

# HTML: c'est quoi

## HyperText Markup Language

- HyperText: lien pour passer d'un document à un autre
- Markup Language: langage de balise

## HTML: un peu d'histoire

- Tim Berners-Lee : 1990
- HTML 2.0 : 1995 (RFC 1886)
- ...
- HTML 4 / XHTML : 1999 (RFC 1886)
- HTML 5 : aujourd'hui ? (draft)

## Introduction

On sépare le *contenu* et la *présentation*

### **Contenu et structure**

: décrit par le langage **HTML** ;

### **Présentation**

: décrite par des feuilles de style CSS : (couleur fonte cadre &hellips;)

Principe :

- En HTML : « ceci est un titre » ;
- En CSS : « les titres sont centrés et composés dans la police helvética » .

## Xhtml et XML

### XHTML

- "Successeur" de HTML compatible XML
- Séparation nette forme et contenu.
- HTML4.01 + règles pour se conformer au XML  
(voir [http://openweb.eu.org/articles/html\\_au\\_xhtml](http://openweb.eu.org/articles/html_au_xhtml))

### XML (Extensible Markup Language)

- standard du World Wide Web Consortium
- base pour créer des langages balise spécialisés
- Objectif : faciliter le partage de textes et d'informations structurés, par exemple au travers d'internet

## Outils pour écrire des pages web

### **Pas besoin d'être connecté !**

- éditeur de texte (notepad++, emacs ...)
- navigateur (firefox, chrome, safari, opera, IE, ...)
- [Documentation HTML4.01 francais](#) :

- [Valdateur de page](#)
  - Sélectionner : validate by file upload.
  - Parcourir : choisir votre fichier puis check.

1. Vous écrivez votre code html dans un fichier .html avec votre éditeur
2. Vous ouvrez la page dans votre navigateur pour afficher le résultat.

## Balises et attributs

- Page HTML : *texte* à afficher *structuré* par des *balises*

### Exemple de balises

```
<h1>Tortues</h1> <p> Les
tortues sont les seuls reptiles ayant une carapace dure et
osseuse.
<br/>ça fait rêver !</p>
```

- Une balise :
  - Commence par un < et se termine par >.
  - Sert à donner des informations au Navigateur.
- Deux types de balises :
  1. balises doubles :<nom\_de\_la\_balise> contenu </nom\_de\_la\_balise>
  2. balises vide :<nom de la balise />

## Balises et attributs

- Les balises peuvent avoir des *attributs*
- les attributs explicitent des propriétés des balises
- les attributs disponibles dépendent des balises utilisées.

exemple :

```

```

## Élément

**Définition** : le mot *élément* désigne une balise et ce qu'elle contient.

Exemple : Dans

```
<h1>Cours de
<em>HTML</em></h1>
```

- On a deux balises : <em> et <h1>
- <h1>Cours de <em>HTML</em></h1> est un élément ;
- <em>HTML</em> aussi ;

## Règles pour l'écriture du code

- le nom des balises est en *minuscules* ;
- toute balise ouverte est fermée
- le noms des attributs est en *minuscules*
- la valeurs des attributs est entre guillemets doubles

- les balises ne se croisent *jamais* :

```
<p> un exemple <em> incorrect</p></em>
<p> un exemple <em> correct</em></em>
```

(Ces règles sont obligatoires en XHTML, mais nous conseillons de les suivre en HTML5)

## Structure d'une page

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8"/>
    <title>Ma belle page</title>
  </head>
  <body>
    <h1>Un exemple de page</h1>
    <p>ô la <em>jolie</em> page</p>
  </body>
</html>
```

## Structure d'une page

### DOCTYPE

HTML 5

### meta charset

précise le codage des caractères (accents, etc...)

### balise ouvrante html

commence le document ; précise éventuellement sa langue (ici français).

### en-tête

balise head. Informations diverses utiles au navigateur.

Au minimum : titre du document.

### corps du document

body contient le texte du document.

## Corps

- Défini par la balise body
- Le corps contient le contenu de votre page

En XHTML, Le corps ne peut contenir que **des blocs** (donc encadré par des balises)

⇒ On ne peut donc pas directement écrire du texte (XHTML)

- Peut se composer de titres, de sous titres, de paragraphes, de listes, de tableaux...

## Texte, titres et paragraphes

- 6 niveaux de titres ⇒ 6 balises h1, ..., h6
  - h1: la plus importante
  - h6 : la moins importante
- paragraphe : balise <p>
  - affiché avec un espacement avant et après
- Mise en évidence de texte (dans un titre ou un paragraphe) :

- balises `<em>` (faible mise en évidence)
- balises `<strong>` (forte mise en évidence).

- commentaires

```
<!--
Texte qui est ignoré
par le navigateur...
à destination du programmeur.
-->
```

- passer à la ligne dans un paragraphe: le dire explicitement avec la balise `<br/>` (à éviter, préférer `<p>`);

## Exemple (source)

```
<body>
<h1>Un gros titre</h1>
<p>Voici le paragraphe associé au
  titre 1. Le saut de ligne précédent ne provoquera pas
  un saut de ligne à l'affichage.</p>
<p>Voici un bout de texte mis en évidence avec la balise em:
  <em>ceci est à faire remarquer</em>.<br/>
  L'usage de la balise
  <code>br</code> n'est
  que rarement utile.
</p>
<p>Et voila le résultat de la balise strong : <strong>très
  important</strong> .</p>
<h2>un titre un peu plus petit</h2>
<h3>encore + petit</h3>
<h4>encore + petit</h4>
<h5>encore + petit</h5>
<h6>encore + petit</h6>
</body>
```

## Exemple (résultat)

### Un gros titre

Voici le paragraphe associé au titre 1. Le saut de ligne précédent ne provoquera pas un saut de ligne à l'affichage.

Voici un bout de texte mis en évidence avec la balise em: *ceci est à faire remarquer*.  
L'usage de la balise `<code>br</code>` n'est que rarement utile.

Et voila le résultat de la balise strong : **très important** .

### un titre un peu plus petit

#### encore + petit

#### encore + petit

#### encore + petit

#### encore + petit

- présentation choisie par le navigateur. Ex. `<em>` en italiques
- modifications possibles par css.

# Les listes

Une liste est

- contenue dans une balise `<ul>` ou `<ol>` ;
- composée de *list items* « entrées de listes », désignés par la balise `<li></li>` ;

Il existe plusieurs sortes de listes :

1. les listes « non ordonnées », ou « unsorted lists » introduites par `ul`;
2. les listes numérotées « ordered lists », introduites par `ol`

## Les listes à puce(source)

```
<h2>listes à puces</h2>
<p>Les animaux domestiques les plus courants sont:</p>
<ul>
  <li>chat</li>
  <li>chien</li>
  <li>poisson rouge</li>
</ul>
<h2>listes ordonnées</h2>
<p>Pour monter des blancs en neige :</p>
<ol>
  <li>Séparer les blancs des jaunes</li>
  <li>Mettre un peu de sel dans les blancs</li>
  <li>Battre les blancs au fouet</li>
</ol>
```

## Les Liste (résultat)

### listes à puces

Les animaux domestiques les plus courants sont:

- chat
- chien
- poisson rouge

### listes ordonnées

Pour monter des blancs en neige :

1. Séparer les blancs des jaunes
2. Mettre un peu de sel dans les blancs
3. Battre les blancs au fouet

## Les listes de définition (source)

```
<h2>listes de définitions</h2>
<p>un exemple :</p>
<dl>
  <dt>Java</dt>
  <dd>langage impératif orienté objet</dd>
  <dt>Scheme</dt>
  <dd>variante du langage fonctionnel Lisp</dd>
</dl>
```

## Les listes de définition (résultat)

### listes de définitions

un exemple :

Java

langage impératif orienté objet

Scheme

variante du langage fonctionnel Lisp

## Listes imbriquées

- Une liste `<ul></ul>` ou `<ol></ol>` ne contient *que* des éléments `li`
- Si on veut mettre un paragraphe, un tableau ou une autre liste dans une liste, elle est rangée **dans** l'un des `<li></li>`

## Listes imbriquées

```
<h2>Plusieurs niveaux de listes</h2>

<p>exemple :</p>
<ol>
  <li>exercice 1
    <ol>
      <li>1er élément</li>
      <li>2ème élément</li>
      <li>3ème élément</li>
    </ol>
  </li>
  <li>exercice 2</li>
</ol>
<p>et non pas :</p>
<ol>
  <li>exercice 1</li>
  erreur Xhtml
  <ol>
    <li>1er élément</li>
    <li>2ème élément</li>
    <li>3ème élément</li>
  </ol>
  <li>exercice 2</li>
  <li>exercice 3</li>
</ol>
```

## Et le résultat

### Plusieurs niveaux de listes

exemple :

1. exercice 1
  1. 1er élément
  2. 2ème élément
  3. 3ème élément
2. exercice 2

et non pas :

1. exercice 1  
erreur Xhtml
  1. 1er élément
  2. 2ème élément
  3. 3ème élément
2. exercice 2
3. exercice 3

## Les tables

```

<table>
  <tr>
    <th></th>
    <th>lundi</th>
    <th>mardi</th>
  </tr>
  <tr>
    <td>Matin</td>
    <td>julie</td>
    <td>martin</td>
  </tr>
  <tr>
    <td>Après-midi</td>
    <td>Julie</td>
    <td>didier</td>
  </tr>
  <tr>
    <td>Soir</td>
    <td>Martin</td>
    <td>Martin</td>
  </tr>
</table>

```

## Et le résultat

	<b>lundi</b>	<b>mardi</b>
Matin	julie	martin
Après-midi	Julie	didier
Soir	Martin	Martin

## Une table

- délimitée par la balise `<table>` ;
- composée de *lignes* `<tr>` (pour *table row*) ;
- chaque ligne est composée de cases :
  - `th` « table header » pour titres de lignes ou de colonnes ;
  - `td` « table data » pour les données.

## Un tableau plus compliqué

	Mon premier tableau	
	<b>lundi</b>	<b>mardi</b>
Matin	julie	martin
Après midi		didier



## Soir                  Martin

- Titre du tableau : caption
- Regroupement de lignes :

La cellule l2c2 doit occuper 2 cellules vers la fin de la colonne attribut `rowspan`

- Regroupement de colonnes :

La cellule l4c2 doit occuper 2 cellules vers la fin de la ligne : attribut `colspan`

## Le source

```
<table summary="résumé du contenu">
  <caption>
    Mon premier tableau
  </caption>
  <tr> <th></th><th>lundi</th> <th>mardi</th> </tr>

  <tr> <td>Matin</td> <td rowspan="2">julie</td> <td>martin</td>
  </tr>

  <tr> <td>Après midi</td><!-- seconde ligne de "julie"-->
    <td>didier</td>
  </tr>
  <tr>
    <td>Soir</td>
    <td colspan="2">Martin</td> <!-- (occupe deux colonnes) -->
  </tr>
</table>
```

## Le résultat

Mon premier tableau		
	<b>lundi</b>	<b>mardi</b>
Matin	julie	martin
Après midi		didier
Soir	Martin	

## Le résultat (bis)

Mon premier tableau		
	<b>lundi</b>	<b>mardi</b>

Matin	julie	martin
Après midi		didier
Soir	Martin	

(limites de cases ajoutées en CSS)

td {border-style :solid}

## Les liens

Permet de naviguer de page en page ou de de morceaux de page en morceaux de page

Définir un lien

```
<a href="URL">texte du lien< /a>
```

**URL** : *Uniform Resource Locator* désigne une ressource, par exemple une autre page web.

## Les URL

*relative*

- ancre dans le même document : #Nom\_ancre ;
- fichier dans le même dossier : autre.html ;
- fichier dans un *autre* dossier : ../autreDossier/fichier.html

*absolue*

protocole + adresse :

- http://www.cnam.fr/salut.html : page web ;
- ftp://ftp.cnam.fr/cours.zip : fichier sur un serveur ftp ;
- file:///home/rosmord/test.html : fichier sur le disque local.

*relative au serveur*

/images/logoCnam.png

## Exemple de liens (sources)

**body de depart.html**

```
<body>
  <h1>Index</h1>
  <ul>
    <li> <a href="chapitre1.html">chapitre 1 </a></li>
    <li> <a href="coursCss/chapitre2.html">chapitre 2</a></li>
  </ul>
</body>
```

**body du fichier chapitre1.html**

```
<h1>chapitre 1</h1>
<p> bla bla bla trtrtrtr trtrtrtr </p>
<p>
  <a href="depart.html">Retour à l'index </a>
</p>
```

## body du fichier `chapitre2.html` du sous repertoire `coursCss`

```
<h1>chapitre 2</h1>
<p> bla bla bla </p>
<p> <a href="../depart.html">Retour à l'index </a>
</p>
```

### Démonstration

## Les ancres

Désignent un point précis dans une page.

Définition d'une ancre

```
<a name="nom_d_ancre">[contenu]</a>
```

Pour `nom_d_ancre` noter un nom. Sans espace, ni accent, avec une lettre comme premier caractère.

Note : on peut utiliser `id` à la place de `name`. `Id` est légal dans *toutes* les balises html.

## Les ancres, exemple (source)

```
<h1><a name="index" id="index">Index</a></h1>

<p><a href="#chapitre1">chapitre 1</a></p>
<p><a href="#chapitre2">chapitre 2</a></p>

<p> un paragraphe </p>
.....
<p> un paragraphe </p>

<h1><a name="chapitre1" id="chapitre1">Chapitre 1</a></h1>
<p> un paragraphe </p>
....
<p> un paragraphe </p>

<h2>bli blo</h2>

<h1><a name="chapitre2" id="chapitre2">Chapitre 2</a></h1>
<p> un paragraphe </p>
....

<a href="#index">retour à l'index</a>
```

### Démonstration

## Les images

Pour insérer une image dans une page : balise `img`

exemple : si on a dans notre repertoire le fichier image `"logoCnam.gif"` :

```

```



## Catégories de balises

- **block et inline**

1. type block : provoque un **retour à la ligne** avant et après.

```
p, h1, ... h6, ul, ol, dl, li, dl, dd, table ...
```

2. type inline : ne provoque **pas de retour à la ligne** : s'insère dans le texte courants. Doit être à l'intérieur d'une balise de type block.

```
a, em, strong, img ...
```

- balises div et span

- div : balise de **type block** n'ayant aucune propriété définie.
- span : balise de **type inline** n'ayant aucune propriété définie.

Servent à ajouter de la structuration au texte.

## Attribut universels

- class et id
- Pour toute les balises.
- Raison d'être de div et span.

```
<!DOCTYPE html>
<html>
<head>
  <title>Page 1</title>
</head>
<body>
  <h1>titre 1</h1>
  <p class="bleu">bla bla bla <em>etc</em></p>
  <div class="important">
    <h1>titre 2</h1>
    <p>encore bla bla bla</p>
    <h2>titre3 <em>+++</em></h2>
  </div>
</body>
</html>
```

