DHTML ou le Html dynamique

1. Définition du DHTML

1.1 Introduction ou ... au risque de vous décevoir

Ne cherchez pas de syntaxe ou de références DHTML, vous n'en trouverez pas. Le DHTML n'est pas un langage de balises, de scripts ou de programmation. Le DHTML n'est pas une quelconque spécification. Il n'est même pas réellement une technologie! Tout simplement,

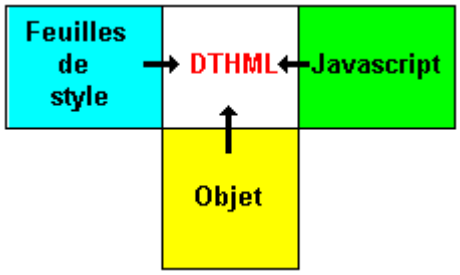
le DHTML ou Html dynamique est, et reste, du Html
où le concepteur a mis l'accent sur les animations.

A l'heure de l'importance croissante de l'impact visuel des sites Internet, le Html dit dynamique est un "plus" incontestable et génial sous certains aspects. Mais il faut raison garder! Le DHTML n'est pas le "futur de l'Internet" comme le prétendent certains mais, à n'en pas douter, il y aura certainement une bonne place.

Bien que peut-être excessifs, je suis même tenté de reproduire les mots de Sally Khudairi [à vos souhaits!] du World Wide Web Consortium (W3C) : "DHTML is nothing more than a marketing term for generic animation and manipulation of multimedia events" (DHTML n'est rien d'autre qu'un terme de marketing pour des animations et des effets multimédia).

1.2. Définition

Le DHTML se définit comme un (savant) mélange de trois développements de la publication sur Internet : les feuilles de style (CSS1), les langages de script, principalement le Javascript, ainsi que les objets et leur positionnement en vue de produire des pages dynamiques et interactives.



Ajoutons car cela n'apparaît pas dans la définition :

- l'aspect dynamique réside dans le fait que la page Html peut être modifiée après le chargement de celle-ci par le navigateur.
- une volonté d'effectuer les animations et interactions côté client (c-à-d côté browser) sans faire appel aux ressources du serveur par des applets Java ou des ActiveX.
- cette technique DTML a un aspect "high-tech" et est réservée aux versions 4.0 (ou plus) de Internet Explorer et Netscape.

1.3. Les feuilles de style

Popularisée par Microsoft avec Internet Explorer 3.0 (bien qu'elles existaient déjà sous d'autres systèmes), les feuilles de style [Cascading Style Sheets] constituent un réel développement dans la conception de pages Web, en séparant la mise en forme du contenu, en proposant des possibilités jusque là inédites en Html strict , etc..

Le concept des feuilles de style est maintenant un standard depuis son officialisation par le Word Web Consortium sous la norme CSS level 1 et son intégration dans le Html 4.0. Et les versions 4.0 de Microsoft Explorer et Netscape Navigator reprennent déjà largement les feuilles de style.

Pour une étude plus poussée, les feuilles de style sont largement abordées dans ce site (www.ccim.be/ccim328/htmlplus/idxstyle.htm).

1.4. Les langages de scripts (surtout Javascript)

Introduit par Netscape avec Navigator 2.0, Javascript a bien mûri au fil des versions 3.0 et 4.0 des browsers de la marque (Javascript 1.1, Javascript 1.2). Enfin repris efficacement par Microsoft sous Explorer 4.0, ce langage de scripts s'est définitivement imposé comme un élément incontournable dans l'élaboration de sites Internet.

Javascript permet de programmer et d'exécuter, côté client c-à-d côté browser et donc sans faire appel à des ressources extérieures, de petites applications à l'intérieur de pages Html. Sans être un véritable langage de programmation, Javascript en possède toutes les fonctionnalités ou presque).

VBscript fait également partie des langages de scripts qui peuvent être utilisés dans le DHTML. Mais propriétaire Microsoft, il n'est que peu utilisé en DHTML au détriment de Javascript.

Javascript est aussi un standard Internet depuis la norme ECMA Script. En outre, le Html 4.0 le reprend largement.

Pour vous guider dans l'apprentissage de ces deux langages de scripts, "Apprendre le Javascript" (www.ccim.be/ccim328/js/index.htm) et "Apprendre le VBscript" (www.ccim.be/ccim328/vb/index.htm) du même auteur, sont à votre disposition.

1.5. Les objets et leur positionnement

L'héritage du Html, à l'origine conçu pour faire transiter du texte structuré, est parfois pesant. Positionner précisément, au pixel près, un texte ou une image, tourne vite au cauchemar. De là est née l'idée d'introduire une sorte de cadre, de rectangle dans lequel on pourrait "encapsuler" des "objets" comme précisément du texte, des images ou tout autre élément.

Comme s'est souvent le cas dans l'histoire de la publication sur Internet, nos deux compères Microsoft et Netscape, se sont attelés à la tâche (sans attendre un quelconque standard) et ce en suivant des modèles différents pour cette insertion d'objets dans les documents Html.

Microsoft fournit un accès aux éléments des pages Html par "ses" feuilles de style (CSS-P). Netscape a choisi un modèle d'insertion d'éléments qui lui est spécifique par le concept de "layer" et sa collection de balises. On se retrouve donc avec deux technologies distinctes pour une même approche.

Le W3C tente de réunir de réunir les deux précédentes technologies pour les faire fonctionner ensemble. C'est le DOM [Document Objet Model]. Cette tentative est encore à ce jour (septembre 98), un document de travail et l'on reste donc encore dans l'attente d'un véritable standard.

Cette lacune ouvre la porte aux problèmes de compatibilité pour les applications DHTML selon quelles soient traitées par Netcape 4.0 ou par Explorer 4.0.

1.6. Déceptions

Si visuellement et techniquement, le DHTML est quelque chose de tout à fait génial, on ne peut qu'être un peu déçu :

- par son manque (actuel?) de compatibilité. Un DHTML qui fonctionne pour IExplorer 4.0 ne tourne que très rarement sous Netscape 4.0. Si le Webmaster est gentil, il aura bien prévu une petite remarque du genre "Il vous faut le browser X". Mais je fais toujours partie de ceux qui n'ont pas de bol et emploient justement le browser Y.
- par sa complexité. Tout en tirant mon chapeau aux concepteurs de DHTML, le Html dynamique n'est pas du tout à la portée d'un débutant et n'est même plus à la portée d'un hobbyiste, fut-il averti. Il faut connaître

sous le bout des doigts, trois techniques : le Html, le Javascript et les feuille de style. Ce qui n'est pas à la portée du premier
- Webmaster venu. A vos cours de programmation... ou à votre "copier/coller" (en respectant le copyright).
- par son temps de chargement plus important bien qu'il ne soit en rien comparable avec la vitesse d'escargot des applets Java.
- par son risque réel de faire planter le browser et même la machine si l'application DHTML est mal écrite.

1.7. Espoirs

Mais terminons par une note positive car :
- le concept DHTML (Feuilles de style - Javascript - Objets) est bien né et n'est pas remis en cause par les actuels problèmes de compatibilité.
- le DHTML est encore jeune et les développeurs ne peuvent que, dans un avenir proche, trouver de nouvelles astuces pour améliorer cette compatibilité.
- après les versions 4.0, Microsoft et Netscape sortiront leurs versions 5.0... Et celles-ci intégreront (peut-être) une gestion commune des objets.

2. Positionner avec Netscape
par la technique des couches [layer]

2.1.La balise <LAYER>

Netscape propose depuis sa version 4.0, une technique pour positionner des éléments. C'est la technique des couches ou des "layers". On peut comparer ces layers aux transparents que l'on utilise avec les rétroprojecteurs.

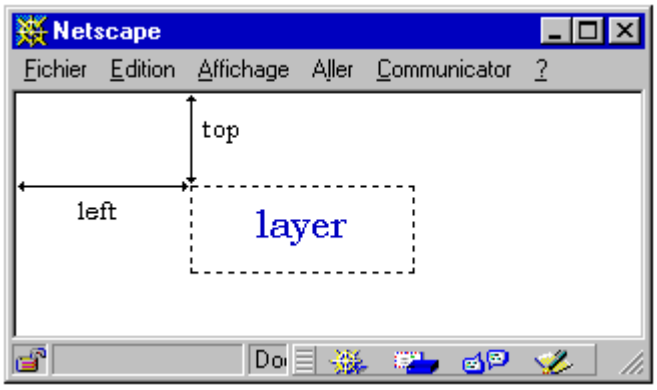
La syntaxe est :

<LAYER name="nom" left=x top=y> ... élément ... </LAYER>

La balise <LAYER> va demander au browser de réserver une sorte de petit rectangle - vide - ou une sorte de container dans lequel on va placer un élément, par exemple une image.

Il sera indispensable de donner un nom au "layer" lorsque plusieurs sont utilisés dans une même page ou un même document.

Les attributs "left" et "top" vont positionner cette zone par rapport au coin supérieur gauche de la fenêtre du browser. Top est la valeur en pixels du haut vers le bas et left, la valeur en pixels de la gauche vers la droite.



Le browser va afficher les layers et leur contenu quoiqu'il arrive! Idéal pour créer selon vos désirs des effets de superpositions. Idéal aussi pour avoir des superpositions inattendues...

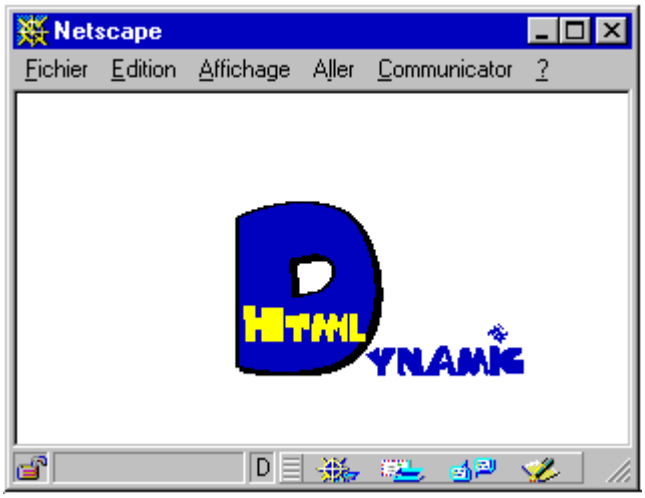
La tag <LAYER> est un tag propriétaire de Netscape 4.0 qui n'a pas été repris par Microsoft. "LAYER" ne fonctionne donc pas sous Eplorer 4.0 et uniquement sous Netscape. Comme les feuilles de style qui permettent aussi de positionner des éléments, sont reprises et par Explorer 4.0 et par Netscape 4.0, cette dernière technique est recommandée pour des raisons de compatibilité.

2.2. Exemple

Pourquoi des couches? Tout simplement (les layers étant affichés quoiqu'il arrive), lorsque vous prévoyez plusieurs layers au même endroit, le premier qui apparaît dans le code, sera affiché directement. Le layer suivant viendra se mettre au-dessus et ainsi de suite. Le premier layer est donc celui qui sera "au fond" de l'écran par rapport autres. En jouant avec la transparence des images gif, on peut ainsi créer des effets surprenants.

Nous allons superposer les deux images suivantes :


```
<LAYER name="layer1" left=100 top=50>
<IMG SRC="dyna.gif" height=97 width=160>
</LAYER>
<LAYER name="layer2" left=110 top=75>
<IMG SRC="html.gif" height=41 width=83>
</LAYER>
```



La couleur verte a disparue car elle a la particularité gif d'être transparente. Le positionnement au pixel près ne se fait pas sans tâtonnement et il faut souvent plusieurs essais avant d'arriver au résultat souhaité.

Le mouvement quant à lui est obtenu avec une bonne dose de Javascript en faisant varier les valeurs des attributs "top" et "left".

3. Positionner avec Explorer

par la technique des feuilles de style (CSS)

3.1. L'approche par les feuilles de style

Quelques mois après la version 4.0 de Netscape, Microsoft sortait la version 4.0 de Explorer. Ce qui lui a permis d'aller plus avant dans le positionnement et la définition des objets.

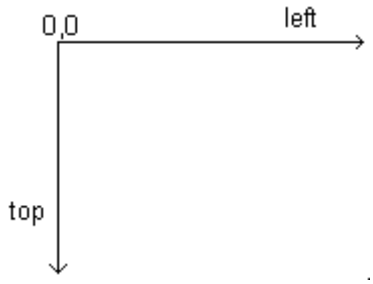
Microsoft a développé "son" concept de feuilles de style (Cascading Style Sheets ou CSS] qui permettait déjà de décomposer une pages Web en blocs avec les balises SPAN et DIV. Restait à pouvoir positionner ces blocs... C'est tout le développement du CSS-P ou du positionnement avec les feuilles de style.

A l'opposé de l'approche de Netscape avec une balise (la balise <LAYER>), les feuilles de style apportent un concept plus vaste, plus complet et plus puissant mais d'un apprentissage (un peu) plus complexe.

Pour les lecteurs qui prennent le site en marche, je rappelle que les feuilles de style sont largement abordées plus avant dans le site (www.ccim.be/ccim328/ htmlplus/idxstyle.htm).

3.2. Position absolue ou relative

La position absolue {position: absolute} se détermine par rapport au coin supérieur gauche de la fenêtre du browser. Les coordonnées de ce point sont top = 0 et left = 0. Les coordonnées d'un point s'expriment en pixels, de haut en bas pour top et de gauche à droite pour left.

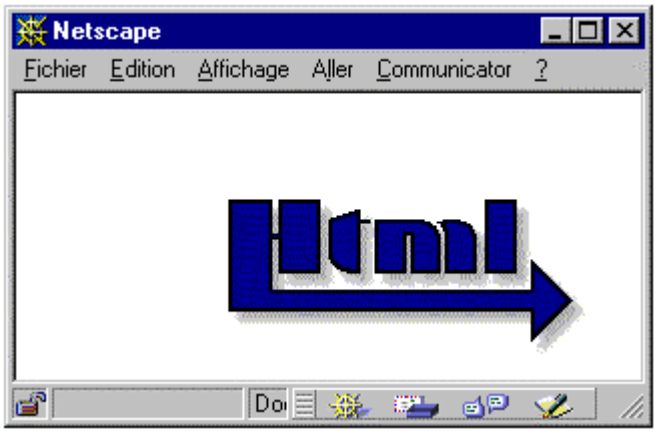


La position relative {position: relative} se détermine par rapport à d'autres éléments de la page, par exemple un élément du code Html.

3.3. Positionner une image

Plaçons l'image htmlplus.gif en position absolue à 50 pixels de haut de la fenêtre (top) et à 100 pixels à gauche (left). Les dimensions de l'image sont width=242 et height=84.

```
<HTML>
<BODY>
<span style="position: absolute; top: 50px; left: 100px; width: 242px;  heigth: 84px;">
<IMG src="htmlplus.gif">
</span>
</BODY>
</HTML>
```



Spécifiez toujours les propriétés width et heigth avec les feuilles de style car par défaut, Netscape 4.0 et Explorer 4.0 ne réagissent pas de la même façon.

3.4. Superposer du texte sur une image

Reprenons l'image htmlplus.gif et on y superposera le nom de l'auteur de ce tutorial, au pixel près dans la barre qui souligne le terme Html.

```
<HTML>
<BODY>
<span style="position: absolute; top: 50px; left: 100px; width: 242px; heigth: 84px;">
<IMG src="htmlplus.gif">
</span>
<span style="position: absolute; top: 96px; left: 145px; color: yellow; font-size: x-small; font-weight:
bold;">
Van Lancker Luc
</span>
</BODY>
</HTML>
```

4. Le "Document Objet Model"

4.1 La notion d'objet.

Conçu dans les années 1980, le Html (HyperText Markup Langage) n'est finalement qu'un long fichier de texte auquel on a ajouté des balises (Markups) - prédéfinies et statiques - afin que les navigateurs puissent afficher ce texte dans une mise en forme ou une mise en page identique sous différentes plates-formes. Html n'a par essence aucune structuration et encore moins d'hiérarchisation de ses composants.

La perception d'une page Html comme un ensemble d'éléments et qui plus est hiérarchisés, est une véritable révolution dans le domaine de la publication Html. Ces éléments sont aussi repris sous le vocable d'objets au sens des langages de programmation comme C ou C++.

A la mi-97, Netscape a entamé le sujet avec sa version 4.0 en créant une balise propriétaire <LAYER>. Explorer 4.0, qui est apparu quelques mois plus tard, a pu aller plus avant dans le concept objet en exploitant quant à lui, le concept des feuilles de style repris et développé par Microsoft.

Il devenait alors évident que ce concept d'objets devait être standardisé par le World Wide Web Consortium (W3C) avec comme objectif de dégager un modèle commun pour la définition des objets dans un document Html. Ce qui a donné naissance au Document Object Model repris sous l'abréviation de DOM.

4.2 Le Document Object Model ou DOM

Ce "Document Object Model" n'est encore à ce jour (septembre 98) qu'un document de travail et n'est donc toujours pas finalisé.

Si les principaux protagonistes, Netscape et Microsoft, sont unanimement d'accord sur la nécessité d'introduire un principe (commun) de la notion d'objets, leurs avis divergent complètement sur les techniques à mettre en oeuvre pour le réaliser.

Leurs principaux points de désaccord peuvent se résumer ainsi :

- Netscape 4.0, conçu plusieurs mois avant son concurrent, a une conception interne assurément moins élaborée et moins souple que Explorer 4.0.
- Netscape 4.0 n'a accès qu'à quelques éléments de la page alors que IExplorer 4.0 a accès à tous les éléments de la page.
- A travers cette implémentation de la notion d'objet, Netscape défend (ou tente de récupérer) le concept de Java et Microsoft prône bien sûr ses ActiveX et autres Active Channels.
- Les versions 5.0 de Microsoft et de Netscape se profilent à l'horizon 99.

Constatant que dans l'état actuel des technologies les avis étaient inconciliables, le W3C a préféré jeter des bases plus solides pour l'avenir en rattachant cette notion commune d'objets au XML (e**X**tensible **M**arkup **L**anguage) qui sera compatible avec les futures évolutions 5.0 des navigateurs.

4.3. Le XML en bref

Il faut bien l'avouer, le Html avec toutes ses balises rajoutées au fil des années, fait un peu désordre. Il subit en outre une demande énorme en termes de besoins en publication sur Internet auxquels il ne peut répondre comme les signes mathématiques, les notes de musique ou le braille.

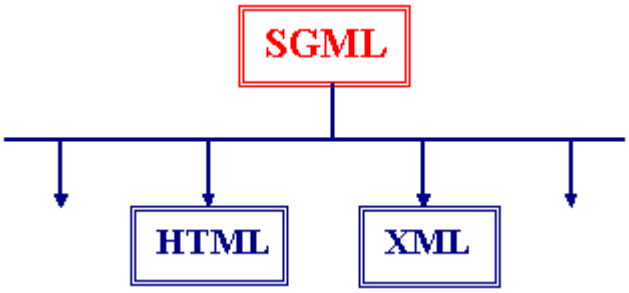
Le XML (e**X**tensible **M**arkup **L**anguage) donne l'impression de repartir avec une feuille blanche. XML n'est en aucune façon, un Html++ mais un nouveau petit frère pour le Html car XML et Html ont la même mère c-à-d le langage SGML.

Pas simple tout cela...

D'abord le SGML (pour **S**tandard **G**eneralized **M**arkup **L**anguage), est une norme internationale (ISO 8879), qui régit la structure et le contenu des différents types de documents électroniques. Le SGML est en quelque sorte un "language-mère" utilisé pour décrire les milliers de documents électroniques que l'on retrouve dans les multiples domaines de l'activité humaine.

Le HTML (**H**yper**T**ext **M**arkup **L**anguage) n'est que un de ces types de documents avec ses balises imposées et qui est utilisé pour les documents électroniques qui circulent sur le Web.

Avec le XML, on retourne aux sources du SGML en permettant au "designer" de définir son propre type de document et, côté spectaculaire de la chose, de créer ses propres balises (voir le **X** de e**X**tensible).



Tant Microsoft que Netscape annoncent que le XML sera disponible avec les versions 5.0 de leur browser.

Nous ne manquerons pas de revenir plus en détail sur le XML, plus tard dans l'élaboration de ce site consacré à la maîtrise du Html.

4.4 En conclusion

Pour en revenir à la définition d'objets, chaque fabricant de browser pourra avec le Xml implémenter des objets, selon sa propre perception et selon sa propre technologie, définir ses propres balises relatives aux objets et le tout sera géré de façon commune par tous les browsers compatibles avec le XML. Et voilà, le tour est joué.

De la compatibilité d'un site

1. Notre démarche

Nous tentons ci-après une réflexion de plusieurs pages sur la compatibilité d'un site par rapport aux différents browsers disponibles sur le Web.

Notre démarche sera la suivante ;

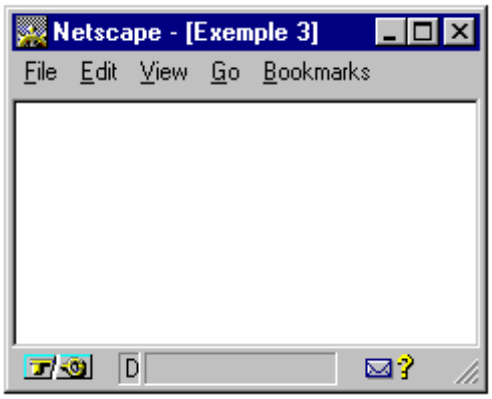
- Nous passons d'abord en revue quelques exemples d'incompatibilité qui peuvent dénaturer l'attrait d'un site.
- Nous présentons un tableau comparatif des navigateurs selon leur compatibilité avec les différents développements du Html et de ses techniques dérivées.
- Nous tenterons ensuite de faire une extrapolation de statistiques disponibles sur le Web concernant les navigateurs utilisés et leurs versions.
- Nous nous attacherons à une réflexion (personnelle) sur la compatibilité d'un site. Avec comme objectif de répondre à la question : Ecrire sur le Web, mais pour quel(s) browser(s) ?
- Nous terminerons par divers conseils relatifs à la compatibilité, glanés ça et là sur le Web.

2. Exemples d'incompatibilité

Avec les éditeurs Html récents, certains perçoivent une page Html comme un document imprimé issu d'un traitement de texte et le Web comme une sorte de fax pour envoyer cette feuille de papier [et j'exagère à peine!...].

N'oubliez jamais qu'une page Web n'est pas un document statique et fixé sur un support. C'est le navigateur ou browser de votre lecteur qui "fixe" le document en fonction de ses propres spécifications et de la configuration de l'ordinateur sur lequel il fonctionne.

Voici quelques exemples de documents avec les mêmes balises, mais visionnés sous deux navigateurs différents.



Acceptez-vous que votre document soit impeccable pour seulement 25% des internautes et un torchon lamentable fait par un incapable pour les autres 75 %. ?

Alors comment allez-vous faire pour "corriger" votre éditeur Html chéri (et cher) ?... Voilà pourquoi il est indispensable d'avoir au moins une connaissance basique du Html et de ses balises si particulières. Voir du même auteur "Apprendre le langage Html" (www.ccim.be/ccim328/html/index.htm).

3. Tableau comparatif

Ci-après un tableau comparatif des différents développements du Html et de ses techniques dérivées

	Netscape 2	**Netscape 3**	**Netscape 4**	**Explorer 3**	**Explorer 4**
Html 3.2	Html 2.0 évolué	Html 3.2	Html 3.2	Html 3.2	Html 3.2
Html 4.0 (balises spécifiques)	NON	NON	oui incomplet	NON	oui incomplet
Javascript	oui version 1.0	oui version 1.1	oui version 1.2	oui incomplet	oui version 1.2
VBscript	NON	NON	NON	oui	oui
Formulaire avec mailto	oui	oui	oui	NON	oui
Feuilles de style CSS1	NON	NON	oui	oui incomplet	oui
Positionner avec CSS1	NON	NON	oui	NON	oui
Positionner avec <layer>	NON	NON	oui	NON	NON
DHTML	NON	NON	oui	NON	oui

Quelques premiers commentaires :

- Plus votre Html sera simple (plus ou moins conforme à la version 3.2), plus votre site sera compatible avec la majorité des browsers. Mais votre site risque d'être un peu triste.
- La véritable poisse pour les concepteurs est un bowser qui ne reprend qu'incomplètement certaines spécifications. En regardant le tableau, vous remarquerez que Microsoft Explorer 3.0 fait partie de cette catégorie (sans compter ses bugs). Ainsi rendre un site compatible Explorer 3.0, est un véritable cauchemar pour les concepteurs de site.
- Que les présentations évoluées passent nécessairement par les versions 4 de Netscape et de Microsoft Internet explorer. Ce qui est finalement logique car ce sont les plus récentes. Revers de la médaille, votre site sera d'autant moins compatible pour les autres browsers.

4. Statistiques

4.1. Précautions préliminaires

Ces chiffres n'ont pas la prétention d'être d'une exactitude absolue
mais de fournir un ordre de grandeur réaliste pour la suite de l'étude !

Une compilation des sites consacrés aux statistiques relatives aux divers navigateurs et leurs versions utilisées sur le Web ne peut que laisser le lecteur perplexe! En effet, si vous voulez connaître précisément la part de marché de Netscape et de Microsoft, vous resterez assurément sur votre faim tant les chiffres peuvent être différents.

Les sites de statistiques, s'ils sont un peu sérieux, ne manquent pas de mettre en garde sur l'interprétation hasardeuse de leurs chiffres car il existe une corrélation évidente entre la sophistication du contenu du site (pas forcément de la présentation) et la version évoluée du navigateur.

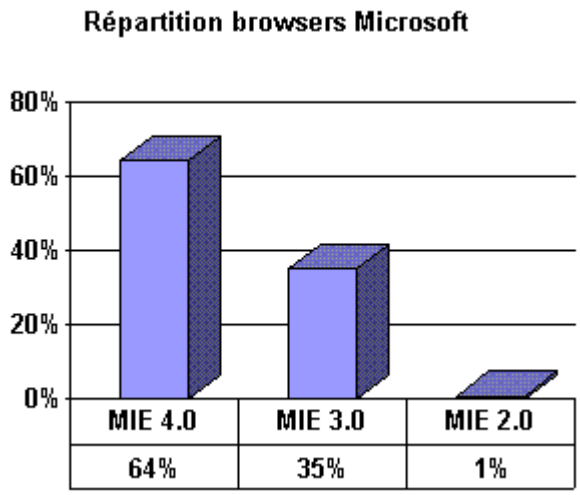
4.2. Microsoft, Netscape et les autres...

Netscape et Microsoft prennent à eux deux 85 à 90 % du marché des browsers. Le reste, soit 10 à 15 %, est occupé par une trentaine de navigateurs plus ou moins connus (moins en ce qui me concerne...). Signalons parmi ceux-ci, la présence d'Opéra-3.0. ce qui est encourageant pour cette jeune initiative.

4.3. Du côté de chez Microsoft

Pour Microsoft Internet Explorer, la répartition entre les versions est à ce jour (septembre 98):

MIE 4.0	MIE 3.0	MIE 2.0
64 %	35 %	1 %



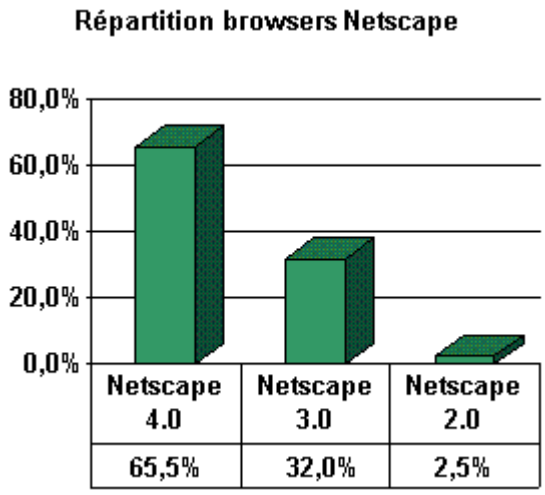
Conclusions :

- La migration de Explorer 3 vers Explorer 4, semble bien effective, ce qui de la part d'un public porté sur l'informatique n'étonne qu'à moitié. Est-ce aussi le signe que ce public aime l'innovation ? Je vous laisse seul juge...
- Explorer 2 (qui pour mémoire ne reconnaissait pas les frames) avec son 1 % des utilisateurs de Microsoft, fait figure de dinosaure et semble en voie de disparition.
- Explorer 3 avec toutes ses imperfections [et j'ai décidé d'être sympa aujourd'hui], est quand même encore utilisé par 35 % des utilisateurs des produits de Microsoft.

4.4. Du côté de chez Netscape

Pour Netscape, la répartition entre les versions est à ce jour (septembre 98) :

Netscape 4.0	Netscape 3.0	Netscape 2.0
65,5 %	32 %	2,5 %



Conclusions :

La répartition est sensiblement identique. Les conclusions le seront aussi.

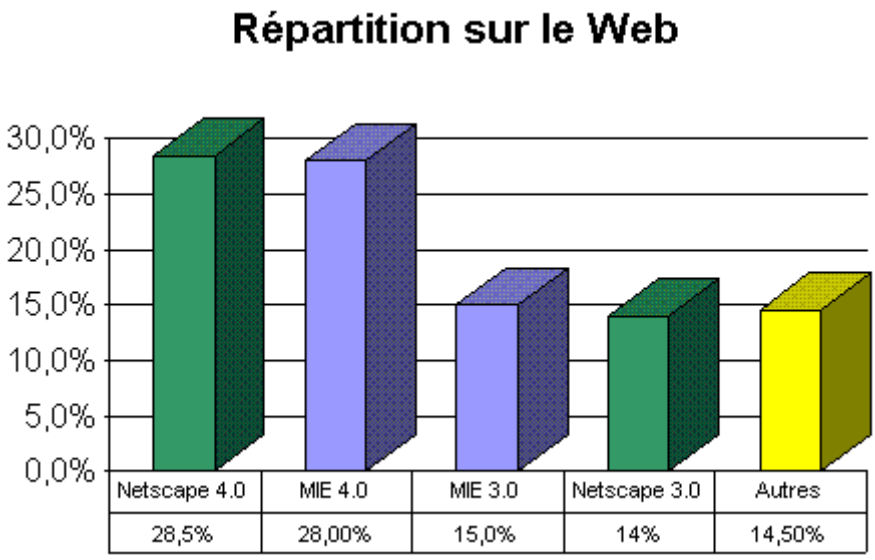
- La migration de Navigator 3.0 à Navigator 4.0 est excellente, avec peut-être un plus grand attrait de l'innovation chez les utilisateurs de Netscape.
- Netscape Navigator 2.0 [qui était déjà un excellent navigateur] garde avec ses 2,5 % quelques irréductibles.
- Navigator 3.0, qui ne reconnaît pas les feuilles de style, est encore utilisé par 32 % des browsers de la marque.

4.5. Du côté du Web

Si on prend comme hypothèses que:

- d'une part les 13 % de browsers qui concurrencent Microsoft et Netscape, auxquels on ajoute Netscape 2.0 et Microsoft 2.0, devenus anecdotiques. Ce qui nous fait un total de 14,5 % dans la catégorie "Autres".
- et d'autre part une répartition de 50 - 50 entre les deux frères ennemis, on pourrait (conditionnel indispensable) avoir à ce jour (septembre 98), la répartition suivante :

Netscape 4.0	MIE 4.0	MIE 3.0	Netscape 3.0	Autres
28,5 %	28 %	15 %	14 %	14,5 %



Ces résultats, nous seront très utiles pour la suite de notre réflexion sur l'importance de la compatibilité des sites Internet

5. Compatibilité d'un site

Publier sur le Web, mais pour quel(s) browser(s) ?

5.1. Attitude 1: Je veux écrire pour tous les browsers.

Autant vous dire que c'est mission impossible! Avec tous les navigateurs sur le marché, avec parfois des déclinaisons selon les systèmes d'exploitation, avec les différentes options de l'utilisateur, avec les différentes résolutions d'écran, il n'est pas du domaine du possible de pouvoir tester et adapter le codage des balises à toutes les situations rencontrées sur le Web.

100% - impossible -

Il faudra se faire une raison... un site ne sera jamais visuellement parfait à 100 % pour tout le monde! Pour comparer avec une situation commerciale, il faut bien accepter qu'un certain pourcentage de la clientèle ne puisse être prospectée car les coûts et les moyens à mettre en oeuvre sont beaucoup trop importants pour le peu de clients potentiels.

5.2 Attitude 2 : Je teste mon site sous un seul navigateur.

On passe d'un extrême à l'autre. Et pourtant... Je pense à des sites impeccables, écrits avec Microsoft Frontpage, testés sous Microsoft Explorer 4 et qui sont épouvantables sous Netscape. L'inverse avec Netscape Communicator (Editor + Navigator) est vrai aussi [mais moins].

Un site testé sous un seul navigateur ne sera parfait que pour (un maximum de) 30% de vos visiteurs. Pour les autres 70%, ce sera "au petit bonheur la chance". Et il ne manquera pas d'y avoir des imperfections... Sur10.000 visiteurs, pour 7000 d'entre eux, votre site sera passable, perfectible ou même désastreux. Pour des pages personnelles, vous pouvez peut-être vous en contenter mais si votre site représente une société commerciale, un organisme, une association, les retombées en terme d'image pour ces 7000 visiteurs insatisfaits, ne seront guère favorables [euphémisme].

max 30%

Et vous ne vous en sortirez pas avec un "Sous Explorer 4.0 seulement". Il faudra déjà que votre site soit plus que passionnant pour que les surfeurs intéressés changent de browser, et ce même dans le cas où il est déjà

implanté sur leur machine. En terme de communication, pour ceux qui ne possèdent pas le browser adéquat l'effet est tout aussi détestable. Autant dire : "T'es déjà pas malin d'avoir choisi le mauvais browser, je parie que tu n'as même pas la toute dernière version sortie sur le marché".

Si je continue ma métaphore commerciale, accepteriez-vous que votre clientèle soit à 70% mal ou incomplètement démarchée?

5.3 Attitude 3 : Site high-tech, écrit pour les browsers de la génération 4.0

Je pense à ces sites bourrés de Javascript 1.2, de positionnement au pixels près, d'effets DHTML et autres techniques de pointes. Ces sites superbes par ailleurs (quand ils fonctionnent...) réclament impérativement les versions 4 de Netscape Navigator et de Microsoft Explorer.

Ces Webmasters doivent savoir que leur site ne s'adresse qu'à un maximum de 55 % des visiteurs du Web. Quid des autres 45 % ? J'espère que le contenu de leur site ne s'adresse lui aussi qu'à 55 % des lecteurs!

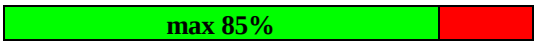
Certains de ses sites dits high-tech ont l'attitude contestable mais au combien réaliste d'exclure (par un petit script) les autres browsers.

5.4. Attitude 4 : Ecrire pour le maximum raisonnable d'internautes.

Vous l'avez compris, la compatibilité d'un site ne peut être qu'un compromis (à la belge... voir nationalité de l'auteur).

Soit votre site sera testé et corrigé pour Netscape 3.0 et 4.0 ainsi que pour Explorer 4.0, dans ce cas il aura un indice de compatibilité (voir statistiques) de 70%.

Soit, mieux encore, Netscape 3.0 et 4.0 ainsi que Explorer 3.0 et 4.0. Dans ce cas, on atteint l'indice maximum raisonnable de 85 %.



Cela vous donnera un site sobre et classique mais qui s'impose pour tout contenu à vocation informative, s'adressant au plus large public possible.

Revenons au cas Explorer 3.0. Pour ne pas répéter tout le mal que j'en pense, celui-ci représente encore à ce jour (septembre 98) 15 % des surfeurs. Soit 1500 sur 10.000 visiteurs. Chacun appréciera le poids de ces 1500 prospects. Pour corser le tout, il vous faudra deux machines pour tester votre site car Explorer 4.0 "mange" Explorer 3.0 à l'installation. J'attends avec impatience le moment où la migration d'explorer 3.0 vers Explorer 4.0 ou 5.0 sera effective mais je crains que l'on doive encore attendre un an ou deux...

6. Conseils

6.1. Compatibilité des fichiers

Si le langage Html est compatible avec toutes les plates-formes, le nom des fichiers de votre site doit cependant respecter les conventions des systèmes d'opération du serveur de votre lecteur et/ou de l'ordinateur de votre lecteur lui-même.

Voici un bref rappel de ces conventions de nom de fichier, en effet, la longueur maximale du nom du fichier, les caractères permis, les signes de ponctuation, et la sensibilité aux majuscules et minuscules [case sensitive] varient d'un "operating system" à l'autre.

DOS et Windows 3.x :

Noms de fichiers avec la convention 8.3 (8 lettres point 3 lettres pour l'extension de fichier), les lettres de A à Z (majuscules et minuscules), les nombres de 0 à 9 et les caractères $ % ' ` @ ^ ! & { } ( ) " ~ et _
Les noms ne sont pas sensibles à la casse.
Ainsi seuls les fichiers avec l'extension htm seront gérés (pas les .html) sous DOS et Windows 3.1 et les fichiers à nom long de Windows 95 entraîneront un message d'erreur.

Windows 95 et NT :
Longueur des noms libre avec un maximum de 255 lettres, l'extension de fichiers n'est plus limitée à 3 lettres, tous les caractères sont permis, même les espaces, sauf \ / : * ? " < > |
Les noms ne sont pas sensibles à la casse.
Sous Windows 95, on peut employer indifféremment l'extension .htm et .html

Macintosh :
Maximum 31 lettres, tous les caractères sont permis excepté le double point (:). Les noms ne sont pas sensibles à la casse. On emploie souvent la notation .html.

Unix :
Maximum 255 lettres, extension de fichier libre (soit .html, . jpeg), tous les caractères sont valides sauf le slach (/). Attention, les noms sont sensibles à la casse [case sensitive]. Ainsi sous Unix MAPAGE.html n'est pas égal à mapage.html.

La seule compatibilité pour tous les noms de fichiers est la convention 8.3, les lettres de A à Z, les nombres de 0 à 9 et l'underscore "_" (ISO 9660).

6.2. Compatibilité des balises

6.2.1. Les balises propriétaires

Ces balises sont propres à une marque de browser et ne sont pas reprises ni par les concurrents de celui-ci, ni par la norme Html.

 Références Explorer uniquement

<BGSOUND>	Son en arrière plan
<COL>	Spécifie l'alignement du texte dans les colonnes d'un tableau.
<COLGROUP>	Groupe des colonnes d'un tableau afin de spécifier leur alignement.
<IFRAME>	Cadre unique.
<MARQUE>	Bannière défilante.
<TBODY>	Corps du tableau.
<VBSCRIPT>	Script en Vbscript.

 Références Netscape uniquement

<BLINK>	Texte clignotant.
<LAYER>	Positionnement d'un objet par la méthode des couches.
<MULTICOL>	Cette commande permet au texte d'être affiché sur plusieurs colonnes.
<SPACER>	Introduit des blocs d'espaces vides.

6.2.2 Plus vicieux, des attributs différents de balises communes.

En voici une liste non exhaustive mais qui reprend les différences les plus spectaculaires. Attention, il est possible que des attributs incompatibles en version 3, soient devenus compatibles avec la 4.

BALISE	 Attributs Explorer seulement
<BODY>	leftmargin=x en pixel(s) : crée une marge à gauche. topmargin=x en pixel(s) : crée une marge dans le haut. bgproperties="fixed" : permet à l'image du fond de rester fixe.
<FRAME>	framespacing=x en pixel(s) : ajuste l'espace entre les cadres.
<HR>	color="#une_couleur " : spécifie une couleur à la ligne horizontale.

<TABLE>	background="une_image" : insère une image en arrière plan.
<TR>	bordercolor="une_couleur" : attribue une couleur à la bordure.
<TD>	bordercolorlight="une_couleur" : attribue une couleur à l'extérieur de la bordure.
	bordercolordark="une_couleur" : attribue une couleur à l'ombre de la bordure.

BALISE	 Attributs Netscape seulement
<IMG>	align (alignement) peut prendre les valeurs : texttop, absmiddle, baseline.
	border=x pixel(s) : permet d'avoir une bordure autour de l'image.
	hspace=x en pixel(s) : spécifie l'espace à laisser à droite et à gauche de l'image.
	vspace=x en pixel(s) : spécifie l'espace à laisser en haut et en bas de l'image.
	lowsrc="image_2" : permet de charger en premier lieu une image de faible résolution et une fois que toutes les informations de la page sont chargées, l'image à plus grande résolution (définie de façon classique) est alors chargée.
<OL>	type=... spécifie un type de puces : A, B, C / a, b, c / i, ii, iii. / I, II, III / 1, 2, 3
<LI>	start=x : permet de démarrer la liste au nombre indiqué.
<UL>	type=disc,circle ou square : spécifie un type pour les puces.
<TABLE>	cellspacing=x : espace entre les cellules du tableau.
	cellpadding=x : espace entre la bordure et le contenu de la cellule du tableau.

Evitez d'utiliser les éditeurs de Microsoft (FrontPage ou FrontPage Express) et de Netscape (Editor ou Composer) car ils utilisent souvent leurs propres balises. A proscrire donc... ou à user avec discernement.

6.3. Compatibilité des graphiques

6.3.1 Conseils usuels

- Utilisez l'attribut "alt" de la balise image pour les browsers dont l'option graphique n'est pas activée.
- Préférez toujours le codage hexadécimal des couleurs plutôt que le nom de couleurs.
- Prenez la main sur l'arrière-plan par défaut du browser de votre visiteur par l'attribut "bgcolor" de la balise <BODY>.
- Utilisez le plus possible des images en 16 couleurs. Cela diminue le temps de chargement et le risque de problèmes d'affichage.
- Testez votre site sous diverses résolutions d'écran 640x480, 800x600 et 1024x768 [pour rappel, sous Windows Panneau de configuration/Affichage/Configuration/Espace du bureau].Certains prétendent que la résolution 800x600 est la plus commune sur le Web mais d'après d'autres sources, c'est la résolution 640x480 qui l'emporte. Alors ?...

6.3.2. Pour les spécialistes

Vous avez peut-être remarqué que vos superbes images, visionnées en interne sur votre ordinateur, deviennent ternes voire banales, une fois que vous visitez votre site à partir du Net. Plusieurs éléments peuvent intervenir :

- Le nombre de couleurs traitées sur votre machine n'est pas forcément identique à celui de l'ordinateur de votre lecteur. Les couleurs ont un format de 8,16 ou 24 bits, ce qui correspond à une palette de 256, 65.536 et 16.777.216 couleurs [pour rappel sous Windows, Panneau de configuration/Affichage/Configuration/Palette de couleurs].

  Votre superbe image jpeg chouchoutée sur votre ordinateur en 16 millions de couleurs sera peut-être vue sur un ordinateur avec un affichage de 256 couleurs. Le système adapte alors les couleurs de votre image selon les couleurs les plus proches de sa palette. Ce qui va entraîner une perte inévitable et incontrôlée de qualité.

- Du temps où les systèmes Windows et Macintoch étaient en réelle concurrence, ils ont retenu chacun leur propre palette de 256 couleurs.

Et hélas (trois fois), il n'y a que 216 couleurs communes à ces deux palettes. A l'époque, Netscape a, assez logiquement, adopté pour son Navigator, une palette avec ces 215 couleurs dites "sûres" [Net safe colors]. Les autres couleurs sont obtenues par un effet de "tramage" qui peut à son tour modifier la qualité de votre image.

Qu'est-ce que ce tramage [dither, dithering] ? Prenons la couleur orange. Mais vous n'avez à votre disposition que des stylos de couleur rouge et jaune. Pour illustrer la couleur orange, la seule solution est de remplir la surface à colorier avec des petits points jaunes et rouges. C'est ce que fait le système lorsqu'il doit afficher une couleur qui n'est pas dans sa palette.

Il est important de noter que cet effet de "tramage" ne se produit pas si votre image est vue avec une palette supérieure en nombre de couleurs. Pour exemple, une image en 256 couleurs sur un système en 65.536 ou en 16 millions de couleurs. Voici la palette des 215 couleurs "sûres" [Net safe colors] pour les images du Web.

Vous pouvez télécharger à l'adresse "www.ccim.be/ccim328/htmlplus/compcons.htm" la palette netscape.pal (1K) ou la palette explorer.pal (1K) [miracle... pour une fois, c'est la même]. Palette que vous pouvez utiliser pour vos images Web, par exemple dans Paint Shop Pro avec le menu Couleurs/Charger une palette. En outre, évitez toujours lorsque vous diminuez le nombre de couleurs d'une image d'utiliser l'option de tramage.

- Et pour vous assommer tout à fait, si dans un affichage en 256 couleurs, vous utilisez 2 images. La première image ayant une première palette de 256 couleurs et la seconde image une autre palette de 256 couleurs différentes de celles de la première, le système ne pourra utiliser, pour la seconde image, que les couleurs de la deuxième images qui sont communes à la première palette. Les autres couleurs seront converties selon les couleurs les plus proches de la première palette... Méfiez-vous donc aussi de l'option "couleurs optimisée" car elle crée une palette différente pour chaque image.

6.4. Moralité

N'oubliez pas qu'**en Html, vous n'avez jamais le contrôle absolu de votre document** (comme avec votre traitement de texte par exemple).

Les formulaires

1. L'interactivité (cgi / mailto / Javascript)

1.1 L'interactivité, mais comment ?

Avec les formulaires, Html vous ouvre les portes de l'interactivité et vous permet de recevoir des informations provenant directement de votre lecteur et éventuellement de lui répondre directement.

Mais encore fallait-il trouver le moyen de renvoyer cette information! En effet, lorsque vous surfez, vous demandez à votre provider (serveur connecté au Web) de vous envoyer la page se trouvant à l'adresse (URL) indiquée. Et vous consultez la page, sur votre ordinateur, en mode **read only** soit en lecture seule.

Pour sortir de la page ou de votre ordinateur, les moyens disponibles sont :

- permettre à certains utilisateurs triés sur le volet, généralement des professionnels, d'écrire sur le serveur et d'en exploiter les ressources. Tout ceci dans les meilleures conditions de sécurité pour le provider. C'est la procédure des CGI.

- utiliser une autre ressource d'Internet disponible pour l'utilisateur, c'est le courrier électronique ou l'e-mail. C'est la procédure mailto.

- une dernière procédure (à laquelle je tiens) qui permet de transférer les informations en interne, à l'intérieur d'une page ou d'un site Web, C'est le Javascript.

1.2. CGI

La procédure de CGI [Common Gateway Interface] est quelque chose de complexe. Nous y consacrons un chapitre, plus loin dans ce site.

Citons quand même ici qu'une CGI est écrite dans un langage de programmation (C, C++, Perl...). Ce programme est alors chargé sur le disque dur du serveur dans un répertoire bien déterminé et qui, généralement, s'appelle
cgi-bin. L'accès à ce répertoire est limité par l'administrateur du serveur, pour d'évidentes raisons de sécurité, aux seuls utilisateurs habilités.
Une fois chargée sur le serveur, on peut y accéder de façon classique par l'URL "http:// www.nom du serveur/ cgi-bin/ ma_cgi.pl". Le programme de la CGI s'effectue en utilisant les ressources de l'ordinateur du serveur, éventuellement pour vous préparer une réponse immédiate (avec si le programme est mal conçu, un risque de plantage complet du serveur lui-même).

1.3 Mailto

Netscape avec Navigator 3 a imaginé le premier ce moyen d'exploiter les formulaires et qui présente l'énorme avantage de pouvoir être utilisé par tout un chacun. Depuis, cette possibilité a été reprise par Microsoft Explorer 4 et bien entendu Netscape 4 (Communicator).

Ainsi, la procédure d'envoi de formulaires par le protocole mailto, ne fonctionne pas sous Microsoft Explorer 3.0. Permettez-moi d'insister pour vous éviter des interrogations inutiles :

Mailto ne fonctionne pas sous Microsoft Explorer 3.0 !

Les formulaires présentent l'avantage par rapport au simple courrier électronique de pouvoir prédéfinir ou de structurer l'information qui vous intéresse. Cette prédéfinition des données est très utilisée dans les applications commerciales du Web

1.4. Javascript

Avec du Javascript, on peut utiliser les formulaires pour transférer des informations à l'intérieur d'une page ou même à l'intérieur d'un site (par l'usage des frames). En outre, Javascript, propose des outils particulièrement adaptés pour la vérification des données introduites par l'utilisateur dans les formulaires avant l'envoi et le traitement de celles-ci.

Vous pouvez en savoir plus sur le Javascript ou sur les formulaires avec Javascript en consultant "Apprendre le Javascript" du même auteur (www.ccim.be/ccim328/js/index.htm).

2. Définition d'un formulaire

2.1. Définition

Avant de pouvoir utiliser les différentes sortes de formulaires (ligne de texte, liste déroulante, cases à cocher...), il faut déclarer au browser qu'il devra gérer des formulaires et ce qu'il devra en faire.

```
<FORM method="post" action="URL d'expédition" enctype="text/plain">
... les formulaires proprement dit ...
</FORM>
```

2.2. Commentaires

- L'attribut "method" vous offre le choix entre get et post. La différence entre ces deux méthodes repose sur la façon dont les données seront transmises au serveur et exploitées par celui-ci. Avec le temps, la méthode post s'est imposée car elle apparaît plus efficace et permet le traitement d'une quantité plus importante de données.

- L'attribut "action" spécifie l'adresse d'expédition des données.

  Dans le cas d'un traitement des données par une CGI, on spécifie le répertoire CGI du serveur et le nom de la CGI.
  <FORM method="post" action="http://www.serveur/cgi-bin/ma_cgi.pl">

  Dans le cas d'un envoi vers en adresse électronique (e-mail), on utilise le protocole mailto: suivi de l'adresse électronique de destinataire (généralement votre adresse e-mail).
  <FORM method="post" action="mailto:Vanlancker.Luc@ccim.be">
  (sans espace entre mailto: et l'adresse e-mail !)

- L'attribut "enctype" (optionnel) spécifie l'encodage utilisé pour le contenu du formulaire. Ce paramètre ne peut être utilisé qu'accompagné par la méthode post. Ainsi enctype="text/plain" encode le contenu du formulaire en format texte lisible par le destinataire. Cette option est particulièrement adaptée à l'action du type mailto.

- Il n'est pas inutile de prévoir l'attribut name="nom" si la page comporte plusieurs formulaires.

- Attention !!! Je suis certain que, emporté par votre impatience à encoder les formulaires, vous allez oublier la balise de fin </FORM>. Dans ce cas, à la visualisation dans le navigateur, rien ne sera affiché.

- Dans le cas de l'utilisation en interne des formulaires par du Javascript, les attributs method, action et enctype sont inutiles car on ne fait pas appel au serveur.

3. Ligne de texte

3.1. Définition

INPUT type="text" indique un champ de saisie d'une seule ligne. C'est assurément le formulaire le plus simple à mettre en oeuvre :

```
<FORM>
```

```
<INPUT type="text" name="nom" size="50">
</FORM>
```

3.2. Commentaires

- L'attribut name="nom du formulaire" est quasiment obligatoire car on n'utilise que rarement un seul formulaire. Le nom va identifier la chaîne de caractères du champ de saisie. De façon schématique, nom = (ce qui est introduit dans la ligne de texte).
- L'attribut size (optionnel) définit la longueur du champ de saisie. Notons que l'on peut introduire un nombre de caractères plus élevé que celui de la longueur.
- Il existe l'attribut maxlength="x" (optionnel) qui limite le nombre réel de caractères que l'on peut introduire dans le champ de saisie. La balise <INPUT> n'a pas de balise de fin.

4. Zone de saisie

4.1. Définition

La balise <TEXTAREA></TEXTAREA> introduit une zone de texte multilignes et non plus une simple ligne de texte. La syntaxe est :

```
<FORM>
<TEXTAREA name="nom" rows=4 cols=40>Valeur par défaut</TEXTAREA>
</FORM>
```

4.2. Commentaires

- L'attribut name permet de donner un nom au formulaire.
- L'attribut rows=x détermine le nombre de lignes de la zone de texte.
- L'attribut cols=y détermine le nombre de caractères visibles sur chaque ligne ou si vous préférez le nombre de colonnes.
- L'attribut wrap (optionnel) détermine la façon dont les sauts de ligne seront traités lors d'un changement de ligne.
    - Avec wrap=virtual, les changements de lignes sont effectués automatiquement dans la zone de texte mais le tout sera transmis en une seule ligne.
    - Avec wrap=physical, les changements de lignes sont effectués automatiquement dans la zone de texte et ceux-ci sont également transmis.
    - Avec wrap=off, il n'y a aucun changement de ligne.
- Attention !!! La balise <TEXTAREA> a une balise de fin, soit </TEXTAREA>.

5. Liste déroulante

5.1. Définition

La balise <SELECT></SELECT> indique au browser l'usage d'une liste déroulante. Les éléments de la liste sont introduits par la balise <OPTION> ... (</OPTION> facultatif).

```
<FORM>
<SELECT name="nom" size="1">
<OPTION>lundi
<OPTION>mardi
<OPTION>mercredi
<OPTION>jeudi
<OPTION>vendredi
</SELECT>
</FORM>
```

Si vous cliquez sur la petite flèche vers le bas, vous obtiendrez la liste déroulante où on retrouve les éléments de la liste (<OPTION>).

5.2. Commentaires

- L'attribut name="nom" définit le nom du formulaire.
- L'attribut size="x" détermine le nombre d'éléments de liste affiché dans la boite d'entrée. En fait, size="1" est optionnel car "1" est la valeur par défaut. Le même exemple avec size="3" donne :

- On peut présélectionner l'élément affiché dans la boite d'entrée (par défaut, le premier élément de la liste sera retenu). On utilise pour ce faire l'attribut selected de la balise <OPTION>. Pour faire afficher d'entrée mercredi au lieu de lundi notre exemple devient :

```
<FORM>
<SELECT name="nom" size="1">
<OPTION>lundi
<OPTION>mardi
<OPTION selected>mercredi
<OPTION>jeudi
<OPTION>vendredi
</SELECT>
</FORM>
```

- La balise <SELECT> a un balise de fin </SELECT> tandis que le balise de fin </OPTION> est facultative.

6. Bouton d'option

6.1. Définition

Il serait plus logique de parler de boutons d'option car ils n'ont de sens que s'ils sont plusieurs. Ainsi on parle d'un groupe de boutons d'options.

Les boutons d'option, aussi appelés boutons radio, ont comme particularité qu'une seule option à la fois peut être activée (le "ou" exclusif).

La syntaxe de base est :

```
<FORM>
<INPUT type="radio" name="nom du groupe" value="valeur du bouton">
</FORM>
```

Ainsi, si on propose un choix entre, ou le tarif de jour ou le tarif de nuit ou le tarif de week-end, l'exemple devient

```
<FORM>
<INPUT type= "radio" name="tarif" value="jour"> tarif de jour
<INPUT type= "radio" name="tarif" value="nuit"> tarif de nuit
<INPUT type= "radio" name="tarif" value="week-end"> tarif de week-end
</FORM>
```

○ tarif de jour ○ tarif de nuit ○ tarif de week-end

Commentaires

- Vous avez compris que l'attribut name="nom" doit avoir le même nom pour tout le groupe de boutons d'option.
- L'attribut "checked" (optionnel) permet de présélectionner un bouton radio. Ainsi

  ```
  <INPUT type= "radio" name="tarif" value="jour" checked> tarif de jour
  ```

  présenterait le bouton radio "tarif de jour" en position présélectionnée.

  ◉ tarif de jour ○ tarif de nuit ○ tarif de week-end
- Le contenu de l'attribut "value" du bouton retenu sera renvoyé par mailto ou utilisé par le Javascript.
- La balise <INPUT> n'a pas de balise de fin.
- Pour la petite histoire le terme radio ferait référence aux anciens postes de radio sur lesquels le fait d'appuyer sur un bouton désactivait le bouton précédemment enfoncé.

7. Case à cocher

7.1. Définition

La philosophie des cases à cocher [checkbox] est assez similaire aux boites d'option. Ici, cependant, plusieurs choix simultanés peuvent être réalisés.

La syntaxe de base est :

```
<FORM>
<INPUT type="checkbox" name="nom" value="valeur attachée au bouton">
</FORM>
```

Comme exemple :

```
<FORM>
<INPUT type="checkbox" name="choix1" value="1"> glace vanille
<INPUT type="checkbox" name="choix2" value="2"> chantilly
<INPUT type="checkbox" name="choix3" value="3"> chocolat chaud
<INPUT type="checkbox" name="choix4" value="4"> biscuit
</FORM>
```

☐ glace vanille ☐ chantilly ☐ chocolat chaud ☐ biscuit

7.2. Commentaires

- Les règles pour l'attribut name="nom" sont moins précises que pour les boutons d'option. Vous pouvez employer des noms identiques ou des noms différents pour chaque case à cocher. Cependant des noms différents sont nécessaires pour l'utilisation d'un script.

- L'attribut checked (optionnel) permet de présélectionner une case à cocher. Ainsi

  ```
  <INPUT type="checkbox" name="choix1" value="1" checked > glace vanille
  ```

  présenterait la case à cocher "glace vanille" en position présélectionnée.

  ☑ glace vanille ☐ chantilly ☐ chocolat chaud ☐ biscuit

- Le contenu de l'attribut "value" du ou des boutons retenus seront renvoyés par mailto ou utilisé par le Javascript.

- La balise <INPUT> n'a pas de balise de fin.

8. Bouton de commande

8.1. Définition

Avec l'introduction des langages de scripts (Javascript et VBscript) l'usage du bouton de commande présente un intérêt certain. Simplement à titre d'illustration ou pour vous donner envie d'en savoir plus sur le Javascript, je vous propose d'en découvrir la syntaxe :

```
<FORM>
<INPUT type="button" name="nom" value="texte du bouton" onclick="fonction Javascript">
</FORM>
```

Voyons un petit exemple.

```
<FORM>
<INPUT type="button" name="nom" value= "Bouton de test" onclick="alert('Test réussi !')">
</FORM>
```

En cliquant sur le bouton "Bouton de test", on va déclencher une fonction Javascript élémentaire [mon cher Watson] qui consiste à afficher une petite boite (dite d'alerte) avec le texte "Test réussi!".

8.2. Commentaires

- L'attribut "value" permet d'adapter le texte du bouton à vos souhaits.

- Avec le bouton de commande, il n'est pas nécessaire d'avertir le browser qu'on utilisera du Javascript par une balise du genre <SCRIPT language="javascript">. En effet, le navigateur (compatible Javascript) reconnaît par défaut les fonctions en Javascript. Par contre, si vous utilisez du VBscript, il faudra utiliser l'encodage <INPUT type="button" name="test" value="Pour un test" language="VBscript" OnClick="MsgBox 'Test réussi!'"> (Pour rappel, Javascript par défaut). Mais je m'emporte et je m'éloigne ainsi des objectifs de ce chapitre...

9. Submit et Reset

9.1. Submit

Le bouton Submit a la tâche spécifique de transmettre toutes les informations contenues dans le formulaire à l'URL désignée dans les attributs ACTION et METHOD du tag <FORM>.

la syntaxe Html est :

```
<FORM>
<INPUT TYPE="submit" NAME="nom" VALUE="texte du bouton">
</FORM>
```

Soit par exemple :

```
<FORM>
<INPUT TYPE="submit" NAME="nom" VALUE=" Envoyer ">
</FORM>
```

Commentaires

- Les modifications seront peu nombreuses car le bouton Submit a une fonction Html très spécifique. Seul le texte du bouton pourra être modifié (par défaut Submit).

- La balise <INPUT> n'a pas de balise de fin.

9.2. Reset

Le bouton Reset permet d'annuler les modifications apportées aux contrôles d'un formulaire et de restaurer les valeurs par défaut.

la syntaxe Html est :

```
<FORM>
<INPUT TYPE="reset" NAME="nom" VALUE="texte du bouton">
</FORM>
```

Soit par exemple :

```
<FORM>
<INPUT TYPE="reset" NAME="nom" VALUE=" Annuler ">
</FORM>
```

Commentaires

- Les modifications seront peu nombreuses car le bouton Reset a une fonction Html très spécifique. Seul le texte du bouton pourra être modifié (par défaut Reset).

- La balise <INPUT> n'a pas de balise de fin.

10. Un livre d'or

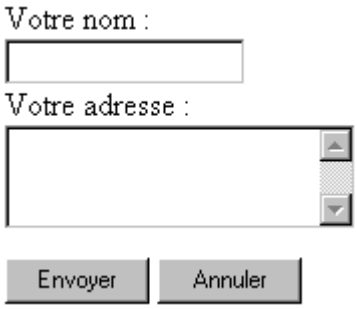
Une des premières applications des formulaires est celle d'un livre d'or [guestbook]. Vous trouverez ci-après un exemple complet de l'utilisation des formulaires par mailto et qui peut être adapté par chacun pour en faire "son" livre d'or.

N'oubliez pas que mailto ne fonctionne pas sous Microsoft Explorer 3.0 mais seulement sous Explorer 4.0 et Netscape 3.0 et plus. Pour être précis, mailto sous Explorer 3.0 n'entraîne pas de message d'erreur mais ouvre simplement une fenêtre de courrier électronique sans tenir compte du formulaire.

Pour ma part, j'ai retiré la page de livre d'or de mon site car à mon avis, pour un site d'amateur ou d'hobbyiste, une prédéfinition de l'information n'était pas forcément adaptée. Il me semble qu'une libre expression du lecteur (concise ou plus étendue) était de loin plus enrichissante. En outre, certains ne tiennent pas à transmettre leur nom ou leur domicile et souhaitent ne s'en tenir qu'à leur adresse émail.

Voici donc un exemple de livre d'or que vous pouvez adapter :

```
<FORM method="post" action="mailto:votre_adresse_E-mail">
Votre nom :<BR>
<INPUT type="text" name="nom"><BR>
Votre adresse :<BR>
<TEXTAREA name="adresse" ROWS=2 COLS=35>
</TEXTAREA><BR>
<INPUT type="submit" value="Envoyer">
<INPUT type="reset" value="Annuler">
</FORM>
```



Vous recevrez dans notre boite de réception, un truc bizarre du genre :

nom=Van+Lancker+Luc&adresse=Rue+des+Brasseurs+2217OD%OA7700+Mouscron.

- où on retrouve les champs nom= et adresse=, •où les champs sont séparés par le signe &,
- où les espaces sont remplacés par le signe +
- et 17%OD%OA correspond à un passage à la ligne.

Il existe heureusement des programmes comme "mailto: Formatter" pour vous aider à décoder tout ceci.

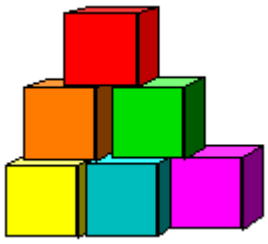
P.S.
Pour définir le sujet d'un mail, on utilisera la notation :
action="mailto:votre_adresse_Email**?Subject="Sujet du mail**">.
On peut aussi envoyer une copie :
action="mailto:votre_adresse_Email**?CC="autre_adresse_Email**">

Images mapées

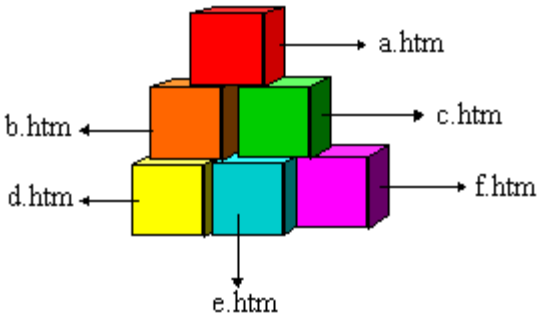
1. Le concept

Les images mapées permettent d'effectuer des liens en fonction de zones [area] prédéfinies de l'image. Cette particularité peut se révéler fort utile pour définir, par exemple, des outils de navigation dans votre site.

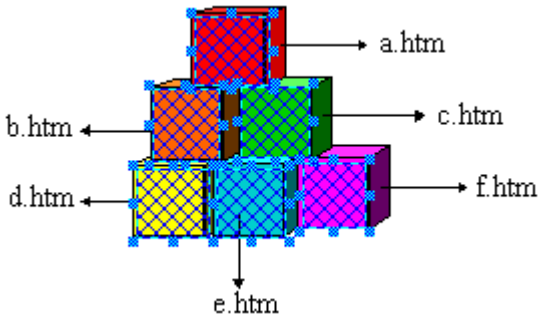
On prend une image.



Pour chaque zone retenue (ici un cube), on va associer un fichier.



Pour ce faire, on va définir des zones dans l'image, un peu comme avec une carte de géographie [map en anglais] et associer à chacune de ces zones, un fichier.



On obtient ainsi une image dite "mapée" car elle est découpée en zones à l'instar des cartes géographiques. On appellera la "map", les balises Html qui reprendront à la fois les zones et les fichiers associés.

Remarque

Il existe plusieurs méthodes pour réaliser cet effet d'images mapées : les méthodes NCSA, CERN et CSIM.

La méthode CSIM (Client Side IMage) est de loin la plus utilisée à l'heure actuelle car elle fait partie, à part entière, du langage Html (spécification Html 3.0). Avec cette méthode, les fichiers de map sont inclus dans la page Html et ne nécessitent pas l'appel de programmes additionnels (CGI) du serveur pour réaliser leurs fonctions.

2. Les balises expliquées

Les balises des images mapées peuvent sembler très bizarres pour l'utilisateur moyen. Cependant, une fois décodées, elles déterminent, en peu d'éléments, tout ce que le navigateur a besoin pour les traiter.

Nous utilisons toujours la méthode CSIM [Client Side Image Maps] qui fait partie du langage Html (Html 3.0).

2.1 La balise de l'image mapée

<IMG SRC="image.gif" USEMAP="#nom_du_map"> pour une map dans le même fichier
ou
<IMG SRC="image.gif" USEMAP="fichier.html#nom_du_map"> pour une map située dans un autre fichier.

En fait, on ajoute simplement à la balise classique de l'image, l'attribut USEMAP pour avertir le navigateur qu'il doit employer pour elle une map et le nom de la map en question.

Remarquez le système de # , propre aux ancres [anchor].

2.2. Les balises de la map

<MAP NAME="nom_du_map">
<AREA SHAPE=méthode COORDS="coordonnées" HREF="lien" ALT="commentaires"
TARGET="target_de_frame">
... autres balises AREA ...
</MAP>

Reprenons les éléments un par un :

L'attribut SHAPE

SHAPE =RECT ou CIRCLE ou POLYGON
RECT pour un rectangle.
CIRCLE pour un cercle.
POLYGON pour un polygone irrégulier.

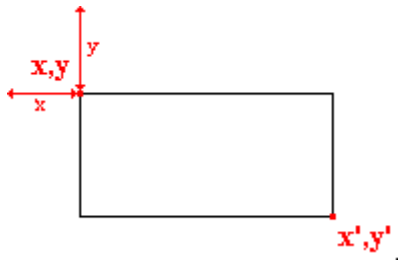
Netscape (pas Explorer) supporte également un quatrième type : DEFAULT qui définit ce qu'il faut faire si l'utilisateur clique hors d'une des zones AREA définies. Pour des raisons de compatibilité, son usage n'est pas conseillé.

L'attribut COORDS

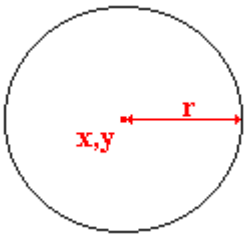
L'attribut COORDS note les coordonnées qui permettront au navigateur de reconstituer la forme géométrique.

Nous détaillons ci-après la façon, spécifique à chaque méthode, de noter ces coordonnées. La véritable difficulté pour le Webdesigner est de trouver les coordonnées des points dans l'image. C'est ici qu'un programme de traitement de l'image comme Paint Shop Pro, se révèle indispensable. Ou mieux encore un programme comme MapEdit ou MapThis (voir plus loin).

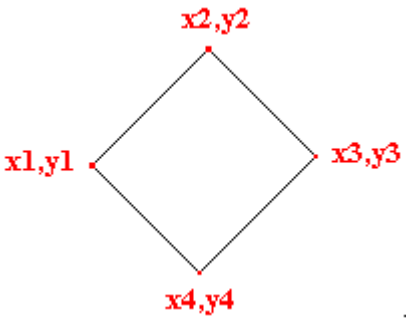
Pour un rectangle : COORDS="x,y,x',y'"



Pour un cercle
COORDS="x,y,r" soit le point central et le rayon



Pour un polygone
Coords="x1,y1,x2,y2,x3,y3,x4,y4" et ce pour autant de points qu'il y a dans le polygone.



L'attribut LINK

L'attribut LINK spécifie le fichier associé à la zone sélectionnée. L'adressage se fait de la façon tout à fait classique en HTML.

L'attribut ALT (facultatif)

L'attribut ALT permet d'ajouter un commentaire. Ce commentaire, tout comme l'attribut ALT des images, sera affiché par certaines versions récentes de navigateur.

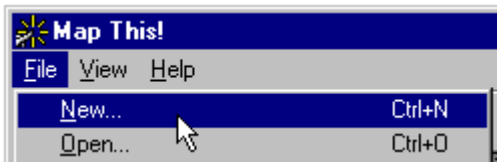
L'attribut TARGET (facultatif)

L'attribut TARGET permet de spécifier la fenêtre de frame dans laquelle doit s'ouvrir le fichier spécifié. Ceci nécessite, bien entendu, une connaissance pointue des frames (voir Apprendre le langage Html : les frames).

3. Avec Mapthis

Blabla sur Mapthis

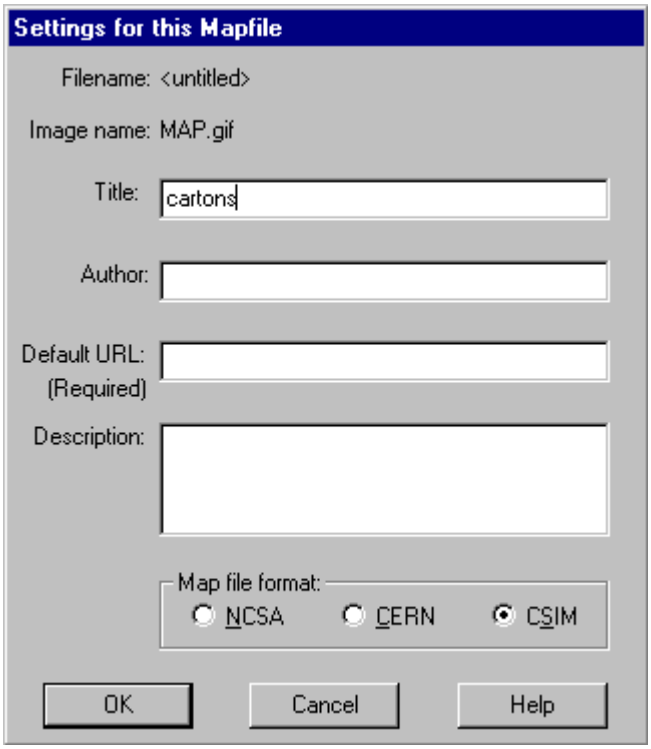
On ouvre le programme MapThis. On fait **File → New**.



On ouvre de façon classique l'image (gif) à "maper".

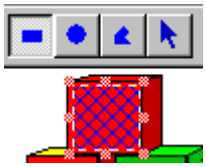
On détermine ensuite les renseignements généraux de la map (nom de la map, méthode).
Mapping → Edit Map Info.

On donne le nom de la map (ici "cartons") et la méthode (ici CSIM). Les autres mentions comme l'auteur, la description et l'URL par défaut (mais oui) sont facultatives.

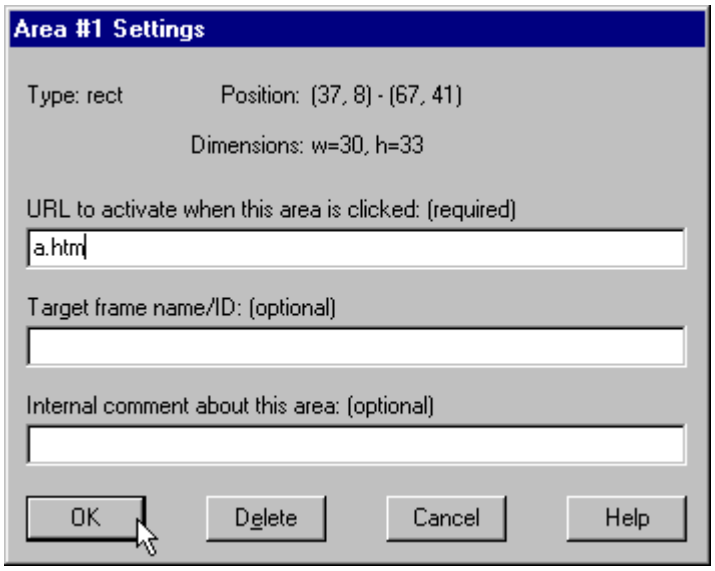


Attaquons nous maintenant aux zones. On sélectionne la forme [shape] désirée, ici le rectangle.

On applique la forme au premier carton.

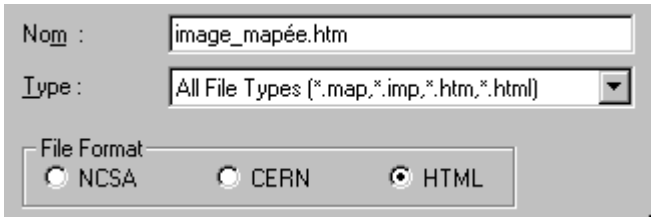


Un écran de zone [area] apparaît simultanément où vous indiquez le fichier associé à la zone (a.htm). les autres mentions sont optionnelles. Remarquez que la position des points de définition de la zone sont indiqués (ici 37,8,67,41)



On procède de même pour les autres zones.

Le travail réalisé, on sauve le fichier, **File → Save as**, sous forme Html.



On obtient ainsi un fichier au format Htm qu'il suffira de fusionner avec la page Html. Ce que nous verrons à la page suivante de ce chapitre.

Le fichier créé par MapThis (image_mapée) ressemble à ceci.

```
<BODY>
<MAP NAME="cartons">
<!-- #$-:Image Map file created by Map THIS! -->
<!-- #$-:Map THIS! free image map editor by Todd C. Wilson -->
<!-- #$-:Please do not edit lines starting with "#$" -->
<!-- #$VERSION:1.30 -->
<!-- #$DATE:Wed Nov 11 10:56:54 1998 -->
<!-- #$GIF:MAP.gif -->
<AREA SHAPE=RECT COORDS="37,9,72,40" HREF="a.htm">
<AREA SHAPE=RECT COORDS="18,46,46,79" HREF="b.htm">
<AREA SHAPE=RECT COORDS="61,43,93,78" HREF="c.htm">
<AREA SHAPE=RECT COORDS="9,84,36,119" HREF="d.htm">
<AREA SHAPE=RECT COORDS="48,85,77,116" HREF="e.htm">
<AREA SHAPE=RECT COORDS="89,81,123,115" HREF="f.htm">
</MAP>
</BODY>
```

Les lignes de commentaires pourront être effacées.

4. En final

Il ne nous reste plus qu'à inclure la map éditée avec MapThis dans notre page Html.

Soit le fichier Html suivant :

```
<HTML>
<BODY>
<CENTER>
<IMG SRC="cubes.gif" HEIGHT=121 WIDTH=136>
</CENTER>
</BODY>
</HTML>
```

Et la map de l'image "cubes.gif" :

```
<MAP NAME="cartons">
<AREA SHAPE=RECT COORDS="37,9,72,40" HREF="a.htm">
<AREA SHAPE=RECT COORDS="18,46,46,79" HREF="b.htm">
<AREA SHAPE=RECT COORDS="61,43,93,78" HREF="c.htm">
<AREA SHAPE=RECT COORDS="9,84,36,119" HREF="d.htm">
<AREA SHAPE=RECT COORDS="48,85,77,116" HREF="e.htm">
<AREA SHAPE=RECT COORDS="89,81,123,115" HREF="f.htm">
</MAP>
```

Cette inclusion se réalise en deux temps :

1. On ajoute l'information pour le browser USEMAP="#cartons" dans la balise IMG de l'image.
2. On inclus la map, n'importe où dans le fichier Html.

Le fichier final devient :

```
<HTML>
<BODY>
<CENTER>
<IMG SRC="cubes.gif" USEMAP="#cartons" HEIGHT=121 WIDTH=136>
</CENTER>
<MAP NAME="cartons">
<AREA SHAPE=RECT COORDS="37,9,72,40" HREF="a.htm">
<AREA SHAPE=RECT COORDS="18,46,46,79" HREF="b.htm">
<AREA SHAPE=RECT COORDS="61,43,93,78" HREF="c.htm">
<AREA SHAPE=RECT COORDS="9,84,36,119" HREF="d.htm">
<AREA SHAPE=RECT COORDS="48,85,77,116" HREF="e.htm">
<AREA SHAPE=RECT COORDS="89,81,123,115" HREF="f.htm">
</MAP>
</BODY>
</HTML>
```

Les Balises META et le référencement

1. Utilité

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

Conformément à la philosophie du Html, les éléments relatifs au document sont dissociés du corps du document lui-même et repris dans la zone d'en-tête (c-à-d entre les balises <HEAD> et </HEAD>). Les balises META apportent une série d'informations relatives à la teneur de votre page Web et sont donc toujours comprises entre les balises <HEAD>.

Les informations ainsi apportées intéressent :
- principalement le référencement de votre site et les moteurs de recherche.
- un peu vos lecteurs (avertis bien entendu).
- et accessoirement le browser ou le navigateur.

1.2. Les moteurs de recherche
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.

Au fil de vos "joggings" sur le Web, vous ne pouvez pas ne pas avoir rencontré des sites référencés de cette façon sibylline et peu informative :

No title
<FRAMESET><FRAME ...><FRAME ...></FRAMESET><NOFRAMES>Ce site comporte des frames</NOFRAMES>

ou si vous souhaitez une variante :

Untitled
<SCRIPT LANGUAGE="JAVASCRIPT"><!--function animate(form){if (!var). etc..//--></SCRIPT>

1.3. Les lecteurs
Ne vous est-il jamais arrivé, surfeur pressé ou économe, d'avoir enregistré un page jugée intéressante et d'avoir, par la suite, voulu désespérément retrouver l'adresse de la page d'accueil du site ou l'adresse e-mail de l'auteur ?

De judicieuses balises META peuvent, entre autres choses, offrir ce petit plus à vos lecteurs.

1.4. Le navigateur
Rares mais au combien spectaculaires, certaines balises META sont directement interprétées par le navigateur pour, par exemple, charger automatiquement un 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.

Mais on entre ici dans le déplaisant domaine du "Cela fonctionne sous le browser Machin version x mais pas sous le browser Chose version y"... Sans commentaires !

2. Les indispensables

2.1. <META NAME="keywords" CONTENT="mot-clé1,mot-clé2,mot-clé3,...">
S'il n'en fallait qu'une, ce serait assurément celle-là ! Tous [ou presque] les moteurs de recherche l'utilisent.

Quelques commentaires s'imposent cependant :
- Les mots-clés doivent être séparés par une virgule. L'auteur préconise de ne pas mettre d'espaces (bien que cela se rencontre chez des webmestres confirmés) afin d'éviter toute confusion lors des recherches par mot-clé.

- On parle dans la littérature d'une limitation à 1000 mots-clés. Plus qu'il n'en faut.
- On conseille aussi de mettre tous les mots clés en minuscules pour éviter les problèmes selon que le moteur de recherche tient compte de la "casse", c'est-à-dire de la différence entre majuscules et minuscules (en anglais « case sensitive » ou non)
-
- Tout l'art consiste à trouver les bons mots-clés relatifs au contenu de votre site. Mettez-vous à la place de vos lecteurs potentiels. Quels mots, quels synonymes, quelles alternatives peuvent être utilisés pour décrire votre site?
- De grâce, éviter de mettre des mots-clés "bidon" qui sont bien entendu très attirants mais sans rapport avec votre site.
- La tentation est grande de répéter un certain nombre de fois un même mot-clé pour espérer un meilleur classement. Par exemple, <META NAME="keywords" CONTENT="html,html,html,html,html>. Désolé pour vous mais ce truc a vécu et est maintenant pénalisé [spam] par les moteurs de recherche.

2.2. <META NAME="description" CONTENT="une brève description de la page">
Cette description sera pertinente, attirante et brève. En effet, selon les moteurs de recherche seuls les 150 à 240 premiers mots seront repris. Bavards s'abstenir.

2.3. <TITLE> ... </TITLE>
Les moteurs de recherche tiennent fortement compte des titres des documents. Il faut impérativement en mettre sur toutes les pages d'un site. Même pour les pages qui n'apparaissent pas directement comme les sous-pages d'une page de frames !

On dit également que le fait de reprendre un ou des mot(s)-clé(s) dans le titre de toutes les pages d'un site est très favorable pour un meilleur classement.

2.4. Et c'est tout pour les "indispensables"
Cela peut paraître léger mais si vous tenez compte de ces trois éléments, 90% d'un bon référencement seront ainsi réalisés. Les autres conseils font partie des "trucs et astuces" dont on ne peut assurer l'efficacité. En effet, dès qu'on tente de pousser plus avant une étude sur le réel fonctionnement d'un robot de recherche par rapport à un autre, on entre dans une zone de brouillard plus ou mois épais...

Nous reviendrons sur ces trucs et astuces du référencement plus loin dans ce chapitre.

3. Les utiles

3.1. <META NAME="author" CONTENT="nom de l'auteur">
Cette balise est d'une utilité discutable car rares sont les moteurs de recherche en tiennent compte. A ma connaissance, 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. N'est-il pas ?

3.2. <META NAME="Copyright" CONTENT="Copyright © date nom">
Il ne m'importe pas de discuter ici de la valeur juridique et pratique d'une mention de copyright sur le Web. D'autres sites s'en chargent.

Il me plaît cependant de noter que dans les e-mails reçus, il n'est pas rare que des étudiants ou des formateurs me demandent de pouvoir reproduire tout ou partie de mon site. Une règle élémentaire de la Netiquette est de respecter le copyright. "Le gratuit n'est pas forcément sans valeur".

3.3. <META NAME="Distribution" CONTENT="Global ou Local">
Cette balise indique la destination de l'information de la page. Soit qu'elle est "Global" et donc destinée à être largement diffusée soit qu'elle est "Local" et donc à diffusion restreinte.

3.4. <META NAME="Generator" CONTENT="nom de l'éditeur Html utilisé">

Aucune influence, ni pour les moteurs de recherche, ni pour le navigateur utilisé. Cette information intéresse cependant au plus au point les responsables marketing des sociétés proposant des éditeurs Html pour calculer la part de marché de leur produit. Alors, pourquoi pas si cela les amuse...

3.5. <META NAME="Rating" CONTENT="Destination de votre audience">
Permet de définir le contenu de votre site. Les appréciations sont General ou Mature ou Restricted ou 14 years pour respectivement tout public, adulte, accès restreint ou 14 ans.
Cet indice d'audience n'a aucune influence directe.

3.6. <META NAME="Robots" CONTENT="instructions pour les robots">
Par cette balise vous pouvez indiquez aux robots de recherche automatique si vous souhaitez que votre site soit ou ne soit pas indexé par eux.

Les instructions sont :
- All (défaut) permet aux robots d'indexer vos pages et de suivre les liens hypertextes d'une page à l'autre.
- None dira aux robots de ne pas indexer vos pages et de ne pas suivre les liens.
- Index indique que vos pages peuvent être indexées par les robots.
- NoIndex pour que le robots ne procèdent à aucune indexation.
- Follow donne la permission aux robots de suivre les liens hypertextes des pages
- NoFollow pour le contraire.

Exemples :
<META NAME="Robots" CONTENT="None">
<META NAME="Robots" CONTENT="Index,Nofollow">

Cette balise est reconnue par tous les robots de recherche automatique.

3.7. <META HTTP-EQUIV="Content-language" CONTENT="fr">

Cette balise déclare la langue utilisée dans le document Html. De plus en plus utile, maintenant que des monteurs de recherche anglo-saxons (et non des moindres comme Altavista et Hotbot) ont inclus la langue dans leurs critères de recherche.

3.8. <META HTTP-EQUIV="Reply-to" CONTENT="votre adresse e-mail">
Cette balise permet au lecteur averti de connaître votre adresse e-mail si elle n'apparaît pas sur la page qu'il consulte (peut-être hors connexion, si elle a été préalablement enregistrée). Finalement, il est peut-être plus utile de donner son adresse électronique que son nom.

3.9. <META HTTP-EQUIV="Reply-to" CONTENT="URL de votre page d'accueil">
Variante de la balise précédente. Il est parfois utile de rappeler l'adresse de la page d'accueil de votre site pour ceux qui y sont entrés par une des pages de celui-ci.

4. Les éventuelles

A côté des balises META indispensables à un bon référencement auprès de moteurs de recherche, à côté des balises META qu'il est bon d'ajouter pour la bonne information du lecteur, il y a encore celles que vous utiliserez de temps en temps lors de l'élaboration de sites. Parcourons en quelques uns.

4.1. Refresh
Un grand classique du Html (2.0 ?) même si cette balise META n'est que rarement utilisé et alimente les trucs et astuces depuis des années. Cette balise META charge automatiquement la page spécifiée à l'attribut URL 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 de votre site à été modifiée.

4.2. Transition

De très jolis effets de transition style PowerPoint sont possibles avec simplement une ligne de code. Mais autant vous le dire tout de suite, cela ne fonctionne **que sous 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 signifie que l'effet de transition se produira à l'entrée de la page ou à la sortie de celle-ci.
- Duration détermine la durée de la transition en secondes. Elle est dans l'exemple de 1 seconde. A l'usage, cette durée n'est pas d'une précision absolue.
- Transition est un nombre de 1 à 23 pour 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 fonctionneront très bien même si PowerPoint n'est pas installé sur la machine de votre visiteur.
- Et pour terminer, vous ne verrez les effets de cette transition que lorsque vous entrez dans la page à partir d'une autre page.

4.3. Expires

Cette balise META "dit" au navigateur la date à laquelle la page Html doit être considérée comme périmée. Avec Netscape, une requête pour un document dont la date est expirée initialisera une recherche sur le réseau au lieu de prendre la page éventuellement présente dans le cache. Ceci est très utile pour les pages fréquemment mises à jour.

Exemple :

```
<META HTTP-EQUIV="expires" CONTENT="Wed, 23 Feb 1999 10:49:02 GMT">
<META HTTP-EQUIV="expires" CONTENT="0">
```

Il faut noter que les robots de recherche peuvent retirer ces pages dites périmées de leur base de donnée.

4.4. Pragma

C'est une autre façon de contrôler le cache du navigateur. Avec ce META, vous pouvez demander au browser de ne pas tenir la page dans le cache.

```
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
```

On rapporte que cela fonctionne sous Netscape mais pas par Explorer...

5. Trucs et astuces de référencement

Alors que les balises META font partie du Html, il existe des tas de trucs et astuces - plus ou moins vérifiables ou vérifiés - qui sont sensés améliorer le rang de votre site parmi les centaines renvoyés lors d'une recherche par un mot-clé. Vérité ou rumeur... Je vous laisse seul juge.

5.1. Référencez-vous

Cela peut paraître une évidence mais si vous attendez qu'un robot de recherche vienne visiter votre site, vous risquez de devoir attendre longtemps... Hotbot, le moteur de recherche le plus complet estime lui-même que sa base de donnée ne reprendrait que 30 à 40% des sites du World Wide Web.

Alors préparez votre liste des moteurs de recherche, prévoyez quelques heures dans votre emploi du temps et au travail ! Sur la page d'accueil de chaque moteur de recherche, vous trouverez "Ajouter un site", "Référencement" ou quelque chose du même acabit. En cliquant sur le lien, vous ouvrirez un formulaire, plus ou moins long, à remplir pour référencer directement votre site.

N'oubliez pas que certains moteurs de recherche style annuaire (comme Yahoo) se réservent de droit de reprendre ou de ne pas reprendre votre site, qu'il faut compter généralement 2 à 3 semaines avant d'apparaître dans la base de donnée et un bon mois avant que les internautes commencent à s'intéresser à votre site. Patience donc ...

5.2. De l'importance des mots-clés

Pour obtenir un bon classement, il faut non seulement définir ces mots-clés dans la balise <META NAME="keywords" CONTENT="mot-clé1,mot-clé2,mot-clé3,..."> mais il est aussi recommandé :

- que le ou les mots-clés soi(en)t repris dans le premier paragraphe de la page.
- que le ou les mots-clés soi(en)t repris dans le titre de la page (<TITLE>).
- que le ou les mots-clés revienne(nt) plus fréquemment que les autres mots de la page.

5.3. La page d'accueil en frames

Les frames ne sont que modérément appréciés par les moteurs de recherche qui ne les tiennent pas ou difficilement en compte (voir tableau comparatif).

Souvent, emporté par le désir d'en finir au plus tôt avec ce fichier délicat, on s'empresse d'écrire les balises de frame soit : <FRAMESET ROWS="30%,70%"> <FRAME SRC="A.htm"> <FRAMESET COLS="30%,70%"> <FRAME SRC="B.htm"> <FRAME SRC="C.htm"> </FRAMESET> </FRAMESET>.

On en oublie les règles les plus élémentaires du référencement.
On oublie généralement de mettre les balises META, plus que jamais indispensables pour les moteurs de recherche.

On néglige de mettre un titre, à priori inutile car il n'apparaît pas, aux fichiers liés dans le cadre (ici a.htm, b.htm, c.htm). On perd ainsi, des points précieux pour le classement du site parce qu'il y a des pages sans titre et sans référence au mots-clés dans le titre.

Les moteurs qui ne suivent pas les liens dans les frames se rabattent sur le contenu des balises <NOFRAMES> ... </NOFRAMES>. Avouez que le classique "Désolé, ce site comporte des frames" ne leur apporte pas grand chose à se mettre sous la dent pour votre référencement ! On conseille de reprendre dans les balises <NOFRAMES> un petit texte descriptif du site (avec si possible un rappel des principaux mots-clés). Le "top" est d'y mette aussi un lien vers les pages principales du site.

5.4. La page d'accueil avec une image

Il est parfois joli de concevoir une page d'accueil composée d'une seule image que le visiteur doit cliquer pour entrer dans le site. Cette situation est assez inconfortable pour le robot à la recherche de mots-clés car il n'a alors aucun texte (et donc de mots) pour faire son référencement...

Il importe dans ce cas extrême de reprendre le mot-clé dans l'attribut ALT de la balise de l'image.

5.5. La page d'accueil avec une image mapée

Pour rappel, les images mapées sont des images découpées en zones. Selon les zones cliquées par le lecteur, celui-ci est dirigé vers une page Html déterminée.
Il est assez risqué, pour obtenir un bon référencement, de prévoir une page d'accueil qui ne comporterait qu'une image mapée comme seul outil de navigation pour votre site. En effet, beaucoup de moteurs de recherche (et pas de moindres comme Hotbot, Excite, Lycos) ne poursuivent pas leur indexation dans les fichiers d'une image mapée. A réserver donc à d'autres usages à l'intérieur de votre site.

Cette caractéristique est reprise dans le tableau comparatif des moteurs de recherche plus loin dans notre étude.

5.6. Et encore

Sont ou seraient bénéfiques pour un bon référencement :
- le fait que votre site soit référencé par d'autres sites.
- le fait que votre site soit mis à jour régulièrement.

5.7. Désolé...

Désolé, mais les trucs suivants ne fonctionnent plus et seraient même pénalisés [spam] par les moteurs de recherche :
- texte invisible dans la même couleur que le fond de la page.
- texte repris en commentaire (balises <!-- ... -->).
- les mêmes mot-clé repris indéfiniment dans la balise <META NAME="keywords" ....>.
- le texte dans l'élément de formulaire <INPUT type="hidden"...> (éléments cachés).

6. Tableau comparatif

Je n'ai pas su résister à la tentation de reproduire ci-après, un excellent comparatif de la prise en compte des balises META et des particularités de référencement des cinq plus grands moteurs de recherche anglo-saxons. Ce comparatif est bien entendu largement inspiré d'un auteur américain dont voici les mentions de copyright :

Prise en compte	**AltaVista**	**Hotbot**	**Excite**	**Infoseek**	**Lycos**
Longueur du titre (mots)	78	115	70	70	60
Si le titre est absent...	No title	Url de la page	Untitled	Premiers mots de la page	Premiers mots de la page
Longueur de la description	150	249	170-240	395	135-200
Balises META retenues	mots-clés description	mots-clés description	description	mots-clés description	aucunes
Balise ROBOTS	Oui	Oui	Oui	Oui	Oui

Prise en compte	**AltaVista**	**Hotbot**	**Excite**	**Infoseek**	**Lycos**
Texte de ALT des images	Oui	Non	Non	Oui	Oui
Texte en commentaire	Non	Oui	Non	Non	Non
Texte invisible	Non [spam]	Non [spam]	Oui	Non [spam]	Non [spam]
Frames	Yes	No	No	No	No
Images mapées	Oui	Non	Non	Oui	Non
Liens dans d'autres sites	Non	Oui	Oui	Non	Oui
Date	Oui	Oui	Non	Oui	Non
Mises à jour	Oui	Oui	Non	Non	Oui

Les feuilles de style

Chapitre 1: Concept et utilité

1.1 Présentation

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 son Internet Explorer 3.0 (mais elles existaient déjà avec Arena sous Unix), ces feuilles de style n'ont connu qu'un intérêt limité du fait que celles-ci n'étaient pas reconnues par Netscape Navigator 3.0.

Depuis les choses ont bien changé. D'abord les browsers 4.0 de Microsoft et de Netscape reconnaissent tous deux les feuilles de style et surtout, la norme Html 4.0 en a repris le concept (CSS version 1) et le recommande d'ailleurs vivement aux "Web designers".

1.2 Concept

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, les noms de chapitres seront mis en police Arial, en gras et en couleur bleue.

On peut imaginer que l'on puisse donner à cette définition de mise en forme un nom soit "titre" et qu'à chaque nouveau chapitre, plutôt que d'écrire chaque fois le nom du titre et puis de devoir le mettre en Arial, gras, bleu, l'on puisse dire à l'ordinateur, nom du chapitre mais dans la mise en forme que j'ai défini sous le nom de "titre". Cette définition de mise en forme particulière, on va l'appeler feuille de style.

Le concept de feuilles de style [Style Sheets] est né. Il existait déjà dans les traitements de texte comme dans Word de Microsoft (comme par hasard...). Allez dans le menu Format de Word, vous y trouvez Style ! Il ne restait plus qu'à coupler le concept au langage Html par des propriétés spécifiques.

<H1><B><FONT COLOR=blue>Titre1</FONT></B></H1>	STYLE des titres
<H2><B><FONT COLOR="green">- A -</FONT></B></H2>	STYLE des sous-titres
<H3><B><FONT COLOR="red">...a....</FONT></B></H3>	STYLE du texte
<H1><B><FONT COLOR=blue>Titre2</FONT></B></H1>	STYLE des titres
<H2><B><FONT COLOR="green">- B-</FONT></B></H2>	STYLE des sous-titres
<H3><B><FONT COLOR="red">...b....</FONT></B></H3>	STYLE du texte

Vous remarquez que l'on parle de feuilles de style [style sheets] car le but du jeu est d'en définir plusieurs. On parle aussi de feuilles de style en cascade [Cascading Style Sheets ou CSS] car en cas de styles identiques, un ordre de priorité sera déterminé par le browser (voir FAQ).

Précisons pour terminer que les feuilles de style ne sont pas une composante directe du langage Html mais un développement à part dans la publication de pages Web.

1.3 Utilité et avantages

- Séparation du contenu et de la mise en forme.
- Cohésion de la présentation tout au long du site avec les feuilles de style externes.
- Modifier l'aspect d'un 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 par rapport au Html et à ses différentes versions.
- Une façon d'écriture concise et nette par rapport au Html qui devient vite fouillis.
- Réduire le temps de chargement des pages.
- Palier certaines 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.

- Permettre le positionnement au pixel près du texte et/ou des images sans passer par les "layers" exclusifs à Netscape 4.0.

1.4 Compatibilité

Les feuilles de style fonctionnent sous :
Explorer 3.0 mais de façon incomplète
Explorer 4.0
Netscape 4.0

Attention !!! Les feuilles de style ne sont pas reprises par Netscape 3.0.

Chapitre 2 : Définition d'un style

La définition de base d'un style est simple :

balise { propriété de style: valeur; propriété de style: valeur }

Exemple :

H3 { font-family: Arial; font-style: italic }

Donc ici, la balise H3 sera en Arial et en italique. Et dans votre document, toutes les balises <H3> auront comme style Arial et italique.

Simple! Mais de nombreux commentaires s'imposent :

- Les feuilles de style portent sur des balises principalement et quelques autres éléments comme par exemple A:link pour un lien non-visité et A:visited pour un lien visité. Comme balises souvent utilisées avec des feuilles de style, on peut citer les en-têtes Hn, P, BODY...
- Les propriétés de style sont entourées par des "{" et pas des [ou des parenthèses.
- Le couple "propriété de style/valeur" est séparé par un double-point (:).
- Chaque couple "propriété de style/valeur" est séparé par un point-virgule (;).
- Il n'y a pas de limite pour le nombre de couples "propriétés de style/valeur".
- L'espace entre propriétés de style et valeur n'est pas obligatoire mais aide fortement à la lisibilité du code source.
- Pour la lisibilité toujours, vous pouvez écrire vos 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}

Chapitre 3 : Styles internes

Il faut maintenant incorporer les styles dans le document Html. Commençons par le plus simple, soit l'incorporation à l'intérieur d'une page. D'où le titre "Styles internes".

3.1 A l'intérieur des balises <HEAD></HEAD>

Cette façon de procéder est de loin la plus commune et la plus logique. D'abord parce que les balises HEAD contiennent des informations pour le browser et les feuilles de style appartiennent à celles-ci. Ensuite parce que l'on rejoint ainsi l'essence même des feuilles de style qui 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 que l'on va utiliser des feuilles de style.
- L'attribut type="text/css" informe que ce qui suit est du texte et qu'il s'agit de cascading style sheets (css). Pour l'instant, il s'agit de la seule possibilité mais on peut prévoir à l'avenir d'autres versions de ce "langage".
- La balise Html de commentaires <!-- ... --> empêche que les browsers qui ne connaissent pas les feuilles de style, tentent d'interpréter ces instructions. Les informations à l'intérieur des tags de commentaires seront ignorées par ces browsers.
- Pour vos propres commentaires à propos des feuilles de style, on utilisera la convention désormais classique (C, C++, Javascript...) de /* commentaires */.

3.2 A l'intérieur des balises <BODY></BODY>

On peut aussi utiliser, au coup par coup, les feuilles de style dans le corps (BODY) du document. Cette façon de faire nous paraît illogique et peu conforme à l'esprit des feuilles de style qui est de définir un style déterminé valable pour la globalité du document. Mais elle existe pour quelques utilisations spécifiques...

```
<HTML>
<BODY>
<H1 style="font-family: Arial; font-style: italic"> blabla </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.
- que l'écriture : <STYLE type="text/css">H1 { "font-family: Arial; font-style: italic" }</STYLE> fonctionne aussi.

Chapitre 4 : Styles externes

C'est déjà bien de pouvoir définir une présentation de style valable pour une page (styles internes). Mais CSS nous propose mieux. Définir une présentation de style valable pour plusieurs pages et même pour toutes les pages d'un site. Ce qui est possible, en créant une page externe qui regroupera toutes les feuilles de style, et en reliant chaque page à cette page de style.

On crée d'abord, dans le répertoire du site, un fichier avec l'extension .css soit styles.css qui contiendra toutes les feuilles de style.

```
<HTML>
<HEAD>
--- Les différentes feuilles de style ---
</HEAD>
<BODY>
</BODY>
</HTML>
```

Ensuite, on crée une page normale soit page1.htm (bien entendu dans le même répertoire que le fichier styles.css).

```
<HTML>
<HEAD>
<LINK rel=stylesheet type="text/css" href="styles.css">
<HEAD>
```

- La balise <LINK> avertit le browser qu'il faudra réaliser un lien.
- L'attribut rel=stylesheet (sans s et sans guillemets) précise qu'il y trouvera une feuille de style externe.
- L'attribut type="text/css" précise que l'information est du texte et du genre cascading style sheets (css).
- L'attribut classique de lien href=" ... " donne le chemin d'accès et le nom du fichier à lier.

Chapitre 5 : Les classes et les ID

5.1 Notion de classes

Mais on désire parfois affecter des styles différents à une même balise. Pas de problèmes, les feuilles de style vous proposent la solution des classes [class].

La syntaxe est ici aussi des plus simple.

La définition d'un style était :
balise { propriété de style: valeur }

Elle devient :
balise.nom_de_classe { propriété de style: valeur }
remarquez le point entre balise et nom_de_classe

Ou, comme la mention de la balise est facultative,
.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.
<balise class="nom_de-classe"> .... </balise>

Un exemple ?

Je souhaite mettre ce qui est important dans le texte en gras et en bleu. Je crée la classe .essentiel :
.essentiel { font-weight: bold; font-color: #000080 }

Et dans le document Html, il suffit d'appeler la classe .important quand cela se révèle nécessaire :

```
<P class=".essentiel"> ... blabla ... </P>
<H1 class=".essentiel">Titre 1</H1>
<TABLE><TR><TD class=".essentiel">cellule</TD></TD>...
```

5.2 Notion des ID

Comme le convention nom/point/nom est utilisée aussi en Javascript (voir Apprendre le Javascript du même auteur : www.ccim.be/ccim328/js/index.htm) , il a fallu trouver une autre convention d'écriture lorsqu'on désire utiliser les feuilles de style avec du Javascript. Ce sont les ID, aussi appelés les identifiants.
Les ID fonctionnent exactement comme les classes. Pas mieux, pas plus. C'est la même chose!

La syntaxe est :
#nom_de_ID { propriété de style: valeur }

Et pour l'appeler :

<balise id="nom_de_ID"> .... </balise>

Notons qu'on ne pourra effectuer qu'un seul appel à #nom_de_ID par page. Ainsi,
Pour #essentiel{ ... }
<P id=essentiel> est correct.
Mais si on rencontre dans la même page
<H1 id=essentiel> ce n'est plus correct !

Si vous pensez utiliser des feuilles de style, mais sans vous compliquer la vie avec des scripts, oubliez au plus vite ID et utilisez exclusivement les classes.

Si par contre vous souhaitez utiliser des scripts avec les feuilles de style pour faire du DHTML par exemple (voir plus loin dans le site), la notion de ID vous sera alors indispensable.

Chapitre 6 : <SPAN> et <DIV>

6.1 Utilité

Dernier point, il fallait penser à un système pour "déconnecter" certains morceaux de paragraphe ou plusieurs paragraphes de cette logique d'écriture avec des feuilles de style. Ce sont respectivement les balises SPAN et DIV qui créaient ainsi des petits blocs particuliers dans le document sans devoir repasser par les éléments structurels du Html classique.
Notons que SPAN et DIV s'utilisent toujours avec les classes et les ID.

6.2 SPAN

La balise <SPAN> ... </SPAN> permet d'appliquer des styles à des éléments de texte d'un paragraphe ou si vous préférez à un morceau de paragraphe. Ainsi je voudrais écrire :

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> ... </DIV> permet de regrouper plusieurs paragraphes ou si vous préférez, 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>
```

Donne comme résultat :

La balise <DIV>

Commentaire :

N'oubliez pas l'attribut class!

Chapitre 7 : Positionner avec CSS

Quel concepteur de pages Web n'a pas laissé échapper quelques jurons bien sentis en essayant, à grand renfort de tableaux, de placer précisément du texte ou une image là où il le désirait ?

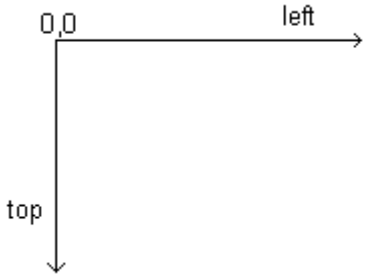
Le miracle est arrivé ! Outre le balise <LAYER> (mais qui n'est comprise que par Netscape 4.0), il est désormais possible de positionner, au pixel près, du texte ou une image avec les feuilles de style.
Notons que ce positionnement n'est possible que sous les versions 4 de Netscape et d'Explorer. Et que cette technique est encore un peu hasardeuse à ce jour, surtout sur le plan de la compatibilité avec les deux browsers susnommés.

Le positionnement des éléments par les feuilles de style est repris sous la spécification CSS-P.

7.1 Position absolue ou relative

La position absolue {position: absolute} se détermine par rapport au coin supérieur gauche de la fenêtre du browser. Les coordonnées de ce point sont top = 0 et left = 0. Les coordonnées d'un point s'expriment en pixels, de haut en bas pour top et de gauche à droite pour left.

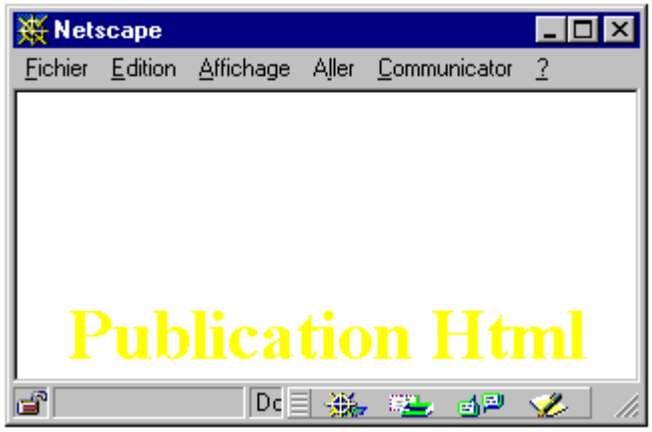
La position relative {position: relative} se détermine par rapport à d'autres éléments de la page, par exemple un élément du code Html.



Précisons que l'on utilisera presque toujours le positionnement absolu car plus facile et plus sûr.

7.2 Positionner du texte

Plaçons en position absolue le texte "Publication Html" à 50 pixels du haut de la fenêtre (top) et à 150 pixels à gauche (left).




```
<HTML>
<HEAD>
<STYLE type="text/css">
.pub{position: absolute; top: 100px; left: 25px;
color: yellow; font-size: x-large; font-weight: bold;}
</STYLE>
</HEAD>
<BODY>
<DIV class=pub> Publication Html </DIV>
</BODY>
</HTML>
```

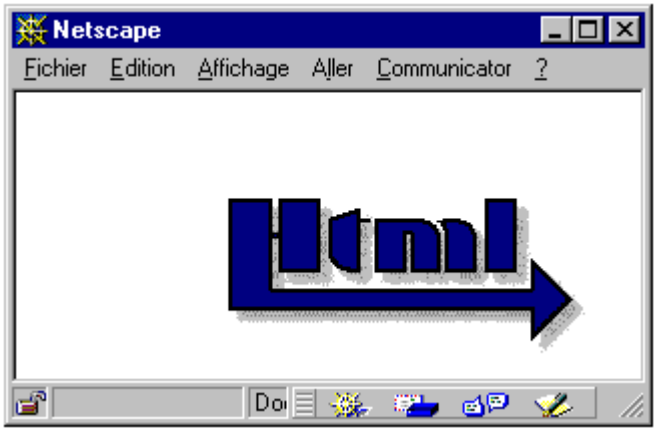
Ajoutons que plusieurs encodages sont possibles.

7.3 Positionner une image

Plaçons l'image htmlplus.gif en position absolue à 50 pixels de haut de la fenêtre (top) et à 100 pixels à gauche (left). Les dimensions de l'image sont width=242 et height=84.

```
<HTML>
<BODY>
<span style="position: absolute; top: 50px; left: 100px; width: 242px;
heigth: 84px;">
<IMG src="htmlplus.gif">
</span>
</BODY>
</HTML>
```

Spécifiez toujours les propriétés width et heigth avec les feuilles de style car par défaut, Netscape 4.0 et Explorer 4.0 ne réagissent pas de la même façon.



Ajoutons que plusieurs encodages sont possibles.

7.4 Superposer du texte sur une image

Reprenons l'image htmlplus.gif et on y superposera le nom de l'auteur de ce tutorial, au pixel près dans la barre qui souligne le terme Html.

```
<HTML>
<BODY>
<span style="position: absolute; top: 50px; left: 100px; width: 242px; heigth: 84px;">
<IMG src="htmlplus.gif">
</span>
<span style="position: absolute; top: 96px; left: 145px;
color: yellow; font-size: x-small; font-weight: bold;">
Van Lancker Luc
</span>
</BODY>
</HTML>
```

Ajoutons que plusieurs encodages sont possibles.

Chapitre 8 : Mini FAQ sur les styles

8.1 Les feuilles de style sont-elles "case sensitive" ?

Non, les feuilles de style ne sont pas sensibles à la case [case insensitive]. Ecrire CLASS ou class ou Class est donc équivalent. Cependant les éléments qui ne sont pas sous le contrôle des feuilles de style comme les noms

de police ou les URLs peuvent être case sensitive. Pour le système d'exploitation (et je pense à Unix), Arial n'est peut-être pas égal à arial, de même IMAGE.gif n'est pas forcément égal à image.gif.

8.2 Quels caractères peut-on utiliser en CSS ?

Les noms des feuilles de style, des sélecteurs, des classes et ID peuvent contenir les lettres de a-z ou de A-Z, les chiffres de 0-9, le trait d'union et le caractère _. 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 chez moi. On prendra vite l'habitude (voir les langages de programmation) de les éviter.

8.3 Peut-on utiliser dans la même page, à la fois des éléments de style CSS et Html ?

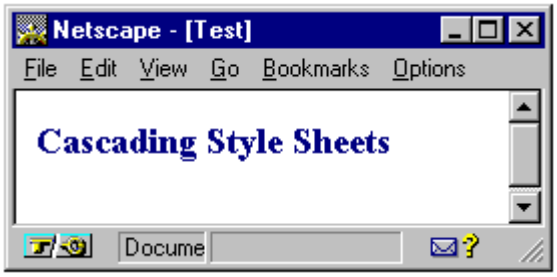
Oui. Les feuilles de style seront ignorées par les browsers qui ne supportent pas les feuilles de style. Et c'est tant mieux pour la compatibilité de votre site sous les différents navigateurs.

8.4 Qui l'emportent : les attributs Html ou les propriétés CSS ?

Les propriétés des feuilles de style l'emportent sur les attributs Html. Si les deux sont spécifiés, les attributs Html seront affichés avec les browsers qui ne supportent pas les CSS et n'auront aucun effet avec les browsers qui supportent les feuilles de style.

Ainsi, <H3 style="color: red"><B><FONT COLOR="navy">Cascading Style Sheets</FONT></B></H3> apparaîtra en rouge sous un browser compatible CSS (par exemple Microsoft Explorer).

et en bleu sous un browser qui ne reconnaît pas les feuilles de style (par exemple Netscape 3.0).



Ce qui est très bien pour la compatibilité mais qui ne simplifie pas la clarté de l'écriture.

8.5 Et s'il y a 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 possibilités de localisation de feuilles de style :

dans un fichier externe avec l'extension css.
dans la balise HEAD du document.

dans le BODY du document.

La règle de priorité appliquée par le browser sera d'appliquer pour la présentation du document la feuille de style la plus proche de l'élément. Ainsi, un style spécifié dans le BODY sera retenu par rapport à un style déclaré dans un fichier externe.

Il y a cependant moyen de contourner ces règles de priorité par la déclaration ! important;. Du genre BODY {background: white ! important; color: black}. Nous en resterons là dans le cadre de ce tutorial.

Chapitre 9 Liste des propriétés

La liste complète (et officielle) des propriétés et recommandations concernant les feuilles de style version 1, peut être trouvée à l'adresse http://www.w3.org/pub/WWW/TR/REC-CSS1 . En voici une sélection :

9.1. Les styles de police

font-family

définit un nom de police ou une famille de police
<nom> ou <famille>
police précise (Arial, Times, Helvetica...) ou
famille (serif, sans-serif, cursive, fantasy, monospace)
H3 {font-family: Arial}

font-style

définit le style de l'écriture
normal ou italique ou oblique
H3 {font-style: italic}

font-weight

définit l'épaisseur de la police
normal ou bold ou bolder ou lighter ou valeur numérique soit
(100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900)
P {font-weight: bold}

font-size

définit la taille de la police
xx-small ou x-small ou small ou médium ou large ou x-large ou xx-large
ou larger ou smaller
ou taille précise en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
P {font-size: 12pt}

font-variant

définit une variante par rapport à la normale
normal ou small-caps
P {font-variant: small-caps}

font

raccourci pour les différentes propriétés de police
P {font: bold italic}

9.2. Les styles du texte

text-align

définit l'alignement du texte
left ou center ou right ou justify
H1 {text-align: center}

text-indent

définit un retrait dans la première ligne d'un bloc de texte
souvent utilisé avec <P>, n'oubliez pas dans ce cas </P>.
spécifié en inches (in) ou en centimètres (cm) ou en pixels (px)
P {text-indent: 1cm}

text-decoration

définit une décoration (?) du texte, soit barré, clignotant, etc.
blink ou underline ou line-through ou overline ou none
A:visited {text-decoration: blink}

text-transform

définit la casse du texte (majuscule, minuscule)
uppercase (met les caractères en majuscules) ou
lowercase (met les caractères en minuscules) ou
capitalize (met le premier caractère en majuscule)
P {text-transform: uppercase}

color

définit la couleur du texte
par exemple en hexadécimal
H3 {color: #000080}

word-spacing

définit l'espace entre les mots
en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
P {word-spacing: 5pt}

letter-spacing

définit l'espace entre les lettres
spécifié en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
P {letter-spacing: 2pt}

line-height

définit l'interligne soit l'espace entre les lignes du texte
en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
P {line-height: 10pt}

width

détermine la longueur d'un élément de texte ou d'une image

en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
H1 {width: 200px}

height

détermine la hauteur d'un élément de texte ou d'une image
en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
H1 {heigh: 100px}

white-space

espace ou blanc
normal ou pre ou nowrap
PRE {white-space: pre}

9.3. Les arrière-plans

background-color

définit la couleur de l'arrière-plan
couleur (par exemple en hexadécimal) ou transparent
H1 {background-color: #000000}

background-image

définit l'image de l'arrière-plan
URL de l'image
BODY {background-image: image.gif}

background-repeat

définit la façon de répéter l'image d'arrière-plan
repeat ou no-repeat ou repeat-x (x = nombre de répétitions horizontales) ou
repeat-y (y = nombre de répétitions verticales
P {background-image: image.gif; background-repeat: repeat-4}

background-attachment

spécifie si l'image d'arrière-plan reste fixe avec les déplacements de l'écran
scroll ou fixed
BODY {background-image: 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
{1, 2}
{top ou center ou bottom , left ou center ou right}
ou en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
BODY {background-image: img.gif; background-position: right top}

background

raccourci pour les différentes propriétés d'arrière-plan
P {background: image.gif fixed repeat}

9.4. Les marges

margin-top

détermine la valeur de la marge supérieure
en unité de longueur ou auto
{ margin-top: 5px }

margin-right

détermine la valeur de la marge droite
en unité de longueur ou auto
{ margin-right: 5px }

margin-bottom

détermine la valeur de la marge inférieure
en unité de longueur ou auto
{ margin-bottom: 5px }

margin-left

détermine la valeur de la marge gauche
en unité de longueur ou auto
{ margin-left: 5px }

margin

regroupe les différentes propriétés de la marge

9.5. Les bords et les "enrobages"

border-top-width

donne l'épaisseur du bord supérieur
thin ou medium ou thick ou spécifié par l'auteur
H3 {border-top-width: thin}

border-right-width

donne l'épaisseur du bord droit
thin ou medium ou thick ou spécifié par l'auteur
H3 {border-right-width: medium}

border-bottom-width

donne l'épaisseur du bord inférieur
thin ou medium ou thick ou spécifié par l'auteur
H3 {border-bottom-width: thick}

border-left-width

donne l'épaisseur du bord gauche
thin ou medium ou thick ou spécifié par l'auteur
H3 {border-left-width: 0.5cm}

border-width

regroupe les différentes propriétés de border-width

border-color

détermine la couleur de la bordure
H3 {border-color: yellow}

border-style

détermine le style du trait de la bordure
none ou solid ou dotted ou dashed ou double
ou groove ou ridge ou inset ou outset
{border-style: solid dashed}

border

regroupe toutes les propriétés des bords

padding-top

valeur de remplissage haut entre l'élément et le bord
en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
H3 {padding-top: 3px}

padding-right

valeur de remplissage droite entre l'élément et le bord
en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
H3 {padding-right: 3px}

padding-bottom

valeur de remplissage bas entre l'élément et le bord
en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
H3 {padding-bottom: 3px}

padding-left

valeur de remplissage gauche entre l'élément et le bord
en points (pt), inches (in), centimètres (cm), pixels (px)
ou pourcentage (%)
H3 {padding-left: 3px}

padding

regroupe les différentes propriétés de remplissage

9.6. Les listes

list-style-type

détermine le type de puces ou de numérotation
disc ou circle ou square
decimal ou lower-roman ou upper-roman ou lower-alpha ou upper-alpha

OL {list-style-type: square}

list-style-image

permet de remplacer les puces par une image
url ou none
OL {list-style-image: image.gif}

list-style-position

spécifie si les puces sont à l'intérieur ou à l'extérieur du texte
inside ou outside
UL {list-style-position: inside}

list-style

regroupe toutes les propriétés de liste
