



Gestion d'identité en ligne

Chapitre 1

Édition de pages web avec HTML



Pablo Rauzy <pr@up8.edu>
pablo.rauzy.name/teaching/gil

Édition de pages web avec HTML

- ▶ *HTML* signifie “*HyperText Markup Language*”, langage de balisage pour l’hypertexte.
- ▶ Inventé à l’origine par Tim Berners-Lee, au CERN, sur la période 1989-1993.
- ▶ Il descend historiquement de SGML (*Standard Generalized Markup Language*).
- ▶ C’est l’une des trois inventions qui composent le *World Wide Web*, avec *HTTP* et les *URL*.
- ▶ Aujourd’hui c’est un standard du W3C.
- ▶ Il est maintenant basé sur XML (*eXtended Markup Language*).

- ▶ Un document HTML est un *arbre*.
- ▶ Les nœuds de l'arbre sont des *éléments* HTML.
- ▶ Sa racine est un élément `html`.
- ▶ Elle contient deux enfants :
 - un élément `head` pour les en-têtes ;
 - un élément `body` pour le corps.
- ▶ Les feuilles de l'arbre sont soit des éléments textuels, soit des éléments "vides".

- ▶ L'en-tête d'un document HTML contient des *méta-données* sur le document :
 - titre,
 - encodage,
 - description,
 - liens vers les feuilles de styles,
 - etc.

- ▶ Le corps du document contient son contenu, ce qui sera affiché dans la page web :
 - titres,
 - paragraphes,
 - listes,
 - tableaux,
 - formulaires,
 - etc.

- ▶ Un document HTML est un fichier texte.
- ▶ Par convention, on nomme ces fichiers texte avec l'extension ".html".
 - Ce n'est qu'une convention, il faut tout de même déclarer le type du document sur la première ligne du fichier avec : "`<!doctype html>`".
- ▶ Dans ce fichier, le code HTML représente la structure du document (l'arbre).
- ▶ Les éléments HTML sont écrits avec des *balises*.
- ▶ Une balise peut avoir des *attributs*.

- ▶ Une *balise* s'ouvre avec "<" suivi de son nom suivi de ">". Elle se ferme avec "</" suivi de son nom suivi de ">".
 - Exemple : `<html> ... </html>`
(élément racine).

- ▶ Une balise vide s'ouvre et se ferme directement avec "<" suivi de son nom suivi de "/>".
 - Exemple : `
`
("br" comme "*break*", un retour à la ligne).

- ▶ Dans la balise ouvrante d'un élément, on peut spécifier des *attributs*.
- ▶ La syntaxe pour les attributs est : nom de l'attribut, suivi de "=" suivi de la valeur de l'attribut entre guillemets doubles "".
 - Exemple : `<meta charset="utf-8" />`
(déclaration de l'encodage Unicode, dans l'en-tête).

- ▶ Les éléments textes peuvent s'écrire directement entre les balises de leur élément parent.
 - Exemple : `<p>Le contenu d'un paragraphe.</p>`
(un élément texte enfant d'un élément paragraphe).
- ▶ Ils peuvent d'être adelphe d'éléments HTML.
 - Exemple : `<p>Un lien dans un paragraphe.</p>`
(un élément paragraphe avec 3 enfants : un élément texte, un élément lien, et un élément texte).

Caractères blancs

- ▶ Les caractères blancs (espaces, tabulations, retours à la ligne) ne sont pas significatifs.
 - Profitez-en pour coder proprement !
 - L'indentation du code HTML doit faire visuellement ressortir la structure d'arbre du document.

Caractères spéciaux

- ▶ Dans les éléments textes et les valeurs des attributs, certains *caractères spéciaux* doivent être *encodés* pour ne pas être interprétés comme de la syntaxe HTML :
 - le caractère "<" s'écrit "<"; (comme "less than");
 - le caractère ">" s'écrit ">"; (comme "greater than");
 - le caractère "\"" s'écrit """; (comme "quote");
 - le caractère "&" s'écrit "&"; (comme "ampersand").
- ▶ Historiquement, tous les caractères spéciaux devaient être encodés de cette façon.
 - Exemple : "à" était "à"; "é" était "é"; etc.
- ▶ Aujourd'hui grâce au standard d'encodage Unicode on a plus ce genre de soucis.
 - On encode plus que les quatre utilisés pour la syntaxe.

Exemple

```
1 <!doctype html>
2 <html lang="fr">
3   <head>
4     <meta charset="utf-8" />
5     <title>Ma première page web</title>
6   </head>
7   <body>
8     <header>
9       <h1>Titre du site</h1>
10      <nav>
11        <ul>
12          <li><a href="index.html">Accueil</a></li>
13          <li><a href="projets.html" class="current">Mes projets</a></li>
14        </ul>
15      </nav>
16    </header>
17    <main>
18      <h2>Mes projets</h2>
19      <article>
20        <h3>Premier projet</h3>
21        <!-- ... -->
22      </article>
23      <!-- ... -->
24    </main>
25    <footer>
26      <p>\_o&lt; coin coin &gt;o_/</p>
27    </footer>
28  </body>
29 </html>
```

- ▶ Un document HTML est *valide* si sa structure respecte la spécification du langage.
 - La spécification ne se limite pas à la syntaxe !
 - L'arbre doit être valide également : le standard HTML définit pour chaque type d'élément les attributs qu'il peut avoir et les types d'éléments qu'il peut avoir comme enfants.

- ▶ La meilleure documentation en ligne est celle de Mozilla :
 - <https://developer.mozilla.org/fr/docs/Web/HTML>

Éléments “en ligne” et éléments “bloc”

- ▶ Certains éléments sont dits *inline* : ils s’intègrent dans le flot du texte.
 - Exemple : Un mot `important` dans un texte.
(un mot marqué comme important dans un texte, qui apparaîtra en gras par défaut)
- ▶ Certains éléments sont dits *block* : ils créent une rupture dans le flot du texte.
 - Exemple : `lalalili`
(une liste à puce avec deux éléments qui seront chacun sur leur ligne par défaut).
- ▶ La spécification précise par exemple qu’un élément de type `ul` ne peut avoir que des enfants de type `li` :
 - https://developer.mozilla.org/fr/docs/Web/HTML/Element/ul#résumé_technique

- ▶ Chaque élément HTML a une *signification*, qu'il faut distinguer de son *rendu visuel*.
 - Par exemple l'élément **strong** signifie qu'un mot est important, par défaut, son rendu visuel est de produire du texte gras, mais cela pourrait être autre chose comme un changement de couleur.
- ▶ On veillera à toujours utiliser les balises pour leur *sens*.
 - On apprendra avec **CSS** à contrôler l'aspect visuel du rendu.
- ▶ C'est très important en terme d'accessibilité et de référencement.
 - Exemple : les titres et sous-titres dans une page sont significatifs, un robot d'indexation (pour déduire des mots-clés) ou un lecteur d'écran (pour la navigation) doivent pouvoir les différencier du reste du contenu.

- ▶ Pour éditer les fichiers textes HTML : **nano**.
- ▶ Pour tester le rendu : Firefox.
 - Si vous travaillez à distance sur les machines du Bocal vous pouvez mettre vos fichiers HTML dans votre répertoire `~/public_html/` ; vous verrez votre site en visitant l'URL `http://www.bocal.cs.univ-paris8.fr/~LOGIN/`.
 - Les *serveurs web* lisent par défaut le fichier `index.html` quand on demande un dossier.
 - En local il suffit de faire "Ouvrir avec..." sur votre fichier HTML et choisir votre navigateur web.
- ▶ Pour comprendre et déboguer : les "outils de développement web".
 - Burger menu > Outils supplémentaires > Outils de développement web.