

IHM web 1 - Cours - Les langages web

1.1 des langages interprétés

Plan du chapitre :

- preconisations du W3C, 3 langages complémentaires
- Le Web et les hyperliens

HTML : HyperTextMarkup Langage ; CSS : Cascading StyleSheets ; Javascript sont des langages interprétés par le navigateur, côté client. Ces 3 langages, en combinaison, vont permettre de créer des *sites internet*.

Les instructions de ces langages sont interprétées par le **navigateur** qui permet l’affichage des pages.

Ces langages sont en constante évolution. Ce sont des langages *open sources*^[5] et de nombreux contributeurs en proposent des améliorations.

Il existe plusieurs logiciels navigateurs (Mozilla, Chrome, Safari,...). Et pourtant, malgré cette diversité et ces évolutions du langage, ceux-ci vont interpréter ces fichiers et afficher les pages (presque) de la même manière car ils se réfèrent tous aux mêmes recommandations, celles du w3c.

Une petite différence peut exister pour le rendu d’une même page, mais avec plusieurs navigateurs différents. Cela est dû aux **propriétés par défaut utilisées par le navigateur** pour l’affichage des différents éléments html de la page. Ces propriétés peuvent être déclarées (CSS) afin d’avoir un rendu toujours identique.

1.2 des tâches différentes

Les langages web se partagent les tâches : Une bonne pratique dans le développement d’un site internet (côté front-end, ce qui est exécuté sur la machine du client) consiste à utiliser :

- HTML pour le **contenu** , et la **structure** (avec un contenu correctement balisé, sémantique, accessible)
- CSS pour la mise en forme et le design de la page, les **propriétés** relatives à chaque élément de Style (balise html).
- et JavaScript pour gérer les **interactions** (qui peuvent éventuellement amener à modifier le contenu via les méthodes du DOM^[1]).

Le langage HTML est un langage constitué de *balises*, comme par exemple : `<title>le titre de ma page</title>` qui permet d’afficher le titre de la page dans l’onglet du navigateur.

Les éléments mis dans le programme à l’aide de ces balises vont permettre d’ajouter et structurer le contenu de la page : des titres de différents niveaux, du texte, des images, des médias (sons, vidéos), mais surtout, des hyperliens :

- entre les pages du sites

- vers les pages de sites externes

Le web est justement basé sur l'utilisation de ces hyperliens, qui permettent de naviguer de pages en pages, sur internet.

Part 2

Le HTML et la page Web

2.1 Les balises principale

Def : un element HTML, comme p, h1, a, ... est défini en HTML par des balises.

Les balises sont les instructions en langage HTML des éléments structurants de la page web.

Balises ouvrantes et fermantes : La plupart des balises (non orphelines) vont par paire, insérant le *contenu*.

Balises avec attribut : Les balises peuvent comporter des **attributs**, auxquels on associe des **valeurs**.

Les balises se distinguent entre celles qui n'ont pas d'attribut obligatoire, et celles qui en ont. Un attribut va ajouter des fonctionnalités à l'élément HTML, et modifier son allure ou son fonctionnement.

Les principales balises structurantes sont :

- h1, h2, h3, ... h6
- div
- p
- span
- ...

element	usage	display	attribut obligatoire
h1	titre niveau 1	block	
h2	titre niveau 2	block	
h3	titre niveau 3	block	
p	paragraphe	block	
div, header, footer, aside, section	container	block	
img	image	inline	src
a	lien	inline	href
span	container	inline	

Elles s'utilisent de la manière suivante :

```
1 <h1>Texte</h1>
```

```
2 <p>Texte</p>
```

Il existe aussi des **balises qui doivent être combinées**, comme celles de listes :

- ul (parent) => li (enfants) pour une liste à puces
- ol (parent) => li (enfants) pour une liste ordonnée (numérotée)

```
1 <ul>
2   <li>Premier</li>
3   <li>Deuxieme</li>
4 </ul>
5
6 <ol>
7   <li>troisieme</li>
8   <li>quatrieme</li>
9 </ol>
```

Résultat :

```
1 * Premier
2 * Deuxieme
3
4 1. troisieme
5 2. quatrieme
```

2.2 Propriétés particulières de certains éléments

Prenons un exemple avec les 2 principales balises du Web : Les éléments p et a diffèrent par leurs propriétés par défaut :

element	display	color	text-decoration
p	block	black	none
a	inline	blue	underline
a :visited	inline	purple	underline

Les éléments de type block prennent toute la largeur de leur parent. Ceux de type inline prennent la dimension de leur contenu.

Il est possible de spécifier une nouvelle largeur et hauteur de boîte, mais **seulement avec les éléments de type block**. La propriété width modifie la largeur, height modifie la hauteur

2.3 balises avec attribut obligatoire

Les attributs seront obligatoires pour certaines balises selon leur fonction. Certains attributs sont facultatifs et vont juste enrichir leur comportement.

2.3.1 lien a

href est un attribut obligatoire. Il redirige , soit :

- vers une adresse interne (voir cours à venir)
- ou bien externe (commence par http)

Modèle :

```
1 <a href="chemin_vers_la_ressource.html">texte cliquable</a>
```

2.3.2 image img

Il s'agit d'une balise *orpheline*. (Sans balise fermante)

src est un attribut obligatoire.

```
1 
```

alt est un attribut facultatif, que l'on ajoute pour afficher un texte lorsque l'adresse de l'image est erronée, ou que celle-ci ne s'affiche pas.

```
1 
```

2.4 le squelette du document

Cette partie du script est le contenu minimum à mettre dans vos pages HTML :

```
1 <!DOCTYPE html>
2
3 <html>
4   <head>
5
6       <!-- encodage des caractères -->
7       <meta charset="UTF-8">
8       <!-- Titre -->
9       <title>Titre de la page web</title>
10      <!-- Lien vers la feuille de style -->
11      <link rel="stylesheet" type="text/css" href="style.css">
12  </head>
13
14  <body>
15      <!-- corps de la page -->
16
17  </body>
18 </html>
```

Le **doctype** indique au navigateur la version HTML utilisée par la page (ici HTML5).

L'élément racine `<html>` : C'est lui qui va recueillir les deux principaux éléments de la hiérarchie : `<head>` et `<body>`.

À ce niveau, le code HTML est alors divisé en deux parties.

On peut lui associer l'attribut `langue`, précisant la langue utilisée dans le document :

```
<html lang="fr">
```

L'en-tête (élément `<head>`) donne l'encodage des caractères (ici UTF-8).

Préciser l'encodage des caractères est primordial pour exploiter la bonne page de code et ne pas se retrouver avec les caractères spéciaux ou accentués. Le choix de l'UTF-8 est désormais préconisé par le W3C pour tous les protocoles échangeant du texte sur internet (dont HTML).

On peut aussi y ajouter des éléments `<style>`, `<link>` et `<script>` :

- `style` permet de placer du code CSS dans la page elle-même
- `link` : permet de mettre en relation la page avec d'autres documents externes, comme une feuille de style :

```
1 <link rel="stylesheet" type="text/css" href="style.css">
```

- `<script>` permet d'ajouter des scripts (JavaScript) qui vont s'exécuter côté client dans le navigateur dès leur chargement, directement dans la page
- `<script>` et fichier externe : là aussi, on peut placer les instructions du langage javascript directement dans la page :

```
1 <script src="chemin/vers/le/script.js"></script>
```

Si vous avez un script qui est très gros mais indépendant, il est préférable de le placer tout à la fin, afin de ne pas retarder le navigateur dans sa construction de l'arbre du DOM et de l'affichage de la page.

2.5 Imbrication et filiation des balises

La première page de l'histoire du Web ne présentait pas de mise en forme particulière. Le code ne contenait pas de balises *fermantes*. Les pages n'étaient pas hiérarchisées.

2.5.1 Mise en boîte

Avec la norme actuelle du HTML, les pages présentent une structure qui propose un parcours de lecture, un design. L'idée est de capter l'attention du lecteur le plus longtemps possible, de transmettre une information dans un certain ordre.

Et pour mettre en forme des éléments à différentes positions, il faut insérer les éléments HTML dans des "boîtes".

D'où l'usage de balises dite "container", qui vont permettre ces structures en lignes et colonnes, comme `div`, `section`, `article`, ...

La boîte principale sera l'élément `body`, qui contiendra toutes les autres.

2.5.2 Arbre du DOM

L'imbrication des balises traduit le lien (la filiation) entre les éléments.

On représente la structure d'un document html à l'aide d'un arbre. On parle d'**arbre DOM** (Document Object Model) du document.

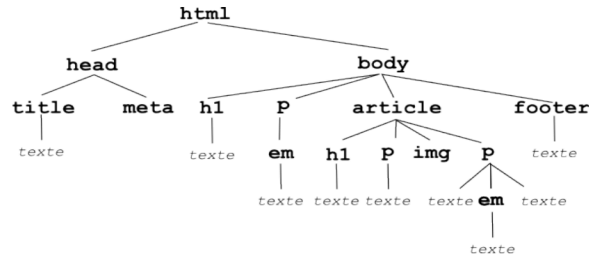


FIGURE 1 – Exemple d'arbre du DOM

Pour cet exemple : Le **nœud** article est le **père** des nœuds h1, p, img et p. Les nœuds h1, p, img et p sont les **fil**s du nœud article.

Le schéma suivant montre la position de ces *boîtes* avec la page 2 vue plus haut en exercices :

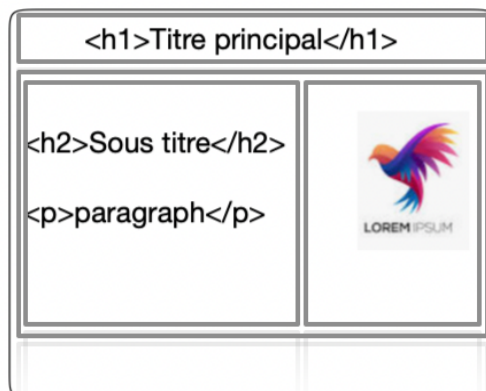


FIGURE 2 – elements dans la page du navigateur

L'image en forme d'arbre, montre la hierarchie entre ces éléments (arbre du DOM)

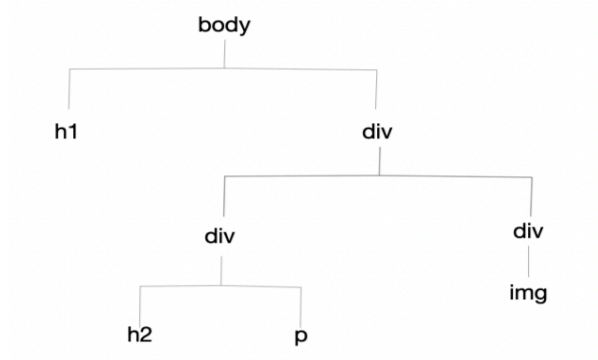


FIGURE 3 – balises sans hierarchie

2.5.3 Quand faut-il recourir à des balises parentes ?

Supposons que l'on ait besoin de structurer la page comme sur l'image suivante :



FIGURE 4 – page avec sections

Les éléments *containers* s'ils sont de type paragraphes, sections, header, nav, article, ... vont se placer naturellement l'un sous l'autre. Il n'y a pas besoin d'imbriquer les balises.

Par contre, s'il faut disposer côte à côte 2 éléments containers, pour faire 2 colonnes, il faudra les disposer à l'intérieur d'un autre élément parent. Puis modifier leur disposition naturelle (paramètre `display`).

Pour cet exemple, au niveau de l'élément au fond blanc, il faudra :

- un container parent, par exemple un élément `div`
- (colonne gauche) : un container fils de `div` qui contiendra lui-même les éléments fils 'section, article, article'
- (colonne droite) : un élément fils de `div` qui sera l'élément `aside`

Part 3

Le CSS

Les sigles « CSS » sont l'abréviation de « Cascading StyleSheets » ou « feuilles de styles en cascade » en français.

3.1 Les 3 manières d'écrire des déclarations en css

3.1.1 directement dans la balise

avec l'attribut optionnel `style`, on peut mettre les déclarations dans la balise elle-même. Pratique pour de petits tests. Cette modification ne concerne alors que l'éléments choisi.

C'est un peu comme la pratique de faire du formatage direct avec Writer, en utilisant la barre d'outils. [Montrer cette pratique avec le logiciel Writer]

3.1.2 Entre les balises `style`, dans la page `css`

Les règles sont alors appliquées à toute la page. Equivalent à la pratique dans un fichier Writer selon laquelle nous avons mis à jour les nouvelles propriétés pour tous les éléments d'un style défini.

3.1.3 Dans un fichier annexe

Ce fichier doit alors être pointé avec la balise `<link rel="stylesheet" type="text/css" href="style.css">`

L'avantage de cette pratique, c'est de pouvoir réutiliser la feuille de style pour TOUTES les pages d'un même site. C'est l'équivalent de la pratique selon laquelle nous avons sauvegardé le fichier comme un modèle de feuille de style. Puis nous l'avons appliqué au prochain fichier ouvert.

3.2 Ecriture d'une déclaration en CSS

Une **déclaration**, c'est :

selecteur { règle ; }

Une règle, c'est : **propriété : valeur**. Il en existe des centaines, toutes référencées sur MDN Mozilla par exemple.

Ces règles peuvent se classer en différentes familles :

- couleur
- effets de texte (police, taille, graisse, soulignement)
- dimensions
- marges internes et externes
- effet de liste (list-style)
- alignements (indent, left, right, center, ..)
- effets graphiques (bordures, bords arrondis, ombres, ...)
- ...

Le choix d'une règle va dépendre de l'élément lui-même, mais aussi de la compatibilité entre règles.

Un même effet peut être obtenu en appliquant une règle à lui-même ou bien à son parent :

- `text-align: center` centre le contenu d'un élément à l'intérieur d'un conteneur, par exemple les lignes d'un paragraphe ou un bouton inline.
- `margin: 0 auto` sert à centrer le bloc lui-même dans son parent, à condition qu'il ait une largeur définie.

Enfin, certaines règles sont héritées de leur parent (toutes les règles ne le sont pas). On place souvent la règle `font` à l'élément `body` afin que tous les éléments d'une même page aient la même police de caractères. D'où le nom CSS, traduit par feuille de style en cascade.

3.3 Les propriétés textuelles

Exemple de propriétés vues avec Writer	équivalent en css	valeurs possibles
police de caracteres	font-family	Helvetica, "Trebuchet MS", Verdana, sans-serif;
taille	font-size	12px ; 1.0 em ; 80% ;
effets de caractere (font effect)	text-decoration	none ; underline ;
effets de caractere (font effect)	font-weight	normal ; bold ;
alignement	text-align	left, center, right
couleur de la police	color	color : #00a400 ; rgb(214, 122, 127) ;

Pour font-family : Les valeurs sont séparées par des virgules, indiquant chacune une police alternative. Le moteur choisira la première valeur pour laquelle la police correspondante est installée sur l'ordinateur

3.4 Les propriétés des boîtes

Exemple de propriétés vues avec Writer	équivalent en css	valeurs possibles
borders	display	block ; inline ; inline-block ; flex ; none ;
effet caractere couleur de fond	border	solid ; 2px dotted ;
marges	background-color	voir css texte color
borders paddings	margin	2px
	padding	10px 50px 30px 0 ;

The display CSS property sets whether an element is treated as a block or inline box.

- Les balises s'affichant en "block" (bloc) : elles prennent toute la largeur disponible et s'affichent avec un saut de ligne avant et après. On peut les dimensionner avec les propriétés width et height. Elles n'acceptent pas vertical-align. Ils peuvent avoir tout type de contenu, inline ou block.
- Les balises s'affichant en "inline" (dans la ligne) : elles prennent uniquement la largeur dont elles ont besoin, sans ajouter de saut de ligne. Elles acceptent vertical-align. Leur taille s'adapte au contenu. Celui-ci ne peut être de type block.
- cas particulier de img : Sa propriété display par défaut vaut inline, mais ses dimensions par défaut sont définies par les valeurs intrinsèques de l'image, à la façon de inline-block. Il est tout à fait possible d'utiliser les propriétés border/border-radius, padding/margin, width, et height sur une image.

Les types de valeur des propriétés width et height sont :

- soit des valeurs de type « longueur » qui vont être généralement exprimées en px ou en em ;
- soit des valeurs en pourcentage, auquel cas le pourcentage donné sera relatif à la dimension de l'élément parent.

3.5 Selection, compléments

selecteur	effet
.important, .alerte	selection multiple avec les éléments séparés par une virgule
td > img	Plusieurs sélecteurs séparés par un caractère supérieur : inclusion directe. Les balises img directement incluses dans une cellule de tableau sont concernées. Il ne doit pas exister de niveaux intermédiaires.
td + img	Plusieurs sélecteurs séparés par un signe + : vient après. Désigne les balises img venant après une balise p.
p ~ img	Plusieurs sélecteurs séparés par un signe ~ : une balise qui suit une autre (directement ou pas)
:hover	Le caractère : (deux points) introduit une pseudo-classe. La pseudo-classe désigne un élément lorsqu'il se trouve dans une certaine situation : ici lorsqu'il est survolé par la souris. Voir la liste complète des pseudo-class.
img[width]	Désigne les balises img comportant un attribut width quelle que soit la valeur de ce dernier. On teste donc seulement la présence de l'attribut.
p[lang = "fr"]	Ici on limite la portée aux balises p qui comportent l'attribut lang avec la valeur "fr". p lang="fr"

3.6 Codage de la couleur

page SNT images, TP données couleur EXIF voir sur [http ://www.proftnj.com/RGB3.htm](http://www.proftnj.com/RGB3.htm)