Chapitre 2

Le langage HTML

Nous avons vu comment mettre en place un serveur Web et dans quel dossier y déposer notre site Web. Voyons maintenant quelle doit être le forme d'une page Web. Celle-ci doit être écrite dans un langage appelé HTML.

HTML (*HyperText Markup Language*, soit langage de balisage hypertexte) est le format de données conçu pour représenter les pages web, créé en 1990, avec HTTP et les adresses Web, pour *WorldWideWeb*. Ce langage permet de décrire de l'hypertexte, d'où son nom. HTML permet également de structurer sémantiquement et de mettre en forme le contenu des pages, d'inclure des ressources multimédias dont des images, des formulaires de saisie, et des éléments programmables.

HTML est initialement dérivé de **SGML** (*Standard Generalized Markup Language*, soit langage de balisage généralisé standard).

2.1 Page de texte

L'intérêt de HTML est de créer des hyperliens mais des hyperliens entre quoi ? Entre des pages de texte pour débuter. Commençons par voir comment afficher une page de texte (bien présentée) dans la fenêtre active d'un navigateur.

2.1.1 Premier exemple

Commençons par écrire une première page qui affiche 'Bonjour'.

Écriture du fichier source.- Avec un éditeur de texte quelconque, écrivons le fichier texte suivant :

```
<HTML>
<HEAD>
<TITLE>Premier exemple</TITLE>
</HEAD>
<BODY>
Bonjour
</BODY>
</HTML>
```

Sauvegarde du fichier.- Sauvegardons notre fichier, par exemple, sous le nom `bonjour.html`. L'extension `.html` est classique depuis un premier fichier de Tim Berners-Lee mais n'est pas obligatoire.

Dépôt du fichier.- Pour pouvoir être accessible, ce fichier doit être déposé dans le répertoire racine adéquat du serveur Web. Nous avons vu au chapitre 1 qu'il s'agit de :

```
repertoire/htdocs
```

dans le cas du serveur Apache.

Accès à la page.- Si nous essayons d'accéder à la page sur la machine qui contient le serveur grâce à l'adresse :

```
http://localhost/bonjour
```

nous verrons 'Bonjour' écrit dans la fenêtre active du navigateur avec le titre 'Premier exemple' dans le bandeau.

Puisque nous nous trouvons sur l'ordinateur contenant le serveur, on peut aussi double-cliquer sur le nom du fichier HTML pour ouvrir le navigateur par défaut et afficher cette page. C'est également une façon de tester les pages si on ne dispose pas d'un serveur Web.

2.1.2 Structure d'un fichier HTML

Ce premier exemple nous montre la structure d'un fichier HTML.

Balise.- Une **balise** (*tag* en anglais) est un mot encadré par les caractères '<' et '>' (appelés *angle brackets* en anglais).

HTML définit un certain nombre de balises. Le mot à l'intérieur d'une telle balise est un mot-clé du langage HTML, dont la casse (majuscule ou minuscule) n'a pas d'importance mais qu'il est traditionnel d'écrire en majuscule pour plus de clarté.

Conteneur.- Un **conteneur** (*container* en anglais) est une portion de fichier HTML comprise entre deux balises appariées : la première contient un mot-clé, la seconde ce même mot-clé précédé d'une oblique '/'.

Le premier exemple nous montre les conteneurs `HTML`, `HEAD`, `TITLE` et `BODY`.

Délimitation d'un fichier HTML.- Comme le montre notre premier exemple, un fichier HTML est un conteneur `HTML`, c'est-à-dire qu'il commence par la balise `<HTML>` et qu'il se termine par la balise `</HTML>`.

Sections d'un fichier HTML.- Comme le montre notre premier exemple, un conteneur `HTML` comprend deux sections : un conteneur `HEAD` suivi d'un conteneur `BODY`.

Le conteneur en-tête `HEAD` est une portion « invisible » du code qui contient des informations administratives et tout code ou script particulier dont la page a besoin alors que le conteneur corps du programme `BODY` contient le code proprement dit.

Titre d'un fichier HTML.- Comme le montre notre premier exemple, l'en-tête peut contenir un conteneur titre `TITLE`, contenant une ligne de texte qui sera reproduite dans le bandeau du navigateur.

2.1.3 Structuration du texte

Un exemple

Nous avons vu comment afficher un texte dans la fenêtre active du navigateur grâce à un fichier HTML. Essayons d'afficher un texte plus long tel que la première page de notre chapitre 1. Si on écrit ce texte tel que (essayer de le faire), on n'obtiendra pas la structuration voulue (titre, titre de section, passage d'un paragraphe à l'autre, mise en évidence de certains mots) et on aura des problèmes avec les éléments diacritiques des lettres françaises. Il faut en fait l'écrire sous la forme ci-dessous :

```
<HTML>
<HEAD>
<TITLE>Mise en place d'un serveur Web</TITLE>
</HEAD>
<BODY>
<H1>Mise en place d'un serveur Web</H1>
<HR>
<H2>Le Web</H2>
<P>
Le <EM>World Wide Web</EM> (<B>WWW</B>), litt&eacute;ralement la &lt;&lt;
toile (d'araign&eacute;e) mondiale &gt;&gt;, commun&eacute;ment appel&eacute
le <B>Web</B>, parfois la <EM>Toile</EM>, est un syst&egrave;me hypertexte
fonctionnant sur Internet qui permet de consulter, avec un logiciel
appel&eacute <B>navigateur</B> (<EM>browser</EM> en anglais), des
informations appel&eacute;es <B>pages Web</B> accessibles sur des
sites (Web). L'image de la toile d'araign&eacute;e vient des
<B>hyperliens</B> qui lient les pages web entre elles.
</P>
<P>
Un <B>serveur HTTP</B> ou <B>d&eacute;mon HTTP</B> ou <B>HTTPd</B> (pour
<EM>HTTP daemon</EM>) ou (moins pr&eacute;cis&eacute;ment) <B>serveur Web</B>,
```

```
est un logiciel servant des requ&ecirc;tes respectant le protocole de
communication client-serveur <EM>Hypertext Transfer Protocol</EM>
(<B> HTTP</B>), qui a &eacute;t&eacute d&eacute;velopp&eacute pour
le World Wide Web.
</P>
<P>
Un ordinateur sur lequel fonctionne un serveur HTTP est &eacute;galement
appel&eacute <B>serveur Web</B>.
</P>

<HR>
<H2>Mise en place d'un serveur Apache</H2>
Le serveur Apache est un logiciel libre disponible &agrave l'adresse :
<P ALIGN=CENTER>
<TT>http://httpd.apache.org</TT>
</P>
<P>pour plusieurs syst&egrave;mes d'exploitation Unix. Il existe
une version pour Windows. C'est celle-ci que nous proposons d'installer
dans une premi&egrave;re &eacute;tape.
</P>
<P>
<U>Premi&egrave;re &eacute;tape : R&eacute;cup&eacute;ration des
binaires</U>.- Sur le site indiqu&eacute ci-dessus, cliquer sur
la colonne de gauche en haut sur &lt;&lt; <EM> Download!</EM> &gt;&gt;
(en fait sur le sous-menu <EM>from a mirror</EM>). Cliquer ensuite sur
la derni&egrave;re version stable (<EM>Stable release</EM>), la version
2.2.21 du 13 septembre 2011 au moment o&ugrave j'&eacute;cris ceci.
Cliquer enfin sur le lien qui suit <EM>Win32 Binary including OpenSSL
0.9.8r (MSI Installer)</EM>. On obtient le fichier :
</P>
<CENTER><TT>httpd-2.2.21-win32-x86-openssl-0.9.8r.msi</TT></CENTER>
<P>
&agrave placer dans le r&eacute;pertoire de votre choix.
</P>
<HR>
</BODY>
</HTML>
```

Lorsqu'on charge cette page, elle se présente comme le montre la figure 2.1, qui est ce que nous désirions.

Syntaxe

Commentons, en suivant notre exemple ligne à ligne, les aspects nouveaux un à un.

Titres de chapitre et de sections.- HTML nous fournit un certain nombre de tailles de texte pour les titres de chapitre, sections et sous-sections. Il s'agit des conteneurs `H` (pour l'anglais *Heading*), suivi d'un chiffre de 1 à 6, ce qui donne `H1` à `H6`. Notre exemple commence par un tel conteneur pour indiquer le titre du contenu de la page.

FIG. 2.1 – Présentation d'un texte structuré

Filet horizontal.- Il est intéressant de séparer les parties d'une page Web par des traits plus ou moins fins, appelés **filets horizontaux** (*horizontal rules* en anglais), comme celui qui apparaît après le titre. On utilise pour cela la balise `HR` (qui n'est pas appariée pour obtenir un conteneur).

On peut utiliser les attributs `WIDTH` (avec un nombre spécifiant la largeur en pixels comme dans `<HR WIDTH=500>` ou un pourcentage de largeur de la fenêtre, celle-ci pouvant être redimensionner par l'utilisateur, comme dans `<HR WIDTH=75%>`), `SIZE` (qui permet de spécifier la hauteur du filet en pixels) et `NOSHADE` (qui supprime l'aspect 3D du filet).

Lignes.- Un texte est formé de lignes. Mais puisque la largeur de la fenêtre active du navigateur dépend de la résolution de l'écran client et que l'utilisateur peut redimensionner celle-ci, il n'est pas astucieux de créer des lignes. On écrit le texte en continu, le navigateur fera passer à la ligne lorsque nécessaire.

Si on veut vraiment qu'il y ait un passage à la ligne à un moment donné (fin de paragraphe, par exemple ; mais nous y reviendrons ci-dessous dans ce cas), on peut utiliser la balise `BR` (pour

l'anglais *line BReak*), qui ne correspond évidemment pas à un conteneur.

Paragraphes.- Si la notion de ligne n'a pas de sens pour une page Web, celle de paragraphe en a bien sûr un. Nous avons vu qu'on peut utiliser la balise `BR` pour spécifier la fin d'un paragraphe. On utilise cependant le plus souvent la balise `P` qui laisse une ligne vide après le paragraphe.

On peut utiliser la balise `P` simple en fin de paragraphe ou comme délimiteur de conteneur encadrant un paragraphe. Cette dernière façon de faire permet de plus d'utiliser l'attribut `ALIGN` avec l'une des trois valeurs : `LEFT` pour justifier à gauche, `CENTER` pour centrer le texte et `RIGHT` pour justifier à droite. Par défaut, on a une justification à gauche.

On pourrait aimer avoir également une justification des deux côtés, comme dans le texte qu'on est en train de lire, mais cela n'est pas possible en HTML.

Pour une ligne centrée on peut également utiliser le conteneur `CENTER`.

Mise en évidence de mots.- On peut mettre en évidence certaines parties dy texte, par exemple en les mettant en gras (conteneur `B` pour l'anglais *Boldface*), en italique (conteneur `I` pour l'anglais *Italics*), en fonte monospace (c'est-à-dire que toutes les lettres, y compris 'i' et 'm', ont la même largeur, grâce au conteneur `TT`, pour *Telescripter Type* ou, plus simplement, *TeleType*), en souligné (conteneur `U` pour l'anglais *Underlined*), en mise en évidence (conteneur `EM` pour l'anglais *EMphasised*, c'est-à-dire en italique dans un texte en romain et en romain dans un texte en italique).

Caractères spéciaux.- Les lettres, chiffres et autres symboles qui peuvent être affichés tels quels dans une page HTML constituent le **jeu de caractères** (*character set* en anglais). Il s'agit en gros des caractères apparaissant sur le clavier (mais pas les caractères accentués français). De plus HTML se réserve quelques caractères pour un usage particulier, en particulier < et >.

Les autres caractères sont affichés grâce à une **séquence d'échappement** (comme dans tout langage de programmation), c'est-à-dire un certain mot du jeu de caractères placé entre une esperluette '&' et un point-virgule ';'.

<	&lt ;	â	&acirc ;
>	&gt ;	à	&agrave ;
©	&copy ;	á	&aacute ;
®	&reg ;	ä	&auml ;
1/2	&frac 1 2 ;	æ	&aelig ;
		ã	&atilde ;
		ç	&ccedil ;
		å	&aring ;

Bien entendu ce qui est valable pour la lettre 'a' l'est également pour les autres lettres, y compris les majuscules.

HTML ignore les espaces multiples dans un texte, seul le premier espace a une importance. On peut utiliser `  ;` (pour l'anglais *No Break SPace*) pour insérer un espace (ou le nombre que l'on veut).

2.1.4 Des textes en couleur

Nous venons de voir comment afficher un texte (bien présenté) en noir et blanc et avec la fonte par défaut. On peut en fait introduire de la couleur et changer de fonte.

Exemple.- Affichons un texte en blanc sur un fond bleu dans lequel on a changé la couleur, la taille ou la fonte de certaines parties du texte :

```
<HTML>
<HEAD>
<TITLE>Essais de couleur</TITLE>
</HEAD>
<BODY BGCOLOR=BLUE TEXT=WHITE>
<H1>Essais de couleur et de fontes</H1>
<HR>
<P> On peut changer la <FONT COLOR=RED>couleur</FONT> d'une partie du texte</P>
<P> sa <FONT SIZE=+2>taille</FONT>,
<P> et m&ecirc;me la <FONT FACE="arial, chicago, jurassic"> fonte</FONT>.
<HR>
</BODY>
</HTML>
```

Lorsqu'on charge cette page, elle se présente comme le montre la figure 2.2, qui est ce que nous désirions.

FIG. 2.2 – Présentation d'un texte en couleur

Couleur de fond.- On peut changer la couleur de fond (*background* en anglais) grâce à l'attribut `BGCOLOR` (évidement pour *BackGround COLOR*) de la balise `BODY`.

Désignation des couleurs.- Les couleurs sont désignées dans le format dit **RGB** (pour l'anglais *Red, Green, and Blue*, soit rouge, vert et bleu). Vous avez certainement appris à l'école primaire comment on peut obtenir des couleurs sur une feuille de papier à partir des trois couleurs « primaires » jaune, bleu et rouge : jaune et bleu permet d'obtenir du vert, par exemple. Pour un écran les couleurs primaires ne sont pas les mêmes : on retrouve bien le rouge et le bleu mais la troisième couleur est le vert et non le jaune.

HTML permet un pourcentage (en fait une intensité) de chacune de ces couleurs primaires allant de 0 à 255. Une couleur est donc décrite par trois nombres hexadécimaux de deux chiffres

(qu'on fait porécédé de '#'pour indiquer qu'il ne s'agit pas de chiffres décimaux), par exemple `#FF0000` désigne le rouge et `#000000` le noir.

On peut donc décrire ainsi $256^3 = 16\,777\,216$ couleurs. Quelques-unes de ces couleurs peuvent être désignées par un nom, par exemple :

Red	FF0000
Green	00FF00
Blue	0000FF
Yellow	FFFF00
Orange	FFA500
Gold	FFD700
Beige	F5F5DC
Gray	808080
Navy	000080
Black	000000
White	FFFFFF

Couleur du texte.- On peut changer la couleur de tout le texte grâce à l'attribut `TEXT` de la balise `BODY`.

Contrôle d'une partie du texte.- Depuis *Netscape Navigator version 2*, on peut changer l'aspect d'une partie du texte grâce au conteneur `FONT` :

- L'attribut `SIZE`, dont la valeur est un entier relatif, permet d'accroître ou de décroître la taille de la fonte.
- L'attribut `COLOR` permet de changer la couleur de la fonte.
- L'attribut `FACE` permet de changer la fonte elle-même, se subsituant à la fonte utilisée par défaut. Par exemple :

  ```
  <FONT FACE = "arial, chicago, jurassic">
  ```

  forcera la navigateur à vérifier la présence d'une fonte appelée *Arial* et de l'utiliser si elle existe. Si elle n'est pas présente, on cherche *Chicago*, puis éventuellement *Jurassic*. Si aucune de ces trois fontes n'est présente, on revient à la fonte par défaut.
  Cependant il n'est pas conseillé en général d'essayer de changer de fonte, car il y a peu de chance qu'elle soit disponible sur le client.

2.2 Les liens

2.2.1 Les liens hypertexte

Maintenant que nous savons comment afficher une page (de texte, bien présenté), voyons comment y placer des hyperliens pour naviguer d'une page à une autre, ce qui est le but du WWW puisqu'il s'agit d'un système hypertexte.

FIG. 2.3 – Présentation d'un texte avec référence

Exemple.- Reprenons notre exemple de texte structuré affichant le début du premier chapitre. Au début second paragraphe nous donnons l'adresse Web du serveur Apache. L'utilisateur doit recopier celle-ci pour accéder à cette page. Changeons la ligne :

```
<P ALIGN=CENTER>
<TT>http://httpd.apache.org</TT>
```

```
</P>
```

par la ligne :

```
<P ALIGN=CENTER>
<A href="http://httpd.apache.org"> <TT>http://httpd.apache.org</TT></A>
</P>
```

Lorsqu'on charge cette page modifiée, elle se présente comme le montre la figure 2.3.

Remarquons que l'adresse Web est maintenant en couleur. Si nous cliquons dessus (avec la souris), nous accédons automatiquement à la page Web désirée.

Remarquons également que si nous revenons à cette page, la couleur du **lien** (cette adresse Web) a changée.

Création d'un lien.- Pour créer un lien sur une page Web, on utilise le conteneur **ancre** (*anchor* en anglais) `A` avec l'attribut `HREF` dont la valeur est une adresse Web, plus généralement une URL. Un élément (du texte en général) doit être insérer dans le conteneur pour qu'on puisse voir quelque chose à l'écran.

Couleur des liens.- Comme nous l'avons vu, les liens ont une certaine couleur et prennent une autre couleur lorsqu'on a cliqué dessus, pour se rappeler sur quels liens on a déjà cliqués.

Les couleurs par défaut peuvent être changées par des attributs du conteneur `BODY` :

- `LINK` permet de définir la couleur d'un hyperlien qui n'a pas encore été sélectionné.
- `ALINK` permet de définir la couleur d'un hyperlien sur lequel on a déjà cliqué.
- `VLINK` permet de définir la couleur d'un hyperlien dont on a déjà visité avec succès la page correspondante.

On pourrait ainsi changer la balise `<BODY>` en, par exemple :

```
<BODY BGCOLOR=BLUE TEXT=WHITE LINK=GREEN ALINK=LIGHTGREEN VLINK=DARKGREEN>
```

Signets.- Cliquer sur un lien nous amène sur une page Web, au début de celle-ci. Ceci n'est pas toujours le plus apprprié lorsque cette page est longue.

On peut parsemer une (longue) page de **signets** (*bookmark* en anglais) grâce à des conteneurs `A` dont l'attribut n'est pas un `HREF` mais un `NAME`.

Si le signet est `monsignet`, on y accède par un conteneur `A` dont la valeur de l'attribut `HREF` est `HREF="#monsignet"` si on se trouve dans la page et par l'adresse suivie de `"#monsignet"` à partir d'une autre page.

Courriel.- On peut créer un lien, non seulement pour accéder à une autre page, mais également pour envoyer un courriel. Lorsqu'on clique sur le lien :

```
<A HREF="mailto:patcep@free.fr">Envoyer vos commentaires</A>
```

on accède au mailer par défaut du poste client avec l'adresse de couriel déjà renseignée.

2.2.2 Les liens hypermédia

Avec les hyperliens, nous savons maintenant créer des hypertextes. Pour passer aux hypermédias, il faut plus que du texte, et des images pour commencer.

2.2.2.1 Les images

Un exemple.- Je prend l'exemple de la page d'accueil de mon site Web :

```
<html>
<head>
<title>Patrick Cegielski's Home Page</title>
</head>
<body BGCOLOR="#FFFFFF" LINK=blue ALINK=red VLINK=green>
<img align=right src="PCegielski.jpg" WIDTH=300 HEIGHT=466>
<H1 ALIGN=Center>
Patrick C&eacute;gielski
</H1>
Professeur de classe exceptionnelle &agrave / Full Professor at
Universit&eacute Paris Est Cr&eacute;teil
<P>
Member of LACL (Laboratoire d'Algorithmique, Complexit&eacute et
Logique, EA 4219)
<P>
<H3> Research Interests </H3>
     <UL>
     <LI> <a href="../jaf"> Weak Arithmetics/Arithm&eacute;tiques faibles </a>
     <LI> Pattern Matching/Recherche de motifs
     <LI> Computability/Calculabilit&eacute (Yuri Gurevich's ASM)
     </UL>
<H3>Publications</H3>
     <UL>
     <LI><a href="books-Pat.html"> Books </a>
     <LI><a href="articles-Pat.html"> Articles </a>
     <LI><a href="preprints-Pat.html"> Preprints </a>
     </UL>
<H3>Congress organization</H3>
     <UL>
     <LI><a href="../jaf"> JAF </a> :
     Journ&eacute;es sur les Arithm&eacute;tiques Faibles <br>
     Weak Arithmetics Days <br><br>
     <LI>CNRIUT
     </UL>
<H3><a href="teach-Pat.html">Enseignement/Teaching</a></H3>
<P>
<address>
Patrick Cegielski<br>
Universit&eacute Paris Est Cr&eacute;teil-IUT<br>
D&eacute;partement Informatique<br>
Route foresti&egrave;re Hurtault<br>
F-77300 Fontainebleau<br>
```

```
France<br></address>
<P>
Telephone:  +33.(0)1.60.74.68.16 (office)<br>
            <br>
Fax: +33.(0)1.60.74.68.28<br><br>
E-mail:
<KBD>
<A HREF="mailto:cegielski@u-pec.fr">cegielski at u-pec.fr</A>
</KBD>
(also <KBD>
 <A HREF="mailto:cep6@orange.fr">patcep at free.fr</A>
</KBD>)
<br>
<br>
<H5> Derni&egrave;re mise &agrave jour : 5 octobre 2011 </H5>
</body>
</html>
```

dont le chargement montre ce qui se voit sur la figure 2.4, avec une image en haut à gauche.

Incrustation d'une image.- Pour placer une image sur une page Web, on utilise la balise `IMG` (évidemment pour *IMaGe*) avec l'attribut `SRC` (pour *SouRCe*), spécifiant le nom du fichier image, sans chemin si celui-ci se trouve dans le même répertoire que celui du fichier HTML qui y fait appel.

Il existe d'autre attributs, non obligatoires :

- Remarquons que, comme dans une revue, l'image n'apparaît pas comme un paragraphe à part entière mais que du texte l'entoure. Par défaut, l'image est placée à gauche mais on peut changer ceci en utilisant l'attribut `ALIGN`, avec l'une des valeurs `LEFT` (valeur par défaut), `CENTER` ou `RIGHT`.
- On peut décider de la hauteur et de la largeur de l'image grâce aux attributs `HEIGHT` et `WIDTH`, dont les valeurs sont des entiers naturels spécifiant le nombre de pixels.
- Un espacement par défaut se trouve entre l'image et le texte qui l'entoure. On peut changer celui-ci en utilisant les attributs `HSPACE` et `VSPACE`, dont les valeurs sont des entiers naturels spécifiant le nombre de pixels entre l'image et le texte, horizontalement et verticalement.
- Certains navigateurs, comme *Lynx*, n'affichent pas les images. Certains utilisateurs refusent de recevoir des images, soit pour des raisons de sécurité, soit parce que la liaison est trop lente. L'attribut `ALT` (pour *ALTernative*), dont la valeur est une chaîne de caractères, permet de donner une description textuelle de l'image. C'est elle qui est affichée, et non l'affreux carré avec une croix rouge, si on se trouve dans une des ces situations.

Format des images incrustées.- Il existe des centaines de formats d'images mais HTML n'en accepte que deux, ce qui exige de convertir l'image si elle n'est pas un de ces formats-là au départ. Les deux formats acceptés au départ sont GIF (*Graphic Interchange Format*) et JPG (*Joint Picture Experts Group*).

Le format GIF fut créé en 1987 par *CompuServe*, le service en ligne dominant aux États-Unis dans les années 1980, comme moyen de transmettre du graphisme *via* un modem. Les images sont restreintes en terme de nombre de couleurs, au maximum 256, mais choisies dans une palette, et de plus bien compressées, ce qui permet une bonne qualité pour un fichier de petite taille.

FIG. 2.4 – Présentation d'un texte avec image

Le format JPG est, quant à lui, bien adapté pour les photographies de grande qualité mais mal adapté là où GIF se distingue.

Au vu du grand nombre d'images GIF diffusées sur le Web, associé au recul des services en ligne devant la concurrence du Web, *CompuServe*, qui possède un brevet sur une partie du format GIF, a menacé de demander une contribution financière sur chaque image GIF affichée sur un site Web. W3C a alors développé son propre format d'image, PNG (*Portable Network Graphic*), qui combine les aspects positifs de GIF et de JPG en un format compact. Ce troisième format est donc accepté par HTML. Par ailleurs les droits de *CompuServe* sur GIF ont expirés en 2004.

2.2.2.2 Les sons et les vidéos

Liens hypermédias.- Il est facile de placer un lien hypermédia, tel qu'un son ou une vidéo, grâce au conteneur `A` :

```
<A HREF="monson.wav">Cliquer ici pour &eacute;couter</A>
```

ou :

```
<A HREF="mavideo.avi">Cliquer ici pour visionner</A>
```

Le navigateur se contentera en général à faire appel au programme associé par défaut au type de fichier chargé, dans une (autre) page Web ou dans une instance du programme. À défaut, il sera proposé de sauvegarder le fichier.

2.3 Formulaires

Nous avons quelquefois besoin de recueillir des informations de la part des visiteurs d'une page Web. C'est en particulier le cas pour le commerce électronique. Mosaic 2.0pre5 a donc introduit la notion de **formulaire** interactif (*form* en anglais).

Les renseignements des formulaires sont des données textuelles. Il n'ya pas de types « int », « date », « float » ou « url ».

2.3.1 Premier exemple

FIG. 2.5 – Présentation d'un formulaire simple

Écrivons une page Web faisant apparaître un formulaire simple demandant notre nom (pour établir, par exemple, la liste des personnes désirant participer à une manifestation) :

```
<HTML>
<HEAD>
<TITLE>Formulaire simple</TITLE>
</HEAD>
<BODY>
<H1>Formulaire simple</H1>
<HR>
<FORM METHOD=POST ACTION="mailto:patcep@free.fr">
Nom : <INPUT TYPE=TEXT NAME="nom" SIZE=30>
<P>
<INPUT TYPE=SUBMIT VALUE="envoi">
<INPUT TYPE=RESET VALUE="annulation">
```

```
</P>
</FORM>
<HR>
</BODY>
</HTML>
```

Lorsqu'on charge cette page, elle se présente comme le montre la figure 2.5.

Le client tape son nom dans la boîte qui apparaît après l'étiquette 'Nom' et appuie sur la touche retour.

S'il a renseigné 'Patrick' alors 'Nom=Patrick' est envoyé par courrier électronique à l'adresse de courriel indiquée (vérifier-le en changeant cette adresse par la vôtre !).

Remarques.- 1^o) Cette page Web peut ne pas fonctionner avec certains navigateurs (par exemple *Internet Explorer*) mais fonctionner avec d'autres (tels que *Google Chrome* ou *Firefox*). Tester-là sur plusieurs navigateurs !

- 2^o) Normalement, le courriel devrait être envoyé en tâche de fond sans en avertir l'utilisateur. Cependant certains mailer demanderont une confirmation avant d'envoyer le courriel.

2.3.2 Syntaxe

Comme le montre notre premier exemple, un formulaire est un conteneur FORM possédant des attributs obligatoires contenant des balises INPUT, possédant également des attributs obligatoires, dont deux au moins sont nécessaires : une pour indiquer un renseignement (sinon à quoi servirait le formulaire ?) et une pour indiquer qu'on a terminé de remplir le formulaire et qu'on peut envoyer les renseignements.

Si, bien entendu, un document peut comporter plusieurs formulaires, ceux-ci ne peuvent en aucun cas être empilés (en d'autres termes, un formulaire ne peut pas en contenir un autre).

Premiers attributs de FORM.- L'attribut `ACTION` spécifie la façon dont les renseignements seront transmis, ici par la méthode `mailto`, c'est-à-dire *via* le mailer par défaut de l'ordinateur client. L'adresse de courriel à laquelle envoyer les renseignements doit alors être indiquée.

L'attribut METHOD spécifie la façon dont les données sont transmises. Il a deux valeurs, POST et GET. Il vaut mieux utiliser la première pour l'instant.

La balise INPUT.- La balise `INPUT`, une des rares balises à ne pas aller en couple pour former un conteneur, est utilisée pour recueillir du texte à partir du clavier de l'ordinateur client. Son attribut principal est `TYPE` prenant un grand nombre de valeurs dont nous allons en étudier trois pour commencer :

- Le type `TEXT` permet d'afficher une boîte d'entrée d'une ligne. L'attribut `NAME`, dont la valeur est un identificateur, est alors obligatoire : nous avons vu que les renseignements seront transmis sous la forme de la valeur choisie pour `NAME`, suivie du signe '=', suivi de la valeur entrée au clavier. Il est évidemment important de choisir une valeur de `NAME` différente pour chaque instance de balise `INPUT`. On peut utiliser d'autres attributs tels que `SIZE` spécifiant la longueur, en caractères, de la boîte à l'écran (il y aura défilement si on a besoin de plus de caractères), `MAXLENGTH` spécifiant la taille maximale de la donnée (20 par défaut) ou `VALUE` qui permet d'afficher un texte dans la boîte.
- Le type `SUBMIT` place un bouton sur le formulaire de nom `Submit`, que l'on peut changer grâce à l'attribut `VALUE`. Le client enverra le formulaire en cliquant sur ce bouton grâce à la souris.

– Le type `RESET` place un bouton sur le formulaire de nom `Reset`, que l'on peut changer grâce à l'attribut `VALUE`. Le client réinitialisera le formulaire, c'est-à-dire qu'il fera disparaître les renseignements déjà remplis, en cliquant sur ce bouton grâce à la souris.

2.3.3 Deuxième exemple

FIG. 2.6 – Présentation d'un formulaire plus élaboré

Écrivons une seconde page Web faisant apparaître un formulaire un peu plus élaboré contenant deux champs de texte pour le nom et le prénom, des **boutons radio** (c'est-à-dire que l'on ne peut en cocher qu'un dans la liste, comme pour le choix de la gamme de fréquence sur les anciens transistors) pour le sexe, des **cases à cocher** (c'est-à-dire que, cette fois-ci, on peut en cocher plusieurs dans la liste ; *checkbox* en anglais) pour déterminer la façon dont on veut obtenir l'information, une **aire de texte** (*textarea* en anglais) permettant d'envoyer un message sur plusieurs lignes, et pour terminer les deux boutons d'envoi et de réinitialisation :

```
<HTML>
<HEAD>
<TITLE>Enregistrement</TITLE>
</HEAD>
<BODY>
<H1>Enregistrement</H1>

<FORM METHOD=POST ENCTYPE="text/plain" ACTION="mailto:patcep@free.fr">
<PRE>
```

```
          Nom: <INPUT TYPE=TEXT NAME=Nom size=30>
       Pr&eacute;nom: <INPUT TYPE=TEXT NAME=Prenom size=30>
         Sexe: <INPUT TYPE=RADIO NAME=Sexe VALUE=Masculin> Masculin
               <INPUT TYPE=RADIO NAME=Sexe VALUE=F&eacute;minin> F&eacute;minin
Recevoir par : <INPUT TYPE=CHECKBOX NAME=send VALUE=Courriel> Courriel
               <INPUT TYPE=CHECKBOX NAME=send VALUE=SMS> SMS
      Message: <TEXTAREA NAME=Message ROWS=5 COLS=30></TEXTAREA>

               <INPUT TYPE=SUBMIT VALUE=Envoyer>
               <INPUT TYPE=RESET VALUE=Effacer
</PRE>
</FORM>
<HR>
</BODY>
</HTML>
```

Lorsqu'on charge cette page, elle se présente comme le montre la figure 2.6.
Commentons la syntaxe des nouveaux éléments :

- L'attribut `ENCTYPE` (pour l'anglais *ENCoding TYPE*, type de codage) du conteneur `FORM` permet de spécifier la façon dont les données seront transmises. La valeur `text/plain` nous permet d'éviter quelques problèmes : sans cet attribut, les renseignements sont transmis les uns à la suite des autres séparés par le caractère '&' ; les lettres accentuées (telle que le 'é' de 'féminin') apparaissent bizarrement dans le message envoyé.
- Le type `RADIO` de `INPUT` permet de faire apparaître une liste (il suffit que plusieurs instances de `INPUT` aient la même valeur pour l'attribut `NAME`) de cases dont on ne peut sélectionner qu'une seule. L'attribut `VALUE` permet de spécifier la valeur renvoyée du bouton sélectionné. On peut aussi positionner l'attribut `CHECKED=TRUE` à l'un (et un seul) des éléments de la liste pour spécifier une valeur par défaut.
- Le type `CHECKBOX` de `INPUT` permet d'afficher des cases à cocher (on remarquera les aspects différents des cases à sélection unique et des cases à sélections multiples sur le navigateur) dont les attributs sont les mêmes que dans le cas du type `RADIO`.
- Le type `TEXTAREA` de `INPUT` permet d'afficher une aire de texte, c'est-à-dire un emplacement que l'on peut renseigner avec du texte sur plusieurs lignes (alors que ceci ne pouvait se faire que sur une ligne avec le type `TEXT`). On a dans ce cas un conteneur et non une simple balise. Les attributs sont `ROWS` et `COLS` spécifiant le nombre de lignes apparaissant sur la boîte et le nombre de caractères par lignes. L'utilisateur peut renseigner la boîte par le nombre de caractères qu'il veut : il y a aura défilement des lignes lorsqu'on atteint la fin d'une ligne et lorsqu'on arrive à la dernière ligne. Des **barres de défilement** (horizontale et verticale ; *scrollbar* en anglais)) apparaîtront alors.

2.3.4 Autres éléments de syntaxe

La balise `INPUT` a d'autres types :

- Le type `PASSWORD` est anlogue au type `TEXT` sauf que le texte renseigné n'apparaît pas à l'écran : chaque caractère est remplacé par '*'. On voit son intérêt et l'origine de son nom pour renseigner un mot de passe.

2.4 Les tableaux

On peut avoir besoin de présenter des données sous forme de tableaux.

FIG. 2.7 – Un tableau

Exemple.- Si on veut une page Web donnant les notes du dernier partiel, le mieux est d'utiliser un tableau pour obtenir la présentation de la figure 2.7. Pour cela on se sert du fichier HTML suivant :

```
<HTML>
<HEAD>
<TITLE>Utilisation d'un tableau</TITLE>
</HEAD>
<BODY>
<H1>Utilisation d'un tableau</H1>
<HR>
<P>
<table Border>
   <caption>Notes d'Algorithmique-Programmation</caption>
   <tr>
       <th>Nom</th>
       <th>Groupe</th>
       <th>Note</th>
   </tr>
   <tr>
       <td>Arthur</td>
       <td>1</td>
```

```
        <td>13</td>
    </tr>
    <tr>
        <td>Michelle</td>
        <td>2</td>
        <td>14,5</td>
    </tr>
</table>
</P>
<HR>
</BODY>
</HTML>
```

Le conteneur tableau.- Un tableau est un élément de conteneur `TABLE`. On peut utiliser les attributs suivants :

- `BORDER` qui permet de dessiner un cadre autour du tableau et de chacune des cellules. On peut spécifier son épaisseur en lui donnant comme valeur un entier naturel, qui indique le nombre de pixels : 0 correspond à pas de bordure (ce qui revient à ne pas utiliser cet attribut), 1 est la valeur par défaut.
- `WIDTH` spécifie la largeur du tableau, en absolu (un entier indique la largeur en pixels) ou en relatif (un pourcentage).
- `HEIGHT` spécifie la hauteur du tableau, en absolu (un entier indique la hauteur en pixels) ou en relatif (un pourcentage).

Un tableau est une suite de lignes (*row* en anglais), composées de **cellules** (*cell* en anglais).

Nom d'un tableau.- On peut donner un nom au tableau en utilisant le conteneur `CAPTION`.

Ligne d'un tableau.- Une ligne d'un tableau est un élément de conteneur `TR` (pour *Table Row*). On peut utiliser les attributs suivants :

- `ALIGN`, pour spécifier la position horizontale du contenu des cellules, avec les trois valeurs `LEFT` (la valeur par défaut), `CENTER` et `RIGHT`.
- `VALIGN`, pour spécifier la position verticale du contenu des cellules, avec les trois valeurs `TOP`, `MIDDLE` (la valeur par défaut) et `BOTTOM`.
- `BGCOLOR` pour spécifier la couleur de fond de la ligne.

Cellule d'un tableau.- Une cellule d'un tableau est un élément de contenur `TD` (pour *Table Data*) ou `TH` (pour *Table Header*). Le contenu de `TH` est un texte (en gras et centré) permettant de donner un titre à la colonne, c'est-à-dire la signification des données de celle-ci. On peut utiliser les attributs suivants :

- Par souci de clarté, on peur vouloir espacer les cellules grâce à l'attribut `CELLSPACING` dont la valeur, un entier, définit l'espacement en pixels entre chaque cellules.
- On peut également spécifier des marges à l'intérieur de la cellule afin d'espacer les données du bord de la cellule grâce à l'attribut `CELLPADDING` dont la valeur, un entier, donne le nombre de pixels.
- `ALIGN`, pour spécifier la position horizontale du contenu de la cellule, avec les trois valeurs `LEFT` (la valeur par défaut), `CENTER` et `RIGHT`.
- `VALIGN`, pour spécifier la position verticale du contenu de la cellule, avec les trois valeurs `TOP`, `MIDDLE` (la valeur par défaut) et `BOTTOM`.

- `BGCOLOR` pour spécifier la couleur de fond de la cellule.
- `WIDTH` spécifie la largeur de la cellule, en absolu (un entier indique la largeur en pixels) ou en relatif (un pourcentage). Mais, bien entendu, le navigateur ne permet qu'une seule largeur par colonne.
- `HEIGHT` spécifie la hauteur d'une cellule, en absolu (un entier indique la hauteur en pixels) ou en relatif (un pourcentage). Mais, bien entendu, le navigateur ne permet qu'une seule hauteur par ligne.
- `NOWRAP` spécifie que le texte de la cellule doit apparaître sur une seule ligne.

2.5 Les cadres

Dans certains cas, on aimerait bien découper la fenêtre du navigateur en plusieurs parties : par exemple, dans le cas d'affichage d'un livre (ou de son analogue), la partie droite contiendrait la table des matières et, en cliquant sur le chapitre, ou la partie voulue, celui-ci s'afficherait dans la partie gauche ; de façon générale, on pourrait avoir une idée des ressources du site en permanence et l'affichage de la ressource qui nous intéresse à un moment donné.

On peut évidemment faire cela en recopiant ce que l'on veut sur chacune des pages, mais la maintenance d'un tel site devient vite compliquée et pénible. *Netscape 2* a introduit pour cela la notion de **cadre** (*frame* en anglais).

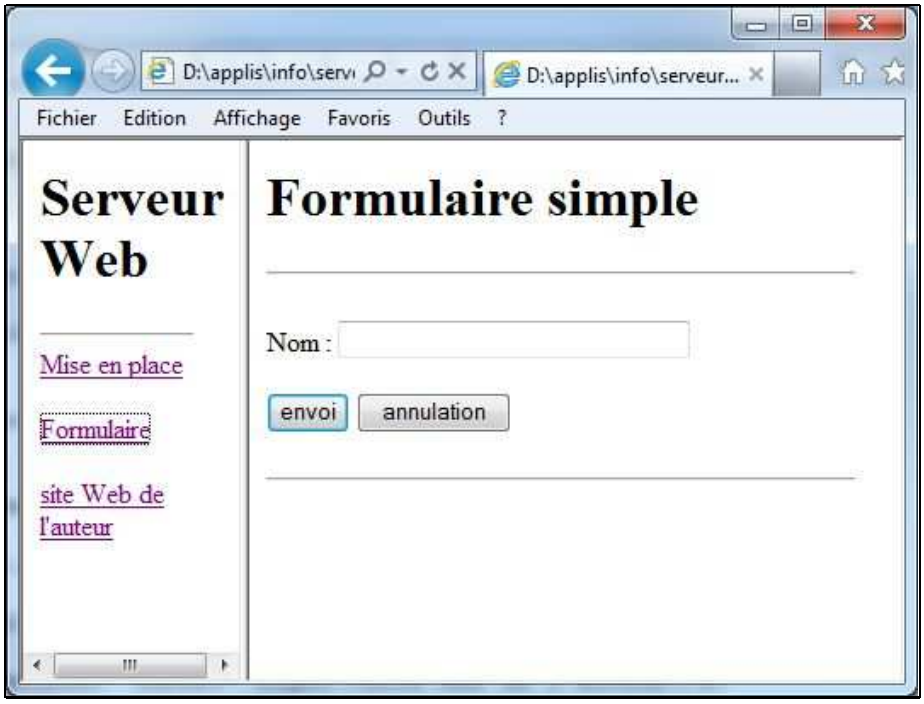

FIG. 2.8 – Une page avec cadres

Exemple.- Essayons d'obtenir ce que l'on voit à la figure 2.8, c'est-à-dire une fenêtre divisée en deux parties : à gauche, le titre et une table des matières abrégée, à droite ce que l'on voit lorsque l'on clique à gauche sur l'item médian.

Pour cela, écrivons un premier fichier, de nom disons « frame.html », contenant le découpage en cadres et les valeurs par défaut dans les deux cadres choisis :

```
<FRAMESET COLS="25%,75%">
<FRAME SRC="toc.html" NAME="left">
<FRAME SRC="text.html" NAME="right">
</FRAMESET>
```

La page de droite n'est rien d'autre que notre premier exemple de formulaire. La page de gauche se trouve dans le fichier « toc.html » :

```
<HTML>
<HEAD>
<TITLE>Serveur Web</TITLE>
</HEAD>
<BODY>
<H1>Serveur Web</H1>
<HR>
<A HREF="text.html" TARGET="right">Mise en place</A>
<P>
<A HREF="form1.html" TARGET="right">Formulaire</A>
<P>
<A HREF="index.html" TARGET="right">site Web de l'auteur</A>
</BODY>
</HTML>
```

Le conteneur principal.- Comme on le voit sur l'exemple, la page de cadres n'est pas un élément `HTML` mais un élément `FRAMESET`, comprenant l'un des attributs suivants :

- `COLS` spécifiant que la page est découpée en colonnes, dont la valeur est une suite de pourcentages séparés par des virgules, des entiers naturels (représentant une valeur absolue en pixels) ou '*' signifiant le reste de l'espace disponible.
- `ROWS` spécifiant que la page est découpée en lignes, la syntaxe de la valeur étant la même que pour les colonnes.

Plusieurs éléments `FRAMESET` peuvent être imbriqués les uns dans les autres pour obtenir un découpage plus complexe, mélangeant découpage en lignes et en colonnes.

Le conteneur cadre.- Comme on le voit sur l'exemple, le contenu d'un conteneur `FRAMESET` est un certain nombre d'éléments `FRAME`, le nombre de virgules plus un dans la valeur de `FRAMESET`, avec les deux attributs suivants :

- `SRC` (pour *SouRCe*) spécifie la page HTML à charger au moment du chargement de la page de cadres.
- `NAME` spécifie un nom pour ce cadre, ce qui sera utile si on doit y faire référence.

D'autres attributs permettent de raffiner son comportement :

- `MARGINWIDTH` spécifie la taille des marges gauche et droite, insérant une espace entre le bord du cadre et le texte ou l'image.
- `MARGINGHEIGHT` spécifie de même la taille des marges haute est basse.
- `SCROLLING`, avec l'une des valeurs `YES`, `NO` et `AUTO` (valeur par défaut), permet de faire apparaître une barre de défilement, telle que celle que l'on voit sur la page de gauche de notre exemple.

– `NORESIZE` interdit à l'utilisateur de redimensionner le cadre manuellement (grâce à la souris).

Le conteneur alternatif.- Pour les navigateurs qui n'acceptent pas accepter les cadres, on peut placer une page HTML qui remplacera la page avec cadres dans l'élément `NOFRAMES`.

L'attribut de renvoi.- Par défaut, lorsqu'on clique sur un lien dans un cadre, la nouvelle page Web se placera dans ce meê cadre. Dans le cas d'une table des matières qui doit rester fixe, la page appelée devant prendre place dans un autre cadre, on utilise l'attribut `TARGET` dans la balise `A`, sa valeur étant le nom d'un cadre.

2.6 Historique

2.6.1 Les origines 1989-1992

HTML est une des trois inventions à la base du World Wide Web, avec HTTP et les adresses web. HTML a été inventé pour pouvoir écrire des documents hypertextuels liant les différentes ressources d'Internet avec des hyperliens. Aujourd'hui, ces documents sont appelés **page web**. En août 1991, lorsque Tim Berners-Lee annonce publiquement le web sur Usenet, il ne cite que le langage SGML, mais donne l'URL d'un document de suffixe .html. Dans son livre [BL-00], Tim Berners-Lee justifie la décision de baser HTML sur SGML comme étant aussi « diplomatique » que technique : techniquement, il trouvait SGML trop complexe, mais il voulait attirer la communauté hypertexte qui considérait que SGML était le langage le plus prometteur pour standardiser le format des documents hypertexte. En outre, SGML était déjà utilisé par son employeur, le CERN.

Les premiers éléments du langage HTML comprennent le titre du document, les hyperliens, la structuration du texte en titres, sous-titres, listes ou texte brut, et un mécanisme rudimentaire de recherche par index. La description de HTML est alors assez informelle et principalement définie par le support des divers navigateurs web contemporains.

Dan Connolly a aidé à faire de HTML une véritable application de SGML [GC-00].

2.6.2 L'apport de Mosaic (1993)

L'état de HTML correspond alors à ce que l'on pourrait rétrospectivement appeler HTML 1.0. Il n'existe cependant aucune spécification portant ce nom, notamment parce que le langage était alors en pleine évolution. Un effort de normalisation était cependant en cours [HTML 1.0]. À partir de fin 1993, le terme HTML+ est utilisé pour désigner la version future de HTML ([HTML-intro], section 2.2.1). Malgré l'effort de normalisation ainsi initié, et jusqu'à la fin des années 1990, HTML est principalement défini par les implémentations des navigateurs.

Avec le navigateur NCSA Mosaic, HTML connaît deux inventions majeures. D'abord l'invention de l'élément IMG permet d'intégrer des images (dans un premier temps, uniquement aux formats GIF et XBM) aux pages web (Mosaic 0.10). Ensuite l'invention des formulaires (Mosaic 2.0pre5) rend le web interactif en permettant aux visiteurs de saisir des données dans les pages et de les envoyer au serveur web. Cette invention permet notamment de passer des commandes, donc d'utiliser le web pour faire du commerce électronique.

2.6.3 L'apport de Netscape Navigator (1994)

Avec l'apparition de Netscape Navigator 0.9 le 13 octobre 1994, le support de nombreux éléments de présentation est ajouté : attributs de texte, clignotement, centrage, etc. Le développement de HTML prend alors deux voies divergentes.

D'une part, les développeurs de navigateurs s'attachent à maximiser l'impact visuel des pages web en réponse aux demandes des utilisateurs. Marc Andreessen, créateur de Netscape Navigator, déclare le 16 juin 1993 sur la liste de discussion www-talk : « *Je pense que s'occuper du SGML en général est une complète perte de temps, et que nous en serions aujourd'hui beaucoup plus loin si nous n'étions pas encombrés avec cet héritage SGML que nous continuons à porter. 99,99 % des gens avec qui je parle veulent mettre en ligne des documents riches, veulent contrôler leur apparence, et se contre-fichent totalement du balisage sémantique ou des différences entre la structure et le rendu d'un document.* »

D'autre part, les concepteurs du web proposent d'étendre les capacités de description sémantique (logos, notes de bas de page, etc.) et les domaines d'applications (formules mathématiques, tables) de HTML. En ceci, ils suivent les principes de SGML consistant à laisser la présentation à un langage de style. En l'occurrence, les feuilles de style en cascade (CSS) sont prévues pour HTML. Seul le support des tables est rapidement intégré aux navigateurs, notamment parce qu'il permet une très nette amélioration de la présentation. Outre la multiplication des éléments de présentation, les logiciels d'alors produisant et consommant du HTML conçoivent souvent les documents comme une suite de commandes de formatage plutôt que comme un marquage représentant la structure en arbre.

2.6.4 HTML 2.0 (1995-1996)

En mars 1995, le *World Wide Web Consortium* (W3C) nouvellement fondé propose le résultat de ses recherches sur HTML+ : le brouillon HTML 3.0. Il comprend notamment le support des tables, des figures et des expressions mathématiques. Ce brouillon expire le 28 septembre 1995 sans donner de suites directes. Fin 1995, la RFC 1866 décrivant HTML 2.0 est finalisée. Le principal éditeur en est Dan Connolly [RFC 1866]. Ce document décrit HTML tel qu'il existait avant juin 1994, donc sans les nombreuses additions de *Netscape Navigator*.

2.6.5 HTML 3.2 et 4.0 (1997)

Le 14 janvier 1997, le W3C publie la spécification HTML 3.2. Elle décrit la pratique courante observée début 1996 [HTM 3.2], donc avec une partie des additions de *Netscape Navigator* et d'*Internet Explorer*. Ses plus importantes nouveautés sont la standardisation des tables et de nombreux éléments de présentation.

HTML 3.2 précède de peu HTML 4.0 et contient des éléments en prévision du support des styles et des scripts. Le 18 décembre 1997, le W3C publie la spécification HTML 4.0 qui standardise de nombreuses extensions supportant les styles et les scripts, les cadres (*frame* en anglais) et les objets (inclusion généralisée de contenu). HTML 4.0 apporte également différentes améliorations pour l'accessibilité des contenus dont principalement la possibilité d'une séparation plus explicite entre structure et présentation du document, ou le support d'informations supplémentaires sur certains contenus complexes tels que les formulaires, les tableaux ou les sigles.