Sommaire

INTRODUCTION

L'Internet est devenu un média à par entière, il est donc nécessaire d'y avoir sa place et ainsi de pouvoir y véhiculer ses idées.
Répondant à ce nouvel adage, le directeur de l'entreprise Inforadour – Adix (spécialisée dans la création de progiciels de comptabilité M9) m'a proposé de réaliser le site Web de sa société.
L'idée m'a évidemment intéressé puisque elle semblait être une suite logique à une formation à l'Internet que j'avais réalisé quelques mois auparavant.
Armé de quelques connaissances sur la manière de créer un site Web et accompagné de quelques livres de formation, je me suis donc plongé pendant trois semaines dans ce travail.
Voici donc résumé en six parties la démarche qui a été la mienne lors de la conception de ce site, d'une première étape imaginaire à sa publication sur le Web.

I LE SUJET

a) Ce qui compte, c'est le contenu

Ça a l'air idiot, mais le plus important doit rester le contenu, le message que l'on veut faire passer ou l'information qu'on veut délivrer.

Le but principal de site d'Inforadour – Adix était de faire connaître sa société, de permettre à ses clients de se tenir informer de la sortie de nouveaux produits et enfin de permettre aux futurs clients de s'informer sur la société et de demander de la documentation à partir du site Web.

b) Une présentation harmonieuse

Les formes, couleurs, et polices de caractères doivent avoir une certaine unité, pour créer une identité propre au site, dont le visiteur se souviendra. Sinon, on risque le fourre-tout, anonyme, lourd voire indigeste. L'idéal est de se cantonner à 3 ou 4 couleurs, et 2 ou 3 polices de caractères.
Par exemple, un fond d'écran bleu marine, un texte noir donnera une page illisible et par voie de conséquence le visiteur s'en va !
De plus, il n'est pas habile de détourner l'attention du lecteur avec des animations ou des couleurs criardes alors que le site est fait pour vendre les produits de la société.

c) Attention aux cadres (frames)

Les frames permettent de créer plusieurs zones indépendantes dans une même page, et de pouvoir les mettre à jour indépendamment les unes des autres. Les frames sont très sensibles aux différences de résolution à l'affichage : on risque donc d'avoir des barres de défilement un peu partout sur la page en 640x480, alors que c'est impeccable en 800x600. Il faut donc savoir à qui sera destiné en priorité le site, quels types d'ordinateurs seront utiliser ; mais surtout il faut faire des tests pour savoir comment le site sera visualisé avec certains écrans (14', 15', …).

d) Des pages légères, donc rapides

Il est nécessaire de se méfier en incluant des applets, images, animations et fonds d'écran sur une page Web. Quand on crée une page en local, le chargement est rapide. Quand on consulte cette page sur le Net à une heure d'affluence, c'est autre chose ! Dommage de faire une superbe page...qui met trois minutes à se charger : le visiteur sera parti, ou exaspéré, avant la fin du chargement.

Une petite astuce est de compter le "poids" de chaque page, en incluant tous les fichiers (images, applets...); elle ne doit pas dépasser 50 Ko. Enfin, une fois que le site est terminer, on peut faire des tests de chargement (15 à vingt secondes maximum) à partir du réseau Intranet de l'entreprise par exemple.

Sur le site d'Inforadour – Adix, seules quelques pages clés de l'arborescence ont bénéficié d'une mise en page animée (incluant images , applets et animations) et ce pour optimiser la navigation.

e) Une arborescence claire et affichée

Il est important que le visiteur puisse trouver rapidement ce qu'il cherche, et revenir tout aussi rapidement au sommaire... Une arborescence fouillis ou incompréhensible, ou qui oblige à utiliser systématiquement les boutons "précédent" et "suivant" (et donc à afficher de nouveau toutes les pages) pour revenir au sommaire, peut décourager le visiteur de s'enfoncer dans la jungle d'un site mal construit.

Les premiers outils dont j'ai eu besoin sont donc... une feuille de papier et un stylo, afin de dessiner l'arborescence du site.
Après ces préliminaires indispensables, le directeur de la société et moi avons discuté des propositions que je lui avaient faites, et après quelques mises au point, je me suis lancé dans la conception des pages du site.

II HTML. LANGAGE DU WEB

a) Avant propos

Il n'est pas nécessaire de connaître le langage HTML sur le bout des doigts pour devenir un auteur de pages Web, mais la connaissance des bases de ce langage permet d'avoir une vision plus intéressante du travail que l'on effectue en tant que Webmaster.
Ce chapitre étudie donc le langage HTML (version 3.2, qui a été normalisé le 15 janvier 1997) et résume les notions importantes que j'ai relevés dans les manuels qui m'ont aidés à créer le site d'Inforadour – Adix.

b) Généralités

Le langage HTML est un langage à balises, avec lequel pour faire une action sur un groupe de mots, vous devez baliser ce groupe de mots. Par exemple, pour mettre en gras le mot David, et le rendre visible sous la forme **David**par les lecteurs de Web, il suffit d'écrire la chose suivante Gilles.
De façon générale les commandes sont de la forme :
<marqueur> texte **</marqueur>**
ou encore
<marqueur attribut=argument> texte **</marqueur>**
ou même
<marqueur>

Les noms de marqueurs sont identiques en majuscules ou minuscules : **** équivaut à <**B**>**.**
On peut attribuer plusieurs attributs à la même balise par la syntaxe suivante :
<marqueur attribut1=argument1 attribut2=argument2> texte **</marqueur>**
Les normes HTML demandent de respecter le codage dans les caractères ASCII 7 bits, c'est à dire sans caractères accentués.
Les logiciels navigateurs actuels sont permissifs et vous pouvez entrer des caractères accentués sous votre éditeur WYSIWYG, mais les caractères accentués sauteront vraisemblablement dans les moteurs de recherche américains, ce point est à méditer.
Pour coder un caractère accentué, on entre une combinaison précédée du caractère **&** et terminé par un ; (point virgule). Ainsi é se code é

Exemples : le caractère > est à remplacer par **>** le caractère < par **<** et le caractère & par **&** .

Les caractères espace consécutifs sont ramenés à un seul espace et pour forcer une suite de caractères espaces, il faut utiliser ** **
Un caractère ASCII peut s'écrire sous la forme &#numéro_du_caractère.
Avant d'entrer dans le vif du sujet il faut connaître les conventions suivantes :

- plusieurs caractères blancs sont remplacés par un seul caractère blanc;
- les retours-chariot que vous entrerez ne seront pas pris en compte, il faudra les expliciter par des balises <P> (changement de paragraphe) ou
 (changement de ligne)


Enfin tout le monde commence par regarder les pages HTML des autres avant d'écrire la sienne (c'est ce que j'ai fait !). Pour cela il vous suffit de cliquer sur le menu *View Source* de votre lecteur de Web pour voir les sources d'une page HTML et vous familiariser avec le langage.
A noter qu'une ligne peut être entrée en commentaires
<! commentaire >

c) Balises de mise en page

Ces balises permettent de faire une action ponctuelle là où elles apparaissent, sans effet sur la suite du document.
<HR> trace un trait de séparation dans le texte, comme ceci.

Cette balise peut être agrémentée des arguments suivants :

- **<HR SIZE=**nombre**>** donne une épaisseur au trait.
- **<HR WIDTH=**nombre|pourcentage**>** donne la longueur du trait
- **<HR ALIGN=left|right|center>** le trait peut être aligné à droite, à gauche ou centré
- **<HR NOSHADE>** le trait n'a pas d'ombre

<P> est une marque de fin de paragraphe.
<P ALIGN=CENTER|LEFT|RIGTH>
permet de rentrer le texte entre les marges gauche et droite ; dans ce cas, la balise **</P>** localisera la fin de la portée de la balise **<P>**
**
** est une marque de fin de ligne sans saut de paragraphe.
Elle connait les arguments suivants :
<BR CLEAR=all|left|right>
CLEAR quand il a la valeur *left* va faire le même saut de ligne et fait un déplacement vers le bas en respectant la marge gauche. **CLEAR** avec la valeur *right* fait la même chose avec la marge droite. Ceci permet aussi de mettre des images autour de texte.
En outre le couple de balises
<DIV ALIGN="left|center|right">...</DIV> permet de justifier un texte à droite, au centre, ou à gauche
De même, le couple de balises **<CENTER>...</CENTER>** permet de centrer le texte qu'il encadre.

d) Balises de documents

On s'attache ici aux couples de balises <xx> ... </xx > qui servent à délimiter une page HTML et à définir son entête et son corps.

- **<HTML> . . . </HTML>** est le couple de balises qui doit marquer le début et la fin du document, le document en HTML 3.2 doit commencer par l'instruction :
  <!**DOCTYPE HTML PUBLIC** "-//W3C//DTD HTML 3.2">
  <TITLE>Exemple</TITLE>
  <**HEAD> . . . </HEAD>** est le couple de balises qui marque l'entête du document
- **<BODY> .. </BODY>** est le couple de balises qui marque le corps du document

Exemple : votre première page HTML visible sera
<!**DOCTYPE HTML PUBLIC** "-//W3C//DTD HTML 3.2">
<HEAD>
<TITLE>Exemple</TITLE>
<**/HEAD>**
<**BODY>**

Ceci est ma première page HTML !
</BODY>
</HTML>

- <**! ....**> est une balise de commentaire qui permet d'inclure des informations non affichées dans vos pages.

e) Balises d'entêtes

Ces commandes se mettent entre le couple de balises <HEAD> et </HEAD>. Elles donnent des informations générales sur toute la page.

- **Exemple :** la page suivante

```
<HTML>
<HEAD>
<BASE HREF="fichier.htm">
</HEAD>
<BODY>
Voici une page d'interrogation très simple
</BODY>
</HTML>
```

- **<TITLE> . . . </TITLE>** donne le titre du document, ce titre apparaît dans le bandeau supérieur de la fenêtre de votre lecteur de Web. Cette balise doit être unique pour respecter la norme HTML 3.2 et elle est obligatoire.
- **<LINK HREF=**adresse **rel=**top|contents|index|glossary|copyright|next|previous|help|search **rev**=top|contents|index|glossary|copyright|next|previous|help|search **title**=valeur>
  établit un lien avec un autre document pour donner le nom du document maître, l'index. Cette balise n'est jamais utilisée mais elle serait très utile pour renseigner sur les enchainements des documents. Ainsi si les liens étaient correctement exploités par les navigateurs, le bouton *suivant* permettrait d'aller à la page suivante de façon automatique.
- **<META>** donne à votre document des informations qui seront lues par le serveur httpd. Ces informations sont généralement votre nom (NAME), le nom de l'auteur (AUTHOR), le contenu (CONTENT), une directive HTTP-EQUIV qui prend souvent la valeur Expires, Keywords, Reply_to. La directive *Expires* est utilisée par la plupart des navigateurs pour donner une date à partir de laquelle la page ne doit plus être conservée dans le cache de votre logiciel mais bien rechargée sur le serveur à chaque passage sur la page.
  La directive META sert également à donner la liste des mots clés du document par l'emploi du mot réservé `KeyWords` :
  Exemple :
  1. `<META HTTP-EQUIV="Expires" CONTENT="Monday, 01-Jan-96" 16:26:30 GMT">`
  2. `<META NAME="KeyWords" CONTENT="saint-martin, pape, pope, touraine,loire-valley, val de loire, tours, france, limousine, limo,loire-valley, chateaux, chateaux-country, driver, car, french wine,wine tastings, val-de-loire">`
- <**BASE HREF= ...**> donne la base de l'adresse URL qui sera placée devant les références relatives dans le document, de façon à ce que hors contexte les fichiers soient cherchés à l'adresse : BASEadresse relative
  Exemple : `<BASE HREF=http://www.pot.fr/>` permettra de référencer les adresses du genre *toto.html* à l'adresse http://www.pot.fr/toto.html

f) Liens et ancres

Un lien permet de définir une région sensible au clic souris dans un document et en même temps, l'endroit ou l'on va se retrouver après ce clic. Cette adresse peut être un document différent, par exemple, mais également un endroit précis (défini par une ancre) dans le document.

La syntaxe générale d'un *lien* est la suivante :

<A HREF=adresse **rel=**top|contents|index|glossary|copyright|next|previous|help|search **rev**=top|contents|index|glossary|copyright|next|previous|help|search **title**=valeur>texte **</A>**

Les attributs rel, rev, et title sont peu usités et ont les mêmes significations que celles exposées dans la balise **LINK**

- **<A NAME="signet"> . . . </A>** définit un nom de référence, c'est à dire un nom de signet ou d'ancre, qui sera atteint par un lien hypertexte.
- **<A HREF="#nom de signet"> . . . </A>** définit un lien vers un signet dans le document, c'est à dire un lien hypertexte, qui conduira vers un signet après un clic dessus.
- **<A HREF="URL"> . . . </A>** définit un lien vers une URL. Une URL est un lien hypertexte vers un document externe qui peut être situé sur un autre serveur. (Voir la définition dans le chapitre sur les Web)
- **<A HREF="URL#signet. </A>** définit un lien vers un URL au signet défini.

Exemple : <A name="CHAP1">Chapitre 1**</A>** définit un signet nommé CHAP1 associé aux mots Chapitre 1

Pour définir un lien hypertexte quelque part ailleurs sur le mot *point*, qui après avoir été cliqué, amène vers les mots Chapitre 1, on fait une définition de lien **<A href="**#CHAP1**">**Point**</A>**, et le mot *Point* sera marqué en bleu.

Rappelons qu'une URL est toujours de la forme : **ressource://host.domaine:port/pathname**

où la ressource peut être : file, http, news, gopher, telnet, wais ou mailto. Les éléments ressources, **port** et **pathname** sont optionnels.

Sur PC, un lien à un document constitué par un fichier situé sur un lecteur de disque défini par l'unité c: se fera par l'URL :

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

Le symbole : est remplacé par | et // est remplacé par /// pour éviter la confusion avec un nom de serveur.

Pour définir les liens URL, généralement en bleu, on utilise la balise **<A HREF=...>** et pour définir l'endroit ou le signet vers lequel on se rend, on utilise la balise **<A NAME=...>**.

Enfin le chargement d'un fichier sons, vidéo, compressé peut être fait au moyen de cette même balise de la façon suivante :

```
<A HREF=fichier>cliquez ici pour charger le fichier</A>
```

L'argument *mailto:adresse* permet d'envoyer un courrier électronique à l'adresse correspondante.

Ainsi:

```
<A HREF=mailto:david.rousse@wanadoo.fr> texte </A>
```

permet d'envoyer un courrier électronique à l'utilisateur david.rousse@wanadoo.fr en cliquant sur le mot *texte*.

Cette directive *mailto* permet de spécifier le champ *sujet* normalement renseigné dans le courrier par ajout de l'argument *subject* sous la syntaxe suivant :

```
<A HREF="mailto:david.rousse@wanadoo.fr?subject=Sujet"> texte </A>
```

g) Balises de mise en forme

Ces commandes permettent de mettre des parties de texte dans un format donné.

Sans attribut, le texte apparaît comme ceci.

Nous allons voir les différentes possibilités d'affichage. Il est à remarquer que ces balises influent sur la *charte graphique* du document mais peuvent ne pas être implémenté par tel ou tel éditeur.

Gras : <B> Ceci permet d'afficher en gras </B>

Italique : <I> Ceci permet d'afficher en italique </I>

Souligné : <U> Ceci permet d'afficher en souligné</U>

Exemple de combinaison : Gras Italique : <B> < I> Gras italique </I></B>

Adresse : <ADDRESS> Ceci permet d'afficher une adresse </ADDRESS>

Blockquote : <BLOCKQUOTE> ceci permet d'afficher un bloc avec retrait à droite</BLOCKQUOTE>

Citation : <CITE> Ceci permet d'afficher des citations </CITE>

```
Code : <CODE> Ceci permet d'afficher du code </CODE>
```

Remarque : cette commande est bien utile dans la mesure où elle respecte les retours-ligne du texte sans demander des marques
 ou <P> en fin de chaque ligne.

```
Machine à écrire : <TT> Affichage style machine à écrire </TT>
```

Variable : <VAR> Ceci permet d'afficher du texte de type variable </VAR>

```
Préformaté : <PRE  WIDTH=nb> Ceci permet de mettre un style préformaté
avec éventuellement une longueur de ligne
si la variable WIDTH=nb </PRE WIDTH>.
```

La balise PRE est la seule qui permet :

- de respecter les caratères retour chariots
- les caractères *espace* d'un fichier texte (car en HTML plusieurs caractères espace sont ramenés à un seul)

Remarque : cette commande est utile dans la mesure où elle respecte les retours-ligne du texte sans demander des marques
 ou <P> en fin de chaque ligne.

```
Exemple : <SAMP> Ceci permet d'afficher un exemple </SAMP>
```

~~Rayé : <STRIKE>Ceci vous permet de barrer un texte </ STRIKE>~~

~~Rayé <S>Cette balise est amenée à remplacer la précédente</S>~~

Epais : <STRONG> Ceci permet de mettre des caractères épais </STRONG>

Clignotant : <blink> Ceci vous permet de mettre du texte en clignotant mais n'est pas dans la norme HTML 3.2!</blink>

<H1>Titre de niveau 1</H1>

<H2>Titre de niveau 2</H2>

<H3>Titre de niveau 3</H3>

<H4>Titre de niveau 4</H4>

<H5>Titre de niveau 5</H5>

<H6>Titre de niveau 6</H6>

Les balises H1 H6 peuvent comporter un attribut **`ALIGN=LEFT|CENTER|RIGHT`** `permettant l'alignement du texte à gauche, au centre ou à droite.`

Emphase : <EM> sert à mettre en relief un groupe de caractères </EM>

`Clavier : <KBD> ceci permet d'afficher les entrées clavier </KBD>`

Définition : <DFN> . . . </DFN> pour afficher une définition

Certaines commandes peuvent ne rien afficher de notable dans votre document. Tout cela dépend de la finesse de votre lecteur de Web.

h) Balises de définitions

Ces commandes permettent de donner des listes de plusieurs types.

Listes de définitions

permet de donner une liste de définitions

```
<DL>
<DT> Terme 1 à définir
<DD> Définition du terme 1
<DT> Terme 2 à définir
<DD> Définition du terme 2
</DL>
```

Exemple :

Terme 1 à définir

Définition du terme 1

Terme 2 à définir

Définition du terme 2

Listes de répertoires

permet de donner une liste de répertoires

```
<DIR>
<LI> texte 1
<LI> texte 2
</DIR>
```

Exemple :

- texte 1
- texte2

Listes de numéros

permet de donner une liste précédée d'un numéro s'incrémentant automatiquement.

```
<OL>
<LI> texte 1
<LI> texte 2
</OL>
```

Exemple :

1. texte 1
2. texte 2

La balise peut prendre les arguments suivants :

```
<OL TYPE=A|a|I|i|1 START=nb COMPACT>
```

Outre les numéros 1, 2, 3 etc.. vous pouvez accéder aux numéros A,B,C etc ... et pouvez commencer à un autre rang que le premier (START).

COMPACT permet l'écriture de numéros plus compacts.

de plus **<LI VALUE=nb>**

permet de changer la valeur de la numérotation en regard d'une ligne.

Liste à puces

permet de donner une liste précédée d'une puce.

```
<UL>
<LI> texte 1
<LI> texte 2
</UL>
```

Exemple :

- texte 1
- texte 2

La balise peut prendre les arguments suivants :

```
<UL TYPE=disc|circle|square>
```

Ainsi il est possible de décider que la puce sera un disque vide, un cercle ou un carré.

```
<UL TYPE=disc>
<LI> texte 1
<LI> texte 2
</UL>
```

Listes de menus

permet de donner une liste menus.

```
<MENU>
<LI> texte 1
```

```
<LI> texte 2
</MENU>
```

Exemple :

- texte 1
- texte 2

Les listes peuvent être emboîtées en niveaux. Par exemple les lignes suivantes seront emboîtées:

- Niveau 1
    - Niveau 2
    - Niveau 2
- Niveau 1

Ceci se programme bien évidemment de la façon suivante :

```
<UL>
<LI>Niveau 1
<UL>
<LI>Niveau 2
<LI>Niveau 2
</UL>
<LI>Niveau 1
</UL>
```

i) Les images

Une des originalités des pages Web est la large utilisation des images. Les images sont au format GIF ou JPG et sont intégrées dans le document par la commande :

```
<IMG SRC="nom du fichier">
```

Cet ordre peut être agrémenté des options suivantes :

ALT : pour afficher un texte si le lecteur de Web ne sait pas lire l'image.

Exemple : `<IMG ALT="texte" SRC="nom du fichier">`

La balise **IMG** peut être agrémentée des arguments suivant :

ALIGN=left|right|top|texttop|middle|absmiddle|baseline|bottom|absbottom

Notons que seules les valeurs soulignées sont admises par la norme HTML 3.2 mais on recontre les autres dans tous les outils de manipulation HTML.

Ces différentes valeurs permettent d'aligner les images à gauche, à droite, en haut, en haut du texte, au milieu, au milieu par rapport à la ligne, en bas, en bas de la ligne etc.

WIDTH=valeur HEIGHT=valeur

Les attributs **WIDTH** et **HEIGHT** permettent de donner la longueur et la largeur de l'image, ce qui a pour effet d'accélérer la visualisation de l'image en évitant les temps de calcul. Cette option permet la visualisation du texte avant la visualisation des images avec les navigateurs les meilleurs. Elle est donc à utiliser.

BORDER=valeur

donne l'épaisseur du trait de marque d'URL autour de l'image.

VSPACE=valeur HSPACE=valeur

VSPACE contrôle l'espacement vertical en pixels au-dessus et en dessous de l'image. **HSPACE** contrôle l'espacement horizontal en pixels à droite et à gauche de l'image.

Les valeur **ISMAP** et **USEMAP** sont également comprises.

La procédure **ISMAP** permet de repérer la position de la souris sur un graphique pour exécuter des actions différentes suivant la région cliquée (IMAGEMAP).

Une des pages du site vous donne un exemple de cette utilisation, puisqu'elle donne la possibilité de cliquer sur des « planètes » pour accéder à de nouvelles pages.

j) Les formulaires

Les formulaires sont les seuls outils permettant de récolter des informations en provenance des lecteurs de Web. Ils seront traités sur le serveur par des programmes appelés CGI.

La puissance des procédures CGI est évidente, elle permet à l'utilisateur d'un lecteur de Web de faire des actions sur le serveur.

Ces actions peuvent être :

- des exécutions de programme sans paramètre,
- des exécutions de programme avec paramètres.

Dans les deux cas, le serveur pourra restituer un texte ou une page HTML de résultat spécialement conçue pour l'utilisateur qui a fait la requête. Mais dans la majeure partie des cas, la procédure CGI ne se contentera pas de restituer une page de compte-rendu, elle aura généralement pour but premier d'enregistrer les informations dans une base de données.

FORM

Les balises FORM permettent de définir les interfaces d'échange d'informations entre un utilisateur et le serveur http. Cette interface est graphique et utilise les listes déroulantes, les boutons poussoirs, les boutons radios et tous les éléments graphiques présents dans les systèmes actuels.

Plusieurs balises FORM peuvent être présentes dans un document HTML mais en aucun cas les balises FORM ne peuvent être emboîtées.

La balise FORM enverra une requête sur le serveur http sur lequel l'utilisateur est connecté lorsque ce dernier appuiera sur le bouton d'envoi de la requête.

La syntaxe d'une balise FORM est la suivante :

<FORM ACTION="url"> ... </FORM>

Les attributs suivants peuvent être utilisés :

- **ACTION** fournit une adresse URL sur laquelle la requête issue de la FORM va être envoyée. Si le champ ACTION est absent, la requête sera envoyée sur le serveur courant.
- **METHOD** est la méthode d'accès au serveur http. On trouve deux méthodes d'accès **POST** et **GET**.
    - GET : provoque une requête pour laquelle le champ de la requête est ajouté à l'URL. Cette méthode limite la taille du message à 200 caractères.
    - POST : Cette méthode utilise un envoi d'un message à part entière et non pas par le biais de l'URL. C'est la méthode recommandée.

- **ENCTYPE** donne la méthode d'encodage pour la FORM. Cet attribut n'est pris en compte que si la méthode est POST. Il y a une seule valeur possible : `application/x-www-form-urlencoded`. Dans les versions futures, ce champ permettra l'encryptage de données.

Le texte encadré par le couple de balises <FORM> et </FORM> peut contenir un ensemble de commandes:

- **INPUT**
- **SELECT**
- **TEXTAREA**.

INPUT

La commande INPUT est utilisée avec des attributs pour spécifier les caractéristiques de la commande clavier (ou souris) désirée.

Les différents attributs que peut utiliser la commande INPUT sont :

- **TYPE** qui peut prendre les valeurs :
  - **text** pour les entrées de type texte. Il est à noter que les caractères entrés par l'utilisateur sont convertis automatiquement avec les quotations (accents, champs &)
  - **hidden** l'entrée ne sera pas affichée dans la page HTML mais grâce à l'ordre VALUE la requête partira vers le serveur http avec la valeur donnée. (ceci permet de forcer une des valeurs qu'attend le serveur sans l'afficher)
  - **password** pour les entrées de type mot de passe, ou les caractères entrés seront cachés par des caractères astérisques.
  - **checkbox** pour les boîtes à cocher.
  - **radio** pour les boutons radios
  - **submit** un bouton poussoir qui provoque l'envoi de la requête FORM.
  - **reset** un bouton poussoir qui provoque la remise aux valeurs initiales de tous les champs de la FORM
  - **image** permet de remplacer le bouton par défaut par une image.
  - **file** permet de soumettre un fichier au serveur.
- **NAME** est le nom d'utilisation par l'auteur de la page, et qui n'est pas le nom affiché à l'utilisateur.
- **VALUE** sert à spécifier la valeur par défaut d'un champ de saisie.
- **CHECKED** indique que le bouton radio ou la boîte à cocher est sur ON.
- **SIZE** est la taille du champ pour les champs caractères. La valeur par défaut est 20. Si la zone valeur de size est remplie avec deux valeurs séparées par une virgule, le champ texte aura une longueur déterminée par la première valeur et un nombre de lignes défini par la deuxième valeur.
- **MAXLENGTH** est le nombre maximum de caractères qui sont acceptés pour les champs textuels. Cet attribut permet de gérer les défilements si MAXLENGTH est plus grand que SIZE.
- **SRC** est l'URL de l'image dans le cas où **TYPE=image** dans ce cas l'attribut **ALIGN** peut prendre les valeurs **LEFT|RIGHT|TOP|MIDDLE|BOTTOM.** Dans ce cas, les paramètres x et y sont passés au serveurs sous la forme du nom de la variable donnée par **NAME** suivi par les extensions .x et .y

- l'attribut **ENCTYPE** prend la valeur *multipart/form-data* dans le cas où **TYPE=file**

Exemple 1:

Entrez la valeur du champ 1

Entrez la valeur du champ 2

Entrez la valeur du champ 3

Envoyer

est obtenue par la programmation suivante :

```
<FORM METHOD="POST" ACTION="http://hoohoo.ncsa.uiuc.edu/htbin-post/post-
query">
Entrez la valeur du champ 1 <INPUT NAME="entry1">
Entrez la valeur du champ 2 <INPUT NAME="entry2">
Entrez la valeur du champ 3 <INPUT NAME="entry3">
<INPUT TYPE="submit" VALUE="Envoyer">
</FORM>
```

- Les champs 1, 2 et 3 s'appellent entry1, entry2 et entry3
- Le résultat de l'appui sur le bouton *Envoyer* donnera si vous avez entré toto, titi dans les deux premiers champs et rien dans le troisième : **"?entry1=toto&entry2=titi&entry3="**
- le champ ACTION est le nom de la procédure qui sera exécutée sur le serveur.

Exemple 2: La forme suivante

Nom

Prénom

1. marié
2. francais

Volontaire

Oui

Non

Ne sait pas.

Ok Annuler

est obtenue par la programmation suivante :

```
<FORM METHOD="POST" ACTION="http://hoohoo.ncsa.uiuc.edu/htbin-post/post-
query">
Nom <INPUT NAME="Nom"><P>
```

```
Prénom <INPUT NAME="Prenom">
<OL>
<LI> <INPUT TYPE="checkbox" CHECKED NAME="topping1" VALUE="Marie">Marié
<LI> <INPUT TYPE="checkbox" NAME="topping2" VALUE="francais"> francais
</OL>
Volontaire
<DL>
<DD> <INPUT TYPE="radio" NAME="callfirst" VALUE="Oui" CHECKED> <I>Oui</I>
<DD> <INPUT TYPE="radio" NAME="callfirst" VALUE="Non"> <I>Non</I>
<DD> <INPUT TYPE="radio" NAME="callfirst" VALUE="Ne sait pas"> <I>Ne sait
pas</I>
</DL>
<INPUT TYPE="submit" VALUE="Ok">
<INPUT TYPE="reset" VALUE="Annuler">
</FORM>
```

- La valeur **CHECKED** initialise la valeur on du champ topping1
- La valeur *reset* du champ **INPUT** permet de remettre aux valeurs initiales les différents champs
- On remarque l'utilisation d'un même nom *callfirst* pour indiquer que cet attribut pourra prendre les valeurs *Oui*, *Non* ou *Ne sait pas.*

Exemple 3 : La forme suivante

Champ 1 (normal):

Champ2 (40 caractères affichés) :

Champ3 (5 caractères seulement) :

OK Annuler

est obtenue par la programmation suivante

```
<FORM METHOD="POST" ACTION="http://hoohoo.ncsa.uiuc.edu/htbin-post/post-
query">
Champ 1 (normal): <INPUT NAME="entry1"> <P>
Champ2 (40 caractères affichés) : <INPUT SIZE=40 NAME="entry2"> <P>
Champ3 (5 caractères seulement) : <INPUT SIZE=5 MAXLENGTH=5 NAME="entry3">
<P>
<INPUT TYPE="submit" VALUE="OK"> <INPUT TYPE="reset" VALUE="Annuler">
</FORM>
```

Exemple de fichier à envoyer

Envoyer le fichier

Cet exemple est obtenu par la syntaxe suivant :

```
<FORM METHOD="POST" ACTION="http://hoohoo.ncsa.uiuc.edu/htbin-post/post-
query">
Exemple de fichier à envoyer <INPUT TYPE=file NAME=fichier> <P>
<INPUT TYPE=submit VALUE="Envoyer le fichier">
</FORM>
```

Remarque : Notons que la pseudo directive *mailto:* peut être utilisée pour faire l'envoi.

SELECT

La commande **SELECT** permet de donner la liste des champs accessibles pour les menus déroulants.

La commande SELECT est utilisée avec la syntaxe suivante :

```
<SELECT NAME="a-menu">
<OPTION VALUE=valeur> Option 1
<OPTION VALUE=valeur> Option 2
</SELECT>
```

Les attributs de la commande **SELECT** sont les suivants :

NAME est le nom symbolique de l'élément. C'est-à-dire le nom du champ utilisé par l'auteur de la page HTML. Ce nom n'est pas visible pour les utilisateurs.

SIZE donne le nombre d'éléments qui seront affichés dans le menu. Les autres valeurs seront accessibles au moyen d'un ascenseur.

MULTIPLE indique que la commande SELECT pourra avoir plusieurs sélections.

SELECTED indique que l'option est sélectionnée par défaut.

VALUE est un paramètre optionnel permettant de donner à la variable en regard de **NAME** la valeur désirée

TEXTAREA

La commande **TEXTAREA** est utilisée avec la syntaxe suivante :

<**TEXTAREA NAME**=... **ROWS**=.. **COLS**=.. > valeur par défaut</**TEXTAREA**>

La commande **TEXTAREA** est utilisée pour les entrées textes et spécifie :

- **NAME** le nom symbolique de l'entrée
- **ROWS** nombre de lignes du champ de saisie
- **COLS** est la longueur de la ligne de saisie du champ texte.

GET

Quand le bouton *submit* est pressé, le contenu de la FORM est assemblé dans la valeur URL qui ressemblera à ceci :

```
action?name=value&name=value&name=value
```

POST

Le contenu de la FORM est passé par un message de requête.

k) Polices de caractères

Les polices de caractères peuvent être modifiées sur l'ensemble d'une page par la directive :

```
<BASEFONT SIZE=valeur>
```

Valeur sert à changer la taille de la police par défaut. Sept tailles sont disponibles et la valeur par défaut est 3, la plus grosse étant de valeur 6.

Une partie de texte peut voir sa taille modifiée si elle est entourée de la balise :

```
<FONT SIZE=valeur> ..</FONT>
```

Valeur sert à spécifier , ici encore, 7 tailles de polices (le défaut étant 3).

La syntaxe **SIZE=+i** ou i peut être compris entre 1 et 7 est admise, elle permet de donner une taille relative par rapport à la taille en cours.

De même les caractères peuvent être modifiés par l'attribut **COLOR**, ainsi

```
<FONT COLOR="#FF0000">permet de colorier les caractères (ici en
rouge)</FONT>
```

Permet de colorier les caractères (ici en rouge)

Notons que les noms de couleurs peuvent être donnés en toutes lettres (Red,Blue,Green,..)

<**FONT COLOR**="Green">**Green**</**FONT**>

I) Les Tableaux

Les tableaux sont une structure importante du langage HTML dans la mesure où ils permettent d'organiser le texte avec ou sans visualisation de cadres. Les tableaux peuvent être emboités.

Un tableau est encadré par la balise **TABLE**

- <**TABLE ...**></**TABLE**> Cette commande permet de définir des tableaux, elle inclut toutes les commandes sur le nombre de lignes et le nombre de colonnes. D'autres attributs (traits, ombres) peuvent être définis.

- <**TR ...**></**TR**> Chaque balise TR définit une ligne du tableau. Donc pour trois lignes dans le tableau, on retrouvera trois balises TR.

- <**TD ...**></**TD**> Chaque balise TD définit une cellule à l'intérieur d'une ligne. On trouve autant de balises TD que de colonnes par ligne.

  Le code :

  ```
  <TABLE BORDER=0>
  <TR>
  <TD>Colonne1 Ligne1</TD>
  <TD>Colonne2 Ligne1</TD>
  <TD>Colonne3 Ligne1</TD>
  </TR>
  <TR>
  <TD>Colonne1 Ligne2</TD>
  <TD>Colonne2 Ligne2</TD>
  <TD>Colonne3 ligne2</TD>
  </TR>
  </TABLE>
  ```

  affichera le tableau suivant :

Colonne1 Ligne1	Colonne2 Ligne1	Colonne3 Ligne1
Colonne1 Ligne2	Colonne2 Ligne2	Colonne3 ligne2

- <**TH ...**></**TH**> Ce couple de balises concerne les entêtes du tableau, c'est-à-dire la première ligne ou la première colonne. La différence avec une balise TD réside dans l'alignement, qui par défaut est centré et la police de caractères qui est en gras. Le code

  ```
  <TABLE BORDER=0>
  <TH> Titre1</TH>  <TH> Titre2 </TH> <TH> Titre3 </TH>
  <TR>
  ```

```
<TD>Colonne1 Ligne1</TD>
<TD>Colonne2 Ligne1</TD>
<TD>Colonne3 Ligne1</TD>
</TR>
<TR>
<TD>Colonne1 Ligne2</TD>
<TD>Colonne2 Ligne2</TD>
<TD>Colonne3 ligne2</TD>
</TR>
</TABLE>
```

affichera le tableau suivant :

Titre1	Titre2	Titre3
Colonne1 Ligne1	Colonne2 Ligne1	Colonne3 Ligne1
Colonne1 Ligne2	Colonne2 Ligne2	Colonne3 ligne2

- <**CAPTION ...**></**CAPTION**> La balise **CAPTION** peut être utilisée dans un texte délimité par les balises **TABLE** (mais pas dans ceux délimités par **TR** ou **TD**), elle sert à donner un titre au tableau et peut être en haut (**ALIGN=TOP**) ou en bas (**ALIGN=BOTTOM**) Ainsi le code :

III LA CRÉATION

Pour chaque paragraphe, j'ai pris le partie d'expliquer le sujet traité de manière générale puis de donner, si besoin, une illustration tirée du site Web d'Inforadour – Adix.

a) Généralités

Une page sur le Web est faite de texte, bien sûr, parmi lequel figure les hyperliens, ces deux types d'éléments formant le fichier HTML. Mais elle comp rend aussi bien souvent des images, des animations... Chacun de ces éléments nécessite un traitement spécifique à l'aide d'un outil spécifique, que je vais présenter. Par ailleurs, afin de visualiser au fur et à mesure l'évolution de son travail, il est évidemment nécessaire de disposer d'un navigateur Web comme Netscape Navigator ou Microsoft Internet Explorer.

b) Les outils de création de pages au format HTML

Un fichier HTML étant un fichier texte, un simple éditeur de texte comme SimpleText (MacOS) ou Bloc-notes (W95) en permet la composition.
Il existe néanmoins des logiciels destinés à simplifier le travail du néophyte et à le rendre plus rapide en automatisant certaines tâches. Ce sont les Editeurs HTML, dont il existe deux versions :

- **Les Editeurs WYSIWYG** (What You See is What You Get), qui permettent de visualiser directement le fichier tel qu'il apparaîtra dans la fenêtre du Navigateur. Ce genre d'éditeur peut sembler plus accessible à un débutant mais il présente plusieurs inconvénients. D'abord celui de n'être absolument pas didactique : vous pouvez passer des heures sur un éditeur WYSIWYG, vous n'en saurez pas plus sur le HTML. Ensuite, ces éditeurs ont souvent la réputation de produire du code HTML un peu fantaisiste, parfois difficilement interprétable par certains navigateurs et en tout cas très difficile à corriger à la main. Pour finir, ils ne permettent pas toujours d'obtenir des mises en pages finement calées. Ces éditeurs peuvent néanmoins être très utiles pour construire rapidement une page simple.

- **Les éditeurs HTML de type textuel**, qui eux proposent de travailler directement sur le fichier texte, en automatisant la plupart des insertions de balises. Le résultat est alors visualisé pour contrôle dans votre navigateur habituel, ou parfois directement à l'intérieur du logiciel qui utilise un

navigateur interne. Ces éditeurs cumulent plusieurs avantages : vous avez toujours le code HTML sous les yeux, ce qui vous permet d'apprendre les balises les plus courantes en même temps que vous travaillez. Ce code est modifiable à tout instant, vous pouvez ainsi l'affiner si vous n'êtes pas satisfait de ce que vous propose le logiciel. C'est une excellent solution pour apprendre le HTML tout en se faisant plaisir en construisant sa première page.

- **Les convertisseurs (ou assistants)** qui , partant d'un texte élaborés avec un traitement de texte font leur possible pour convertir leur formatage en langage HTML.

Pour ma part, j'ai utilisé deux éditeurs WYSIWYG (en l'occurrence Microsoft FrontPage 98 et Symantec Visual Page) et Notepad pour visualiser les codes – sources de certaines pages ou de certains scripts.

c) Les outils graphiques

Les images qui apparaissent au sein des pages Web doivent être au format JPEG ou GIF, porter un nom ne comportant ni accent ni caractères spéciaux (l'idéal étant un nom d'une dizaine de caractères maximum plus une extension, le tout en minuscule), être correctement cadrées et mesurées, et peser un poids raisonnable (un poids dépassant 30 Ko est fortement déconseillé).

Pour retravailler les images afin qu'elles répondent aux critères que l'on désire, il faut utiliser des **éditeurs graphiques**. Ces derniers permettent également de travailler des animations (GIF animés par exemple), de construire des Images Map à partir desquelles on peut naviguer facilement, etc.

Pour ma part, j'ai utilisé les outils qui étaient mis à ma disposition, à savoir :

- **Paint Shop Pro 4.14**, logiciel de traitement d'images
- **Photoshop 4.0**, un concurrent de Paint Shop Pro 4.14
- **Microsoft Gif Animator** , qui permet de créer facilement des icônes ou des images animées.

IV LES LANGAGES DU WEB

A côté du langage HTML qui est la base du Web, d'autres langages sont couramment utilisés dans la création de pages Web.

a) JAVA

JAVA est un langage compilé et orienté objet, qui permet de créer de vrais programmes. Ces programmes, ou *applets*, sont appelés par les pages Web, et s'exécutent dans la fenêtre du navigateur. Ils permettent entre autres de créer des animations complexes comme le défilement d'un texte. Contrairement à HTML et JavaScript, le langage JAVA demande un gros travail d'apprentissage, pour qui souhaite créer ses propres applets.
Côté page HTML, une nouvelle balise appelée *APPLET* est utilisée pour insérer des applets. La syntaxe de cette balise est :

```
<APPLET code="nom_applet.class"  width=20 height=20>
</APPLET>
```

Pour ma part, j'ai téléchargé plusieurs applets sur mon PC et j'en ai utilisé trois sur le site : un premier sur la page d'accueil, un second sur la page centrale du site et enfin un troisième pour une table des matières.

b) JavaScript

Si le langage Java impose le téléchargement de classes, c'est à dire de programmes précompilés depuis le serveur, JavaScript provoque l'exécution de programmes non compilés mais interprétés et contenus dans le corps de la page HTML.

Alors que Java fut proposé par SUN et ensuite adopté par Netscape 2.0 , Javascript n'est compris que par Netscape et par Internet Explorer V3.0.

Pour passer des considérations marketing aux considérations techniques, revenons sur le fait que JavaScript est interprété là où Java est compilé. Ceci a plusieurs implications :

- le code Java est protégé des actes de copie frauduleuse,
- le code JavaScript est moins *typé* donc moins robuste mais plus accessible à des non informaticiens.
- JavaScript est accessible à des auteurs de pages HTML alors que Java est réservé à un public plus professionnel.

Les principales sources d'information sur JavaScript sont aujourd'hui :

- la FAQ [www.freqgrafx.com/411/jsfaq.html] JavaScript (en anglais)
- la newsgroup comp.lang.javascript [news:comp.lang.javascript]
- la page Yahoo relative à JavaScript [www.yahoo.com/Computers_and_Internet/Languages/JavaScript/]

J'ajoute la page de Timothy : Timothy's JavaScript Examples [www.essex1.com/people/timothy/js-index.htm] qui est contient un grand nombre d'exemples de programmes en JavaScript.

En pratique, JavaScript peut être implanté dans une page HTML de 3 façons :

- par la balise SCRIPT :

```
<SCRIPT LANGAGE="JavaScript">
code
</SCRIPT>
```

On le voit, le choix du langage ne se limite pas à JavaScript et reste ouvert à d'autres langages comme Visual Basic, que supporte Internet Explorer.

- en utilisant les événements :

Les événements sont les résultats d'une action de l'utilisateur, comme par exemple un clic sur l'un des boutons de la souris.

La syntaxe de ces événements est :

```
<balise eventHandler="code JavaScript">
```

où *balise* est le nom d'une balise et eventHandler est le nom d'un événement.

Par exemple, pour utiliser la fonction *exemple* quand un utilisateur appuie sur un bouton, l'instruction suivante

```
<INPUT TYPE=BUTTON VALUE=Essai onClick="exemple(this.form)">
```

Une fonction ou bien un code JavaScript peut être inséré comme valeur de l'argument *exemple*

- en mettant le code dans un fichier séparé uniquement à partir de la version 3 de Netscape :

Utiliser alors la balise SCRIPT comme suit :

```
<SCRIPT LANGAGE=JavaScript SRC=source.js> </SCRIPT>
```

source.js peut être un fichier dans le répertoire courant ou bien une URL pointant vers un fichier distant.

Sur le site d'Inforadour – Adix, j'ai inséré un script pour faire défiler du texte dans la barre d'état (en appliquant la 1° méthode).

c) Programmes CGI

Ce sont des programmes exécutés sur les serveurs de pages WEB. Les traitements d'informations et les modifications de mise en page sont effectués avant la composition ou le transfert des pages Web vers le navigateur client.
Leurs champs d'application sont très vastes. Ils sont utilisés, par exemple, pour le stockage, le traitement et, ou, la restitution d'informations, l'interrogation des bases de données, etc..
Perl est un langage de programmation couramment utilisé pour créer des programmes CGI (mais on peut également utiliser le C ou d'autres langages) sur les systèmes UNIX (il est aussi disponible pour d'autres systèmes d'exploitation). Il est un peu compliqué pour les gens qui, comme moi, ne sont pas habitués au monde Unix, mais il est très simple à mettre en œuvre. Son principal problème est son temps d'exécution (C'est un langage interprété).
On peut reconnaître l'appel à un programme CGI par l'extension des liens appelés (en .cgi, .pl, .exe, .asp etc..).

Quelques références en anglais:

CGI Programming Techniques in Perl : Pour un apprentissage de Perl en douceur.

ScriptSearch : Des centaines de scripts dans tous les domaines et dans tous les langages.

The CGI Resource Index: Encore plus de scripts, plus des sources de documentation.

Sur le site que j'ai réalisé, une de pages contenait un formulaire ; j'ai donc recherché sur l'Internet un script CGI qui puis se traiter ce dernier.

V PUBLICATION DU SITE

a) Transfert du site

Une fois les pages créées, il faut les télécharger (ou uploader) sur le serveur qui va héberger votre site (voir rubrique hébergement)

Pour ce faire, on utilise le protocole FTP (File Transfert Protocol), qui nécessite l'emploi d'un programme dédié. Actuellement 3 logiciels se partagent le marché; tous les trois sont des shareware ou des freeware, au moins dans leur version light, largement suffisante pour cet usage.

- **Principes d'utilisation**

Le serveur FTP du site qui vous héberge sert exclusivement à télécharger vos pages. On y accède via le logiciel avec une adresse de type ftp.serveur.com. Il faut configurer la connexion au serveur FTP en indiquant son login (ou pseudo) et son mot de passe, pour accéder à la zone disque qui vous est réservée pour le stockage des pages.

- **Les logiciels de téléchargement**

Ces trois logiciels proposent des fonctions sensiblement équivalentes, le choix se fera donc principalement sur l'interface.

FTP Expert (Visicom) : le logiciel de Visicom présente l'avantage d'être totalement en français.

Cute FTP : L'interface graphique (icônes) est bien faite. La licence coûte 35 $…

WS FTP : WS FTP est en anglais, mais il est facile a configurer, et son utilisation est assez intuitive. De plus, il est peu gourmand en espace disque. La version limitée (WS FTP LE) est gratuite pour les "home users".

Pour ma part, je ne me suis pas chargé du transfert du site car, l'entreprise faisant partie d'un groupe informatique, des discussions devaient avoir lieu à propos de la manière d'héberger le sites des différents filiales.

b) L'hébergement

Les offres d'hébergement gratuit sur le Web existent bien - sûr (Mygale) mais j'ai surtout recherché les offres pour les professionnels.
Les offres d'hébergement professionnel, comme leur nom l'indique, sont adaptées à une utilisation professionnelle d'Internet. Elles offrent donc un service différent des services d'hébergement gratuits. Pour ne pas se tromper, il est important de savoir **avant de choisir** à quoi va servir cet espace, et ce qu'il va contenir.

Voici quelques uns des points qui me semblent importants:

Evaluer l'espace disque nécessaire lors du lancement du site
Les tarifs d'hébergement varient significativement en fonction de la taille du site

Evaluer les perspectives d'évolution de l'espace disque nécessaire à moyen terme (6 mois à un an)
Si le site est appelé à grossir, vérifier les tarifs de l'hébergeur pour les tailles supérieures : il serait dommage de bénéficier d'une offre d'appel sur les petits sites, et de devoir payer une grosse majoration si le site grossit !

Le site proposera-t-il du téléchargement ?
Plusieurs hébergeurs refusent les sites qui proposent du téléchargement, qui sont très gourmands en bande passante (le téléchargement monopolise une grosse partie de la capacité du serveur à envoyer des données à plusieurs utilisateurs en même temps)

Evaluer le trafic que va générer le site
Le trafic, c'est la quantité de données que le serveur va transférer du fait de l'activité du site. Certaines offres d'hébergement prennent en compte ce critère dans la détermination du tarif. Il faut donc considérer la taille des fichiers, et le nombre de visites.

Combien de comptes E-mail sont nécessaires au site ?
Dans la plupart des cas, une seule adresse suffit. Elles sont en général facturées, au-delà de la première adresse.

Quelques exemples (renseignements trouvés sur le Net)

Rega : Rega est une offre d'hébergement créée par Infonie, service en ligne français.
Prix pour un site de 10 Mo : 4800 FHT.
Prix de la création de nom de domaine : en .fr , 1200 FHT

RapidSite : le poids lourd de l'hébergement pro. Rapidsite offre une gamme très complète, pour des sites de 10 à 200 Mo.

Prix pour un site de 10 Mo : 200 FHT / mois, avec une adresse E-mail.
Prix de la création de nom de domaine :
En .fr , 1300 FHT + 200 FHT par an
En .com, 70 $, payable directement à l'Internic

Netlink : Filiale française d'un gros fournisseur d'hébergement anglais, Netlink propose des tarifs très intéressants :

Prix pour un site de 10 Mo : 2490 FHT, création de nom de domaine comprise.

VI EN RESUME

LES « DIX COMMANDEMENTS » DE LA CREATION D'UNE PAGE WEB

Trouvez le sujet du site et posez-vous des questions sur la pertinence de ce site. Fixez-vous des objectifs !

Créez le principe de navigation en fonction de votre arborescence (Barre de boutons, images cliquables, palette de navigation dans une fenêtre externe,…)

Rassemblez les informations que vous voulez diffuser (Textes, images, sons, animations).

Ecrivez l'HTML en se conformant aux recommandations du W3C et contrôler en fonction du ou des logiciels visés (Internet Explorer ou Netscape Navigator/Communicator).

Créez l'arborescence du site et classer les informations dans chaque rubrique du site. Faire une ébauche papier.

Vérifiez les liens et contrôlez la taille des pages, images (<=40 Ko).

Définissez graphiquement une métaphore afin de lui donner une identité forte et reconnaissable. Créez une charte graphique à laquelle vous vous conformerez tout le long de la création ET de la mise à jour. Pensez aussi à l'avenir de manière à ne pas rester bloqué ! !

Nettoyer votre dossier des médias et des fichiers HTML inutilisés.

Préparez vos médias pour le Web (Scans, conversion des médias aux formats du Web).

Diffuser l'adresse de votre site et communiquez sur celui-ci.

CONCLUSION

Au fils de la conception du site Web d'Inforadour – Adix, j'ai pu emprunter de nouveaux chemins dans le monde complexe de l'Informatique : la découverte du HTML, l'utilisation de nouveaux logiciels, l'initiation à de nouveaux langages de programmation sont, entre autres, des domaines d'un indéniable intérêt.
Je tiens donc à remercier le personnel d'Inforadour et plus particulièrement son directeur, **Mr Garros**, pour leur accueil pendant un mois.

Plus généralement, cette activité m'a permis de prendre conscience que la réalisation d'un projet, quel qu'il soit, est une opération délicate : une méthode claire et précise de travail s'impose donc.

Dans tous les domaines, la fin dépend des moyens…