Content-Style-Type" CONTENT="text/css">
```

La liste des mots-clés relatifs à l'attribut **http-equiv** n'est pas exhaustive. Il en existe d'autres encore moins usités ou propriétaires (IBM, Microsoft, etc...), ou spécifiques à un logiciel donné (Lotus, etc...).

## 2.4 Fonctionnement du Serveur Web

Avant que le concepteur de site décide du bien fondé de l'insertion de telle ou telle balise META dans le site, il parait judicieux de s'interroger sur l'attitude du serveur à l'égard de ces balises.

La formulation de la **réponse HTTP** apportée à la requête d'un client est normalement laissée à l'initiative du serveur. Dans le cas le plus simple, seuls le type MIME du fichier demandé et sa taille sont reportés respectivement dans les champs **Content-Type** et **Content-Length** de la réponse.

En y ajoutant le code de statut, on obtient un en-tête complet :

```
HTTP/1.0 200 OK
Content-Type: text/html
Content-Length: 4023
```

Un serveur Web dont le contenu est sans cesse mis à jour peut associer une date d'expiration à chaque document. Dans ce cas, chaque fois que le fichier est mis à jour, le serveur doit modifier l'enregistrement adéquat au niveau de sa base de données.

La balise META correspondante (Expires) offre une autre solution. Elle permet d'influencer directement les champs incorporés dans la réponse HTTP du serveur.

Destinées la plupart du temps à être reprises telles quelles par le serveur, les balises META doivent respecter le format prévu, notamment pour les indications de date :

```
<META HTTP-EQUIV="Creation-Date"  CONTENT="Tue, 1 Apr 1997 10:00:26 GMT">
<META HTTP-EQUIV="Last-Modified" CONTENT="Sat, 5 Apr 1997 12:45:26  GMT">
<META HTTP-EQUIV="Expires"           CONTENT="Thu, 1 Dec 1997 00:00:00 GMT">
```

La présence de balises META oblige le serveur Web hébergeur du site à scruter, puis traiter chaque page Web avant de l'envoyer à l'internaute. Un excès de balises META peut ralentir l'activité du serveur dans la livraison des pages Web.

En raison du supplément de traitement nécessaire, les serveurs HTTP ont développé au moins deux techniques pour repérer et traiter les documents possédant des balises META :

- Ranger ces documents Web dans des dossiers à part,
- Mettre en œuvre un type MIME supplémentaire (text/x-scan-meta) obligeant le concepteur à utiliser une extension spéciale (*.mhtml).

**Pour les concepteurs de pages Web, les balises META constituent la seule possibilité d'influencer la réponse du serveur HTTP.**

## 2.5 Les générateurs de balises <META>

Insérer des balises <META> dans une page n'est pas complexe.

Tous les éditeurs HTML avancés proposent cette fonctionnalité. Il est également possible de le faire « à la main » aisément et directement dans le code.

**Attention** :

Certains éditeurs « polluent » légèrement (ou fortement) le code HTML généré en insérant des balises qui ne servent à rien pour l'optimisation du code (ce qui ne signifie pas qu'elles ne servent à rien dans l'absolu). Il faudra donc, de façon obligatoire, nettoyer « à la main » le code des pages si l'on désire l'optimiser au mieux et obtenir un code « propre ».

Si le concepteur ne désire vraiment pas insérer lui-même ces balises dans les pages, un certain nombre de sites (ou de logiciels) permet également de générer automatiquement des balises <META>. Il suffit d'indiquer un certain nombre d'informations (mots clés, phrase de description, etc.) et l'on obtient, sur le Web ou par courrier, les balises correspondantes.

La plupart de ces services ajoutent une ligne de commentaire afin de préciser que les balises ont été créées grâce à un utilitaire :

```
<!—Meta-tags created by the Meta-Tag Generator
                                    http://www.websitepromote.com/resources/meta →
```

Cette ligne peut, bien entendu, être supprimée du code final.

## 2.6 Exemples Fréquemment Utilisés

### 2.6.1 Balise <META NAME>

```
<META name="Description"
content="Un Webzine mensuel et gratuit sur les actualités et nouveautés informatiques.">
<META name="Keywords"
content="Artisan2k, RIQ, informaticien, mensuelle, gratuite, gratis, free, list, diffusion,
mailing, abonnement, newsletter, news, ezine, magazine, webzine, neuf, nouveau, cours,
html, ouebemestre, webmestre, webmaster, prix, cout, achat, ordinateur, memoire, ram,
materiel, comparaison, CD-Info, cdinfo, videotron, cable, internet, antivirus, actualite, alerte,
```

```
virus, logiciel, gratuiciel, graticiel, partagiciel, freeware, shareware, conseil, truc, recherche,
inocculateit, xdrive, zden, cofa, comclick,  godado, Quebec, Canada, Monteregie, Montreal,
St, saint, Bruno, Rive-Sud, Windows, COFA, membre, fournisseur, provider, francophone,
francais." >
<META name="Author"            content="Rene Leclerc" >
<META name="copyright"         content="Artisan2k"        >
<META name="Publisher"         content="Artisan2k.com" >
<META name="Identifier-URL" content="http://artisan2k.com" >
<META name="GENERATOR" content="MSHTML 5.00.2919.6307" >
<META name="language"          content="FR" >
<META name="robots"            content="ALL">
<META name="revisit-after"     content="7 days" >
```

### 2.6.2 Balise <HTTP-EQUIV>

```
<META HTTP-EQUIV="Content-Type"       CONTENT="text/html; charset=iso-8859-1">
<META HTTP-EQUIV="Content-language" CONTENT="fr">
<META HTTP-EQUIV="Reply-to"           CONTENT="ouebemestre@artisan2k.com">
<META HTTP-EQUIV="Expires"            CONTENT="Wed, 23 Feb 1999 10:49:02
GMT">
```

**Arrivé à ce niveau du chapitre, les connaissances sont suffisantes pour faire d'un site Web un site de professionnel. Les balises META judicieusement choisies, peuvent à la fois :**

- **référencer correctement votre site (exemple ci-dessus),**
- **influencer la réponse du serveur Web hébergeur du site,**
- **influencer le comportement du navigateur du client.**

# 3. Balise LINK

**Description**

La balise LINK (HTML 4.0, Netscape 2.0 et Internet Explorer 3.0) indique la présence d'un **lien entre le document courant** et un ou plusieurs autres pouvant être utilisés par un outil d'automatisation quelconque, par opposition à <A>, qui marque une action de la part de l'utilisateur dans le corps de la page.

Un document peut présenter un nombre quelconque de liens (balise **link)** associés à d'autres documents. Chaque lien est déclaré séparément par une balise **link**.

La balise link est ouverte (sans balise de fermeture).

**Attributs**

| <link | | | |
|---|---|---|---|
| type | = | nom | Type MIME du document lié. |
| rel | = | nom | Type de relation entre le document courant et la cible. L'attribut HREF doit être présent. |
| rev | = | nom | Même chose que REL, mais la relation est inversée. |
| urn | = | nom | URN (uniform resource name) vers un document cible. |
| href | = | URL | Lien URL en relation avec un autre attribut. |
| hreflang | = | nom | Langage des documents associés en utilisant le code international standard de deux lettres. |
| charset | = | code | Encodage des caractères utilisés dans la page. |
| title | = | nom | Titre du document associé. |
| name | = | nom | Nom de l'objet ou contrôle associé. |
| methods | = | nom | Informations sur les fonctions que les utilisateurs peuvent utiliser sur un objet. |
| media | = | nom | Type de média support pour l'affichage. |
| target | = | cible | Cible de tous les liens d'un document. |
| > | | | |

**Exemples**

Le chapitre consacré aux feuilles de style mentionne cette balise pour appeler des feuilles de style externes à la page Web appelante. Voici sans doute l'exemple le plus couramment utilisé :

```
 <HTML>
<HEAD>
<LINK HREF="feuillestyle1.css" REL="stylesheet" TYPE="text/css" media="screen">
</HEAD>
<BODY>
…
</BODY>
</HTML>
```

Le deuxième exemple ci-dessous, venant du site Paris Match, indique l'adresse de courriel du webmestre (l'URI de l'auteur). Ceci permet d'avoir des statistiques sur la visite du site dans la mesure où la lecture de cette page lui renvoie un courrier.

```
<LINK REV=MADE HREF="mailto:lesommier@parismatch.com">
```

Cet exemple venant du site AllHtml permet de télécharger l'icône du site que l'on visualise, soit dans la barre d'adresses du navigateur, soit dans les favoris si le client enregistre cette page.

```
<LINK REL="SHORTCUT ICON" HREF="http://www.allhtml.com/favicon.ico">
```

Il s'agit toujours du fichier "favicon.ico" ayant les caractéristiques suivantes : 16x16 pixels sur 256 couleurs, 1,2Ko.

L'exemple cité ci-dessous est issu d'un site multilingue. Cette balise déclare le **titre** du document actuel et fournit une association ou lien avec la version espagnole du document.

```
<HEAD>
<TITLE>Référence-web</TITLE>
<LINK title="Référencia web" type="text/html" rel="alternate" hreflang="es"
                                    href="http://www.reference-web.com/espagnol/index.html">
</HEAD>
```

L'élément link déclare : le titre espagnol (*Référencia web*), le type du document (*text/html*), la relation (*alternate*) avec le document actuel, le langage (*es* - Espagnol), l'adresse URL du document associé (*http://www.reference-web.com/espagnol/index.html*).

Ajoutons un exemple sur l'utilisation de l'attribut charset (une balise META remplit la même fonction) :

```
<link charset="iso-8859-1">
```

Voici d'autres exemples pour compléter cette première version du chapitre.

| | |
|---|---|
| **<link rel="prev"** href="intro.htm"> | le document courant est précédé par le document *intro.htm*. |
| **<link rel="next"** href="suite.htm"> | le document courant précède le document *suite.htm*. |
| **<link rel="history"** href="versions.htm"> | le document *versions.htm* contient l'historique des versions du document courant. |
| **<link rel="glossary"** href="glossaire.html"> | le document *glossaire.html* est un glossaire pour le document courant. |
| <**link rel="top"** href="http://www.anima.net"> | Spécifie l'URI de la page principale du site ou sommet du site (cf. home). |
| **<link rel="contents"** href="http://www.anima.net/tdm.html"> | Spécifie l'URI de la page de la table des matières (cf. toc). |
| **<link rel="copyright"** href="http://www.anima.net/copyright.html"> | Spécifie l'URI de la page contenant la mise en forme du copyright. |
| **<link rel="help"** href="http://www.anima.net/aide.html"> | Spécifie l'URI de la page contenant l'aide. |
| **<link rel="search"** | Spécifie l'URI du moteur de recherche |

href="http://www.anima.net/recherche.html"> spécifique au site.

L'attribut REL peut aussi prendre les valeurs suivantes :

| | |
|---|---|
| Start | : référence au premier document d'une série de documents. |
| Home | : lien vers une page d'accueil ou vers une page située au sommet de l'arborescence locale du site. |
| ToC | : lien vers une table de contenu ou table des matières, |
| Index | : lien vers un index, |
| Up | : lien vers le document supérieur dans une structure hiérarchique, |
| Bookmark | : lien vers des pages favoris, |
| Banner | : lien vers un document à utiliser en tant qu'en-tête. |

L'inventaire des valeurs possibles pour les attributs REL et REV n'est certes pas exhaustif ainsi que celui des attributs possibles pour la balise LINK.

En fait, cette balise reste peu utilisée, parce qu'elle n'était pas reconnue par les navigateurs de première génération, et que son emploi reste relativement sybillin.

# 4. Autres balises

## 4.1 Balise ISINDEX

### Description

Cette balise (norme HTML2.0) informe le client que le document peut être parcouru à l'aide d'une recherche par mots-clés. Elle permet d'afficher un champ recherche comme pour un formulaire mais de façon simplifiée.

L'internaute valide par la touche Entrée la saisie du mot-clé. Un programme CGI doit résider sur le serveur pour traiter la requête, et renvoyer une page de réponses HTML dans la fenêtre ou le cadre actif. Le serveur,via le programme CGI, recherche le mot-clé au sein d'un index interne avec répertoire alphabétique.

La balise <ISINDEX> était proposée dans la version 1.0 du langage HTML notamment pour interfacer une application de base de données sur un serveur. Elle a depuis été remplacée par la balise <FORM>. La version HTML 4.0 annonce que cette balise est périmée.

**Syntaxe : <ISINDEX PROMPT="chaine" ACTION="URL">**

### Attributs

L'attribut **action** spécifie l'adresse URL du programme qui traitera la chaîne de caractères entrée dans la boîte texte. L'adresse URL peut être suivie de mots-clés destinés au moteur de recherche. Sans attribut action et mot-clé indiqué, il y a une insertion automatique des mot(s) saisi(s) par le visiteur dans le champ texte.

L'attribut optionnel **prompt** permet d'insérer une légende avant la zone d'index pour la décrire.

### Exemples

La page décrite dans le 1er exemple affichera un masque de saisie d'une clé de façon à lancer la requête *fichier.htm?cle* . L'attribut href ici présent désigne le document lui-même.

```
<HTML>
<HEAD>
<BASE HREF="fichier.htm">
<ISINDEX>
</HEAD>
<BODY>
Voici une page d'interrogation très simple
</BODY>
</HTML>
```

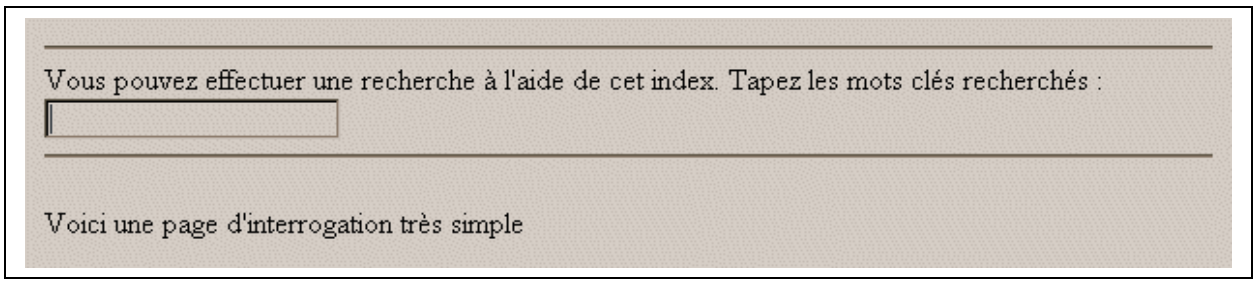



**Figure 1**

Le 2ème exemple permet d'afficher un masque de saisie de façon à lancer le programme requête « cherche » .

```
<HTML>
<HEAD>
<TITLE>Document 1</TITLE>
<ISINDEX ACTION="http://demain.com/cherche" PROMPT="Entrez la recherche : ">
</HEAD>
<BODY>
…
</BODY>
</HTML>
```


**Figure 2**

Après avoir saisi la clé "les copains", la validation par touche Entrée lance la requête suivante.


**Figure 3**

## 4.2 Balise BASEFONT

### Description

La balise **BASEFONT** (**HTML 3.2**), employée aussi bien dans la section du corps (BODY) de la page web ou dans l'en-tête (HEAD) permet de définir les propriétés relatives à la police de caractères pour l'ensemble du document.

### Attributs

| <basefont | | | |
|---|---|---|---|
| face | = | police(s) | Désigne la (ou les) police(s) de caractères utilisées. Plusieurs valeurs sont séparées par une virgule. |
| size | = | nombre | Spécifie la taille de la police de caractères. |
| color | = | couleur | Désigne la couleur en choisissant une couleur prédéfinie ou une couleur en valeur hexadécimale précédée du caractère "#". |
| style, class, id | | | Voir chapitre sur les feuilles de style. |
| > | | | |

Une valeur donnée à l'attribut SIZE, par exemple, permet de fixer la dimension du texte à une autre valeur (1 à 7) que la valeur par défaut égale à 3.

### Exemples

Voici un exemple utilisé dans le corps de la page.

```
<BASEFONT size="1">Ce texte est petit<FONT size="4"> Ce texte est plus grand</FONT>
et celui-ci est à nouveau petit.
```

La balise BASEFONT est ouverte (sans balise de fermeture).

La version HTML 4.0 annonce que cette balise est aussi périmée.

# Conclusion

D'autres éléments existent tels que l'élément **STYLE** (voir chapitre spécifique consacré aux feuilles de style), ou l'élément script (voir chapitre "Langages de Scripts" dans le module "HTML Dynamique").

Un élément NEXTID n'a pas été présenté : il est reconnu uniquement par Internet Explorer, son emploi reste confus.

Il n'est pas aisé de trouver un exemple pour l'emploi de chaque balise ou attribut. Malgré les exemples présents, l'auteur du document regrette d'être aussi succinct sur l'emploi de la balise link qui une fois bien assimilée peut rendre bien des services.

Cependant, ce chapitre offre un tour d'horizon relativement exhaustif sur l'ensemble des balises que l'on peut rencontrer dans un en-tête de document HTML.

# Complément 3 : Feuilles de style

# Introduction

L'objet du chapitre est de parcourir toutes les possibilités d'une balise particulière placée en général dans l'en-tête du document HTML : l'élément **style**.

Cette balise rassemble les éléments de description du style d'une page Web allant de la police du texte à celle des titres, en passant par l'arrière-plan. Elle permet aussi le positionnement d'un objet (image, texte,...), l'espacement entre les paragraphes, etc… Si ce style est commun à plusieurs pages Web du site, on peut même créer un fichier externe spécifique (extension *.css) appelé par chacune de ces pages. Ce fichier, décrivant l'ensemble des composantes du style choisi, s'appelle "feuille de style".

Le concept des **feuilles de style** n'est pas à proprement parler une nouveauté dans le domaine de la publication HTML. Introduit en 1997 par Microsoft avec Internet Explorer 3.0 (elles existaient déjà avec Arena sous Unix), ces feuilles de style n'ont connu au départ qu'un intérêt limité du fait que celles-ci n'étaient qu'en partie reconnues par Netscape Navigator 3.0.

Depuis les choses ont bien évolué. Les versions 4.0 des navigateurs de Microsoft et de Netscape ont reconnu l'élément STYLE. Si la norme HTML3.2 a présenté l'élément STYLE, la norme **HTML 4.0** a assis le concept des feuilles de style (CSS version 1), concept qu'elle recommande vivement aux "Web designers".

# 1. Définitions

## 1.1 Concept

### 1.1.1 Origine

Dans un document d'une certaine importance, il arrive fréquemment que l'on attribue à certains éléments des caractéristiques de mise en forme identiques. Par exemple, il est décidé que les noms de chapitres utiliseraient les éléments de style suivants : police Arial, style gras et couleur bleue.

Imaginons que l'on donne à cette définition de mise en forme un nom, soit "titre", et qu'à chaque nouveau chapitre de la page HTML, plutôt que reécrire le nom du titre et puis de le mettre en "Arial, gras, bleu", l'on puisse indiquer simplement au navigateur le nom de la mise en forme "titre" à appliquer. Cette définition de mise en forme particulière s'appelle feuille de style.

Le concept de feuilles de style (**Style Sheets**) était né. Il était déjà présent dans les logiciels de traitements de texte comme Word de Microsoft (menu Format de Word, rubrique Style). Il ne restait plus qu'à coupler ce concept à celui du langage HTML par ses propriétés spécifiques.

Les éléments HTML **<H1>** à **<H6>** correspondent respectivement aux titres 1 à 6 du logiciel de traitement Word. Les balises <P> et <DIV> correspondent à quelque chose près pour la première aux paragraphes et pour la seconde aux sections d'un document classique.

L'exemple suivant rappelle comment définir des éléments de style avec de simples balises HTML sans l'intervention des feuilles de style.

| | |
|---|---|
| <H1><font color="red">Titre1</font></H1> | STYLE des titres (titre 1) |
| <H2>< font color="blue">- A </font></H2> | STYLE des sous-titres (titre 2) |
| <DIV><font color="black">...a....</font></DIV> | STYLE du texte (normal) |

### 1.1.2 Evolutions

De la notion d'éléments de **style internes** à une page, on évolue progressivement vers celle de la feuille de **style externe** à la page si l'on rend homogène le style de plusieurs pages d'un site. On peut concevoir d'étendre l'homogénéité d'un style à l'ensemble des pages d'une rubrique, et même décider d'étendre ce style à l'ensemble du site.

Cette externalisation des éléments de style n'empêche pas le concepteur de pages de redéfinir pour une page donnée, quelques propriétés à l'aide d'un élément HTML **style**.

La deuxième évolution consiste à déclarer ses feuilles de style suivant un certain ordre de priorité déterminé (voir paragraphe FAQ) : on parle alors de **feuilles de style en cascade** (**Cascading Style Sheets ou CSS**).

Le cas le plus simple est de définir plusieurs styles à la suite pour une même balise HTML, de mettre en œuvre des propriétés à valeurs relatives ou des propriétés dont les valeurs sont héritables.

Une troisième évolution consiste à utiliser les apports de la norme **CSS 2.0**. Celle-ci sera décrite dans un prochain chapitre consacré au DHTML. Il s'agit non seulement d'utiliser les nouvelles propriétés ou les nouvelles valeurs de propriétés de la norme CSS1.0, mais de les coupler avec des scripts écrits en général en javascript.

Nous précisons pour terminer que les feuilles de style ne sont pas une composante directe du langage HTML mais plutôt un développement particulier dans la publication de pages Web.

## 1.2 Avantages

Voici quelques arguments qui peuvent convaincre les indécis. L'emploi des feuilles de style procure les avantages suivants :

- La séparation du contenu et de la mise en forme.
- La cohésion de la présentation tout au long du site avec les feuilles de style externes.

- La modification de l'aspect d'une page ou d'un site sans en modifier le contenu et cela en quelques lignes plutôt que de devoir changer un grand nombre de balises.
- Un "langage" neuf, compréhensible, simple et logique.
- La réduction du temps de chargement des pages.
- Un palliatif aux insuffisances du langage HTML (contrôle des polices, contrôle de la distance entre les lignes, contrôle des marges et des indentations ), sans devoir utiliser de tableaux ou de balise <DD>, et ainsi augmenter la créativité des écrivains du Web.
- Le positionnement au pixel près du texte et /ou des images sans passer par le principe des "layers" exclusif à Netscape 4.0.

## 1.3 Compatibilité et Références

Les feuilles de style fonctionnent à partir des versions suivantes de navigateur :

- Internet Explorer 3.0 mais de façon incomplète,
- Internet Explorer 4.0,
- Netscape 4.0.

La norme CSS1.0 (Cascading Style Sheets, niveau 1) devint une recommandation W3C en Décembre 1996. Le document de référence issu du consortium W3C est le suivant :

| Numéro | Titre | Auteur ou Editeur | Date |
|---|---|---|---|
| RFC2318 | *The text/css Media Type* | Lie, Bos, Lilley | Mars 1998 |

La liste complète (et officielle) des propriétés et recommandations concernant les feuilles de style version 1, peut être trouvée à l'adresse ***www.w3.org/TR/REC-CSS1***.

La norme **CSS2.0** (***www.w3.org/TR/REC-CSS2***), qui devint une recommendation W3C en Mai 1998, s'est construite à partir de la norme CSS1.0. Elle ajoute le soutien des feuilles de style pour des media spécifiques (par exemple, les imprimantes et les appareils vocaux), les polices téléchargeables, le positionnement d'éléments et des tableaux.

La spécification **"CSS Mobile Profile"** (***www.w3.org/TR/css-mobile***) postule comme recommandation depuis Octobre 2001.

La norme **CSS3.0** est en cours de développement.

## 1.4 Attributs

Nous présentons ci-dessous les attributs de l'élément style tel qu'il peut être utilisé, dans l'en-tête ou dans le corps de la page. La page peut soit utiliser ces éléments de style internes ou aller chercher ces éléments dans une feuille de style externe. Ces possibilités ne sont pas exclusives entre elles.

```
<style
```

| | | | |
|---|---|---|---|
| type | = | mots | Format des données internes à l'élément style : **text/css** en général. |
| media | = | mot | Médium pour lequel cette information de style est applicable (CSS2.0). |
| src | = | URL | Adresse URL d'une feuille de style externe à la page courante (valide sous Netscape). Sous Internet Explorer, il spécifie plutôt: **@import: url(URL);** |

```
>
<!--
… sélecteurs de style ou instructions …
-->
</style>
```

La balise de fermeture est obligatoire. Les attributs de la balise d'ouverture ne sont pas obligatoires.

L'ensemble des sélecteurs ou éléments de style doit être inséré au sein d'une balise de commentaires HTML au bénéfice des navigateurs qui ne prennent toujours pas en charge les feuilles de style.

# 2. Caractéristiques

## 2.1 L'attribut MEDIA

On utilise l'attribut **media** de la balise STYLE pour spécifier le support cible des éléments de style définis à l'intérieur de la balise (ordinateur, synthétiseurs vocaux, ...). On peut décrire des styles spécifiques à chaque support. Le navigateur du support ne prendra en compte que la feuille de style qui lui est destinée. Le tableau ci-dessous résume les valeurs disponibles pour cet attribut.

| Valeur | Information(s) |
|---|---|
| **all** | Pour tous les types de périphériques |
| **screen** | Type classique, pour tous les moniteurs courants |
| **tty** | Type télévision (basse résolution) |
| **projection** | Type projecteur |
| **handheld** | Type PC de poche (Psion, Windows CE, Pilot ...) |
| **print** | Type impression (pour les imprimantes) |
| **braille** | Pour les dispositifs de représentation en braille |
| **aural** | Pour les synthétiseurs vocaux |

L'exemple suivant concerne un PC de poche :

```
<STYLE TYPE="text/css" MEDIA="handheld">
<!--
H2 { font-size: 10pt; color: black }
//-->
</STYLE>
</HEAD>
```

## 2.2 Eléments de style HTML

La définition et la syntaxe de base d'un élément de style sont simples :

**balise { propriété 1 de style: valeur 1; propriété 2 de style: valeur 2 }**

Exemple :

```
H3 { font-family: Arial; font-style: italic }
```

Dans cet exemple, la balise "H3" est définie avec la police Arial et le style "italique". En conséquence, toutes les balises <H3> du document utiliseront cette police et style en question, exceptées si l'on redéfinit les propriétés de style au niveau d'une balise donnée (voir paragraphe 3.2).

A priori, cela reste simple! Mais de nombreux commentaires sur les conditions d'emploi et la syntaxe s'imposent :

- Les feuilles de style portent principalement sur des balises HTML courantes comme : Hn, P, BODY, A, DIV, OL et UL, FORM et INPUT et SELECT et TEXTAREA, IMG, TABLE, TFOOT et THEAD, DL, etc...
- Elles peuvent utiliser d'autres éléments comme par exemple "A:link" pour un lien non-visité et "A:visited" pour un lien visité, ou encore "A:hover" pour un lien survolé (le survol est aussi possible pour du texte, comme par exemple "P:hover" si l'on veut changer de style lorsque l'utilisateur survole un paragraphe donné).
- Les éléments de style (ou sélecteurs) sont entourés par des "**{**" et non par des [ ou des parenthèses.
- La propriété de style est séparée de sa valeur par un double-point ":".
  - Chaque couple "propriété de style/valeur" est séparé du couple suivant par un point-virgule ";".
  - Les valeurs peuvent être précédées d'appels à des fonctions comme la fonction url ou rgb.
- Le caractère "espace" entre la propriété de style et la valeur n'est pas obligatoire mais aide fortement à la lisibilité du code source.
- Pour la lisibilité toujours, il est conseillé d'écrire les styles sur plusieurs lignes :

```
H3 {
font-family: Arial;
font-style: italic;
font-color: green
}
```

- On peut attribuer plusieurs valeurs à une même propriété. Dans ce cas, on séparera les différentes valeurs par des virgules.

```
H3 {font-family: Arial, Helvetica, sans-serif}
```

- On peut attribuer un même style à plusieurs balises (séparées par des virgules).

```
H1, H2, H3 {font-family: Arial; font-style: italic}
```

- Les propriétés peuvent elles même être regroupées, peut -être dans un certain ordre :

```
H1 { font: bold 10pt/12pt time }
```

- Quand le séparateur est un ou plusieurs **espaces**, la propriété n'affecte une balise que lorsqu'elle apparaît dans l'autre. La règle suivante modifie le comportement de la balise EM, **uniquement quand elle est à l'intérieur** d'un bloc défini par H3:

```
H3 EM { color:red; font-size:larger }
```

Les différentes propriétés de CSS1.0 sont énumérées dans la dernière section du document.

## 2.3 Les longueurs

Les valeurs exprimant des longueurs positives représentent des valeurs relatives (en pourcentage) ou des valeurs absolues.

Les valeurs relatives peuvent être :

| Valeur | Exemple | Explication |
|---|---|---|
| **em** | 0.5em | pourcentage par rapport à la taille de la police |
| **ex** | 1ex | pourcentage par rapport à la taille de la lettre x |
| **px** | 12px | pourcentage par rapport à la résolution d'une image en pixel |
| **%** | 50% | pourcentage par rapport à la totalité de l'espace |

Dans l'exemple suivant, *text-indent* désigne 3 x 12 = 36 points :

```
BODY { font-size: 12pt; text-indent: 3em; }
```

**Les valeurs absolues peuvent être :**

- **pt** : les points
- **in** : inches ou pouces (2,54cm)
- **cm** : les centimètres
- **mm** : les millimètres
- **pc** : les picas

# 2.4 Appels aux fonctions

Les valeurs peuvent être précédées d'appels à des fonctions comme la fonction url( ) pour chercher un fichier image ou rgb( ) pour désigner d'une autre façon la couleur, rect( ) pour dessiner une bordure rectangulaire autour d'un texte (norme CSS2.0).

## 2.4.1 Les couleurs

Il est possible d'employer des noms de couleurs prédéfinis. Leur nombre débute à 16 : aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white et yellow.

La couleur peut aussi être désignée par sa valeur hexadécimale selon le principe RGB précédé du caractère "#".

La couleur peut enfin être représentée par l'appel de la fonction **rgb** fonctionnant suivant le même principe. Elle renseigne sur les valeurs des couleurs élémentaires rouge, vert et bleu. Les valeurs sont soit des nombres absolus du style rgb(10,20,255) ou des nombres relatifs exprimés en pourcentage, par exemple rgb(10%,20%,70%).

## 2.4.2 Les URL

Les adresses URL sont généralement utilisées pour insérer des images d'arrière-plan ou des puces graphiques. Elles **doivent** être précédées de l'appel de fonction **url** selon la syntaxe ci-dessous :

```
BODY { background: url(http://www.ungi.com/gif/caution.gif) }
```

**Les adresses URL relatives sont interprétées par rapport à la source de la feuille de style et non pas par rapport à la source du document.**

# 2.5 Opérateurs sur les valeurs

Le concept des styles autorise l'indication de plusieurs valeurs à une propriété. Le tableau donné ci-dessous renseigne sur les conventions à adopter :

- **propriété** : valeur1 | valeur2 | valeur3 : *propriété* peut prendre l'une des trois valeurs

- **propriété** : valeur1 | valeur2 | valeur3 {1,4} : *propriété* peut prendre l'une des trois valeurs de un à quatre fois.
- **propriété** : valeur1 || valeur2 || valeur3 : les valeurs *valeur1*, *valeur2* et *valeur3* peuvent être présentes simultanément

# 3. Styles internes

La première utilisation des styles et la plus simple consiste à incorporer la balise style à l'intérieur d'une page, d'où l'appellation de "Styles internes".

Le développeur de site peut indiquer d'entrée de jeu au serveur Web d'hébergement comme au navigateur du client que ses pages utilisent le principe des styles grâce à l'insertion d'une balise META :

```
<META HTTP-EQUIV="Content-Style-Type" content="text/CSS">
```

Comme il existe d'autres langages de style, il s'agit par cette balise d'indiquer lequel est utilisé par défaut dans la page.

## 3.1 A l'intérieur de l'entête

Le procédé de loin le plus courant est d'incorporer cette balise dans l'en-tête de la page. Il est aussi le plus logique parce que l'essence même des feuilles de style est de séparer les éléments de mise en forme du contenu.

```
<HTML>
<HEAD>
<STYLE type="text/css">
<!--
La ou les feuille(s) de style
-->
</STYLE>
</HEAD>
<BODY>
```

La balise **<STYLE>** avertit le navigateur de l'utilisation des feuilles de style.

L'attribut **type="text/css"** informe que :

- le style décrit est à appliquer sur du texte,
- il s'agit de **cascading style sheets** (css).

La balise HTML de commentaires **<!-- ... -->,** insérée à l'intérieur de l'élément style, est destinée aux navigateurs qui ne reconnaissent pas les feuilles de style, pour les empêcher

d'interpréter les éléments de style décrits. Les informations situées à l'intérieur de la balise de commentaires seront ainsi ignorées par ces navigateurs.

Les commentaires spécifiques des feuilles de style reprennent la convention désormais classique de **/* commentaires */** appartenant aux langages de script ou de programmation (C, C++, Javascript...).

## 3.2 A l'intérieur du corps de page

Il est possible d'utiliser, au coup par coup, les feuilles de style dans le corps (BODY) du document. Cette façon de faire est peu conforme à l'esprit des feuilles de style qui est de définir un style déterminé global au document. Mais cette possibilité de spécifier une feuille de style intraligne existe pour quelques utilisations particulières...

```
<HTML>
    <BODY>
    <H1 style="font-family: Arial; font-style: italic"> mon titre de page </H1>
    </BODY>
</HTML>
```

Signalons:

- que le style "Arial, italique" n'affectera que cette seule balise H1.
- que la syntaxe est légèrement différente de la précédente.

La variante syntaxique qui suit fonctionne tout aussi bien.

```
<STYLE type="text/css">H1 { "font-family: Arial; font-style: italic" }</STYLE>
```

**Exemple d'emploi :**

Le développeur de sites peut être amené à définir non seulement un style global à la page (balise style dans l'en-tête), mais aussi un style unique spécifique à un titre donné ou autre. Au lieu de décrire ce deuxième style, appliqué une fois seulement, au niveau de l'en-tête, le concepteur aura tendance à le mettre en place au niveau de la balise elle-même.

# 4. Styles externes

Pouvoir définir une présentation de style valable pour toute une page (styles internes) est déjà très intéressant. Mais les fonctionnalités de CSS nous proposent mieux : définir une présentation de style valable pour plusieurs pages si ce n'est pour toutes les pages d'un site.

Ceci est possible en créant une page externe (ou feuille de style) regroupant toutes les éléments de style désirés.

Cette page externe est reliée par un lien à chaque page du site.

# 4.1 Fichier CSS

On crée d'abord, un répertoire du site nommé *styles* par exemple, puis un fichier texte avec l'extension **.css** soit *styles.css* qui contiendra tous les éléments de style.

Ce fichier peut être édité à l'aide d'un éditeur de texte simple comme Bloc-Notes, ou à l'aide d'éditeurs spécialisés comme Topstyle de Bradsoft, ou encore à l'aide d'éditeurs HTML avancés. Il se présente simplement sous la forme d'un fichier texte comme dans l'exemple ci-dessous.

```
h1 {  font-family: Arial, Helvetica, sans-serif; font-size: large; line-height: normal;
font-weight: bold; color: #FFFFFF; text-decoration: none; text-align: left; clip:  rect(  )}

h2 {  font-family: Arial, Helvetica, sans-serif; font-size: medium; line-height: normal;
font-weight: bold; color: #FF3333; text-decoration: none; text-align: left; }

h3 {  font-family: Arial, Helvetica, sans-serif; font-size: medium; margin-left:1cm ;
font-weight: bold; color: #FFFFFF; text-decoration: none; text-align: left; }

h4 {  font-family: Arial, Helvetica, sans-serif; font-size: medium; margin-left:2cm ;
font-weight: bold; color: #FFFFFF; text-decoration: none; text-align: left; }

h5 {  font-family: Arial, Helvetica, sans-serif; font-size: medium;margin-left:3cm ; line-
height: normal; font-weight: bold; color: #FFFFFF; text-decoration: none; text-align: left; }

h6 {  font-family: Arial, Helvetica, sans-serif; font-size: medium;margin-left:4cm ; line-
height: normal; font-weight: bold; color: #FFFFFF; text-decoration: none; text-align: left; }

p {font-family: Arial, Helvetica, sans-serif; margin-left:1cm; margin-right:1cm;
text-align:justify}

ul {font-family: Arial, Helvetica, sans-serif; margin-left:1cm; margin-right:1cm;
text-align:justify; font-size: medium}

ol {font-family: Arial, Helvetica, sans-serif; margin-left:1cm; margin-right:1cm;
text-align:justify; font-size: medium}
```

# 4.2 Appel du fichier CSS

## Balise LINK

Ensuite, une page Web du site, soit *page1.htm*, située pour simplifier dans le même répertoire que le fichier *styles.css*, établit un lien avec ce fichier de style en utilisant la balise **link**.

Le but de ce paragraphe n'est pas de passer en revue toutes les possibilités de cette balise, mais d'expliciter uniquement son emploi spécifique aux feuilles de style.

```
<HTML>
<HEAD>
      <LINK rel="stylesheet" type="text/css" href="styles.css">
</HEAD>
```

Quelques commentaires s'imposent :

- La balise **<LINK>** avertit le navigateur qu'il faut réaliser un lien.

- L'attribut **rel="stylesheet"** (sans "s") précise que la page courante importe une feuille de style externe.
- L'attribut **type="text/css"** précise que le format des données importées est du texte et du genre css (cascading style sheets).
- L'attribut classique de lien **href** donne l'adresse URL de la feuille de style externe ou le chemin d'accès et le nom du fichier à lier si ce dernier est présent sur le site.

Si l'attribut **rel** est présent, l'attribut **href** est obligatoire, alors que l'attribut **type** est optionnel.

## Autres moyens

Pour appeler un fichier externe, une autre possibilité existe avec la balise **style** grâce à la directive **@import**. Voici un exemple d'importation combinant à la fois une feuille externe, et complétant le style par un nouvel élément pour la page en question.

```
<HEAD>
<STYLE TYPE="text/css">
@import url(http://....);
H1 { color: blue }
</STYLE>
</HEAD>
```

La syntaxe de la directive demande à ce que la ligne se termine par un point-virgule.

**Remarque :**

En dehors des fichiers à extension *.css, il est tout aussi possible de charger de simples fichiers textes ayant pour extension *.txt.

**Exemple**

```
<style><!--
@import "style/style-web.txt";
--></style>
```

**Remarque :**

A travers l'avant-dernier exemple, l'on voit qu'il est possible de définir et **de composer des feuilles de style à trois niveaux** : des feuilles de style externes pour tout ou partie des pages d'un site, des feuilles de style internes (ou éléments de style) pour une page donnée, un élément de style spécifique pour une balise donnée.

**Note :**

Une directive **!important** permet d'augmenter l'importance d'une valeur par rapport aux précédentes dans le cas de styles en cascade. La syntaxe consiste à l'accoler à la valeur en question.

# 5. Eléments de style CSS

## 5.1 Notion de classes

Le concepteur de site peut désirer affecter des styles différents à une même balise. Plutôt que de définir un style global à la page puis le redéfinir à un endroit particulier de la page, le concept des feuilles de style nous propose la solution des **classes [class]**.

La syntaxe reste aussi simple. La définition d'un style était :

**balise { propriété de style: valeur }**

Elle devient :

**balise.nom_de_classe { propriété de style: valeur }**

Remarquons le point entre "balise" et "nom_de_classe".

La mention de la balise est facultative. Dans ce cas, la classe de style ainsi définie peut être sollicitée par n'importe quelle balise.

**.nom_de_classe { propriété de style: valeur }**

Attention ! L'emploi du point (.) devant le nom de classe est indispensable.

Pour appeler l'effet de style dans le document, on ajoute le nom de la classe à la balise appelante.

**<balise class="nom_de-classe"> .... </balise>**

Proposons un exemple où l'on souligne l'importance des mots en les mettant en gras et en appliquant la couleur bleu. Pour cela, créons d'abord la classe *.essentiel* :

```
.essentiel { font-weight: bold; color: #000080 }
```

Ensuite, dans le document HTML, il suffit d'appeler la classe quand cela se révèle nécessaire :

```
<P class="essentiel"> ... blabla ... </P>
<H1 class="essentiel">Titre 1</H1>
<TABLE><TR><TD class="essentiel">cellule</TD></TR>...
```

Cela donne le résultat suivant :


Un deuxième exemple pris sur un site donne un éventail des possibilités offertes par les classes.

```
.Texte { font-variant: normal; font-size: 11px; font-family: Verdana; color: #000000; }
.Texte:hover { font-variant: normal; font-size: 11px; font-family: Verdana; color: #FF0000; }
.Titre {font: 17px Verdana; color: #000000; font-weight: bold; }
.Hypertext {font: 24px Verdana; color: #8997D4; font-weight: bold; text-decoration: none}
.Hypertext:hover {font: 24px Verdana; color: #8997D4; font-weight: bold; text-decoration: underline overline}
.LI    {font: 13px Verdana; color: #000000; text-decoration: none}
```

## 5.2 Notion des ID

Comme la convention nom/point/nom est utilisée aussi en Javascript, il a fallu trouver une autre convention d'écriture lorsqu'on désire utiliser les feuilles de style avec des scripts du langage Javascript. Ce sont les attributs **ID**, aussi appelés les **identifiants**.

Les attributs **ID** fonctionnent exactement comme les attributs **classes** : c'est la même chose!

La syntaxe de déclaration est :

**#nom_de_ID { propriété de style: valeur }**

La syntaxe d'appel dans la page est :

**<balise id="nom_de_ID"> .... </balise>**

Si le concepteur de site pense utiliser des feuilles de style sans scripts, il est plutôt conseillé d'utiliser les attributs classes.

Si par contre, le concepteur de site souhaite associer des scripts aux feuilles de style pour faire du DHTML par exemple, la notion d'attribut ID est alors indispensable.

# 6. Morceaux de style

## 6.1 Utilité

Si le concepteur de site veut appliquer un effet de style à certains **morceaux de paragraphe** ou à **plusieurs paragraphes**, sans devoir repasser par les éléments structurels du langage HTML classique, les balises **SPAN** et **DIV** insérées au sein d'un document permettent de réaliser respectivement l'un et l'autre. Les balises DIV et SPAN permettent aussi de contrarier certaines instructions de la feuille de style en appliquant un style différent pour quelques morceaux de la page.

Ces balises s'utilisent toujours avec les attributs classes et ID.

## 6.2 Balise SPAN

La balise **<SPAN>** (norme HTML4.0, Netscape 4 et I. Explorer 3.0) permet d'appliquer des styles à des éléments de texte d'un paragraphe ou à un morceau de paragraphe.

Le concepteur dans l'exemple qui suit veut attirer l'attention de l'internaute sur un (groupe de) mot(s) :

Un monde de **géants**.

```
<HTML>
 <HEAD>
      <STYLE type="text/css">
      .element {font-size: x-large; color:  navy}
      </STYLE>
 </HEAD>
 <BODY>
      <P>Un monde de <SPAN class="element">géants</SPAN>.</P>
 </BODY>
</HTML>
```

## 6.3 DIV

La balise **<DIV>** permet de délimiter une zone comportant plusieurs paragraphes.

```
<HTML>
 <HEAD>
      <STYLE type="text/css">
      .zone{font-size: x-small}
      </STYLE>
 </HEAD>
 <BODY>
      La balise <DIV>
```

```
      <DIV class=zone>
      <P>Commentaire :</P>
      <P>N'oubliez pas l'attribut class!</P>
      </DIV>
  </BODY>
</HTML>
```

Le code donne comme résultat :

La balise <DIV>

Commentaire :

N'oubliez pas l'attribut class!

# 7. Mini FAQ sur les styles

## 7.1 Les feuilles de style sont-elles "case sensitive" ?

Les feuilles de style ne sont pas sensibles à la casse (case insensitive). Ecrire CLASS ou class ou Class est donc équivalent.

Cependant les valeurs des propriétés comme les noms de police ou les URLs peuvent être "case sensitive".

Pour le système d'exploitation Unix, *Arial* n'est pas égal à *arial*, de même *IMAGE.gif* n'est pas forcément égal à *image.gif*.

## 7.2 Quels caractères peut-on utiliser en CSS ?

Les noms des feuilles de style, des sélecteurs, des classes et ID peuvent contenir l'ensemble des lettres minuscules ou majuscules, les chiffres, le trait d'union et le caractère "_" ou souligné. Les noms ne peuvent commencer par un chiffre ou un tiret.

La documentation officielle affirme que les caractères spéciaux ASCII 160-255 peuvent être utilisés, mais cela ne fonctionne pas toujours. Aussi, le concepteur prendra vite l'habitude de les éviter.

## 7.3 Peut-on utiliser dans la même page, à la fois des éléments de style et des propriétés d'éléments HTML ?

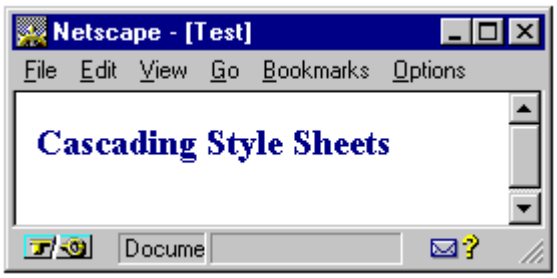
Les feuilles de style externes seront ignorées par les navigateurs qui ne supportent pas les feuilles de style, alors que les attributs de style du langage HTML le seront. Pour assurer une portabilité maximum du site, il est vivement conseillé d'utiliser les deux.

## 7.4 Qui l'emportent : les attributs HTML ou les propriétés CSS ?

Les attributs de style purement HTML sont traités de manière égale avec les propriétés des feuilles de style. La règle qui prévaut reste comme pour la question suivante, la **règle de proximité**. C'est la définition de style la plus proche de la cible qui est prise en compte.

Ainsi l'exemple suivant apparaîtra en bleu si l'on intègre le code source tel quel dans la page. L'exemple apparaîtra en rouge si l'élément H3 est imbriqué dans la balise <font>, donc au plus près de cette dernière.

```
<H3 style="color: red"><B><FONT COLOR="navy">Cascading Style
Sheets</FONT></B></H3>
```




## 7.5 Concurrence entre plusieurs éléments de style

En cas de concurrence entre plusieurs éléments de style, intervient alors la notion de "cascade" ou d'ordre de priorité.

La concurrence entre plusieurs éléments de style peut provenir des différentes localisations des éléments de style :

- le fichier externe,
- l'en-tête du document,
- le corps du document.

Elle peut aussi provenir de la succession d'éléments de style HTML destinés au même élément HTML.

La règle de priorité définie au niveau du navigateur sera d'appliquer, pour la présentation du document, **la feuille de style la plus proche de l'élément**. Ainsi, un style spécifié au niveau de l'élément BODY sera retenu par rapport à un style déclaré dans un fichier externe.

On peut, au contraire, profiter de cette règle de proximité pour qu'un élément de style contienne des propriétés dont les valeurs sont héritées des précédents éléments, ou dont les valeurs modifient une valeur précédemment définie (par exemple la valeur ***larger*** ou ***smaller***).

# 8. Liste des propriétés

Ces propriétés de style sont naturellement regroupées par type de modification. Elles sont à l'image de l'organisation des éditeurs HTML avancés ou des logicels spécialisés.

Dans un premier temps, sont regroupés les styles de police (**famille font**), puis les styles de texte (**famille text** principalement).

Dans un deuxième temps, sont rassemblées les propriétés concernant l'arrière-plan (**famille background**), auxquelles font suite celles des marges avec la **famille margin** pour la mise en page.

Dans un troisième temps, sont énumérées les propriétés des bordures et espacements (respectivement les **familles border** et **padding**) au profit de textes ou d'éléments de tableaux, ou d'images.

Dans un quatrième temps, ce sont les propriétés des listes (**famille list-style**) qui sont passées en revue.

Enfin, ce sont des propriétés d'alignement relatif entre les objets qui terminent cette liste exhaustive.

Par convention d'écriture, les valeurs par défaut sont indiquées en gras.

## 8.1. Les styles de police

La famille complète de ces propriétés, exceptée la propriété font-variant pour Netscape, est interprétée à partir des versions 4 des navigateurs Netscape et Internet Explorer.

Toutes les valeurs de cette famille de propriétés, définies dans un élément de style donné, sont attribuées en héritage aux éléments de style suivants, à moins d'être redéfinies dans ces derniers.

**font-family**

- définit un nom de police ou une famille de polices **<nom>** ou **<famille>**
- police précise (Arial, Times, Helvetica...) ou famille de polices (serif, sans-serif, cursive, fantasy, monospace)
- Si un nom complet comprend des espaces, il doit être côté.
- Cette propriété admet plusieurs valeurs séparées par une virgule, à l'image de l'attribut *face* de l'élément HTML *font*.
- Lorsqu'on spécifie une liste, le navigateur vérifie la présence de la première police listée sur la station de l'internaute, sinon passe à la seconde, et ainsi de suite.
- exemple : BODY {font-family: "Century Schoolbook", Times, serif}

**font-style**

- définit le style de l'écriture (normal ou italique)

- valeurs : **normal**, italic ou oblique
- exemple : H3 {font-style: italic}

**font-weight**

- définit l'épaisseur de la police
- valeurs :
  - mots prédéfinis : **normal** ou bold ou bolder ou lighter (les deux dernières sont à utiliser dans le cas de styles en cascade)
  - valeur numérique : 100, 200, 300, 400, 500, 600, 700, 800, 900.
    - Elles permettent de mieux graduer l'épaisseur en fonction de l'effet désiré.
- exemple : P {font-weight: bold}

**font-size**

- définit la taille de la police
- valeurs :
  - mots prédéfinis : xx-small ou x-small ou small ou **médium** ou large ou x-large ou xx-large (ces valeurs correspondent respectivement aux tailles 1 à 7 du langage HTML), ou larger ou smaller
  - valeur numérique :
    - taille précise en points (pt), inches (in), centimètres (cm), pixels (px), millimètres (mm), pica (pc)
    - taille relative en pourcentage (%), em ou ex
- exemple : P {font-size: 12pt}

**font-variant**

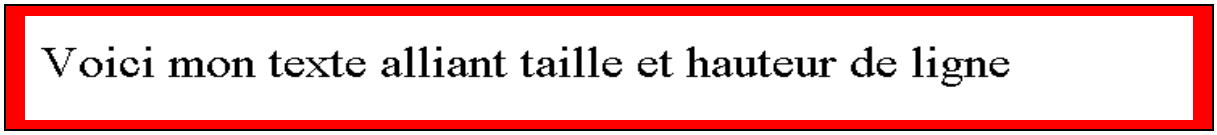
- définit une variante par rapport à la normale
- valeurs : **normal** ou small-caps (petites lettres capitales ou petites majuscules)
- exemple : P {font-variant: small-caps}

**font**

- raccourci pour les différentes propriétés de police à utiliser dans l'ordre suivant :
  - [ <font-style> || <font-variant> || <font-weight> ]? <font-size> [/<line-height>]? <font-family>

- Il est possible de combiner les caractéristiques d'une police et celle de la hauteur de ligne réservée (espace supérieur). La propriété line-height est commentée dans le paragraphe suivant.

- exemple : P { font: x-large/110% "new century schoolbook", serif }




## 8.2. Les styles du texte

La famille complète de ces propriétés est interprétée à partir des versions 4 des navigateurs Netscape et Internet Explorer. Toutes les valeurs de cette famille de propriétés peuvent être attribuées en héritage.

Dans le cas contraire, l'exception sera mentionnée.

**text-align**

- définit l'alignement du texte (attribut align du langage HTML)
- valeurs : left, center, right ou justify
  - La dernière valeur, admise par certains navigateurs comme IE, ne fait pas partie des recommandations CSS1.0.
- exemple : H1 {text-align: center}

**text-indent**

- définit un **retrait dans la première ligne** d'un bloc de texte
- souvent utilisé avec l'élément **p**, il ne faut pas oublier dans ce cas la balise de fermeture </p>.
- valeurs : en inches (in) ou en centimètres (cm) ou en pixels (px), par défaut **0**
- exemple : P {text-indent: 1cm},

**text-decoration**

- définit une décoration du texte : soit barré, soit clignotant, etc.
- valeurs : blink (Netscape) ou underline (souligné) ou line-through (barré) ou overline (surligné ou souligné au-dessus du texte) ou **none**
- les valeurs de cette propriété ne peuvent pas être données en héritage
- exemple : p.old {text-decoration: line-through}

### text-transform

- définit la casse du texte (majuscule, minuscule)
- valeurs : uppercase (majuscules), lowercase (minuscules), capitalize (1er caractère en majuscule) ou **none**
- exemple : P {text-transform: uppercase}

### vertical-align

- spécifie l'alignement vertical du texte par rapport au reste du texte
- valeurs : **baseline** (même ligne de base) | sub (indice) | super (exposant) | top (sommet) | text-top (aligne le haut de l'élément avec l'élément parent)| middle (milieu de l'élément avec la ligne de base) | bottom (bas) | text-bottom (aligne le bas de l'élément avec l'élément parent)
- cette propriété est limitée à Internet Explorer, et n'est pas donnée en héritage.
- exemple : EM {vertical-align: sub}

### color

- définit la couleur du texte
- valeur initiale (**black**), propriété avec héritage.
- exemple : H3 {color: #000080}, P {color: red}, EM {color:rgb(255,0,0)}

### word-spacing

- définit l'espace entre les mots si l'attribut align avec valeur *justify* n'est pas employé
- valeur par défaut (**normal**) ou valeurs numériques exprimées en points (pt), inches (in), centimètres (cm), pixels (px) ou en pourcentage (%)
  - dans le cas d'une valeur par défaut, le navigateur calcule l'espacement.
- exemple : P {word-spacing: 5pt}

### letter-spacing

- définit l'espace entre les lettres
- valeur par défaut (**normal**) ou valeurs numériques exprimées en points (pt), inches (in), centimètres (cm), pixels (px) ou en pourcentage (%)
- propriété absente sous Netscape, sans héritage.
- exemple : P {letter-spacing: 2pt}

**line-height**

- définit l'espace entre les lignes du texte (hauteur de ligne par rapport à l'élément de police de plus haute taille)
- valeurs : **normal**, ou nombre, ou longueur [points (pt), inches (in), centimètres (cm), pixels (px)], ou pourcentage (%)
- exemple : P {line-height: 10pt}

**white-space**

- définit la façon d'afficher le caractère "espace" (espaces insécables et retour chariots)
  - par défaut, le navigateur réduit à un seul espace une suite de caractères de ce type. De même, pour forcer un saut de ligne, il faut insérer l'élément HTML br.
  - propriété absente sous Internet Explorer.
- valeurs : pre, nowrap, **normal**
  - la valeur pre permet de conserver les espaces sans avoir à insérer des espaces insécables.
- exemple : div.example { white-space: pre}

**width**

- détermine la longueur d'un élément de texte ou d'une image
- valeurs par défaut (auto) ou numériques exprimées en points (pt), inches (in), centimètres (cm), pixels (px) ou pourcentage (%)
- exemple : H1 {width: 200px}

**height**

- détermine la hauteur d'un élément de texte ou d'une image

## 8.3 Les arrière-plans

La famille complète de ces propriétés est interprétée à partir de la version 4 des navigateurs Internet Explorer et Netscape. Aucune valeur de cette famille de propriétés **ne** peut être donnée en héritage.

**background-color**

- définit la couleur de l'arrière-plan
- valeurs : couleur (par exemple en hexadécimal) ou **transparent**
- exemple : H1 {background-color: #000000}

**background-image**

- définit l'image de l'arrière-plan
- valeur : URL de l'image précédé de l'appel à la fonction url( )
- exemple : BODY {background-image: url(image.gif) }

**background-repeat**

- définit la façon de répéter l'image d'arrière-plan
- valeurs : **repeat** ou no-repeat ou repeat-x (x = nombre de répétitions horizontales), repeat-y (y = nombre de répétitions verticales)
- exemple : P {background-image: url(image.gif); background-repeat: repeat-4}

**background-attachment**

- spécifie si l'image d'arrière-plan reste fixe avec les déplacements de l'écran
- valeurs : **scroll** ou fixed
- non valide sous Netscape
- exemple : BODY {background-image: url(image.gif) ; background-attachement: fixed}

**background-position**

- spécifie la position de l'image d'arrière-plan par rapport au coin supérieur gauche de la fenêtre
- valeurs : {**0%,0%**}
  - mots réservés : {top, center, bottom}, {left, center ou right}
  - valeurs numériques absolues exprimées en points (pt), inches (in), centimètres (cm), pixels (px), soit par exemple {1, 2}
  - valeurs numériques en pourcentage (%), soit par exemple {20%,30%}
- exemple : BODY {background-image: url(img.gif); background-position: right top}

- deux valeurs en pourcentage expriment d'abord la largeur avant la hauteur. Si un seul est présent, il s'applique à la largeur.
- on peut combiner pourcentage et longueur.
- on peut combiner deux mots réservés séparés par un espace, comme dans l'exemple ci-dessus où l'image est positionnée en haut à droite.

**background**

- raccourci pour les différentes propriétés d'arrière-plan
- background : background-attachment background-color background-image background-position background-repeat
- non valide sous Netscape
- exemple : P {background: url(image.gif) fixed repeat}

## 8.4 Les marges

La famille complète de ces propriétés est interprétée à partir de la version 4 des navigateurs Internet Explorer et Netscape. Aucune valeur de cette famille de propriétés **ne** peut être donnée en héritage.

**margin-top**

- détermine la valeur de la marge supérieure
- valeur par défaut (**auto**) ou valeurs numériques, en unités de longueur ou un pourcentage de la marge de l'élément parent immédiat
- exemple de marge absolue : { margin-top: 5px }

**margin-right, margin-bottom, margin-left**

- détermine de la même façon respectivement la valeur de la marge droite, de la marge inférieure, de la marge gauche

**margin**

- regroupe les différentes propriétés de la marge
- valeur initiale : 0
- exemple : BODY { margin: 1em 2em 3em 2em }
  - les marges sont indiquées dans l'ordre : supérieure=1em, droite=2em, inférieure=3em, gauche=2em.
  - si une seule est mentionnée, la même marge est appliquée sur les quatre bords.
  - si deux sont mentionnées, la première marge est appliquée sur les bords supérieurs et inférieurs, la seconde sur les autres.

- si trois sont mentionnées, la première marge est appliquée sur le bord supérieur, la seconde sur les bords droit et gauche, la dernière sur la marge inférieure.

## 8.5 Les bords et les "enrobages"

La famille complète de ces propriétés est interprétée à partir de la version 4 des navigateurs Internet Explorer et Netscape. Aucune valeur de cette famille de propriétés ne peut être donnée en héritage.

**border-top-width, border-right-width, border-bottom-width, border-left-width**

- donne respectivement l'épaisseur du bord supérieur, du bord droit, du bord inférieur, du bord gauche d'un texte ou autre
- valeurs :
  - mots réservés : thin ou **medium** ou thick
  - valeurs numériques abolues ou relatives
- exemples : H3 {border-top-width: thin}, H1 { border-top-width: 0.5em }

**border-width**

- regroupe les différentes propriétés de border-width
- exemple : H4 { border-width: thin medium thick medium }
  - les marges sont indiquées dans l'ordre : supérieure=1em, droite=2em, inférieure=3em, gauche=2em.
  - les même règles que celles des marges s'appliquent en l'absence d'une dimension.

...blabla...

**border-color**

- détermine la couleur de la bordure à l'aide des mots réservés ou de la fonction rgb( ) ou du code #RVB.
- Netscape n'accepte qu'une seule valeur pour les quatre bordures, alors que Internet Explorer et la norme CSS autorisent quatre valeurs différentes séparées par un espace, ceci quel que soit le mode utilisé.
- exemple :
  - H3 {border-color: yellow},
  - DIV { border-color: red rgb(0,0,255) red}

**border-style**

- détermine le style du trait de la bordure
- valeurs : **none,** solid, dotted (pointillé), dashed (tirets), double, groove (crevassé), ridge (crête), inset (incrusté) ou outset (en relief)
- exemple : P {border-color: red brown red; border-style: solid dashed ridge outset}
  - plusieurs valeurs peuvent être indiquées. Les même règles que celles des marges s'appliquent.

Exemple de couleur et de style de bordure !

**border-top, border-right, border-bottom, border-left**

- regroupe toutes les propriétés des bordures en question dans l'ordre suivant : épaisseur, style et couleur
- valeur par défaut: aucune
- H1 { border-top: thick }

**border**

- regroupe toutes les propriétés des bords
- un attribut destiné à un bord donné a priorité sur un attribut global aux bordures dans la mesure où il est défini après.

**padding-top, padding-right, padding-bottom, padding-left**

- valeur de remplissage entre l'élément HTML et le bord désigné
- valeurs numériques en points (pt), inches (in), centimètres (cm), pixels (px) ou pourcentage (%),
- valeur nulle par défaut
- exemple : H3 {padding-top: 3px}

**padding**

- regroupe les différentes propriétés de remplissage
- plusieurs valeurs peuvent être indiquées. Les même règles que celles des marges s'appliquent.
- exemple : **H1** { padding: 1em 2em }

## 8.6 Les listes

La famille complète de ces propriétés est interprétée à partir de la version 4 des navigateurs Internet Explorer et Netscape. Toutes les valeurs de cette famille de propriétés sont attribuées en **héritage**.

Ces propriétés s'appliquent non seulement aux éléments OL ou UL et LI, mais aussi aux éléments DT et DD.

**list-style-type**

- détermine le type de puces dans le cas d'une liste non numérotée (ou liste simple) ou le type de numérotation dans le cas d'une liste numérotée
- valeurs :
  - liste simple : **disc** ou circle ou square
  - liste ordonnée : decimal ou lower-roman ou upper-roman ou lower-alpha ou upper-alpha
- exemple : OL {list-style-type: square}

**list-style-image**

- permet de remplacer les puces par une image
- valeurs : url ou **none**
- exemple : OL {list-style-image: url(image.gif)}

**list-style-position**

- spécifie si les puces sont à l'intérieur ou à l'extérieur du texte
- valeurs : inside ou **outside**
- exemple : UL {list-style-position: inside}

**list-style**

- regroupe toutes les propriétés de liste dans l'ordre type, position puis image si besoin
- exemple : UL { list-style: upper-roman inside none}

## 8.7 Mise en page

La famille complète de ces propriétés est interprétée à partir de la version 4 des navigateurs Internet Explorer et Netscape. Toutes les valeurs de cette famille de propriétés sont attribuées en **héritage**.

**float**

- permet de définir l'habillage de l'élément.
- valeurs : left (élément à gauche, le reste à droite), right (élément à droite, le reste à gauche), **none** (pas d'habillage)
- exemple :

```
  <span style="float:right">Texte habillé</span>
<span style="color=green">Texte habillant le texte par la gauche</span>
```

Texte habillant le texte par la gauche Texte habillé

**clear**

- permet d'annuler l'habillage d'un élément (***propriété float***) par du texte et de déterminer comment poursuivre la présentation.
  - pour empêcher l'élément courant d'être affiché sur la même ligne que l'élément flottant précédent, il faut employer la propriété ***clear*** avec la même valeur, dans le doute avec la valeur ***both***.
  - La propriété clear ainsi positionnée, l'élément courant se placera sur la première ligne disponible sous l'élément flottant.
- valeurs : left, right, both , **none** (pas d'habillage)
- exemple : l'emploi de cette propriété fait suite à celle de la propriété **float**

```
<span style="float:left; width:100 px">
<img src="tdm.gif">
</span>
<span>Le texte habille l'image par la droite</span>
<span style="clear:left"> et continue sous l'image</span>
```


Le texte habille l'image par la droite
et continue sous l'image

**display**

- détermine si l'élément doit être affiché, et si oui, la place qui lui est réservée (type de boite).
  - A ne pas confondre avec la propriété **visibility** de CSS2.0 qui réserve un espace même si ce dernier n'est pas affiché.

- valeurs : block (boite de type bloc), **inline** (boîte intra-ligne), list-item (boîte de type bloc incluant une boîte de puces), none (pas de boîte, contenu invisible)
  - Avec CSS2.0, d'autres valeurs viennent compléter cette liste sommaire.
- exemple :

```
<p style="display:none">Texte non visible</P>
<p style="display:block">Texte visible dans un bloc</P>
```

Texte visible dans un bloc

# Conclusion

Même si l'étude des feuilles de style telle qu'elle a été exposée peut paraître fastidieuse, les éditeurs HTML avancés tels que Frontpage ou Dreamweaver proposent des menus conviviaux susceptibles d'intégrer ces éléments dans les pages Web, aussi bien en styles internes qu'en styles externes, mais aussi au niveau des balises sélectionnées au préalable.

Les feuilles de style représentent une étape importante dans l'étude des techniques d'élaboration de sites Web. Elles préparent conceptuellement le développeur à séparer le contenu de la page (l'information) de la mise en forme (présentation de l'information) selon les principes du langage XML.

Dans un deuxième temps, elles préparent techniquement le développeur qui maîtrise toutes les techniques du langage HMTL V3.2 à évoluer vers la version 4.0 du même langage, voire le DHTML.

# Complément 4 : Langage HTML V4.0

## Introduction

La version 4.0 du langage HTML est sortie en juillet **1997**, six mois seulement après la normalisation de la version 3.2. Hélas, en dehors de légers additifs survenus en décembre 1999 (v4.01), c'était la **dernière version** de ce langage.

Le langage HTML, côté **interopérabilité**, s'est attaché à prendre en compte chaque machine susceptible d'exploiter l'information présente sur le Web : PCs dotés d'écrans de résolutions et systèmes de couleurs distincts, téléphones cellulaires, portables, appareils à interface vocale, ordinateurs bas débits, etc.

L'**internationalisation** du langage a été obtenue par l'incorporation de la **RFC2070** traitant du sujet, et l'adoption de la norme **ISO10646** définissant le codage de caractères pour les documents HTML.

Le langage HTML a continué ses efforts pour rendre les pages Web plus accessibles aux personnes subissant certaines déficiences physiques.

Le nouveau modèle des tableaux HTML s'appuie sur le document **RFC1942**.

Le langage HTML dispose désormais d'un mécanisme standard pour inclure des objets multimédia génériques ainsi que des applications dans des documents HTML. L'élément OBJECT (conjointement à ses ancêtres plus spécialisés IMG et APPLET) procure un mécanisme pour inclure dans un document des images, de la vidéo, du son, des formules mathématiques, des applications spécialisées, et d'autres objets. Il permet de plus de hiérarchiser les diverses représentations d'un même objet à destination d'agents utilisateurs de nature et de capacités différentes.

Le but de ce chapitre est de passer en revue les nouveautés techniques apportées par le langage HTML 4.0 et de démontrer leurs réels apports dans le développement de pages web :

- rendre systématique la présence d'un prologue HTML (voir balise !DOCTYPE dans le paragraphe suivant) renseignant sur le modèle de document utilisé, ainsi que l'indication de la langue (attribut lang)
- signifier les versions de documents (éléments ins et del accompagnées de l'attribut datetime)
- pour les sites à vocation internationale, gérer le sens de l'écriture (attribut dir et élément bdo)
- associer systématiquement du texte à des éléments HTML (attribut title, élément acronym)
- améliorer la gestion des tableaux en structurant les lignes et les colonnes (thead,…, colgroup), en spécifiant une mise en forme de ces éléments, en détaillant la présentation de ces derniers (bordures et gouttières)
- améliorer les formulaires grâce aux boutons à image incorporée (élément button), aux étiquettes pour nommer les champs des formulaires ou groupes de champs (éléments label et legend)

- permettre une plus grande indépendance entre le langage HTML et le navigateur en activant les éléments de formulaire par les raccourcis clavier (attributs tabindex et accesskey) ou par des scripts (attributs readonly et disabled).
- insérer des éléments de style dans 70 éléments HTML (attributs style, class et id) ou remplacer les tableaux par les feuilles de style (cf. complément "Feuilles de style" du même module).

Les éditeurs HTML avancés tels que FrontPage 2000 et Dreamweaver 4.0 ne proposent toujours pas dans leurs menus ces nouvelles possibilités. L'utilisateur doit donc, s'il veut les exploiter, les saisir directement dans le code source.

# 1. Prologue HTML

Tout document suivant la norme HTML doit désormais débuter par l'une des instructions suivantes depuis la version 3.2. Cette instruction prenant le nom de **prologue** est tirée d'une construction **SGML**.

Limité à une seule instruction au sein du langage HTML, il n'est pas indispensable à l'interprétation par le navigateur des éléments de la page. Cela reste une convention.

Le prologue prend toute son importance au sein du langage **XML** ou de son dérivé XHTML. Il devient alors obligatoire et se voit compléter par d'autres instructions.

L'élément DOCTYPE mentionne en final, juste avant le paramètre de langage EN (pour english), la version de la norme de référence : celle en cours de normalisation (Draft), celle qui est finalisée (Final peut alors être omis), ou même la norme la plus stricte.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Draft//EN">
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Final//EN">
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Strict//EN">
```

Le dernier type est approprié pour valider des documents qui n'utilisent aucun élément ni attribut HTML de présentation tels que l'élément FONT ou l'attribut align.

Le dernier mot interne à la balise indique la langue de la **DTD HTML** (vocabulaire XML spécifiant le modèle de référence du document), dans ce cas l'anglais ("EN").

L'exemple ci-dessous impose une validation technique du document par la DTD officielle du site Web W3C. Cette façon de fonctionner est obligatoire dans le cas de documents XHTML ou XML.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN" "http://www.w3.org/TR/REC-
html40/strict.dtd">
```

# 2. Nouveaux attributs de éléments existants

## 2.1 Langue et orientation d'écriture

Il est maintenant possible de spécifier la **langue et l'orientation d'écriture** des documents HTML, respectivement grâce aux attributs **lang** et **dir**.

Ces deux attributs peuvent être utilisés dans bon nombre d'éléments HTML : HTML, TITLE, HEAD, H1..H6, EM, STRONG, DFN, CODE, SAMP, KBD, VAR, CITE, ACRONYM, BLOCKQUOTE, Q, SUB, SUP, P, PRE, etc…

### Attribut LANG

Il peut figurer dans l'entête pour indiquer au navigateur la langue d'utilisation de toute la page : balise META, balise link, etc... Il peut aussi figurer dans le corps de la page au profit d'une section écrite dans une autre langue que celle de la page.

La connaissance de la langue sert bien évidemment :

- aux moteurs de recherche,
- à la sélection de la typographie des polices,
- au choix des caractères de cotation,
- à connaître les emplacements de rupture des mots et les espacements inter-caractères.

Elle peut servir dans une utilisation future à la correction automatique des fautes d'orthographe des pages HTML.

Les valeurs possibles sont : EN, FR, EN-US, DE, AR, RU, JA etc, … FR représente bien sûr le français, EN-US l'américain, RU le russe, AR l'arabe et JA le japonais.

La liste complète des abréviations des langues utilisées des pays peut être consultée dans les documents de référence **RFC1766** et **ISO639**.

### Attribut DIR

L'orientation permet de spécifier si l'écriture de la page se fait de droite à gauche ou de gauche à droite et permet maintenant aux pages de langue arabe de voir le jour sans les artifices de textes en bitmap.

L'attribut correspondant est **DIR**. Les deux valeurs possibles sont **RTL** ou **LTR**, indiquant respectivement la direction d'écriture de droite à gauche (Right To Left) ou de gauche à droite (Left To Right). La valeur par défaut est LTR.

### Remarque

L'élément **BDO** (Bidirectional Override) permet aux auteurs de désactiver l'algorithme bidirectionnel pour certains fragments de texte.

…français1 HEBREU2 français3 **<BDO** dir="LTR">4UERBEH**</BDO>** français5

HEBREU6…

## 2.2 Datage de document

L'attribut **datetime** attribue une heure et une date, par exemple à une nouvelle version d'un document. Le concepteur de pages le rajoute directement dans le code source.

Le format de DATETIME est **AAAA-MM-JJThh:mm:ssTZD** où :

- **AAAA** = année sur quatre digits
- **MM** = mois sur deux digits (01=Janvier)
- **JJ** = jour sur deux digits (01 à 31)
- **hh** = heure sur deux digits (01 à 24)
- **mm** = minute sur deux digits (00 à 59)
- **ss** = seconde sur deux digits (00 à 59)
- **TZD** = zone horaire de la forme 00 (pour UTC), +hh:mm ou -hh:mm pour les décalages horaires

Exemple :

```
<INS datetime="1997-08-05T08:08:30Z"> Ajouté le 1 août 1997 à 8 h 30 Temps universel
</INS>
```

Cet attribut ne s'utilise qu'au sein des éléments HTML **ins** et **del** apparus avec la version 4.0, éléments traités dans la section suivante.

# 3. Nouveaux éléments

Les nouveaux éléments apportés par cette version d'HTML sont Q, INS, DEL, ABBR, ACRONYM, BDO, LEGEND, COLGROUP, BUTTON, FIELDSET, NOFRAMES, NOSCRIPT et SPAN.

Les trois derniers éléments ne sont pas présentés ici. L'élément NOFRAMES est employé de manière conjointe avec l'élément FRAME ou cadre, dans le cas où le navigateur ne sait pas interpréter les cadres. Il en est de même de l'élément NOSCRIPT pour les scripts. L'élément SPAN, permettant d'appliquer un effet de style sur un morceau de texte, est traité dans le chapitre consacré aux feuilles de style.

D'autres éléments introduits par Microsoft ou … ont été normalisés par la version HTML4.0 : IFRAME, OBJECT, etc…

**Remarque** :

La norme HTML4.0 n'est reconnue par Internet Explorer qu'à partir de la version 5.5, et par Netscape à partir de sa version 6.0.

# 3.1 Élément Q

### Description

Les éléments BLOCKQUOTE et Q encadrent du texte cité. La première est destinée aux citations longues alors que la deuxième est plutôt destinée aux citations courtes qui ne contiennent pas de ruptures de paragraphes. L'élément Q est appelé à remplacer à terme l'élément BLOCKQUOTE.

Le texte marqué par <Q> apparaît généralement indenté et placé entre guillemets.

**Remarque** :

La balise Q n'entraîne plus d'indentation comme la balise BLOCKQUOTE, que ce soit sous IE5.5 ou Netscape 6.

### Attribut

| | | | |
|---|---|---|---|
| <Q | | | |
| cite | = | URL | Indique la source de la citation. |
| >…texte à citer… | | | |
| </Q> | | | |

# 3.2 Élément ACRONYM

### Description

L'élément ACRONYM (Internet Explorer 4.0) est utilisé pour signifier que le mot encadré est un acronyme. Une telle aide pour la détection des acronymes est d'une grande utilité pour les correcteurs orthographiques, les synthétiseurs vocaux, et autres outils et agents utilisateurs.

### Attributs

| | | | |
|---|---|---|---|
| <ACRONYM | | | |
| title | = | chaîne | Information affichée lorsque la souris survole l'acronyme.<br>Elle sert à expliciter l'acronyme en question. |
| ID, CLASS, LANG , DIR, STYLE | | | Autres attributs génériques possibles. |
| ONCLICK, ONDBLCLICK, ONMOUSEDOWN, ONMOUSEUP, ONMOUSEOVER, ONMOUSEMOVE, ONMOUSEOUT, ONKEYPRESS, ONKEYDOWN, ONKEYUP. | | | Evènements associables. |
| >…acronyme à citer… | | | |
| </ACRONYM> | | | |

### Exemple

```
<acronym title="acronyme de Un Nouveau Guide Internet">UNGI</acronym>
```

Le contenu d'un élément ACRONYM est l'acronyme lui-même. L'attribut **title** est utilisé pour signaler en info-bulle la signification complète de l'acronyme correspondant.

### Résultat


De même, l'élément **ABBR** indique une forme abrégée (e.g., WWW, HTTP, URI, Mass., etc.). L'emploi de ces deux structures fournit une information utile aux navigateurs et aux outils tels que les moteurs de recherche, les synthétiseurs vocaux et systèmes de traduction.

## 3.3 Élément INS

### Description

L'élément INS (Internet Explorer 4.0) est utilisé pour mettre en valeur des éléments de textes nouveaux par rapport à la version antérieure du document. Cet élément est prévu pour les navigateurs qui sont capables de visualiser dans une police particulière les nouveautés d'un document.

### Attributs

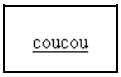
| | | | |
|---|---|---|---|
| <INS | | | |
| datetime | = | date formatée | Voir attribut DATETIME de la seconde section. |
| cite | = | URL | Pointe l'information qui explique pourquoi ou en quoi le document a changé : URL du document ou message source. |
| ID, CLASS, LANG , DIR, STYLE, TITLE | | | Autres attributs génériques possibles. |
| ONCLICK, ONDBLCLICK, ONMOUSEDOWN, ONMOUSEUP, ONMOUSEOVER, ONMOUSEMOVE, ONMOUSEOUT, ONKEYPRESS, ONKEYDOWN, ONKEYUP. | | | Evénements associables. |
| >…nouveau texte inséré… | | | |
| </INS> | | | |

Les principaux attributs utilisés sont la date de modification, un effet de style réalisé directement (style) ou indirectement (class ou id), des évènements associés à des scripts.

### Exemple

```
<ins datetime="2002-04-07T20:38:30Z">coucou</ins>
```

### Résultat




Le texte affiché apparait souligné sous Internet Explorer 5.5 et sous Netscape 6.0.

## 3.4 Élément DEL

### Description

L'élément DEL est utilisé pour mettre en valeur des éléments de textes supprimés par rapport à la version antérieure du document. Cet élément est prévu pour les navigateurs qui seront ainsi capables de visualiser dans une police particulière les parties supprimées d'un document.

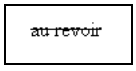
### Attributs

| | | | |
|---|---|---|---|
| <DEL | | | |
| datetime | = | date formatée | Voir attribut DATETIME de la première section. |
| cite | = | URL | Pointe l'information qui explique pourquoi ou en quoi le document a changé : URL du document ou message source. |
| ID, CLASS, LANG , DIR, STYLE | | | Autres attributs génériques possibles. |
| ONCLICK, ONDBLCLICK, ONMOUSEDOWN, ONMOUSEUP, ONMOUSEOVER, ONMOUSEMOVE, ONMOUSEOUT, ONKEYPRESS, ONKEYDOWN, ONKEYUP. | | | Evènements associables. |
| >…ancien texte modifié… | | | |
| </DEL> | | | |

### Exemple

```
<del datetime="2002-04-07T20:38:30Z">au revoir</del>
```

### Résultat




Le texte affiché apparaît barré sous Internet Explorer 5.5 et sous Netscape 6.0.

# 3.5 Élément BDO

L'algorithme bidirectionnel et l'attribut **dir** suffisent en général à gérer les inclusions avec changement de direction d'écriture de texte. Il existe cependant quelques situations dans lesquelles l'utilisation de l'algorithme bidirectionnel provoque une représentation incorrecte.

L'élément **BDO** permet aux auteurs de désactiver l'algorithme bidirectionnel pour certains fragments de texte.

Les deux attributs possibles sont **lang** et **dir**.

La balise de fin est requise.

# 3.6 Élément IFRAME

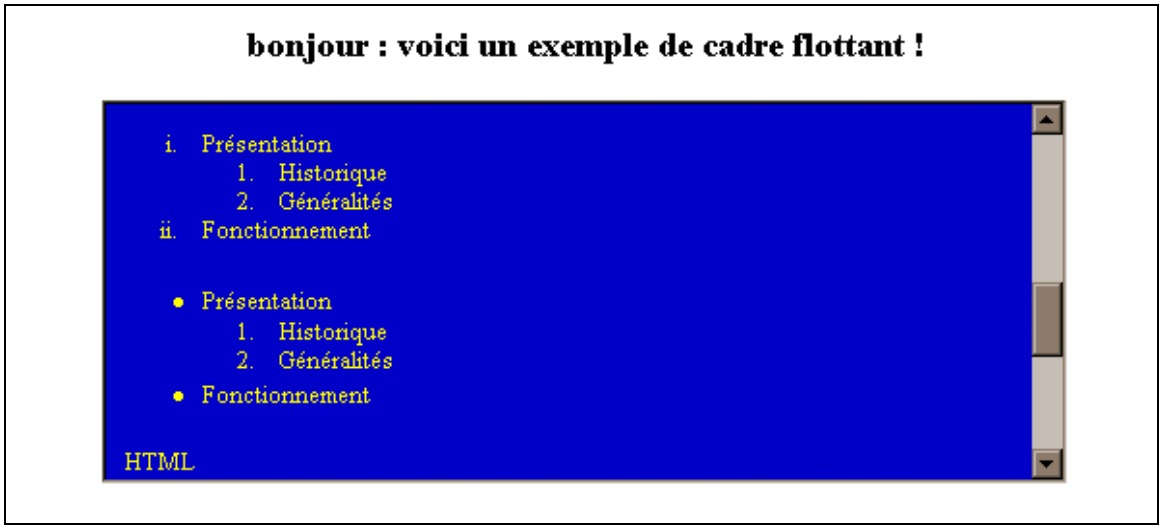
### Description

L'élément IFRAME (Internet Explorer 3.0), présentant les mêmes attributs que ceux de l'élément FRAME, est introduit pour permettre d'inclure une fenêtre au milieu d'un document HTML.

### Attributs

| `iframe` | | | |
|---|---|---|---|
| `align` | = | `center`, `left` *ou* `right` | Précise l'alignement du cadre flottant dans le document. |
| `frameborder` | = | `0` *ou* `1` | Indique si une bordure est tracée en 3D ou en aplat. |
| `height` | = | `nombre` | Hauteur du cadre flottant en pixels. |
| `marginheight` | = | `nombre` | Hauteur de marge en pixels. |
| `marginwidth` | = | `nombre` | Largeur de marge en pixels. |
| `hspace` | = | `nombre` | Conjointement à l'attribut *align*, spécifie, en pixels, l'espace à ménager à droite et à gauche du cadre. |
| `vspace` | = | `nombre` | Conjointement à l'attribut *align*, spécifie, en pixels, l'espace à ménager au dessus du cadre. |
| `name` | = | `texte` | Nom de l'élément utilisé pour des contrôles. |
| `scrolling` | = | `yes` *ou* `no` *ou* `auto` | Autorise ou non la présence d'ascenseurs. |
| `src` | = | `URL` | Document HTML à afficher. |
| `width` | = | `nombre` | Largeur du cadre flottant en pixels. |
| `>`*texte pour les navigateurs qui ne supportent pas cet élément*`</iframe>` | | | |

### Résultat




### Exemple

```
<IFRAME SRC="listes.htm" WIDTH="500" HEIGHT="200"
SCROLLING="AUTO" FRAMEBORDER="1">
[votre agent utilisateur ne supporte pas les cadres ou n'est pas configuré pour les afficher.<br>
Cliquez pour récupérer <A href="liste.htm">le document associé.</A>]
</IFRAME>
```

## 3.7 Élément OBJECT

### Description

La norme HTML 4.0 préconise l'élément **object** (Netscape 4 et Internet Explorer 3.0) pour remplacer désormais l'élément **embed** et l'élément **applet**. L'élément **object** existait déjà dans la version 3.2 du langage HTML mais était réservée uniquement à l'insertion des contrôles ActiveX.

Cet élément a pour but essentiel d'incorporer des objets dans un document HTML. Ces objets peuvent être associés à une vidéo, une application multimédia ou un contrôle ActiveX ou même à un autre document HTML.

Pour visualiser des données qu'ils ne peuvent traiter nativement, les navigateurs (agents utilisateurs) exécutent en général des applications externes. L'élément OBJECT permet aux auteurs de contrôler si les objets inclus doivent être gérés par les navigateurs, ou de façon externe.

Dans le cas général, le système de visualisation de l'objet nécessite trois types d'informations :

- des données d'implémentation du système de visualisation lui-même (caractéristiques de l'application elle-même),
- les données devant être visualisées (fichier texte ou base de données),

- des données supplémentaires nécessaires à l'exécution du visualisateur (éléments param).

Dans certains cas, il n'est pas nécessaire de fournir ces trois informations. Par exemple, certains systèmes de visualisation ne nécessitent pas de données en entrée (par exemple une applet affichant une petite animation directement codée dans son programme).

D'autres systèmes ne nécessiteront pas de données supplémentaires d'initialisation.

Certains autres enfin, ne demanderont pas de données d'implémentation supplémentaires, par exemple, parce que l'agent utilisateur (le navigateur) sait déjà lui-même comment visualiser des données de ce type de média (les images par exemple).

En langage HTML, l'élément OBJECT indique où trouver le code du visualisateur ainsi que les données dont ce dernier a besoin.

Toutes les informations sont données par les divers attributs de l'élément OBJECT. L'élément PARAM permet de définir un certain nombre de paramètres d'exécution. Cet élément sera décrit par la suite, dans la section concernant l'initialisation des objets.

### Attributs

| <object | | | |
|---|---|---|---|
| name | = | texte | Spécifie un nom d'identification de l'objet pour les scripts. |
| classid | = | texte | Cet attribut spécifie l'adresse du système de visualisation, ou identifie le contrôle ActiveX. |
| codebase | = | URL | Cet attribut spécifie le chemin d'accès de base utilisé pour exprimer des URL relatives spécifiées par **classid.** Il peut être suivi de paramètres. |
| codetype | = | MIME | Cet attribut optionnel spécifie le type MIME des données attendues par le système de visualisation défini par **classid.** Mais il est recommandé lorsque classid est explicité, car il permet d'éviter le chargement de ressources ou parties de ressources non supportées par l'agent utilisateur. Si aucune valeur explicite ne lui est donnée, sa valeur par défaut est alors celle définie pour l'attribut **type.** |
| type | = | MIME | Cet attribut optionnel fixe le type MIME des données spécifiées par **data.** Si aucune valeur explicite n'est donnée, le navigateur cherchera a déterminer par lui-même le type des données à visualiser. |
| data | = | URL | Identifie une adresse URL d'où l'objet pourra importer les données nécessaires à son initialisation. |
| archive | = | URL | Indique le fichier d'archives si les données sont disponibles sous une forme compressée. |
| declare | | (+ v4) | Ce booléen permet de déclarer l'objet (ou de la placer dans la page) sans l'exécuter. L'objet devra être instancié par une définition d'**OBJECT** ultérieure, se référant à cette déclaration (où seront définis les paramètres d'exécution). |
| shapes | | (+ v4) | Ce booléen spécifie que l'objet en cours de définition est une image |

| | | | |
|---|---|---|---|
| | | | Map (ou carte graphique). |
| usemap | = | URL (+ v4) | Il spécifie la carte graphique externe à l'image représentative de l'objet. |
| standby | = | texte (+ v4) | Il spécifie le message à afficher pendant que l'objet est chargé. |
| tabindex | = | nombre | Détermine la tabulation pour l'objet sélectionné. |
| width, height | = | nombre | Donne les dimensions initiales en pixels de la surface ou l'objet sera affiché. |
| border | = | nombre (+ v4) | Spécifie la largeur de la bordure de l'objet (pour les liens). |
| vspace, hspace | = | Nombre (+ v4) | Marge verticale et horizontale de l'objet (optionnel). |
| align | = | valeur | Valeurs possibles = { left \| center \| right \| top \| middle \| bottom \| texttop \| absmiddle \| baseline \| absbottom }, alignement par défaut à gauche. |
| ID, CLASS, LANG , DIR, STYLE, TITLE | | | Autres attributs génériques possibles. |
| ONCLICK, ONDBLCLICK, ONMOUSEDOWN, ONMOUSEUP, ONMOUSEOVER, ONMOUSEMOVE, ONMOUSEOUT, ONKEYPRESS, ONKEYDOWN, ONKEYUP. | | | Evènements intrinsèques associables. |
| > *balise <param> éventuelle répétée autant de fois que le permet le développeur de l'application.* | | | |
| </object> | | | |

L'initialisation des objets s'effectue à l'aide de la balise PARAM. Si la balise de début est obligatoire, la balise de fin est interdite.

| <param | | | |
|---|---|---|---|
| name | = | texte | Cet attribut définit le nom d'un paramètre passé à l'objet. La sensibilité à la casse (minuscule/majuscule) de ce paramètre dépend de l'implémentation interne de l'objet. |
| value | = | texte | Cet attribut définit la valeur du paramètre spécifié par l'attribut **name.** |
| type | = | MIME | Cet attribut optionnel fixe le type MIME des données visées par **value** ***uniquement*** dans le cas où l'attribut **valuetype** prend la valeur "ref". Cet attribut indique au navigateur le type de variables que ce dernier trouvera à l'URL donnée par l'attribut value. |
| valuetype | = | MIME | Cet attribut spécifie le type de l'attribut **value.** Les valeurs possibles sont :<br>• data (chaîne de caractères), ref (adresse URL non résolue), |

| | | | |
|---|---|---|---|
| | | | • object (fragment de déclaration d'élément OBJECT du même document),<br>La valeur (#nom) sert à établir un lien avec l'autre déclaration d'élément OBJECT (attribut id). |
| > | | | |

### Exemple 1

Dans l'exemple suivant, nous insérons un visualisateur fictif écrit en langage Python qui affiche une horloge analogique. Cette applet ne nécessite aucune donnée d'implémentation additionnelle, ni paramètres d'exécution. L'attribut **classid** donne l'emplacement de l'applet. Nous recommandons que cette déclaration soit complétée par un texte alternatif défini dans la définition de l'élément **OBJECT**, au cas où l'agent utilisateur ne pourrait visualiser l'horloge.

```
<OBJECT classid="http://www.miamachina.it/analogclock.py">
Une horloge animée.
</OBJECT>
```

### Exemple 2

Nous pouvons insérer plusieurs déclarations d'OBJECT les unes dans les autres. Elles seront interprétées dans l'ordre de lecture : (1) une applet "Terre" écrite en Python, (2) une animation MPEG de la Terre Earth, (3) une image GIF de la Terre, (4) un texte alternatif.

```
<OBJECT title="La Terre vue de l'Espace"
classid="http://www.observer.mars/TheEarth.py">
  <OBJECT data="TheEarth.mpeg" type="application/mpeg">
    <OBJECT src="TheEarth.gif">La <STRONG>Terre</STRONG> vue de l'espace.
    </OBJECT>
  </OBJECT>
</OBJECT>
```

### Exemple 3

Dans l'exemple qui suit, le paramètre d'initialisation est défini comme étant une ressource extérieure (un fichier GIF). La valeur de l'attribut *valuetype* vaut de ce fait "ref", et l'attribut **value** est une adresse URL qui désigne cette ressource.

```
<OBJECT classid="http://www.gifstuff.com/gifappli" standby="Loading Elvis...">
<PARAM name="Init_values" value="./images/elvis.gif" valuetype="ref">
</OBJECT>
```

### Exemple 4

La première partie d'une adresse URL absolue désigne le protocole utilisé pour transférer les données pointées par la partie adresse de l'URL. Pour les documents HTML, ce protocole est pratiquement toujours "http".

Certains systèmes de visualisation peuvent être amenés à utiliser d'autres protocoles. Par exemple, lorsqu'un visualisateur Java est invoqué, les adresses URL appelées commencent par **java**, et pour des applets ActiveX, par **clsid**.

Dans l'exemple qui suit, nous insérons une applet java dans un document HTML.

```
<OBJECT classid="java:program.start">
</OBJECT>
```

### Exemple 5

En définissant l'attribut **codetype**, le navigateur décide de télécharger ou non l'application Java selon qu'il sait l'exécuter ou non.

```
<OBJECT codetype="application/octet-stream" classid="java:program.start"
        codebase="http://foooo.bar.com/java/myimplementation/">
</OBJECT>
```

### Exemple 6

L'exemple suivant montre comment accéder à un visualisateur ActiveX via une URL commençant par le protocole propriétaire "clsid". L'attribut **data** donne la localisation des données à visualiser (une autre horloge, en l'occurrence).

```
<OBJECT classid="clsid:663C8FEF-1EF9-11CF-A3DB-080036F12502"
        data="http://www.acme.com/ole/clock.stm">
This application is not supported.
</OBJECT>
```

### Exemple 7

Lorsqu'un document doit contenir plus d'une seule instance d'un même objet, il est possible de séparer la déclaration de l'objet de ses instanciations. Cette façon de faire présente un certain nombre d'avantages :

- Les données nécessaires ne seront téléchargées du réseau *qu'une seule fois* (à la déclaration) et seront utilisées (les mêmes) pour chaque instanciation.
- Il est possible d'instancier un objet à partir de différents endroits d'un document, par exemple, en activant un hyperlien.
- Il est possible de définir des objets comme paramètres d'exécution pour d'autres objets.

Dans l'exemple qui suit, nous déclarons un élément *OBJECT* sans l'exécuter grâce à l'attribut booléen declare, puis nous provoquons son exécution depuis une instance de l'objet en activant un hyperlien. Les deux parties de la déclaration sont bien reliées : id="nom" et href="#nom".

```
<OBJECT declare id="earth_declaration" data="TheEarth.mpeg" type="application/mpeg">
  <OBJECT src="TheEarth.gif">La <STRONG>Terre</STRONG> vu de
l'espace.</OBJECT>
</OBJECT>
...plus loin dans le document...
Cliquer pour voir une animation nette <A href="#earth_declaration">de la Terre!</A>
```

**Exemple 8**

Parfois, plutôt que d'établir un lien vers un autre document, il peut être plus judicieux d'inclure le contenu d'un document HTML dans un autre document HTML. Nous recommandons d'utiliser un élément **OBJECT** dont on aurait défini l'attribut **data** à cette fin.

```
...texte précédent...
<OBJECT data="file_to_include.html">
Erreur : file_to_include.html n'a pas pu être inclus.
</OBJECT>
...texte suivant...
```

**Exemple 9**

Les images Map côté serveur peuvent être intéressantes à exploiter lorsque la définition géométrique est trop compliquée pour être définie par les mécanismes côté client.

Pour indiquer au navigateur qui exploite les liens de la carte graphique, il faut mentionner l'attribut booléen **ismap** pour les hyperliens résolus par le serveur, dans l'élément ***IMG*** ou ***OBJECT***.

Dans l'exemple suivant, la première région définie est un lien côté client. La seconde définit un lien traité côté serveur, sans qu'une forme de zone spécifique ne soit indiquée.

```
<OBJECT data="game.gif" shapes>
  <A href="guide.html" shape="rect" coords="0,0,118,28">Rules of the Game</A>
  <A href="http://www.acme.com/cgi-bin/competition" ismap shape="default">
      Guess the location</a>
</OBJECT>
```

# 4. Éléments obsolètes et périmés

## 4.1 Eléments obsolètes

Les éléments suivants sont désormais obsolètes : ISINDEX, APPLET, CENTER, FONT, BASEFONT, STRIKE, S, U, DIR, et MENU.

## 4.2 Eléments périmés

Les éléments suivants sont désormais périmés : XMP, PLAINTEXT, et LISTING. A la place de ces éléments, il conviendra d'utiliser l'élément PRE.

# 5. Compléments aux tableaux

Les premiers exemples de tableaux formaient un ensemble unique. Il est désormais possible, depuis la version HTML3.2, de structurer les tableaux, en au moins trois groupes : l'entête, le corps et le "pied de tableau".

Les tableaux connaissent d'autres améliorations : la possibilité de préciser le nombre de colonnes (attribut cols) afin que le navigateur n'ait pas à le calculer en parcourant tout le tableau et puisse ainsi afficher le tableau dès les premières lignes lues.

Ensuite les colonnes peuvent être groupées (balise colgroup) pour permettre :

- d'appliquer une présentation différente (largeur, alignement) pour chaque groupe ou colonne du groupe (balise col)
- de spécifier des bordures et gouttières personnalisées pour chaque ligne ou chaque colonne ou chaque groupe (attributs frames et rules).

## 5.1 Attribut COLS

### Description

Lorsqu'un tableau est affiché, le navigateur doit en analyser toutes les lignes pour compter le nombre de colonnes. Cela provoque des délais d'affichage pénalisants pour des gros tableaux.

L'attribut **COLS** ajoute à l'élément TABLE le nombre de colonnes du tableau.

### Exemple

```
<TABLE COLS="3" border="1">
<tr><td>colonne 1 </td><td>colonne 2 </td><td>colonne 3</td></tr>
</TABLE>
```

## 5.2 Parties d'un tableau

Cette partie se justifie ici dans la mesure où ces possibilités n'ont été vraiment exploitées qu'à partir de la version HTML4.0. Les lignes peuvent être groupées en trois ensembles qui ont chacun leur spécificité et leur style : l'**entête**, le **corps** et le "**pied de tableau**" ou piètement.

### Description

L'entête reprend l'idée initiale des éléments **th** qui avaient pour but de présenter les têtes de colonnes. Ce groupe, comprenant éventuellement plusieurs lignes, est défini grâce aux éléments **thead** (head of table). Il est reconnu par Internet Explorer 3.0.

Le "pied de tableau" a pour but de rehausser le style des résultats ou totaux d'un tableau à calculs. Ce groupe, pouvant englober plusieurs lignes, est défini avec les éléments **tfoot** (foot of table).

Notons enfin que si l'un de ces deux groupes est présent, le reste des lignes du tableau définit lui-même le corps du tableau encadré par l'élément **tbody** (body of table) sans style particulier.

Si l'entête est présent, le pied du tableau n'est pas obligatoire, et vice versa. Il est aussi possible d'utiliser les deux sans créer le groupe du corps de tableau. Enfin, nous pouvons insérer plusieurs groupes **tbody** afin de définir plusieurs sections dans notre tableau.

Il est préférable pour le navigateur que les groupes *thead* et *tfoot* apparaissent avant le groupe tbody.

Les balises de fermeture </thead>, </tbody> et </tfoot> sont optionnelles mais aident à la lisibilité du code.

### Attributs

| <THEAD | | | |
|---|---|---|---|
| align | = | center,left, right, justify, char | Cet attribut spécifie l'alignement d'un groupe de colonnes. La valeur par défaut est "center".<br>Depuis la version 4, il est possible de **justifier** comme il est possible de définir un caractère d'alignement (**char**). |
| valign | = | BASELINE, TOP BOTTOM, CENTER | Cet attribut définit l'alignement vertical du texte d'un groupe de colonnes. |
| char | = | caractère | Cet attribut spécifie dans un fragment de texte le caractère utilisé comme repère d'alignement.<br>Par défaut, l'alignement est toujours réalisé sur la virgule décimale propre à chacune des langues courantes (attribut **lang**) : le point "." en anglais et la virgule "," en français. |
| charoff | = | nombre | Lorsque présent, cet attribut spécifie une longueur de décalage comptée à partir de la première occurrence du caractère d'alignement. |

| | |
|---|---|
| ID, CLASS, LANG , DIR, STYLE, TITLE | Autres attributs génériques possibles. |
| ONCLICK, ONDBLCLICK, ONMOUSEDOWN, ONMOUSEUP, ONMOUSEOVER, ONMOUSEMOVE, ONMOUSEOUT, ONKEYPRESS, ONKEYDOWN, ONKEYUP. | Evènements associables. |
| >… </THEAD> | |

Il en est de même pour les groupes tbody et tfoot.

**Exemple**

```
<table border rules="groups">
<thead>
  <tr><th>Ent&ecirc;te</th></tr>
</thead>
<tbody>
  <tr><td>Les lignes</td></tr>
  <tr><td>du tableau</td></tr>
</tbody>
<tfoot>
  <tr><th>Pied</th></tr>
</tfoot>
</table>
```

**Résultat**

| Entête |
|---|
| Les lignes |
| du tableau |
| Pied |

## 5.2 Attributs FRAME et RULES

De même, cette partie trouve sa place dans ce document dans la mesure où les différents écrits à ce sujet les classent plus dans la version HTML4.0 que dans la version HTML3.2.

**Descriptions et valeurs**

Le premier attribut **frame** contrôle le tracé des différentes parties de la bordure autour du tableau (**bordure externe**) et doit être utilisé conjointement avec l'attribut **border.**

| | | | |
|---|---|---|---|
| frame | = | void | supprime le cadre autour du tableau |

| | | | |
|---|---|---|---|
| | | above | trace le bord supérieur |
| | | below | trace le bord inférieur |
| | | hsides | trace les bords supérieur et inférieur |
| | | lhs | trace le bord gauche (“left hand side”) |
| | | rhs | trace le bord droit (“right hand side”) |
| | | vsides | trace les bords gauche et droit |
| | | box ou border | trace les quatre bords |

Quant à lui, l’attribut **rules** contrôle le tracé des lignes de division à l’intérieur du cadre (appelées aussi **gouttières** par les textes de référence par opposition aux **bordures** obligatoirement externes) et doit être utilisé conjointement avec l'attribut **border.**

| | | | |
|---|---|---|---|
| rules | = | none | supprime le quadrillage interne |
| | | groups | trace une ligne horizontale entre chaque groupes de lignes (cf. THEAD, TFOOT, et TBODY) et les groupes de colonnes (cf. COLGROUP et COL). |
| | | rows | trace une ligne horizontale autour de chaque ligne |
| | | cols | trace une ligne verticale autour de chaque colonne |
| | | all | dessine toutes les divisions internes |

**Exemple**

```
<TABLE BORDER="2" WIDTH="200" FRAME="HSIDES" RULES="COLS">
<THEAD>
<COLGROUP SPAN="3" ALIGN="LEFT" VALIGN="TOP">
</THEAD>
<TBODY>
<TR>
<TD><B>Ligne 1</B></TD>
<TD>Cellule 1</TD>
<TD>Cellule 2</TD>
</TR>
<TR>
<TD><B>Ligne 2</B></TD>
<TD>Cellule 3</TD>
<TD>Cellule 4</TD>
</TR>
</TR>
</TBODY>
</TABLE>
```

### Résultat

| | | |
|---|---|---|
| **Ligne 1** | Cellule 1 | Cellule 2 |
| **Ligne 2** | Cellule 3 | Cellule 4 |

### Remarques

- Nous pouvons combiner plusieurs styles ensemble en encadrant la valeur par des guillemets et en séparant les styles par une virgule, exemple : rules="rows, cols".
- Les réglages suivants devront être compris par les agents utilisateurs pour des raisons de compatibilité ascendante.
  - Donner une valeur nulle à l'attribut **border** implique la valeur "void" à l'attribut **frame** et, sauf indication contraire, la valeur "none" à l'attribut **rules**.
  - Une valeur non nulle de **border** implique que l'attribut **frame** a la valeur "border" et, sauf indication contraire, que l'attribut **rules** a la valeur "all".
- Notons que ces deux attributs peuvent aussi être utilisés avec les éléments THEAD, TBODY et TFOOT.

## 5.3 Groupe de colonnes COLGROUP et COL

### 5.3.1 Élément COLGROUP

### Description

Les colonnes d'un tableau peuvent être groupées. Chaque groupement est défini par une balise COLGROUP (Internet Explorer 3.0) et deux attributs obligatoires SPAN et WIDTH.

**<COLGROUP SPAN=***valeur* **WIDTH=***valeur***>**

### Attributs

| | | | |
|---|---|---|---|
| <COLGROUP | | | |
| span | = | nombre | Cet attribut sert à définir le nombre de colonnes qui seront affectées par l'attribut ALIGN (un par défaut). |
| width | = | nombre | Il fixe la largeur de chaque colonne du groupement en pourcentage ou en valeur absolue.<br>La forme "0*" signifie une largeur minimale à chacune des colonnes du groupement. |
| align, valign, char, charoff | | | Ces attributs spécifient l'alignement des groupes de cellules. |
| ID, CLASS, LANG , DIR, STYLE, TITLE | | | Autres attributs génériques possibles. |

| | |
|---|---|
| ONCLICK, ONDBLCLICK,<br>ONMOUSEDOWN, ONMOUSEUP,<br>ONMOUSEOVER, ONMOUSEMOVE,<br>ONMOUSEOUT, ONKEYPRESS,<br>ONKEYDOWN, ONKEYUP. | Evènements associables. |
| > | |

Cette balise doit être utilisée avant le groupement THEAD comme le montre l'exemple suivant et avant l'élément CAPTION s'il est utilisé.

**Exemple**

```
<TABLE>
<COLGROUP SPAN="5" WIDTH="50">
<COLGROUP SPAN="6" WIDTH="20">
<THEAD>
<TR>
<td>a</td><td>b</td><td>c</td><td>d</td><td>e</td><td>f</td><td>g</td><td>h</td>
<td>i</td><td>j</td><td>k</td>
</tr>
<TR>
<td>l</td><td>m</td><td>n</td><td>o</td><td>p</td><td>q</td><td>r</td><td>s</td>
<td>t</td><td>u</td><td>v</td>
</tr>
</TABLE>
```

**Résultat**

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i | j | k |
| l | m | n | o | p | q | r | s | t | u | v |

### 5.3.2 Balise COL

**Description**

La balise COL (Internet Explorer 3.0) et ses deux arguments obligatoires WIDTH et SPAN sert à définir chacune des colonnes à l'intérieur du groupement .

**<COL SPAN=*valeur* WIDTH=*valeur*>**

**Attributs**

Cette balise présente les mêmes attributs que la balise précédente.

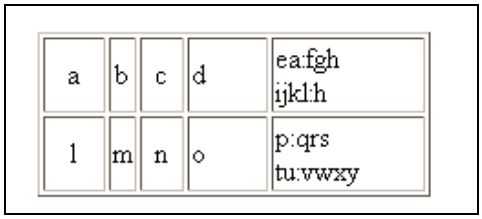
En plus des valeurs absolues et relatives, de la valeur "0*", on trouve la valeur "i*" qui permet de donner une valeur relative dans le groupement.

**Exemple**

```
<TABLE>
<COLGROUP>
<COL WIDTH="30">
<COL WIDTH="0*">
<COL WIDTH="2*">
<COLGROUP ALIGN="CENTER">
<COL WIDTH="1*">
<COL WIDTH="3*" ALIGN="CHAR" CHAR=":" CHAROFF="5">
<TR><td>a</td><td>b</td><td>c</td><td>d</td><td>ea:fgh<br>ijkl:h</td></TR>
<TR><td>l</td><td>m</td><td>n</td><td>o</td><td>p:qrs<br>tu:vwxy</td></TR>
</TABLE>
```

**Remarque** :

L'alignement indiqué ci-dessus à l'aide du caractère ":" ne fonctionne pas sous IE5.5.




# 6. Améliorations des formulaires

**L'attribut GET** est déprécié alors qu'il reste utile pour les tests d'applications CGI et autres scripts SSI (Server Sides Includes).

## 6.1 Élément BUTTON

**Description**

L'introduction d'un **élément BUTTON** (Internet Explorer 4.0) ajoute un bouton graphique dans un formulaire pour permettre d'enclencher des actions à l'aide de la souris. Le but de cet élément est de se substituer progressivement à l'élément INPUT. L'élément BUTTON est un élément fermé.

**Attributs**

| <BUTTON | | | |
|---|---|---|---|
| name | = | nom | Nom du bouton. |
| value | = | nom | Valeur renvoyée par le bouton. |
| type | = | nom | Type du bouton ( button, submit, reset ). |
| disabled | | | Désactive le bouton. |

| | | | |
|---|---|---|---|
| tabindex | = | nombre | Définit l'ordre de tabulation entre les éléments possédant une propriété *tabindex*. |
| onfocus | = | script | Evénement qui se déclenche lorsque l'élément est sélectionné. |
| onblur | = | script | Evénement qui se déclenche lorsque l'élément est désélectionné. |
| > *...étiquette du bouton accompagnée éventuellement d'une image...* | | | |
| </BUTTON> | | | |

### Exemples

Cet élément est plus riche que l'élément INPUT puisqu'il permet d'encapsuler des zones se comportant comme un bouton d'action. Le texte et l'image de l'exemple ci-dessous s'affichent à l'intérieur du bouton.

```
<form method="post" action="tags-v4.htm">
  <BUTTON name="submit" value="submit" type="submit">
Suivant <IMG src="triangles.gif" alt="Envoi"></BUTTON>
  </form>
```

Voici le résultat du code source ci-dessus !


Voici un deuxième exemple :

```
<BUTTON name="reset" value="effacer"  type="reset" > Annule <IMG
src="../images/stop.gif" alt="annuler" style="left: 10px; position: relative" </BUTTON>
```

Le navigateur de Netscape semble ne pas le reconnaître.

Une image de type USEMAP ne peut être incluse dans un couple de balises BUTTON.

## 6.2 Élément LABEL

### Description

L'élément LABEL ou étiquette (Internet Explorer 4.0) permet d'associer une zone de texte avec un élément de formulaire de type "champ de texte".

### Attributs

| | | | |
|---|---|---|---|
| <LABEL | | | |
| for | = | nom | Nom du champ texte associé. |

| | |
|---|---|
| class, lang, dir, title, style, id | Autres attributs génériques possibles. |
| tabindex, disabled, | Attributs spécifiques aux éléments de formulaires (c.f. partie "Activation des zones de saisie"). |
| accesskey = lettre | Touche du clavier associée à l'objet pour déclencher une action (c.f. partie "Activation des zones de saisie"). |
| onclick, onfocus, onblur, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout, onkeypress, onkeydown, onkeyup. | Evènements associables. |
| > *...légende du champ texte associé...* | |
| </LABEL> | |

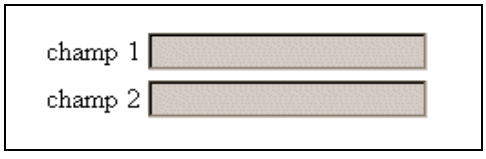
L'attribut **for** de la balise **label** permet d'associer cette dernière à un élément de formulaire défini dans la même page par son nom de contrôle (valeur de son attribut id ou name). L'élément label fait office de définition.

Ainsi, quand un élément de nom *label* est activé, il passe le contrôle à l'élément associé défini par for. On peut ainsi cliquer sur le contrôle comme sur son étiquette.

**Exemple**

```
<FORM ACTION="..." METHOD="post">
<TABLE>
 <TR>
  <TD><LABEL FOR="premier">champ 1</LABEL>
  <TD><INPUT TYPE="texte" NAME="premiernom" ID="premier">
 <TR>
  <TD><LABEL FOR="deuxieme">champ 2</LABEL>
  <TD><INPUT TYPE="texte" NAME="deuxiemenom" ID="deuxieme">
</TABLE>
</FORM>
```

**Résultat**



# 6.3 Éléments FIELDSET et LEGEND

### Descriptions

L'élément **FIELDSET** (Internet Explorer 4.0) permet de regrouper plusieurs champs ensemble pour leur donner une cohérence.

L'élément **LEGEND** permet de donner une légende à un groupe de champs regroupés par un élément FIELDSET.

### Attributs

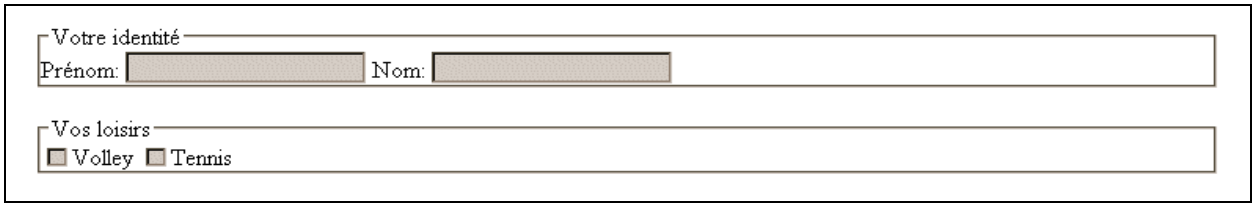
| | | | |
|---|---|---|---|
| `<LEGEND` | | | |
| `class, lang, dir, title, style, id` | | | Autres attributs génériques possibles. |
| `align` | = | `top, bottom, left, ou right` | Valeur d'alignement selon que la légende est placée en haut, en bas, à gauche ou à droite du champ associé. |
| `accesskey` | = | `lettre` | Touche du clavier associée à l'objet pour déclencher une action (c.f. partie "Activation des zones de saisie"). |
| `onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout, onkeypress, onkeydown, onkeyup.` | | | Evènements associables. |
| `>` *...légende du champ texte associé...* | | | |
| `</LEGEND>` | | | |

L'élément FIELDSET présente une balise de fermeture, mais pas d'attribut. Il encadre simplement les éléments à regrouper, comme dans l'exemple ci-dessous.

**Exemple**

```
<FORM ACTION="..." METHOD="post">
<FIELDSET>
 <LEGEND ALIGN="top">Votre identité</LEGEND>
 Prénom: <INPUT NAME="Prenom" TYPE="text" TABINDEX="1">
 Nom: <INPUT NAME="Nom" TYPE="text" TABINDEX="2">
</FIELDSET>
 <FIELDSET>
 <LEGEND ALIGN="top">Vos loisirs</LEGEND>
 <INPUT NAME="Sports" TYPE="checkbox" VALUE="Volley" TABINDEX="20">Volley
 </INPUT>
 <INPUT NAME="Sports" TYPE="checkbox" VALUE="Tennis" TABINDEX="20">Tennis
</INPUT>
 </FIELDSET>
 </FORM>
```

**Résultat**




# 6.4 Elément OPTGROUP

Il faut signaler un dernier élément très peu cité en dehors du document de normalisation. L'élément **OPTGROUP** permet aux auteurs de grouper des choix logiques au sein d'un élément **SELECT**. Ceci est particulièrement utile lorsque l'utilisateur doit choisir dans une longue liste d'options; les groupes de choix liés étant plus faciles à manipuler et à se remémorer qu'une simple et longue liste d'options.

# 6.5 Activation des zones de saisie

Auparavant, seule l'action de la souris rendait possible la sélection d'une zone de saisie ou le passage d'une zone à une autre.

Désormais, il est possible de changer l'ordre activé par les touches de tabulation (attribut **tabindex**) et de définir des raccourcis clavier qui permettront d'aller directement sur une zone (attribut **accesskey**).

### Attribut TABINDEX

Description

Chaque fois que cet attribut est présent dans une balise, il prend une valeur entière qui est la position (ou le rang) de l'élément.
Les valeurs négatives sont admises mais ces valeurs n'entrent pas dans l'ordonnancement des zones associées.
Les valeurs n'ont pas besoin d'être consécutives.

Si deux valeurs sont identiques, l'ordre de leur apparition dans le document sera pris en compte.

Éléments

Les éléments qui supportent l'attribut **tabindex** sont *A, AREA, OBJECT, INPUT, SELECT, TEXTAREA et BUTTON***.**

Exemple :

```
<A TABINDEX="10" HREF="...">URL</A>
  <BUTTON TYPE="button" NAME="get-database" TABINDEX="1">
  Cliquez ici
  </BUTTON>
```

```
<FORM ACTION="..." METHOD="post">
<INPUT TABINDEX="2" TYPE="text" NAME="champ1">
<INPUT TABINDEX="3" TYPE="text" NAME="champ2">
</FORM>
```

### Attribut ACCESSKEY

Description

Cet attribut assigne une touche du clavier à une zone. Cette touche est en général choisie parmi les caractères alphabétiques du clavier. Le navigateur interprète de manière équivalente minuscules ou majuscules.

Le fait de presser une touche du clavier (touche ALT sur un PC, touche CTRL sur Macintosh), sélectionne (focus) l'élément ou la balise possédant l'attribut accesskey associé.

Éléments

Les éléments qui supportent l'attribut *ACCESSKEYS* sont *LABEL, A, CAPTION* et *LEGEND.*

Exemple :

```
<A ACCESSKEY="C" HREF="...">Cliquez ici</A>
```

Remarque :

Sous IE5.5, l'action de la touche ALT affiche dans la barre d'état "Contient les commandes pour les éléments sélectionnés".
Dans l'exemple ci-dessus, l'action de la touche "C" affiche dans la barre d'état l'adresse du lien visé. La validation (touche Entrée) entraîne l'activation du lien.
Sous Netscape 6.0, la combinaison de touches suffit à activer le lien.

### Attribut DISABLED

Description

Cet attribut permet de ne pas sélectionner un élément (mettre le focus sur), ni par la souris, ni par les touches clavier.

Les éléments qui sont en mode **disabled** ne sont pas transmis avec le formulaire.

Éléments

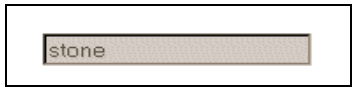
Les éléments suivantes supportent l'argument **disabled** : *INPUT, TEXTAREA, SELECT, OPTION, OBJECT, LABEL et BUTTON.*

L'élément considéré comme inactif (disabled) sera en général **grisé** dans le navigateur (voir résultat ci-dessous).

Un script (JavaScript ou VBScript) peut rendre actif un élément inactif et vice versa.

Exemple :

```
<INPUT DISABLED NAME="fred" VALUE="stone">
```




**Attribut READONLY**

Description

Un élément marqué **readonly** ne peut pas être modifié par un utilisateur, par contre il est transmis avec le formulaire à la différence de l'attribut **disabled**.
Un élément marqué en lecture seule (readonly) peut être activé mais pas modifié. Il peut donc être inclus dans la liste des éléments indexés par **tabindex**.

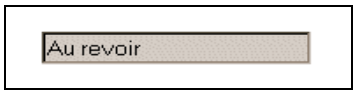
Éléments

Les éléments pouvant recevoir l'attribut readonly sont les éléments de formulaires **INPUT** de type **text** ou **password** et les éléments **TEXTAREA**.

Un script (JavaScript ou VBScript) peut rendre actif un élément.

Exemple

```
<INPUT READONLY NAME="pat" VALUE="Au revoir">
```




L'utilisateur ne peut accéder au champ ci-dessus ni en modifier le contenu.

# Conclusion

Le document présent, dans sa première version, ne prétend pas être exhaustif sur les possibilités du langage HMTL v4.0.

Celles qui ne seraient pas présentes ont été considérées par le rédacteur comme étant soit pratiquement inutilisées, soit difficiles à expliquer aux concepteurs non programmeurs d'applications web.

Certains diront qu'il est trop complet.

Pour terminer, nous disons que ce chapitre offre un tour d'horizon relativement solide et suffisant sur le sujet.

Une deuxième version du même document permettrait de le compléter (exemples avec résultats) ou de l'alléger (aspect international) selon les besoins.

**Bibliographie :**

- LEMAY L. - Le programmeur HTML4. - Simon & Schuster MacMillan (France), 2000
- Le dictionnaire du HTML, XML, VML - Micro Application

Il est néanmoins dommage de constater qu'une norme sortie en 1997 ne soit pas encore prise en compte par les éditeurs HTML classiques, qu'il existe peu de textes pour expliquer l'emploi de certaines éléments ou attributs, que les navigateurs n'interprètent pas encore certains éléments de référence.

# Annexe 1 : Noms des 140 couleurs prédéfinies par Netscape

| | |
|---|---|
| aliceblue | #F0F8FF |
| aqua | #00FFFF |
| azure | #F0FFFF |
| bisque | #FFE4C4 |
| blanchedalmond | #FFEBCD |
| blueviolet | #8A2BE2 |
| burlywood | #DEB887 |
| chartreuse | #7FFF00 |
| coral | #FF7F50 |
| cornsilk | #FFF8DC |
| cyan | #00FFFF |
| darkcyan | #008B8B |
| darkgray | #A9A9A9 |
| darkkhaki | #BDB76B |
| darkolivegreen | #556B2F |
| darkorchid | #9932CC |
| darksalmon | #E9967A |
| darkslateblue | #483D8B |
| darkturquoise | #00CED1 |
| deeppink | #FF1493 |
| dimgray | #696969 |
| firebrick | #B22222 |
| forestgreen | #228B22 |
| ghostwhite | #F8F8FF |
| gold | #FFD700 |
| gray | #808080 |
| greenyellow | #ADFF2F |
| hotpink | #FF69B4 |
| indigo | #4B0082 |
| khaki | #F0E68C |

| | |
|---|---|
| antiquewhite | #FAEBD7 |
| aquamarine | #7FFFD4 |
| beige | #F5F5DC |
| black | #000000 |
| blue | #0000FF |
| brown | #A52A2A |
| cadetblue | #5F9EA0 |
| chocolate | #D2691E |
| cornflowerblue | #6495ED |
| crimson | #DC143C |
| darkblue | #00008B |
| darkgoldenrod | #B8860B |
| darkgreen | #006400 |
| darkmagenta | #8B008B |
| darkorange | #FF8C00 |
| darkred | #8B0000 |
| darkseagreen | #8FBC8F |
| darkslategray | #2F4F4F |
| darkviolet | #9400D3 |
| deepskyblue | #00BFBF |
| dodgerblue | #1E90FF |
| floralwhite | #FFFAF0 |
| fuchsia | #FF00FF |
| gainsboro | #DCDCDC |
| goldenrod | #DAA520 |
| green | #008000 |
| honeydew | #F0FFF0 |
| indianred | #CD5C5C |
| ivory | #FFFFF0 |
| lavender | #E6E6FA |

| | |
|---|---|
| lavenderblush | #FFF0F5 |
| lemonchiffon | #FFFACD |
| lightcoral | #F08080 |
| lightgoldenrodyellow | #FAFAD2 |
| lightgrey | #D3D3D3 |
| lightsalmon | #FFA07A |
| lightskyblue | #87CEFA |
| lightsteelblue | #B0C4DE |
| lime | #00FF00 |
| linen | #FAF0E6 |
| maroon | #800000 |
| mediumblue | #0000CD |
| mediumpurple | #9370DB |
| mediumslateblue | #7B68EE |
| mediumturquoise | #48D1CC |
| midnightblue | #191970 |
| mistyrose | #FFE4E1 |
| navajowhite | #FFDEAD |
| oldlace | #FDE5E6 |
| olivedrab | #6B8E23 |
| orangered | #FF4500 |
| palegoldenrod | #EEE8AA |
| paleturquoise | #AFEEEE |
| papayawhip | #FFEFD5 |
| peru | #CD853F |
| plum | #DDA0DD |
| purple | #800080 |
| rosybrown | #BC8F8F |
| saddlebrown | #8B4513 |
| sandybrown | #F4A460 |
| seashell | #FFF5EE |
| silver | #C0C0C0 |
| slateblue | #6A5ACD |
| snow | #FFFAFA |

| | |
|---|---|
| lawngreen | #7CFC00 |
| lightblue | #ADD8E6 |
| lightcyan | #E0FFFF |
| lightgreen | #90EE90 |
| lightpink | #FFB6C1 |
| lightseagreen | #20B2AA |
| lightslategray | #778899 |
| lightyellow | #FFFFE0 |
| limegreen | #32CD32 |
| magenta | #FF00FF |
| mediumaquamarine | #66CDAA |
| mediumorchid | #BA55D3 |
| mediumseagreen | #3CB371 |
| mediumspringgreen | #00FA9A |
| mediumvioletred | #C71585 |
| mintcream | #F5FFFA |
| moccasin | #FFE4B5 |
| navy | #000080 |
| olive | #808000 |
| orange | #FFA500 |
| orchid | #DA70D6 |
| palegreen | #98FB98 |
| palevioletred | #DB7093 |
| peachpuff | #FFDAB9 |
| pink | #FFC0CB |
| powderblue | #B0E0E6 |
| red | #FF0000 |
| royalblue | #4169E1 |
| salmon | #FA8072 |
| seagreen | #2E8B57 |
| sienna | #A0522D |
| skyblue | #87CEEB |
| slategray | #708090 |
| springgreen | #00FF7F |

| | |
|---|---|
| steelblue | #4682B4 |
| teal | #008080 |
| tomato | #FF6347 |
| violet | #EE82EE |
| white | #FFFFFF |
| yellow | #FFFF00 |

| | |
|---|---|
| tan | #D2B48C |
| thistle | #D8BFD8 |
| turquoise | #40E0D0 |
| wheat | #F5DEB3 |
| whitesmoke | #F5F5F5 |
| yellowgreen | #9ACD32 |

# Annexe 2 : Liste des caractères spéciaux

*Cette liste s'inspire du guide de référence simplifié du langage HTML.*

## A.Caractères ISO Latin-1

| | | | | | |
|---|---|---|---|---|---|
| À | À | capital A, grave accent | ï | ï | small i, diæresis/umlaut |
| à | à | small a, grave accent | Ð | Ð | capital Eth, Icelandic |
| Á | Á | capital A, acute accent | ð | đ | small eth, Icelandic |
| á | á | small a, acute accent | Ñ | Ñ | capital N, tilde |
| Â | Â | capital A, circumflex | ñ | ñ | small n, tilde |
| â | â | small a, circumflex | Ò | Ò | capital O, grave accent |
| Ã | Ã | capital A, tilde | Ò | ò | small o, grave accent |
| ã | ã | small a, tilde | Ó | Ó | capital O, acute accent |
| Ä | Ä | capital A, diæresis/umlaut | ó | ó | small o, acute accent |
| ä | ä | small a, diæresis/umlaut | Ô | Ô | capital O, circumflex |
| Å | Å | capital A, ring | ô | ô | small o, circumflex |
| å | å | small a, ring | Õ | Õ | capital O tilde |
| Æ | Æ | capital AE ligature | õ | õ | small o, tilde |
| æ | æ | small ae ligature | Ö | Ö | capital O, diæresis/umlaut |
| Ç | Ç | capital C, cedilla | ö | ö | small o, diæresis/umlaut |
| ç | ç | small c, cedilla | Ø | Ø | capital O, slash |
| È | È | capital E, grave accent | ø | ø | small o, slash |
| è | è | small e, grave accent | Ù | Ù | capital U, grave accent |
| É | É | capital E, acute accent | Ù | ù | small u, grave accent |
| é | é | small e, acute accent | Ú | Ú | capital U, acute accent |
| Ê | Ê | capital E, circumflex | ú | ú | small u, acute accent |
| ê | ê | small e, circumflex | Û | Û | capital U, circumflex |
| Ë | Ë | capital E, diæresis/umlaut | û | û | small u, circumflex |
| ë | ë | small e, diæresis/umlaut | Ü | Ü | capital U, diæresis/umlaut |
| Ì | Ì | capital I, grave accent | ü | ü | small u, diæresis/umlaut |
| ì | ì | small i, grave accent | Ý | Ý | capital Y, acute accent |
| Í | Í | capital I, acute accent | ý | ý | small y, actite accent |
| í | í | small i, acute accent | Þ | Þ | capital thorn, Icelandic |

| | | | | | |
|---|---|---|---|---|---|
| Î | Î | capital I, circumflex | þ | þ | small thorn, Icelandic |
| î | î | small i, circumflex | ß | ß | small sharp s, German sz |
| Ï | Ï | capital I, diæresis/umlaut | ÿ | ÿ | small y, diæresis/umlaut |

# B. Caractères Additionnels ISO 8859-1

| | | | | | | | |
|---|---|---|---|---|---|---|---|
|   |   | | non-breaking space | ± | ± | ± | plus-or-minus sig |
| ¡ | ¡ | ¡ | inverted exclamation mark | ² | ² | ² | superscript two |
| ¢ | ¢ | ¢ | cent sign | ³ | ³ | ³ | superscript three |
| £ | £ | £ | pound sign | ´ | ´ | ´ | acute accent |
| ¤ | ¤ | ¤ | general currency sign | µ | µ | µ | gmicro sign |
| ¥ | ¥ | ¥ | yen sign | ¶ | ¶ | ¶ | pilcrow (paragrap |
| ¦ | ¦ | ¦ | broken (vertical) bar | · | · | · | middle dot |
| § | § | § | section sign | ¸ | ¸ | ¸ | cedilla |
| ¨ | ¨ | ¨ | umlaut/dieresis | ¹ | &supl; | ¹ | superscript one |
| © | © | © | copyright sign | º | º | º | ordinal indicator |
| ª | ª | ª | ordinal indicator, fem | » | » | » | angle quotation m |
| « | &Iaquo; | « | angle quotation mark, left | ¼ | &fracl4; | ¼ | fraction one-quar |
| ¬ | ¬ | ¬ | not sign | ½ | &fracl2; | ½ | fraction one-half |
|  |  | | soft hyphen | ¾ | ¾ | ¾ | fraction three-qua |
| ® | ® | ® | registered sign | ¿ | ¿ | ¿ | inverted question |
| ¯ | ¯ | ¯ | macron | × | × | · | x multiply sign |
| ° | ° | ° | degree sign | ÷ | ÷ | ÷ | division sign |