

HTML

HTML

- Basé sur SGML (Standard Generalized Markup Language)
- Créé par Tim Berners-Lee (1993: premier draft IETF)
- Problème majeur: divergence entre implémentations
 - ajout de tags non standards
 - différence de comportement entre navigateurs
- Standardisation reprise en 1995 par W3C
- Dernière version HTML 4.01 (2001)
 - simplification par rapport à HTML 3.2
 - Existe en 3 versions: strict, transitionnal et frameset

Exemple :

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

Il y a 3 types de documents HTML normalisés :

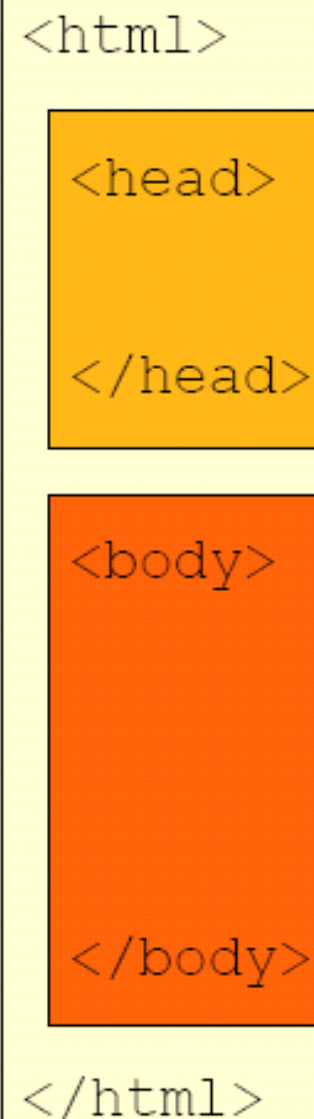
- HTML Strict : c'est une spécification de laquelle ont été retirées des balises liées à l'aspect visuel d'une page et les balises liées aux cadres (*frames*). Il s'agit de la définition de type de document (DTD : *Document Type Definition*) la plus restrictive des trois.
- HTML Transitional : il s'agit d'une spécification conservant des balises permettant essentiellement des effets graphiques (balises ayant souvent été introduites par des navigateurs sans passer par un processus de normalisation). Cette norme doit permettre aux webmestres de faire progressivement une transition de leurs « vieilles » pages HTML à la norme HTML Strict. Cette spécification ne contient pas les balises en rapport avec la gestion des cadres.
- HTML Frameset : Il s'agit de la norme Transitional à laquelle les balises concernant les cadres ont été ajoutées.

Les balises ont
une signification
Sémantique

Les balises ont
une signification de
Présentation

Éléments Documents

- `<html>...</html>`
 - délimite le document HTML
- `<head>...</head>`
 - contient les méta-données associées au document
- `<body>... </body>`
 - délimite le corps du document



The diagram illustrates the structure of an HTML document. It is represented as a large yellow rectangle. At the top left of this rectangle is the opening tag `<html>` and at the bottom right is the closing tag `</html>`. Inside the yellow rectangle, there are two nested orange rectangles. The top orange rectangle contains the opening tag `<head>` at its top left and the closing tag `</head>` at its bottom right. The bottom orange rectangle contains the opening tag `<body>` at its top left and the closing tag `</body>` at its bottom right. This visualizes the hierarchical nesting of the HTML document elements.

```
<html>
```

```
<head>
```

```
</head>
```

```
<body>
```

```
</body>
```

```
</html>
```

Remarques :

- Dans une partie textuelle d'un document HTML, plusieurs blancs (c'est-à-dire des espaces, des tabulations, des retours à la ligne) seront reproduits par le navigateur comme un seul espace (Les blancs suivant une balise n'étant même pas pris en compte). Les tabulations et les retours à la ligne servent donc essentiellement à rendre le source HTML plus lisible.

Un document HTML étant de type texte (au sens MIME), on a vu que selon la norme HTTP, le codage des caractères devrait être par défaut ISO-8859-1 (Latin1) pour le navigateur. On devrait donc pouvoir insérer des lettres accentuées comme « é » ou « à » directement dans le source HTML. Mais la norme pour HTML 4.01 est **Unicode** et non pas Latin1. De plus, le navigateur peut proposer à l'utilisateur de fixer le jeu de caractères utilisé pour l'affichage (*Affichage/Encodage des caractères* dans Firefox, *Affichage/Codage* dans Internet Explorer). Afin d'être certain d'obtenir les caractères que l'on désire, HTML définit des « entités de caractères ». Celles-ci commencent par le symbole « & » et se terminent par un « ; ». Le principe de construction de ces entités est résumé dans le tableau ci-dessous :

	Lettre de base	Modificateur		
&	a	A	acute	accent aigu
	e	E	grave	accent grave
	i	I	circ	accent circonflexe
	o	O	uml	tréma
	u	U	cedil	cédille
	c	C		
				;

Exemple : « carré » s'écrit « carré ».

Caractères utilisés

■ = entité HTML

• caractère	code ISO8859-1	entité
• guillemet	"	" "
• et commercial	&	& &
• signe plus petit que <	<	<
• plus grand que	>	> >
• espace non sécable	 	 _
• livre sterling	£	£ £
• c cédille	ç	ç ç
• e accent grave	è	è è
• e accent aigu	é	é é

■ Exemple

- | | | |
|------------------|----------------------|--------------------|
| • a < b if c > d | a < b if c>d | a < b if c>d |
|------------------|----------------------|--------------------|

- Il est possible d'insérer des commentaires dans une page HTML. Les commentaires en HTML prennent la forme suivante :

`<!-- note -->`

ou, sur plusieurs lignes :

`<!-- remarquez
bien cela -->`

- De façon générale, les navigateurs sont **très** tolérants avec les erreurs contenues dans les pages HTML. Ils préfèrent afficher ce qu'ils peuvent « comprendre » du document plutôt que de produire des messages d'erreurs.

Éléments Blocs

- `<p>...</p>` : paragraphe
- `<blockquote>...</blockquote>` : citation importante
- `<address>...</address>` : information sur l'auteur
- `<div>...</div>` : générique bloc (voir CSS)
- `<pre>...</pre>` : texte préformaté

Éléments Structure (1/2)

- Base

- `<abbr>...</abbr>` : **abréviation**
- `<acronym>...</acronym>` : **acronyme**
- `<cite>...</cite>` : **citation**
- `<dfn>...</dfn>` : **définition**
- `<em>...</em>` : **mise en valeur**
- `<q>...</q>` : **courte citation**
- `<span>...</span>` : **pour utilisation avec CSS**
- `<strong>...</strong>` : **mise en valeur importante**
- `<br/>` : **retour à la ligne**

Les balises ont
une signification
Sémantique

Elements Structure (2/2)

- Code

- `<code>...</code>` : code
- `<samp>...</samp>` : exemple de sortie
- `<kbd>...</kbd>` : texte entré par l'utilisateur
- `<var>...</var>` : variable dans un programme

- Entêtes

- `<h1>...</h1>` : du plus important...
- `<h2>...</h2>`
- `<h3>...</h3>`
- ...
- `<h6>...</h6>` : ... au moins important

Les balises ont
une signification
Sémantique

Exemple :

<h1>Travaux de recherches</h1>

<h2>I. Introduction.</h2>

<p>

Nous pensions faire cela.

Mais nous avons fait ceci.

</p>

<h3>II. Conclusion.</h3>

<p>

Comme disait Truc :

<cite>On fera mieux la prochaine fois.</cite>

</p>

Résultat obtenu :

Travaux de recherches

I. Introduction.

Nous pensions faire *cela*.

Mais nous avons fait *ceci*.

II. Conclusion.

Comme disait Truc : *On fera mieux la prochaine fois.*

Les balises permettant de positionner des attributs physiques (choix d'une fonte, couleur, etc.) pour une partie d'un document sont présentées au chapitre 11. Il est préférable de ne pas les mélanger avec les balises que nous venons de voir qui sont en rapport avec la structure logique du document. Pour les attributs physiques, il vaut mieux recourir aux feuilles de style (CSS) pour définir ceux-ci.

Éléments Présentation

- Usage déconseillé: utiliser plutôt les balises structures avec un CSS

- `<b>...</b>` : texte en gras
- `<big>...</big>` : texte agrandi
- `<hr/>` : ligne horizontale
- `<i>...</i>` : texte en italique
- `<small>...</small>` : texte réduit
- `<sub>...</sub>` : texte en retrait (M_{lle})
- `<sup>...</sup>` : texte en puissance (x²)
- `<tt>...</tt>` : texte monospace

Les balises ont
une signification de
Présentation

Listes (1/2)

- Liste sans ordre

```
<ul>
  <li>item 1</li>
  <li>item 2</li>
  <li>item 3</li>
</ul>
```

- item 1
- item 2
- item 3

- Liste ordonnée

```
<ol>
  <li>item 1</li>
  <li>item 2</li>
  <li>item 3</li>
</ol>
```

- 1.item 1
- 2.item 2
- 3.item 3

Listes (2/2)

- Liste de définition

```
<dl>
  <dt>item 1</dt>
  <dd>Definition 1</dd>
  <dt>item 2</dt>
  <dd>Definition 2</dd>
  <dd>More...</dd>
  <dt>item 3</dt>
  <dd>Definition 3</dd>
</dl>
```

item 1

Definition 1

item 2

Definition 2

More...

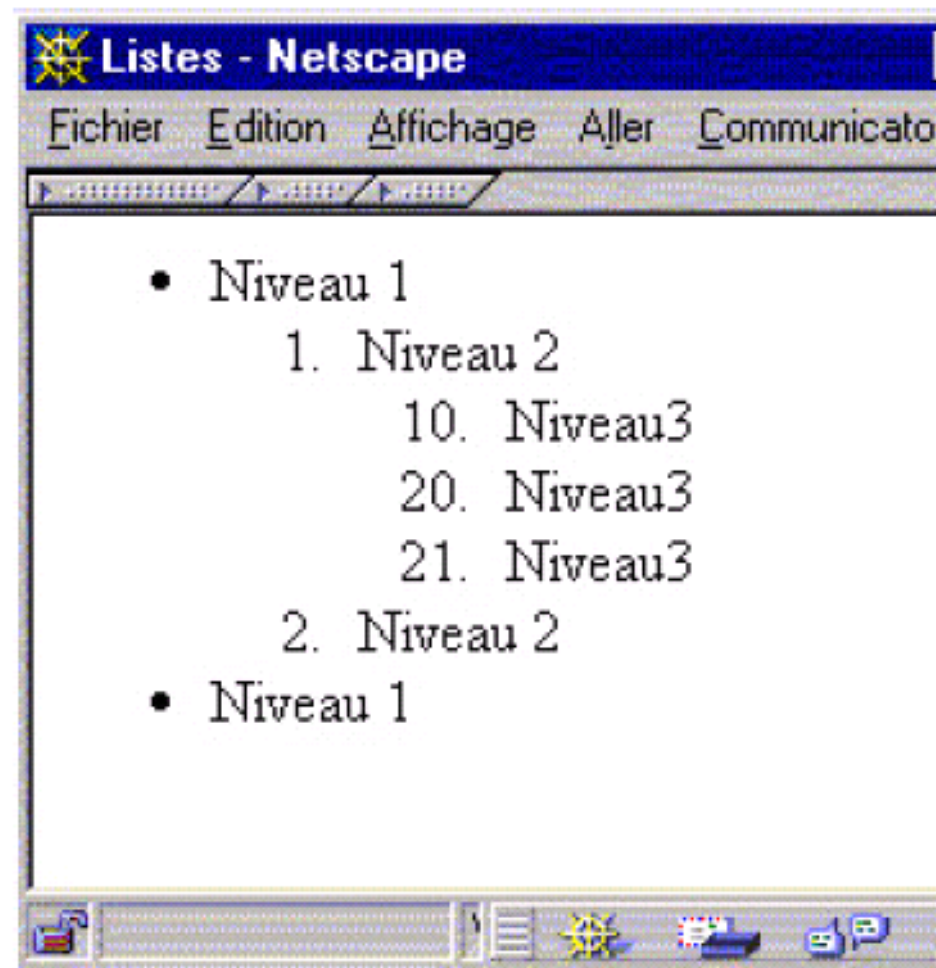
item 3

Definition 3

Les Listes

- Liste sans ordre , Liste Ordonnée

```
<UL>
  <LI>Niveau 1
  <OL>
    <LI>Niveau 2
    <OL>
      <LI VALUE="10">Niveau3
      <LI VALUE="20">Niveau3
      <LI>Niveau3
    </OL>
    <LI>Niveau 2
  </OL>
  <LI>Niveau 1
</UL>
```



Images (1/2)

- **Syntaxe:** `<img />`
- **Attributs:**
 - `src` : adresse de l'image
 - `alt` : texte alternatif
 - `width` **et** `height` : dimension de l'image

```

```

- Formats possibles: GIF, JPEG, PNG...

Inclusion d 'Images

■ Élément IMG

```
<IMG SRC="/icon/logo.gif" ALT="My Company Logo">
```

- image incluse dans le document
- Attribut optionnel

positionnement par rapport au texte

» ALIGN = TOP, MIDDLE, BOTTOM.

Forcer la taille de l 'image

» width=180 height=60

» *évite les reformatages intempestifs
au chargement du document*

- Remarque

- Type d 'image généralement supporté
GIF, JPEG, GIF Animé, XBM
- Plug-In pour les autres
signifie le chargement du PlugIn et son installation (Fat-Browser)

Hyperlien

Un lien hypertexte, dont l'affichage peut être un texte ou une image, est une référence à une autre ressource Web. En cliquant sur le lien, l'internaute est conduit à cette autre ressource. Les liens hypertextes constituent donc la base du concept de navigation sur le Web.

Exemple :

```
<a href="http://www.msi.unilim.fr/">
```

Vers le site du LMSI

```
</a>
```

L'URI indiquée avec href peut aussi être relative à la page contenant le lien. Par exemple, depuis la page stockée à l'URL : `http://www.msi.unilim.fr/index.html`, le lien :

```
< a href="themes.html">
```

référence la ressource associée à l'URL : `http://www.msi.unilim.fr/themes.html`

Hyperlien

La ressource désignée par href n'est pas nécessairement une ressource HTTP (puisque'il s'agit d'une URI).

Exemples :

```
<a href="ftp://ftp.lip6.fr/pub/python/2.4/Python-2.4.tgz">  
    Téléchargez la dernière version  
    de Python.  
</a>
```

```
<a href="mailto:hegel@berlin-uni.de">  
    Écrivez moi pour obtenir des renseignements  
    sur le Système du Savoir Absolu.  
</a>
```

Liens locaux

Exemple :

```
<a name="intro">  
    <h1>Introduction</h1>  
</a>
```

Exemple :

```
<a href="#intro">Lien vers l'introduction</a>
```

Exemple :

```
<a href="cours.htm#intro">Lien vers l'introduction du cours</a>
```

L'attribut name peut être remplacé par l'attribut id. Cet attribut id peut de plus être utilisé pour définir une ancre avec d'autres balises. Ainsi :

```
<h2 id="section2">Section 2</h2>
```

crée une ancre dans le document, ancre qui peut ensuite être référencée par :

```
<a href="#section2">Retour &agrave; la section 2 du document</a>
```

Tables (1/3)

```
<table border="1">  
  <tr>  
    <td>Row 1, cell 1</td>  
    <td>Row 1, cell 2</td>  
    <td>Row 1, cell 3</td>  
  </tr>  
  <tr>  
    <td>Row 2, cell 1</td>  
    <td>Row 2, cell 2</td>  
    <td>Row 2, cell 3</td>  
  </tr>  
</table>
```

Row 1, cell 1	Row 1, cell 2	Row 1, cell 2
Row 2, cell 1	Row 2, cell 2	Row 2, cell 3

Tables (2/3)

```
<table border="1">
  <tr>
    <td>Row 1, cell 1</td>
    <td colspan="2">Row 1, cell 2 and 3</td>
  </tr>
  <tr>
    <td rowspan="2">Row 2 and 3, cell 1</td>
    <td>Row 2, cell 2</td>
    <td>Row 2, cell 3</td>
  </tr>
  <tr>
    <td>Row 3, cell 2</td>
    <td>Row 3, cell 3</td>
  </tr>
</table>
```

Row 1, cell 1 and 2		Row 1, cell 3
Row 2 and 3, cell 1	Row 2, cell 2	Row 2, cell 3
	Row 3, cell 2	Row 3, cell 2

On peut procéder à une fusion de lignes ou de colonnes dans un tableau. On dispose pour cela des attributs `rowspan` et `colspan` pour les balises `<td>` et `<th>`.

```

<table border="1">
    <tr> <td rowspan="3">La</td>    <td>Chartreuse</td> </tr>
    <tr>                                <td>de</td>          </tr>
    <tr>                                <td>Parme</td>        </tr>
</table>

```

Affichage obtenu :

La	Chartreuse
	de
	Parme

```

<table border="1">
    <tr> <td>Mort</td> <td>&agrave;</td> </tr>
    <tr> <td colspan="2">Cr&eacute;dit</td> </tr>
</table>

```

Affichage obtenu :

Mort	à
Crédit	

Exemple :

```
<table border="1">  
  <caption>Années de naissance</caption>  
  <tr> <th>Nom</th>   <th>Année</th> </tr>  
  <tr> <td>Dupont</td> <td>1964</td>   </tr>  
  <tr> <td>Durand</td> <td>1960</td>   </tr>  
</table>
```

Résultat obtenu :

Années de naissance	
Nom	Année
Dupont	1964
Durand	1960

Exemple :

```
<table>
  <tr>
    <td>
      
    </td>
    <td>
      
    </td>
  </tr>
</table>
```

Résultat obtenu :



FORMULAIRE

Un formulaire est délimité dans le code HTML par les balises : `<form>` et `</form>`. La balise `<form>` dispose des attributs suivants :

- `name` : pour une gestion du formulaire avec JavaScript
- `action` : indique l'URL ou l'adresse e-mail où sont envoyées les données
- `method` : choix entre les méthodes GET ou POST
- `enctype` : type MIME indiquant l'encodage des données du formulaire

Exemples :

```
<form name="donneesClient"
      action="cgi/verifClient.pl"
      method="GET">
  <!-- Contenu du formulaire -->
</form>
```

```
<form name="saisieDonnees"
      action="mailto:machin@domaine.org"
      method="POST"
      enctype="text/plain">
  <!-- Contenu du formulaire -->
</form>
```

Les éléments d'un formulaire

■ FORM

- action, method (GET|POST), enctype, événements, ..

■ INPUT TYPE = (TEXT | PASSWORD | CHECKBOX | RADIO | SUBMIT | RESET | FILE | HIDDEN | IMAGE | BUTTON)

- Événements : *onfocus*, *onblur*, *onselect*, *onchange*, *accept*

■ TEXTAREA

■ BUTTON

■ SELECT, OPTGROUP, OPTION

■ FIELDSET *cadre de regroupement*

■ ISINDEX *obsolète*

Exemple :

```
<input type="radio"
      name="qualite" value="Mme" checked="checked">
Madame<br>
<input type="radio"
      name="qualite" value="Mlle">
Mademoiselle<br>
<input type="radio"
      name="qualite" value="M.">
Monsieur<br>
```

Affichage obtenu :



The screenshot shows a gray rectangular area containing three radio button options. The first option, 'Madame', is selected, indicated by a green dot in the center of the radio button. The second option, 'Mademoiselle', and the third option, 'Monsieur', are not selected, indicated by empty white radio buttons with blue outlines.

- ☒ Madame
- ☐ Mademoiselle
- ☐ Monsieur

Exemple :

Votre langage favori :

<select name="langage">

<option value="BASIC">BASIC</option>

<option value="Forth">Forth</option>

<option value="Lisp">Lisp</option>

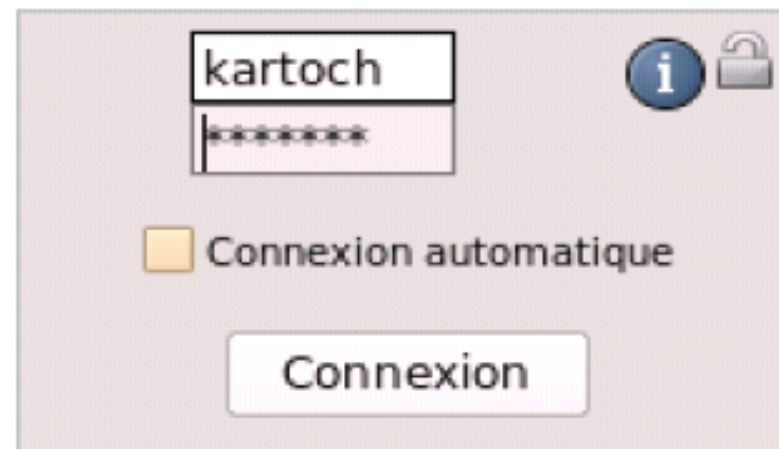
</select>

Affichage obtenu :

A screenshot of a web form on a grey background. The text 'Votre langage favori :' is displayed in a black serif font. Below it is a dropdown menu with a white background and a blue border. The word 'BASIC' is visible in the menu, and a blue arrow points downwards to the right of the text.

Formulaires: exemple

- Entrée d'informations dans un formulaire
 - les informations sont envoyés au serveur puis traités par un script



A screenshot of a web login form. It features a text input field containing 'kartoch' and a password input field with masked characters. To the right of the password field is a blue information icon and a lock icon. Below the password field is a checkbox labeled 'Connexion automatique'. At the bottom is a button labeled 'Connexion'.

```
<form method="post" action="/login.html" id="formulaire">
  <input type="hidden" name="url" value="http://linuxfr.org/pub/" />
  <input type="text" size="8" name="login" value="kartoch"/> <br />
  <input type="password" size="8" name="passwd" value="passwo"/>
  <br />
  <input type="checkbox" name="isauto" value="1" /> Connection Auto.
  <br />
  <input type="submit" name="Envoyer" value="Connexion"/>
</form>
```

Les Formulaires HTML

■ Entrée d'informations dans un formulaire

- *Evitez de rentrer votre numéro de carte de crédit !!!!!*
- Le contenu du formulaire est envoyé à un serveur puis traité par un script du serveur

```
<HTML><HEAD><TITLE>Formulaire</TITLE></HEAD>
```

```
<BODY><H1 align=center>Formulaire</H1><hr>
```

```
<FORM ENCTYPE="multipart/form-data" METHOD="POST" ACTION="/servlet/formprocess">
```

```
Ligne de Texte : <INPUT TYPE=text VALUE=" A Remplir" NAME=lig>
```

```
Zone de Texte : <TEXTAREA NAME=zon COLS=40 ROWS=5>
```

```
A Remplir
```

```
</TEXTAREA>
```

```
<INPUT TYPE=checkbox VALUE="Opt1" NAME=opt> Option 1
```

```
<INPUT TYPE=checkbox VALUE="Opt2" NAME=opt> Option 2<BR>
```

```
<INPUT Type=submit Value="Envoi">
```

```
<INPUT Type=reset Value="Reinit">
```

```
</FORM>
```

```
</BODY></HTML>
```

Les Formulaires HTML

■ Assure l'interactivité avec l'utilisateur

- Entrée d'information dans un formulaire
 - *Evitez de rentrer votre numéro de carte de crédit !!!!!*
- Le contenu du formulaire est envoyé à un serveur puis traité par un programme CGI du serveur

```
<FORM action="http://somesite.com/prog/adduser" method="post">
  <P>
    First name: <INPUT type="text" name="firstname"><BR>
    Last name: <INPUT type="text" name="lastname"><BR>
    email: <INPUT type="text" name="email"><BR>
    <INPUT type="radio" name="sex" value="Male"> Male<BR>
    <INPUT type="radio" name="sex" value="Female"> Female<BR>
    <INPUT type="submit" value="Send"> <INPUT type="reset">
  </P>
</FORM>
```

First name:

Last name:

email:

☐ Male

☐ Female

Le format HTML — les cadres

Les cadres, “frame”, permettent de subdiviser la fenêtre du navigateur en plusieurs cadres indépendants, chacun de ces cadres pouvant afficher un document HTML différent.

Pour utiliser les cadres, il existe deux balises :

- la balise FRAMESET qui débute la définition d'un ensemble de cadres
- la balise FRAME qui définit chaque cadre

La définition de la taille de chacun de ces cadres peut être faite de manière relative en utilisant la notation en «%» par rapport à la taille de la fenêtre du navigateur, ou bien en pixel.

Pourquoi ne pas les utilisés

L'utilisation des cadres **détruit l'expérience de navigation** de l'utilisateur :

- si l'utilisateur arrive sur une page qui doit faire partie d'un document comportant plusieurs cadres, il lui manque aux mieux des informations, au pire **il ne peut naviguer** sur le site.

Exemple : Le document est décomposé en deux cadres :

un cadre est employé pour donner une liste de liens (cadre menu), un autre cadre pour l'affichage du contenu de chaque lien (cadre information).

Problème : l'utilisateur arrive directement sur une page «contenu» normalement affichée dans le cadre information, et ne voit pas le cadre menu. La navigation est alors impossible.

- l'utilisateur ne peut mettre de **signet** sur un document utilisant des cadres de telles manières qu'il retrouve le contenu de chaque cadre tel qu'il était à la pose du signet. Il ne mémorise que l'état initial des cadres obtenu lors du premier chargement.

Les logiciels de conception de pages HTML permettent de définir une **page type** et de réutiliser cette page type pour toutes les pages d'un site. Il suffit alors de construire le menu de navigation du site dans cette partie commune, ce qui annule le besoin de cadres.

Le format HTML — les cadres

Les cadres sont définis en terme de ligne et de colonnes. Chacune des cases définies possède une dimension qui peut être fixée.

Exemple de définition

```
<FRAMESET ROWS="60%,*" COLS="65%,20%,*">
```

Le «*» permet d'utiliser la place restante.

Sur l'exemple, un ensemble de 6 cadres a été défini, sous forme de deux lignes de 3 colonnes.

La balise FRAMESET doit être employée avant le corps (balise BODY) du document.

Il doit y avoir pour chaque cadre défini, une balise FRAME qui permet d'associer au cadre :

- un document HTML;
- un nom qui permet de désigner dans un lien le cadre où doit s'afficher le document désigné par le lien.

Par défaut, c'est le cadre où se trouve le lien.

Exemple

```
<HTML><HEAD> ... </HEAD>
```

```
<FRAMESET ROWS="30%,*">
```

```
  <FRAME SRC="frame1.html" NAME="a_droite">
```

```
  <FRAME SRC="frame2.html" NAME="a_gauche">
```

```
</FRAMESET>
```

```
<NOFRAMES>
```

Ce texte est affiché parceque votre navigateur ne g&eagrave;re pas les lien

```
  <A HREF="...">version sans frame</A>
```

```
</NOFRAMES> </HTML>
```

L'attribut SCROLLING de la balise FRAME permet d'avoir des barres de défilement ou non pour le cadre.

Utilisation de la notion de cible pour les liens

```
<A HREF="..." TARGET="a_gauche">
```

Exemple :

```
<frameset cols="20%,80%">  
    <frame name="menu" src="menu.html">  
    <frame name="article" src="introduction.html">  
</frameset>
```

Dans l'exemple ci-dessus, la page menu.html contiendra probablement des liens vers des articles contenus sur le serveur. Si les liens ont la forme :

```
<a href="article1.html">Article 1</a>
```

la page de l'article s'affichera dans la frame du menu (ce qui n'est certainement pas l'effet souhaité). Pour que les pages HTML correspondant aux articles apparaissent dans l'autre frame, on va utiliser l'attribut target et le nom de la frame dans laquelle on désire que l'affichage ait lieu :

```
<a href="article1.html" target="article">Article 1</a>
```

.

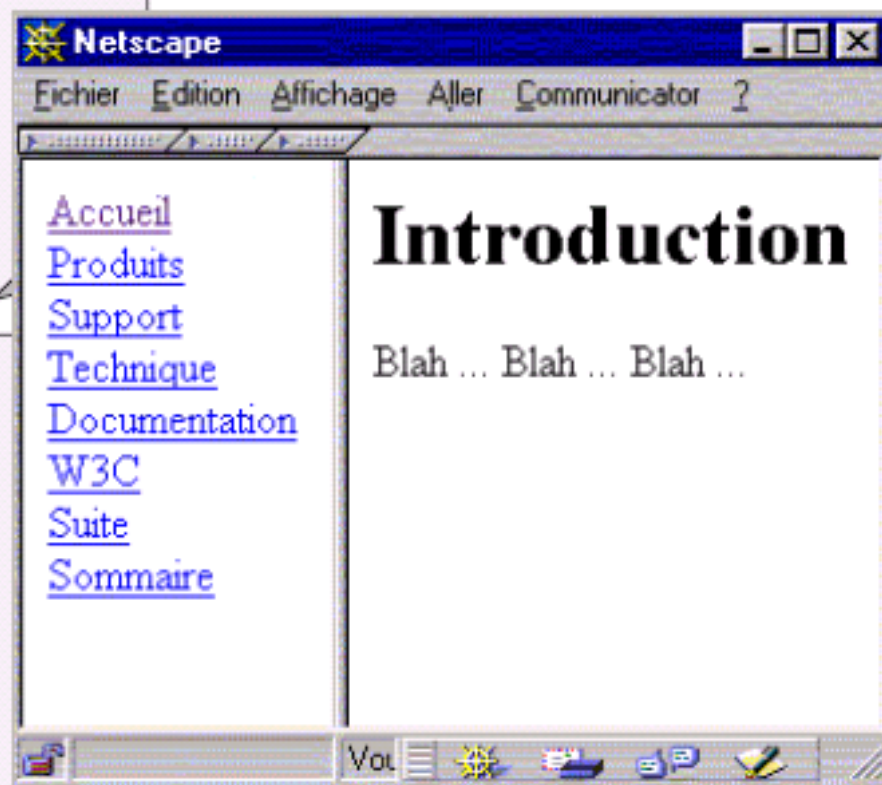
Les cibles dans les Frames HTML

indexframe.htm

```
<FRAMESET COLS="100, *">  
  <FRAME SRC="som1.htm" NAME="fr1">  
  <FRAME SRC="intro.htm" NAME="fr2">  
</FRAMESET>
```

som1.htm

```
<HTML><HEAD><BASE TARGET="fr2"></HEAD>  
<BODY>  
  <A HREF="intro.htm">  
    Accueil</A><BR>  
  <A HREF="produit.htm" TARGET="fr2">  
    Produits</A><BR>  
  <A HREF="sommaire.htm" TARGET="_top">  
    Support Technique</A><BR>  
  <A HREF="http://www.w3c.org/TR" TARGET="_self">  
    Documentation W3C</A><BR>  
  <A HREF="som2.htm" TARGET="fr1">  
    Suite Sommaire</A><BR>  
</BODY></HTML>
```



On peut aussi avoir des cadres locaux à une page HTML avec l'élément `iframe`.

Exemple :

```
<table border="2">
  <tr>
    <td>

      <iframe src="http://www.msi.unilim.fr/" width="250" height="250">
      </iframe>
    </td>
    <td>
      <iframe src="http://www.google.fr/" width="250" height="250">
      </iframe>
    </td>
  </tr>

  <tr>
    <td>
      <iframe src="http://www.alternatives87.eu.org/"
        width="250" height="250">
      </iframe>
    </td>
    <td>
      <iframe src="http://www.python.org/" width="250" height="250">
      </iframe>
    </td>
  </tr>
</table>
```

Résultat obtenu :



- Frames:

- « *In general, frames do nothing but **add complexity** and **subtract usability**.* »
 - ~~frame~~, ~~frameset~~, ~~iframe~~, ~~noframe~~

La partie <head> (2/2)

- <meta/>

- spécifier des méta-informations sous la forme clé/valeur:

- <meta name="author" content="F **damien** ">

- <meta name="keywords" content="Flower, Kitty">

- <meta name="description" content="My page
about Flower and Kitty (don't laugh) ">

- spécifier des entêtes HTTP:

- <meta http-equiv="expires" content="Tue, 20
Aug 1996 14:25:27 GMT">

- <meta http-equiv="Content-Type"
content="text/html; charset=ISO-8859-5">

Une balise <meta> a la forme suivante :

```
<meta name="identifiantInformation" content="information">
```

L'information sur le document HTML est identifiée par l'attribut name ; le contenu associé à cette information est fourni par l'attribut content.

Exemple :

```
<meta name="author" content="Karl Popper">
```

La norme HTML 4.01 ne définit aucun standard pour les noms et les contenus de la balise <meta>. Plusieurs balises de méta-données peuvent être présentes dans le document.

On peut aussi spécifier la langue utilisée par l'information de l'attribut content.

Exemple :

```
<meta name="keywords" lang="fr" content="Epistemologie, Falsifiabilité">
```

```
<meta name="keywords" lang="en" content="Epistemology, Falsifiability">
```

La balise <meta> a aussi un intérêt plus « pratique ». Elle permet d'insérer un en-tête HTTP dans la réponse du serveur. Pour cela, on substitue l'attribut http-equiv à l'attribut name.

Exemple :

```
<meta http-equiv="Expires" content="Tue, 20 Aug 1996 14:25:27 GMT">
```

a pour conséquence l'ajout de l'en-tête suivant à la réponse du serveur Web :

```
Expires: Tue, 20 Aug 1996 14:25:27 GMT
```

Remarque : l'en-tête HTTP ci-dessus peut être utile à des systèmes de cache (*proxy* par exemple) pour déterminer quand ils ont besoin de redemander un document.

Autre exemple :

```
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
```

12. XHTML.

XHTML (*eXtensible HyperText Markup Language*) est l'évolution du HTML préconisée par le W3C. De façon très simplifiée, on peut voir XHTML comme une reformulation en XML (*eXtensible Markup Language*) de HTML, XML étant une sorte de SGML allégé. Les buts principaux de XHTML sont :

- de séparer plus clairement que ne le fait HTML contenu et forme d'un document,
- d'utiliser les outils standards pour éditer et travailler sur des documents XML,
- de permettre un traitement automatisé plus simple des documents, notamment avec l'aide des outils XML (parsers, etc.),
- de rationaliser et faciliter les extensions du langage.

Il y a aujourd'hui deux normes concernant XHTML : XHTML 1.0 et XHTML 1.1.

XHTML 1.0 regroupe en fait trois DTDs qui correspondent à celles de HTML : Strict, Transitional et Frameset. XHTML 1.0 permet aux webmestres de migrer progressivement leurs sites en HTML vers XHTML.

XHTML 1.1 est conçu comme un ensemble de modules. XHTML Basic est un ensemble minimal de modules qui permettent un affichage même sur des appareils limités en capacités matérielles (comme les smartphones ou certains PDA). On peut ensuite adjoindre à XHTML Basic des modules étendant ses fonctionnalités ou en apportant de nouvelles. On trouve ainsi un module de script (pour le JavaScript), un module de style (pour les CSS), etc.

XHTML

- Reprise de HTML avec la grammaire XML (plus stricte)
mais considéré comme une branche différente: le développement de HTML n'est pas arrêté
 - XHTML 1:
 - reformulation XML de HTML 4
 - XHTML 1.1:
 - introduction de la modularité
 - disparition des 3 versions (strict, transitional et frameset)
 - XHTML 2 (en préparation):
 - Support Xform
 - XML event pour la gestion des événements
 - Nouveaux éléments, liste de navigation...



Différences HTML 4.01 / XHTML 1

- Grammaire plus stricte

- Les documents doivent respecter la hiérarchie

```
<p>here is an emphasized <em>paragraph</em>.</p>
```

- Les éléments non vides doivent posséder une balise de fin

```
<p>here is a paragraph.</p><p>here is another paragraph.</p>
```

- Les éléments vides (sans balise de fin) doivent se fermer avec `</>`

```
<dl compact="compact"/>
```

- La valeur d'un attribut doit être présente et entre guillemets
- L'attribut `name` est remplacé par `id`

XML (2)

- Exemple de « technologies » web basés sur XML :
 - SVG (*Scalable Vector Graphic*)
 - XUL (*XML User Interface Language*)
 - XBL (*XML Binding Language*)
 - RDF (*Ressource Description Framework*)
 - RSS (*Really Simple Syndication*)
 - Xpath (*XML Path Language*)
 - XSLT (*Extensible Stylesheet Language Transformations*)
 - XML Web Services: SOAP, XML-RPC
- Ce sont « juste » des schémas XML garantissant l'interopérabilité

