

HTML 4

Introduction



SYLVAIN
RENARD

Le présent document est une simple introduction à HTML 4. Il utilise la norme HTML 4.01 dont il reprend les points principaux dans un but pédagogique. Il omet volontairement de nombreuses questions et de nombreux détails. On se reportera aux ouvrages spécialisés ou à la norme disponible sur www.w3c.org pour en savoir plus.

Généralités

Le World Wide Web repose sur **3 mécanismes** :

- un **système de nommage uniforme** pour la localisation des ressources sur le web.
- **des protocoles** pour accéder à ces ressources.
- **l'hypertexte** pour naviguer entre elles.

Chaque ressource sur le web possède un identifiant de ressource uniforme ou URI.

Un URI est composé de 3 parties :

- le nom du protocole utilisé pour accéder à la ressource
- le nom de la machine hébergeant la ressource
- le nom de la ressource sous forme d'un chemin.

Exemple :

`http://www.ruses.com/estienne/estienne.pdf`

`http` désigne le protocole

`www.ruses.com` la machine

`/estienne/estienne.pdf` le chemin du document.

Les URL sont un sous-ensemble des URI.

Le HTML a été créé par **Tim Berners-Lee** au CERN et utilisé d'abord par le navigateur Mosaic. Son importance s'est développée dans les années 1990 d'où la nécessité d'une spécification.

La spécification HTML 2.0 a été publiée en 1995 par l'IETF, la spécification HTML 3.2 en 1997 par le World Wide Web Consortium. La spécification qui fait l'objet de cet exposé est la 4.01, elle date du 24 décembre 1999.

HTML a été développé pour que des appareils différents puissent utiliser le web : les ordinateurs mais aussi des appareils de synthèse et de reconnaissance de la parole. Il a aussi été conçu pour que l'on puisse diffuser des documents dans toutes les langues.

Cette spécification permet l'utilisation des **feuilles de style** dans le but de séparer la structure du document de sa présentation. Pour autant HTML arrive au bout de son évolution dans une certaine confusion. Les balises de structure coexistent avec des balises de formatage physique sans espoir de clarification. L'avenir est à XML et d'abord au langage de transition qu'est XHTML.

HTML autorise l'utilisation de scripts dans différents langages (on notera, dans la pratique, l'importance de Javascript).

Les auteurs de la norme conseillent de séparer la structure et la présentation, de tenir compte des besoins des personnes handicapées et d'aider les agents utilisateurs dans la restitution progressive.

Rapport avec SGML

SGML est un système qui permet de **définir des langages de balisage**.

Chaque langage défini en SGML est une **application SGML**. Une application se caractérise par une déclaration SGML (définition des caractères et des délimiteurs qui peuvent apparaître dans l'application), une définition de type de document, la **DTD** (définition de la syntaxe du balisage), une spécification qui décrit la sémantique imputée au balisage et les instances du document contenant les données et le balisage.

HTML est défini par une DTD.

La structure d'un document HTML

Un document HTML 4 se compose de 3 parties :

- une ligne contenant les informations de version HTML,
- une section HEAD qui est déclarative, elle n'affiche rien dans le navigateur,
- le corps du document implémenté par l'élément BODY ou l'élément FRAMESET.

Les sections 2 et 3 sont entourées par l'élément HTML.

Voici un premier exemple de document HTML :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<HTML>
  <HEAD>
    <TITLE>Ceci est un document HTML</TITLE>
  </HEAD>
  <BODY>
    <P>Bonjour tout le monde !</P>
  </BODY>
</HTML>
```

La déclaration de type de document indique la DTD utilisée pour le document.

Il existe 3 DTD pour la norme HTML 4.01

- **DTD HTML 4.01 strict**

La déclaration correspondante est la suivante :

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

- **DTD HTML 4.01 transitoire** (DTD strict augmentée des éléments et attributs déconseillés)

La déclaration correspondante est la suivante :

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

- **DTD HTML 4.01 de jeu d'encadrement** (DTD transitoire augmentée des cadres)

La déclaration correspondante est la suivante :

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

Les URI dans ces déclarations permettent aux navigateurs de télécharger les DTD qui leur sont nécessaires pour interpréter les documents.

L'élément HTML

La balise **<HTML>** vient immédiatement après la déclaration de type de document. La balise fermante correspondante **</HTML>** termine le document.

L'en-tête du document

L'élément HEAD

Les balises **<HEAD>** et **</HEAD>** définissent l'en-tête du document.

Les éléments contenus entre ces deux balises ne sont pas affichés par le navigateur mais lui fournissent des informations utiles.

L'élément TITLE

Les balises **<TITLE></TITLE>** permettent de définir un titre unique pour le document. C'est celui qui sera affiché par le navigateur comme titre de la fenêtre.

Il est conseillé pour faciliter la lecture et surtout l'indexation des documents de donner des titres précis qui évoquent le contenu de la page. Préférer **La maison de la Bretagne : introduction à introduction**. Préférer **Graphiloup : les images des loups et des renards à Les illustrations**.

L'élément META

Les balises META permettent de publier des métadonnées. La balise **</META>** est interdite.

Voici quelques exemples courants :

```
<META name="Auteur" content="Isabelle Boulay">
```

```
<META name="Auteur" lang="fr" content="Jeanne Cherhal">
```

```
<META http-equiv="Expires" content="TUE, 20 Aug 2005 12:00:00
GMT">
```

```
<META name="keywords" lang="fr" content="violon,mandoline,
contrebasse,tbouret">
```

```
<meta name="description" content="Devenez acteur de la
révolution de la musique numérique. Simplement."
```

En 3 étapes : Importez toute votre musique sur votre Mac ou PC. Organisez-la avec iTunes, le meilleur juke-box numérique au monde. Transférez-la sur votre iPod et c'est parti !">

```
<meta name="keywords" content="f&eacute;te de la musique, fete de la musique, f&eacute;te musique, fete musique (Paris), mp3, concert, 21 juin, mixer, instrument, instruments, enregistrer, graver un CD, ipod, lecteur de musique num&eacute;rique, itunes, garaband, liste de lecture, juke-box, partager la musique, Mac, PC">
```

```
<meta name="keywords" content="PAO, DTP, formation, art, graphique, communication, multimedia, hebergement, publicit&eacute;, XPress, Illustrator, Photoshop, Painter, Golive, bibliographie, lexique, acrobat, typographie, interfacage, transcodage, photographie, graphisme, calibreteur, infographie, compressions, compression, mise en page">
```

```
<meta name="description" content="Ressources et services pour les arts graphiques, la PAO et le multimedia. Lexique, bibliographie, infos sur XPress, Illustrator et Photoshop. Formation, conseil, production d'imprimes, conception et hebergement de sites web.">
```

Le corps du document

L'élément BODY

Les balises **<BODY></BODY>** encadrent le contenu du document. Tous les éléments situés entre ces balises seront affichés par le navigateur.

Il est actuellement déconseillé d'utiliser des attributs de la balise BODY pour formater la page. Il faut utiliser une feuille de style (interne ou externe) dans toute la mesure du possible.

Les attributs id et class.

L'attribut **id** assigne un nom unique à un élément.

Il peut étre utilisé comme un sélecteur de feuille de style (avec grande prudence), comme ancre, comme moyen d'appeler un élément particulier à partir d'un script, pour un traitement universel par les agents utilisateurs...

L'attribut **class** assigne un ou plusieurs noms de classes à un élément. Plusieurs éléments peuvent avoir le m&eacme; nom de classe.

Il est utilisé comme sélecteur dans une feuille de style et pour un traitement universel par les agents utilisateurs.

Eléments de bloc et éléments en-ligne.

Le corps de la page accueille des éléments de type bloc (block-level) et des éléments de niveau « en-ligne » (inline)

Les éléments de bloc peuvent contenir des éléments en-ligne et d'autres éléments de bloc. Les éléments en-ligne ne peuvent contenir que des données et d'autres éléments en-ligne. Les éléments de bloc créent des structures « plus grande » que les éléments en-ligne.

Les éléments de bloc sont formatés différemment des éléments en-ligne. En général les éléments de bloc commencent sur une nouvelle ligne.

Les éléments DIV et SPAN

Les balises `<DIV></DIV>` définissent un élément comme étant un bloc.

Les balises `` définissent un élément comme étant en-ligne

Ces deux types de balises sont utilisés en conjonction avec les feuilles de style pour formater les éléments.

Les éléments H1, H2, H3, H4, H5, H6

Ces balises permettent de définir 6 niveaux de titrage. H1 est le niveau le plus important.

Exemple de synthèse :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Une page</title>
<meta name="description" content="Une page en licence flux
numerique">
</head>
<body>
<h1>Voici un premier exemple</h1>
<h2>Voici un premier exemple</h2>
<h3>Voici un premier exemple</h3>
<h4>Voici un premier exemple</h4>
<h5>Voici un premier exemple</h5>
<h6>Voici un premier exemple</h6>
</body>
</html>
```

Le texte

Le texte structuré

Les éléments EM, STRONG, DFN, CODE, SAMP, KBD, VAR, CITE, ABBR et ACRONYM

`` Mise en exergue

`` Mise en exergue plus forte

`<CITE></CITE>` Citation

`<DFN></DFN>` Définition

`<CODE></CODE>` Code informatique

`<SAMP></SAMP>` Exemple de sortie d'un programme.

`<KBD></KBD>` Indique un texte que doit saisir l'utilisateur

`<VAR></VAR>` Variable ou paramètre

`<ABBR></ABBR>` Forme abrégée

<ACRONYM></ACRONYM> Acronyme

Les éléments BLOCKQUOTE et Q

BLOCKQUOTE indique une citation longue tandis que Q indique une citation brève.

Les navigateurs restituent généralement BLOCKQUOTE sous la forme d'un bloc de texte en retrait.

Les éléments SUB ET SUP

Les balises et permettent de définir des indices et des exposants.

Les lignes et les paragraphes

L'élément P

Les balises <P> et </P> définissent un paragraphe.

L'élément BR

L'élément BR coupe la ligne de texte courante.

Pour éviter une coupure de ligne on utilise l'entité (No Break SPace).

L'élément PRE

Les balises <PRE></PRE> indiquent que le texte situé entre les deux balises est préformaté.

Le navigateur peut laisser les blancs intacts, peut restituer le texte avec une police à chasse fixe, peut désactiver les retours à la ligne automatique.

Les listes

Les éléments UL OL LI

Les balises permettent de définir une liste non-ordonnée

Les balises permettent de définir une liste ordonnée

Les balises permettent de définir les items de listes

Exemple de synthèse :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Une autre page</title>
<meta name="description" content="Une page en licence flux
numerique">
</head>
<body>
<ol>
<li>Marie</li>
<li>Karine</li>
</ol>
```

```
<li>sa vie</li>
<li>son oeuvre</li>
</ol>
<li>Annabelle</li>
</ol>
</body>
</html>
```

Les tables

L'élément TABLE

Les balises `<TABLE></TABLE>` encadrent tout le contenu d'une table

L'élément CAPTION

Les balises `<CAPTION></CAPTION>` permettent de définir la légende unique du tableau.

Des balises permettent de regrouper des lignes et des colonnes (voir la norme)

L'élément TR

Les balises `<TR></TR>` permettent de créer des lignes

Les balises `<TD></TD>` définissent les cellules à l'intérieur des lignes.

Le nombre de colonnes ou de lignes occupées par une cellule est indiqué par les attributs `rowspan` et `colspan`.

L'attribut **border** de l'élément TABLE sert à définir l'épaisseur du cadre autour de la table.

L'attribut **align** (alignement horizontal) peut prendre les valeurs **left**, **center**, **right**, **justify** ou **char** (alignement sur un caractère)

L'attribut **valign** (alignement vertical) peut prendre les valeurs :

top les données sont placées en haut de la cellule,

middle les données sont centrées verticalement (valeur par défaut),

bottom les valeurs sont repoussées en bas de la cellule,

baseline les premières lignes de texte sont alignées pour toutes les cellules de la ligne qui ont le même attribut.

L'attribut **cellspacing** définit l'espacement entre les cellules.

L'attribut **cellpadding** définit l'espace entre le bord des cellules et leur contenu.

Exemple de synthèse :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Une autre page</title>
<meta name="description" content="Une page en licence flux
numerique">
</head>
<body>
```

```

<table border = '0'>
<tr bgcolor='#ffddcc'>
<td width = '200'>Nom</td>
<td width = '100' align = 'center'>Exercice 1</td>
<td width = '100' align = 'center'>Exercice 2</td>
<td width = '100' align = 'center'>Exercice 3</td>
</tr>
<tr bgcolor='#ffffcc' >
<td width = '200' >Marie</td>
<td width = '100' align = 'center'>15</td>
<td width = '100' align = 'center'>18</td>
<td width = '100' align = 'center'>17</td>
</tr>
<tr bgcolor='#ffffcc'>
<td width = '200'>Karine</td>
<td width = '100' align = 'center'>10</td>
<td width = '100' align = 'center'>8</td>
<td width = '100' align = 'center'>12</td></tr>
<tr bgcolor='#ffffcc'>
<td width = '200'>Ingrid</td>
<td width = '100' align = 'center'>12</td>
<td width = '100' align = 'center'>14</td>
<td width = '100' align = 'center'>17</td>
</tr>
</table>
</body>
</html>

```

Les liens

L'élément A

Les liens sont principalement créés par les balises `<A>` uniquement dans le corps du document et par la balise `<LINK>` uniquement dans l'en-tête.

L'attribut **href** de l'élément A définit l'ancre source d'un lien que l'utilisateur peut activer pour ramener une ressource web.

L'élément LINK

L'élément **LINK** définit une relation entre le document courant et une autre ressource. Il peut apparaître un nombre quelconque de fois.

Quand l'élément **LINK** relie une feuille de style externe à un document, l'attribut **type** spécifie le langage de feuille de style et l'attribut **media** spécifie les medias de restitution prévus.

Exemple de synthèse :

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Une autre page</title>
<meta name="description" content="Une page en licence flux
numerique">
</head>

```

```
<body>
Pour ruser cliquez <a href = http://www.ruses.com>ici</a>
</body>
</html>
```

L'élément BASE

La balise **<BASE>** permet de spécifier l'URI de base du document, qui permet de résoudre les URI relatifs.

Exemple :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.
w3.org/TR/html4/strict.dtd">
<HTML>
<HEAD>
<TITLE>Nos produits</TITLE>
<BASE href="http://www.poulets.com/produits/intro.html">
</HEAD>
<BODY>
<P>Avez-vous vu nos <A href=" ../cages/poules.gif">cages &agrave;
poules</A> ?
</BODY>
</HTML>
```

Objets, images et applets

L'élément IMG

La balise **** (balise fermante interdite) permet d'insérer une image.

L'attribut **alt** permet d'indiquer un texte de remplacement qui sera affiché si l'image ne peut l'être.

L'attribut **name** permet de donner un nom à l'image.

L'attribut **src** permet de localiser la ressource par son chemin.

L'attribut **width** permet d'indiquer la largeur d'une image (elle est appliquée à la place de la largeur réelle de l'image)

L'attribut **height** permet d'indiquer la hauteur de l'image. (Elle est appliquée à la place de la hauteur de l'image)

Une image peut également être insérée grâce à la balise **<object>**.

L'élément OBJECT

Les balises **<OBJECT></OBJECT>** permettent de réaliser une inclusion générique c'est à dire qu'elles permettent d'insérer des objets de tous types (images, applets, modules d'extension, gestionnaires de média, etc. Elles permettent également d'indiquer ce qui est nécessaire à l'objet pour qu'il soit affiché : le code source, les valeurs initiales et les données d'exécution. Elles peuvent donc se substituer aux balises ****, **<APPLET>** et **<IFRAME>**.

L'attribut **classid** sert à localiser l'implémentation d'un objet par un URI.

L'attribut **codebase** indique le chemin utilisé pour résoudre les URI relatifs spécifiés par les attributs **classid**, **data** et **archive**.

L'attribut **codetype** indique le type de contenu de données attendues lors du chargement de l'objet spécifié par **classid**.

L'attribut **data** peut spécifier la localisation des données de l'objet.

L'attribut **type** spécifie le contenu des données spécifiées par data.

L'attribut **archive** peut indiquer la liste des URI, séparés par des espaces, des archives contenant des ressources concernant l'objet.

L'élément PARAM

La balise <PARAM> indique les valeurs qui sont nécessaires à l'exécution d'un objet. On peut l'utiliser plusieurs fois de suite mais dans le contenu d'un élément OBJECT ou APPLET et au début du contenu.

L'attribut **name** définit le nom d'un paramètre d'exécution connu des objets insérés.

L'attribut **value** indique la valeur d'un paramètre d'exécution spécifié par l'attribut name.

L'attribut **valuetype** peut prendre les valeurs **data**, **ref** ou **object**.

data indique la valeur par défaut de l'attribut.

ref : la valeur spécifiée par l'attribut value est un URI qui désigne la ressource dans laquelle les valeurs d'exécution sont stockées.

object : la valeur spécifiée par value est un identifiant qui se réfère à la déclaration d'un objet OBJECT dans le même document. L'identifiant doit être la valeur de l'attribut id mis dans l'élément OBJECT déclaré.

L'attribut **type** indique le type de contenu de la ressource spécifiée par l'attribut value seulement dans le cas où l'attribut valuetype a la valeur ref.

Exemple :

```
<P><OBJECT classid="http://www.miamachina.it/orologianalogico.py">
<PARAM name="height" value="40" valuetype="data">
<PARAM name="width" value="40" valuetype="data">
L'agent utilisateur ne reconnaît pas les applications Python.
</OBJECT>
```

Les éléments MAP et AREA

La balise <MAP> permet de créer une image cliquable côté client qui est associée à un élément IMG ou plus rarement à un élément OBJECT ou INPUT.

L'attribut **usemap** de l'élément concerné permet d'associer l'image cliquable à l'élément.

Exemple de synthèse :

```
<td align="left" valign="top" width="160" height="128"
xpos="624">
</td>
<map name="panneau1b5f42f67"><area shape="rect"
coords="2,110,158,135" href="00000.htm"><area shape="rect"
coords="2,92,158,109" href="0005.htm"><area shape="rect"
coords="1,73,157,92" href="0004.htm"><area shape="rect"
coords="2,56,158,73" href="0003.htm"><area shape="rect"
coords="2,37,159,56" href="0002.htm"><area shape="rect"
coords="2,20,158,37" href="0001.htm"><area shape="rect"
coords="1,1,160,21" href="000.htm"></map></div>
```

La présentation visuelle des images, objets et applets.

Nous avons déjà vu les attributs **width** et **height**.

L'attribut **hspace** indique la taille d'espace à insérer à gauche et à droite des éléments IMG, APPLET et OBJECT.

L'attribut **vspace** indique la quantité d'espace à insérer au-dessus et en-dessous des éléments IMG, APPLET ou OBJECT.

L'attribut **border** indique l'épaisseur de la bordure autour d'un élément IMG ou OBJECT.

L'attribut **align** indique la position d'un élément IMG, OBJECT ou APPLET par rapport à son contexte.

bottom : le bas de l'objet devrait s'aligner verticalement sur la ligne de base courante.

middle : le centre de l'objet devrait s'aligner verticalement avec la ligne courante

top : le haut de l'objet devrait s'aligner verticalement sur la ligne de base courante.

Tous les attributs concernant la présentation visuelle des images, des applets et des objets sont déconseillés. Dans toute la mesure du possible, il faut utiliser les feuilles de style.

L'attribut **alt** permet de spécifier un texte de remplacement.

Exemple de synthèse :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Une autre page</title>
<meta name="description" content="Une page en licence flux
numerique">
</head>
<body>
<table border = '0' cellspacing="0" cellpadding="0">
<tr><td rowspan = '2'></td>
<td></td></tr>
<tr><td></td></tr>
</table>
</body>
</html>
```

Les feuilles de style

Les feuilles de style constituent un progrès majeur. Elles permettent de contrôler de manière précise la manière dont les pages HTML sont affichées.

Les documents HTML peuvent contenir directement les règles de feuilles de style ou bien les importer (feuille de style externe). Il est recommandé d'utiliser une seule feuille de style pour l'ensemble d'un site ou au moins pour un ensemble de pages. On aura un code plus efficace et on pourra modifier la présentation du site rapidement et sans retoucher au code des pages HTML.

On utilise généralement le langage de style CSS.

Le langage de feuille de style par défaut du document se définit en utilisant l'élément META.

Exemple :

```
<META http-equiv="Content-Style-Type" content="text/css">
```

Héritage et cascade

Le mécanisme de cascade est utilisé quand plusieurs règles de style s'appliquent toutes directement à un élément.

Si une propriété peut s'hériter (ce n'est pas le cas de toutes les propriétés), l'agent utilisateur examine l'élément englobant immédiat pour voir si une règle s'y applique. Ce processus continue jusqu'à ce qu'une règle applicable soit trouvée.

Exemple : si la famille de la police est spécifiée grâce à l'élément BODY, cette famille s'appliquera à tous les éléments contenus dans le document pour peu qu'aucune autre information de famille de police ne soit présente à un niveau inférieur.

On se reportera aux normes :

www.w3c.org/Style/CSS/

et dans la pratique on utilisera les éditeurs généraux comme DreamWeaver ou Adobe Golive ou encore des éditeurs spécialisés pour réaliser les feuilles de style.

On utilisera ces techniques de préférence aux balises de formatage physique mais on le fera avec précaution en raison des différences d'interprétation entre les différents navigateurs qui restent préoccupantes.

Il est nécessaire, même pour des travaux simples de réaliser des tests à l'aide des principaux navigateurs du marché.

Exemple de synthèse 1

Une feuille de style CSS :

```
body { font-size: 12px; font-family: Helvetica, Geneva, Arial, SunSans-Regular, sans-serif }
a:link { color: maroon; text-decoration: none }
a:hover { color: red; text-decoration: none }
a:visited { color: gray; text-decoration: none }
.serif { color: maroon; font-size: 10px; font-family: Georgia, "Times New Roman", Times, serif }
```

Le code HTML qui l'utilise :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Une autre page</title>
<meta name="description" content="Une page en licence flux
numerique">
<link href="styles.css" rel="stylesheet" type="text/css"
media="all">
</head>
<body>
Il est donc possible de faire des <a href="http://www.ruses.
com">liens</a>
<br>
<br>
<div class="serif">
Un paragraphe particulier.

</div>
```

```
<br>
Et du texte normal.
</body>
</html>
```

Exemple de synthèse 2

Une feuille de style qui utilise un positionnement absolu :

```
#id01 { position: absolute; top: 200px; left: 400px }
#id02 { position: absolute; top: 380px; left: 420px }
```

Le code HTML qui l'utilise (superposition de texte à une image) :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
    <title>Page sans titre</title>
    <link href="style2.css" rel="stylesheet" type="text/css" media="all">
  </head>
  <body bgcolor="#ffffff">
    
    <div id="id02">Légende</div>
  </body>
</html>
```

Alignement, styles de police et règles horizontales

On aborde ici des éléments attributs qui peuvent être utilisés pour la mise en forme visuelle des éléments. Ils sont souvent déconseillés. On doit en effet les remplacer par l'utilisation des feuilles de style.

Formatage

Couleur d'arrière-plan

L'attribut `bgcolor` (utilisé avec `BODY`, `TABLE`, `TR`, `TH` ou `TD`) fixe la couleur de l'arrière-plan.

Alignement

L'attribut **align** (valeurs possibles : **left**, **center**, **right** ou **justify**) indique l'alignement horizontal de son élément par rapport à son environnement.

Polices

Les éléments de style de police. **TT**, **I**, **B**, **BIG**, **SMALL**, **STRIKE**, **S** et **U**

TT : généralement police à chasse fixe.

I : généralement texte en italique.

B : Généralement texte en gras.

BIG : généralement augmentation du corps.

SMALL : généralement corps plus petit.

STRIKE ET S : généralement texte barré. Déconseillé

U : généralement texte souligné. Déconseillé.

Règles : élément HR

La balise <HR> insère une règle horizontale.

Les cadres

Introduction

Les cadres sont utilisés aujourd'hui beaucoup moins systématiquement qu'il y a quelques années. Ils présentent des inconvénients de navigation et pour le référencement.

Le principe consiste à diviser un document en plusieurs « vues ». On a alors un document qui détermine les cadres et un document par cadre.

La disposition des cadres

Techniquement, la section BODY est remplacée par une section FRAMESET. Cette section indique la disposition des vues dans la fenêtre principale. Cette section FRAMESET peut contenir un élément NOFRAMES afin de proposer un contenu de remplacement.

L'élément FRAMESET

Il admet pour attributs :

rows indique la disposition des cadres horizontaux. On utilise une liste de longueurs en pixels, pourcentages ou relatives, séparées par des virgules.

cols indique la disposition des cadres verticaux sous la forme d'une liste de même nature.

Exemples :

```
<FRAMESET rows="50%, 50%">
...le reste de la définition...
</FRAMESET>
```

(Division de l'écran en deux verticalement)

```
<FRAMESET cols="1*,250,3*">
...le reste de la définition...
</FRAMESET>
```

(Création de colonnes, la deuxième a une largeur fixe de 250 pixels, la première reçoit 25% de l'espace disponible et la troisième 75 % de l'espace restant)

```
<FRAMESET rows="30%,70%" cols="33%,34%,33%">
...le reste de la définition...
</FRAMESET>
```

(Création d'une grille de 2 lignes et 3 colonnes)

Les jeux d'encadrement peuvent s'imbriquer.

Exemple :

```
<FRAMESET cols="33%, 33%, 34%">
  ...le contenu du premier cadre...
  <FRAMESET rows="40%, 50%">
    ...le contenu du deuxième cadre, première rangée...
    ...le contenu du deuxième cadre, seconde rangée...
  </FRAMESET>
  ...le contenu du troisième cadre...
</FRAMESET>
```

(Le premier élément FRAMESET divise l'espace de la fenêtre en trois colonnes égales. Le deuxième élément FRAMESET divise la deuxième colonne en deux rangées de hauteur inégale.)

L'élément FRAME

Il permet de définir le contenu et l'apparence d'un seul cadre.

L'attribut **name** assigne un nom au cadre courant.

L'attribut **longdesc** spécifie une description longue du cadre (utile pour les agents utilisateurs non-visuels)

L'attribut **src** spécifie la localisation du contenu initial à placer dans le cadre.

L'attribut **noresize** (booléen) indique que le cadre ne doit pas être redimensionné.

L'attribut **scrolling** qui peut prendre les valeurs **auto**, **yes** ou **no** spécifie comment doit être géré le défilement pour le cadre en cause.

auto : le navigateur permet le défilement si c'est nécessaire

yes : le navigateur fournit toujours un mécanisme de défilement.

no : le navigateur ne fournit pas de mécanisme de défilement.

L'attribut **frameborder** permet de rendre visible (valeur 1) ou invisible (valeur 0) la bordure du cadre.

L'attribut **marginwidth** indique la quantité d'espace à laisser entre le contenu du cadre et ses marges gauche et droite.

L'attribut **marginheight** indique la quantité d'espace à laisser entre le contenu du cadre et ses marges haute et basse.

L'élément IFRAME

Cet élément permet d'insérer un cadre dans un bloc de texte. On insère ainsi un document HTML au sein d'un autre.

Noter que les cadres en-ligne ne peuvent pas être redimensionnés.

Les formulaires

Introduction

Un formulaire est une partie de document qui a une fonction particulière. Il est doté de contenu normal mais aussi de commandes (cases à cocher, boutons radio, menus déroulants,...)

L'utilisateur « remplit » le formulaire en modifiant les commandes (saisie de texte, sélection d'un article de menu, etc.) avant de soumettre le formulaire à un agent pour son traitement en cliquant sur un bouton.

Les données envoyées sont ensuite traitées, généralement par un serveur web ou un serveur de courrier.

Les commandes.

Le nom de chaque commande est donné par l'attribut **name**.

Chaque commande possède une valeur initiale et une valeur courante. La valeur initiale est donnée par l'attribut **value** sauf pour les éléments **TEXTAREA** et **OBJECT**.

La valeur courante est égale à la valeur initiale tant qu'elle n'a pas été modifiée par l'utilisateur ou un script.

La réinitialisation d'un formulaire permet de retrouver les valeurs initiales.

Les types de commandes

Les boutons

Il en existe 3 types :

- **Les boutons de soumission.** Ils déclenchent bien entendu la soumission du formulaire. Noter qu'il peut en exister plusieurs pour le même formulaire.
- **Les boutons de réinitialisation.** Ils permettent de réinitialiser toutes les commandes à leur valeur initiale.
- **Les boutons poussoirs.** Ils n'ont pas de comportement par défaut. On peut associer des scripts côté client aux attributs d'événement de l'élément.

Exemple : on peut associer un script en javascript à l'attribut onclick du bouton. Le script sera alors exécuté à chaque fois qu'on cliquera sur le bouton.

Les cases à cocher.

Ce sont des interrupteurs marche/arrêt (elles permettent de transmettre une information booléenne).

Les boutons radio

Ils sont analogues aux cases à cocher mais quand plusieurs boutons partagent le même nom de commande, ils s'excluent mutuellement.

Les menus

Ils proposent des options aux utilisateurs.

La saisie de texte

L'élément **INPUT** crée une commande de saisie sur une seule ligne et l'élément **TEXTAREA** crée une commande pour une saisie sur plusieurs lignes.

La sélection d'un fichier.

Créé avec l'élément **INPUT**, ce type de commande permet à l'utilisateur de localiser un fichier sur sa machine de manière à ce que son contenu puisse être soumis avec le formulaire.

Les commandes cachées

Elles permettent au développeur de transmettre au serveur des informations qui ne sont pas saisies par l'utilisateur et qui sont nécessaires au traitement du formulaire.

Les commandes d'objets.

Elles permettent d'insérer des objets génériques dans les formulaires.

L'élément FORM

Les formulaires doivent commencer par **<FORM>** et se terminer par **</FORM>**.

L'élément FORM spécifie la disposition du formulaire (par son contenu), le programme qui va manipuler le formulaire (par l'attribut **action**), la méthode de transmission (par l'attribut **method**) et l'encodage des caractères par l'attribut **accept-charset**).

L'élément INPUT

Les types de commandes créés par INPUT

Le type de commande créé par l'élément INPUT est fonction de la valeur de l'attribut **type** :

text crée une commande de saisie de texte sur une ligne,

password fait la même chose mais le texte saisi sera dissimulé,

checkbox crée une case à cocher,

radio crée un bouton radio,

submit crée un bouton de soumission,

image crée un bouton de soumission graphique. La valeur de l'attribut **src** indique l'URI de l'image qui va décorer le bouton. L'attribut **alt** permet de spécifier un texte de remplacement. Noter que les valeurs des coordonnées du clic sont transmises au serveur si un dispositif de pointage (souris) est utilisé.

reset crée un bouton de réinitialisation,

button crée un bouton poussoir,

hidden crée une commande cachée,

file crée une commande de sélection de fichier.

L'élément BUTTON

Les boutons créés par l'élément BUTTON fonctionnent comme ceux créés avec l'élément INPUT mais ils permettent des possibilités de restitution plus variées.

Les éléments SELECT, OPTGROUP et OPTION

L'élément SELECT crée un menu, chaque option est représentée par un élément OPTION.

L'élément SELECT admet les attributs suivants :

name donne un nom à la commande.

size indique le nombre de rangées qui doivent être visibles en même temps.

multiple indique qu'une sélection multiple est possible.

L'élément OPTGROUP permet aux auteurs un regroupement logique des options.

L'élément OPTION admet les attributs suivants :

selected indique que l'option est présélectionnée.

value indique la valeur initiale de la commande

label permet de spécifier un intitulé pour l'option

L'élément TEXTAREA

Il admet les attributs suivants :

name donne un nom à la commande,

rows indique le nombre de lignes de texte visibles,

cols indique la largeur visible en fonction de la chasse moyenne des caractères.

Les scripts

Introduction

Un script est un programme qui s'exécute sur la machine cliente au chargement du document ou à un autre moment comme l'activation d'un lien.

Les scripts sont utilisés pour la vérification des formulaires, pour modifier dynamiquement le contenu du document ou produire les éléments d'une interface graphique en étant attachés aux commandes d'un formulaire.

Les documents pour les navigateurs qui gèrent les scripts

L'élément SCRIPT

Il permet de définir le contenu du script. Si l'attribut **src** n'est pas présent, le contenu de l'élément est le script, sinon le navigateur va chercher le script à l'URI spécifié.

L'attribut **src** permet de localiser un script externe.

L'attribut **type** permet de spécifier le langage du contenu de l'élément. (Exemple : "text/javascript")

Les événements intrinsèques

onload : fin de chargement (utilisé avec BODY et FRAMESET)

onunload : disparition du document (utilisé avec BODY et FRAMESET)

onclick : clic sur un élément (utilisé avec la plupart des éléments)

ondblclick : double-clic sur un élément (utilisé avec la plupart des éléments)

onmousedown : appui sur le bouton de la souris au-dessus de l'élément (utilisé avec la plupart des éléments)

onmouseup : relâchement du bouton de la souris au-dessus d'un élément. (Utilisé avec la plupart des éléments)

onmouseover : déplacement de la souris sur un élément. (Utilisé avec la plupart des éléments)

onmousemove : déplacement de la souris au-dessus d'un élément. (Utilisé avec la plupart des éléments)

onmouseout : déplacement de la souris en dehors de l'élément (utilisé avec la plupart des éléments)

onfocus : réception du focus (de l'attention) par l'intermédiaire de la souris ou de la touche tabulation (utilisé avec A, AREA, LABEL, INPUT, SELECT, TEXTAREA et BUTTON)

onblur : perte du focus (utilisé avec les mêmes éléments que onfocus)

onkeypress : une touche est pressée (utilisé avec la plupart des éléments)

onkeydown : une touche est appuyée (utilisé avec la plupart des éléments)

onkeyup : une touche est relâchée (utilisé avec la plupart des éléments)

onsubmit : le formulaire est soumis (utilisé uniquement avec l'élément FORM)

onreset : le formulaire est réinitialisé (utilisé uniquement avec l'élément FORM)

onchange : une commande perd le focus et sa valeur a été modifiée depuis l'instant où elle a reçu le focus. (Utilisé avec INPUT, SELECT et TEXTAREA)

Exemple de synthèse :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>

<head>
<title>Un formulaire</title>
</head>

<body bgcolor="#ffffee">
<div align="center">
<form action="http://www.ruses.com/traitement.php" method="post"
name="F1">
<table bgcolor = '#ffffaa'>

<tr>
<td align = "right" width="250">Nom :</td>
<td width="250"><input type="text" size="40" name ="nom"></td>
</tr>

<tr>
<td align = "right" >Prénom :</td>
<td><input type="text" size="40" name ="prenom"></td>
</tr>

<tr>
<td align = "right" >code :</td>
<td width="200"><input type="password" size="40" name ="code">
</td>
</tr>

<tr>
<td align = "right" >Profession :</td>
<td><select name = type_client>
<option value ="i">Imprimeur</option>
<option selected value ="g">Graphiste</option>
<option value ="p">Photographe</option>
<option value ="l">Illustrateur</option>
</select>
</td>
</tr>

<tr>
<td align = "right" >Equipé ADSL : </td>
<td width="200"><input type="checkbox" name="adsl"></td>
</tr>
```

```

<tr>
<td align = "right" >Sexe : </td>
<td width="200">Masculin<input type="radio" name="sexe" value="f">
F&eacute;minin<input type="radio" name="sexe" value="m"></td>
</tr>

<tr>
<td align = "right" >Intérêt :</td>
<td><select name = type_client size="4" multiple>
<option value = "pr">Presse offset</option>
<option selected value = "t">Imprimeuse</option>
<option value = "p">Photocopieur</option>
<option value = "e" selected>Encres</option>
<option value = "v">vernis</option>
<option value = "s">s&egrave;cheurs</option>
</select></td>
</tr>

<tr>
<td align = "right" >Commentaires :</td>
<td><textarea name="commentaires" rows="10" cols="40">Saisissez
ici vos commentaires !</textarea></td>
</tr>

<tr>
<td colspan = "2" align="right"><input type="reset"><input
type="submit"></td>
</tr>

</table>
</form>
</div>
</body>
</html>

```

Les références des entités de caractères dans HTML 4

HTML 4 reconnaît plusieurs jeux de références d'entités de caractère :

- les caractères ISO 8859-1 qui possèdent des noms pratiques issus des annexes de SGML.
- les symboles, symboles mathématiques et lettres grecques représentés grâce à la police Symbol d'Adobe.
- les caractères significatifs pour le balisage et pour l'internationalisation (par exemple pour un texte bi-directionnel)

Pour des raisons de place, on ne donne ci-dessous que la liste des caractères ISO 8859-1 qui sont fréquemment utilisés.

```

<!ENTITY nbsp CDATA «&#160;» -- espace insécable, U+00A0 ISOnum -->
<!ENTITY iexcl CDATA «&#161;» -- point d'exclamation renversé, U+00A1 ISOnum -->
<!ENTITY cent CDATA «&#162;» -- symbole centime, U+00A2 ISOnum -->
<!ENTITY pound CDATA «&#163;» -- symbole livre, U+00A3 ISOnum -->
<!ENTITY curren CDATA «&#164;» -- symbole monétaire, U+00A4 ISOnum -->
<!ENTITY yen CDATA «&#165;» -- symbole yen, U+00A5 ISOnum -->
<!ENTITY brvbar CDATA «&#166;» -- barre verticale discontinue, U+00A6 ISOnum -->

```

<!ENTITY sect CDATA «§» -- paragraphe, U+00A7 ISOnum -->
 <!ENTITY uml CDATA «¨» -- tréma, U+00A8 ISODia -->
 <!ENTITY copy CDATA «©» -- symbole copyright, U+00A9 ISOnum -->
 <!ENTITY ordf CDATA «ª» -- indicateur ordinal féminin, U+00AA ISOnum -->
 <!ENTITY laquo CDATA ««» -- guillemet gauche, U+00AB ISOnum -->
 <!ENTITY not CDATA «¬» -- signe négation, U+00AC ISOnum -->
 <!ENTITY shy CDATA «­» -- trait d'union virtuel, U+00AD ISOnum -->
 <!ENTITY reg CDATA «®» -- symbole marque déposée, U+00AE ISOnum -->
 <!ENTITY macr CDATA «¯» -- macron, U+00AF ISODia -->
 <!ENTITY deg CDATA «°» -- symbole degré, U+00B0 ISOnum -->
 <!ENTITY plusmn CDATA «±» -- signe plus-ou-moins, U+00B1 ISOnum -->
 <!ENTITY sup2 CDATA «²» -- exposant deux, U+00B2 ISOnum -->
 <!ENTITY sup3 CDATA «³» -- exposant trois, U+00B3 ISOnum -->
 <!ENTITY acute CDATA «´» -- accent aigu, U+00B4 ISODia -->
 <!ENTITY micro CDATA «µ» -- symbole micro, U+00B5 ISOnum -->
 <!ENTITY para CDATA «¶» -- pied de mouche, U+00B6 ISOnum -->
 <!ENTITY middot CDATA «·» -- point médian, U+00B7 ISOnum -->
 <!ENTITY cedil CDATA «¸» -- cédille, U+00B8 ISODia -->
 <!ENTITY sup1 CDATA «¹» -- exposant un, U+00B9 ISOnum -->
 <!ENTITY ordm CDATA «º» -- indicateur ordinal masculin, U+00BA ISOnum -->
 <!ENTITY raquo CDATA «»» -- guillemet droite, U+00BB ISOnum -->
 <!ENTITY frac14 CDATA «¼» -- fraction un quart, U+00BC ISOnum -->
 <!ENTITY frac12 CDATA «½» -- fraction un demi, U+00BD ISOnum -->
 <!ENTITY frac34 CDATA «¾» -- fraction trois quarts, U+00BE ISOnum -->
 <!ENTITY iquest CDATA «¿» -- point d'interrogation renversé, U+00BF ISOnum -->
 <!ENTITY Agrave CDATA «À» -- lettre majuscule latine A accent grave, U+00C0 ISolat1 -->
 <!ENTITY Aacute CDATA «Á» -- lettre majuscule latine A accent aigu, U+00C1 ISolat1 -->
 <!ENTITY Acirc CDATA «Â» -- lettre majuscule latine A accent circonflexe, U+00C2 ISolat1 -->
 <!ENTITY Atilde CDATA «Ã» -- lettre majuscule latine A tilde, U+00C3 ISolat1 -->
 <!ENTITY Auml CDATA «Ä» -- lettre majuscule latine A tréma, U+00C4 ISolat1 -->
 <!ENTITY Aring CDATA «Å» -- lettre majuscule latine A rond en chef, U+00C5 ISolat1 -->
 <!ENTITY Aelig CDATA «Æ» -- lettre majuscule latine AE, U+00C6 ISolat1 -->
 <!ENTITY Ccedil CDATA «Ç» -- lettre majuscule latine C cédille, U+00C7 ISolat1 -->
 <!ENTITY Egrave CDATA «È» -- lettre majuscule latine E accent grave, U+00C8 ISolat1 -->
 <!ENTITY Eacute CDATA «É» -- lettre majuscule latine E accent aigu, U+00C9 ISolat1 -->
 <!ENTITY Ecirc CDATA «Ê» -- lettre majuscule latine E accent circonflexe, U+00CA ISolat1 -->
 <!ENTITY Euml CDATA «Ë» -- lettre majuscule latine E tréma, U+00CB ISolat1 -->
 <!ENTITY Igrave CDATA «Ì» -- lettre majuscule latine I accent grave, U+00CC ISolat1 -->
 <!ENTITY Iacute CDATA «Í» -- lettre majuscule latine I accent aigu, U+00CD ISolat1 -->
 <!ENTITY Icirc CDATA «Î» -- lettre majuscule latine I accent circonflexe, U+00CE ISolat1 -->
 <!ENTITY Iuml CDATA «Ï» -- lettre majuscule latine I tréma, U+00CF ISolat1 -->
 <!ENTITY ETH CDATA «Ð» -- lettre majuscule latine ED, U+00D0 ISolat1 -->
 <!ENTITY Ntilde CDATA «Ñ» -- lettre majuscule latine N tilde, U+00D1 ISolat1 -->
 <!ENTITY Ograve CDATA «Ò» -- lettre majuscule latine O accent grave, U+00D2 ISolat1 -->
 <!ENTITY Oacute CDATA «Ó» -- lettre majuscule latine O accent aigu, U+00D3 ISolat1 -->
 <!ENTITY Ocirc CDATA «Ô» -- lettre majuscule latine O accent circonflexe, U+00D4 ISolat1 -->
 <!ENTITY Otilde CDATA «Õ» -- lettre majuscule latine O tilde, U+00D5 ISolat1 -->
 <!ENTITY Ouml CDATA «Ö» -- lettre majuscule latine O tréma, U+00D6 ISolat1 -->
 <!ENTITY times CDATA «×» -- signe multiplication, U+00D7 ISOnum -->
 <!ENTITY Oslash CDATA «Ø» -- lettre majuscule latine O barré obliquement, U+00D8 ISolat1 -->
 <!ENTITY Ugrave CDATA «Ù» -- lettre majuscule latine U accent grave, U+00D9 ISolat1 -->
 <!ENTITY Uacute CDATA «Ú» -- lettre majuscule latine U accent aigu, U+00DA ISolat1 -->
 <!ENTITY Ucirc CDATA «Û» -- lettre majuscule latine U accent circonflexe, U+00DB ISolat1 -->
 <!ENTITY Uuml CDATA «Ü» -- lettre majuscule latine U tréma, U+00DC ISolat1 -->
 <!ENTITY Yacute CDATA «Ý» -- lettre majuscule latine Y accent aigu, U+00DD ISolat1 -->
 <!ENTITY THORN CDATA «Þ» -- lettre majuscule latine THORN, U+00DE ISolat1 -->
 <!ENTITY szlig CDATA «ß» -- lettre minuscule latine s dur, U+00DF ISolat1 -->
 <!ENTITY agrave CDATA «à» -- lettre minuscule latine a accent grave, U+00E0 ISolat1 -->
 <!ENTITY aacute CDATA «á» -- lettre minuscule latine a accent aigu, U+00E1 ISolat1 -->
 <!ENTITY acirc CDATA «â» -- lettre minuscule latine a accent circonflexe, U+00E2 ISolat1 -->
 <!ENTITY atilde CDATA «ã» -- lettre minuscule latine a tilde, U+00E3 ISolat1 -->
 <!ENTITY auml CDATA «ä» -- lettre minuscule latine a tréma, U+00E4 ISolat1 -->
 <!ENTITY aring CDATA «å» -- lettre minuscule latine a rond en chef, U+00E5 ISolat1 -->

```

<!ENTITY aelig CDATA «&#230;» -- lettre minuscule latine ae, U+00E6 ISolat1 -->
<!ENTITY ccedil CDATA «&#231;» -- lettre minuscule latine c cédille, U+00E7 ISolat1 -->
<!ENTITY egrave CDATA «&#232;» -- lettre minuscule latine e accent grave, U+00E8 ISolat1 -->
<!ENTITY eacute CDATA «&#233;» -- lettre minuscule latine e accent aigu, U+00E9 ISolat1 -->
<!ENTITY ecirc CDATA «&#234;» -- lettre minuscule latine e accent circonflexe, U+00EA ISolat1 -->
<!ENTITY euml CDATA «&#235;» -- lettre minuscule latine e tréma, U+00EB ISolat1 -->
<!ENTITY igrave CDATA «&#236;» -- lettre minuscule latine i accent grave, U+00EC ISolat1 -->
<!ENTITY iacute CDATA «&#237;» -- lettre minuscule latine i accent aigu, U+00ED ISolat1 -->
<!ENTITY icirc CDATA «&#238;» -- lettre minuscule latine i accent circonflexe, U+00EE ISolat1 -->
<!ENTITY iuml CDATA «&#239;» -- lettre minuscule latine i tréma, U+00EF ISolat1 -->
<!ENTITY eth CDATA «&#240;» -- lettre minuscule latine ed, U+00F0 ISolat1 -->
<!ENTITY ntilde CDATA «&#241;» -- lettre minuscule latine n tilde, U+00F1 ISolat1 -->
<!ENTITY ograve CDATA «&#242;» -- lettre minuscule latine o accent grave, U+00F2 ISolat1 -->
<!ENTITY oacute CDATA «&#243;» -- lettre minuscule latine o accent aigu, U+00F3 ISolat1 -->
<!ENTITY ocirc CDATA «&#244;» -- lettre minuscule latine o accent circonflexe, U+00F4 ISolat1 -->
<!ENTITY otilde CDATA «&#245;» -- lettre minuscule latine o tilde, U+00F5 ISolat1 -->
<!ENTITY ouml CDATA «&#246;» -- lettre minuscule latine o tréma, U+00F6 ISolat1 -->
<!ENTITY divide CDATA «&#247;» -- signe division, U+00F7 ISOnum -->
<!ENTITY oslash CDATA «&#248;» -- lettre minuscule latine o barré obliquement, U+00F8 ISolat1 -->
<!ENTITY ugrave CDATA «&#249;» -- lettre minuscule latine u accent grave, U+00F9 ISolat1 -->
<!ENTITY uacute CDATA «&#250;» -- lettre minuscule latine u accent aigu, U+00FA ISolat1 -->
<!ENTITY ucirc CDATA «&#251;» -- lettre minuscule latine u accent circonflexe, U+00FB ISolat1 -->
<!ENTITY uuml CDATA «&#252;» -- lettre minuscule latine u tréma, U+00FC ISolat1 -->
<!ENTITY yacute CDATA «&#253;» -- lettre minuscule latine y accent aigu, U+00FD ISolat1 -->
<!ENTITY thorn CDATA «&#254;» -- lettre minuscule latine thorn, U+00FE ISolat1 -->
<!ENTITY yuml CDATA «&#255;» -- lettre minuscule latine y tréma, U+00FF ISolat1 -->

```

Comment utiliser cette liste ?

Sa présentation un peu curieuse vient du fait qu'il s'agit d'un extrait d'un document SGML.

On recherche le caractère à afficher dans la colonne des commentaires, (celle qui suit --), on lit son nom au début de la ligne après <!ENTITY, on ajoute « & » devant le nom et « ; » après et ça marche.

Exemples :

1.

On veut afficher une **espace insécable**. On cherche ce nom dans la colonne des commentaires, on le trouve sur la première ligne.

On note le nom de l'entité « nbsp », on ajoute les caractères qui transforment ce nom en entité et on obtient l'entité recherchée :

2.

On veut afficher un pied de mouche. On trouve le nom de cette entité « para ».

L'entité recherchée est : ¶



SYLVAIN
RENARD

*Formation et réalisations multimédias
Développement de sites web dynamiques*

**58, Bd Henri Dunant
Bâtiment A
91100 Corbeil-Essonnes**

**Tél. : 01 64 96 02 31
Mobile : 06 60 45 19 73**

**www.ruses.com
srenard@ruses.com**