1 Introduction

1.1 XHTML et CSS

L'écriture de pages web d'appuie sur deux langages :

1. XHTML, *eXtensible HyperText Markup Language*, qui permet de décrire le *contenu logique* du document. Par exemple : la page contient le texte "Ma première page" qui est un titre de niveau 1, le texte "blablabla" qui est un paragraphe, la photo "maphoto.jpg" qui est une image, l'acronyme "XHTML", le texte "eXtensible HyperText Markup Language" qui est une expression en langue étrangère, le texte "blablabla" qui est une définition, une section de conclusion, *etc.*

2. CSS, *Cascading StyleSheet*, qui permet de décrire la *présentation* du document. Par exemple, les titres de niveau 1 s'affichent en police de taille 16 et de couleur rouge, le texte d'un paragraphe doit être justifié, les images ont une largeur de 200 pixels, les acronymes s'affichent en caractères gras, les expressions en langue étrangère sont inscrites entre guillemets, les définitions s'affichent en caractères italiques, la conclusion est ponctuée du caractère spécial "•", *etc.*

Pour faire une analogie avec un éditeur de traitement de texte : une chose est d'écrire une phrase à laquelle on applique le style "Titre 1", une autre est de définir les paramètres de mise en forme du style "Titre 1".

1.2 Histoire et origine

Au départ... il y avait SGML (pour *Standard Generalized Markup Language*), qui est un méta-langage, c'est-à-dire un langage permettant d'écrire d'autres langages, dits *de structuration* ou *à balises*. On décrit en SGML un ensemble de balises ou d'éléments (identifiés par leur nom), ainsi que leur contenu. Le contenu d'un élément est défini par une liste de sous-éléments (éventuellement hiérarchisés) et le nombre d'occurrences de ceux-ci, ainsi que par une liste d'attributs et les valeurs pouvant être prises par ceux-ci. L'ensemble de ces spécifications forme ce qu'on appelle une DTD (pour *Document Type Definition*). Par exemple, l'élément "tableau" contient forcément un ensemble d'éléments "ligne", l'élément "ligne" comporte lui-même nécessairement un ensemble d'éléments "cellule" (imbrication des éléments) ; par ailleurs, un tableau peut ou non être quadrillé, avoir des bordures, *etc.* (valeur d'attributs).

Le langage HTML est une application de SGML. Conçu à l'origine pour l'échange de documents scientifiques, HTML a connu un tel succès qu'il s'est développé de façon quelque peu anarchique. Ce n'est que récemment que l'on parvient à une certaine harmonisation, essentiellement due au *World Wide Web Consortium* W3C qui en édicte les standards, mais aussi à la bonne volonté des éditeurs de navigateurs web et des développeurs de sites web à s'y conformer. Deux grandes réussites de l'évolution du langage HTML sont, d'une part, *la séparation nette du contenu logique de la présentation* par l'externalisation des spécifications de style dans les fichiers CSS et, d'autre part, *le caractère vérifiable* du langage *via* l'utilisation de DTD. Le langage HTML disparaît récemment au profit du langage XHTML (pour *Extensible HyperText Markup Language*), à la fois *plus simple* - XHTML est écrit en XML (pour *Extensible Markup Language*) qui est une simplification de SGML - et *plus universel* - les outils XML étant de plus en plus standard dans les applications informatiques.

1.3 Environnement logiciel

1.3.1 Pour *"exécuter"*

Les formats XHTML et CSS sont directement interprétés par les navigateurs web (*ie.*, il n'y a pas de phase de compilation comme en C). Parmi les navigateurs les plus connus, citons Mozilla Firefox, Internet Explorer, Opéra, ... À noter sur l'utilisation des formats XHTML et CSS :

- XHTML et CSS sont des langages universels, supportés par toutes les plateformes (Windows, Unix, ...) ;
- néanmoins, le rendu visuel dépend du navigateur utilisé (tous les navigateurs n'interprètent pas de la même façon les balises XHTML, ni les propriétés de style CSS).

1.3.2 Pour développer

- Extension des fichiers XHTML Deux extensions sont possibles : ".html" ou ".htm".
- Quel éditeur ? On peut développer des pages web sous n'importe quel éditeur de texte (*ex.* le bloc-notes windows fait très bien l'affaire !). Néanmoins, pour se faciliter la tâche, on pourra préférer un éditeur qui reconnaît la syntaxe XHTML (*ex.* : Notepad++, Kate, emacs, gedit, ...).
- Quel navigateur ? Évidemment, lorsque l'on développe des pages web, on ne peut préjuger du navigateur qui sera utilisé par l'internaute ! Aussi, il faut, dans la mesure du possible, maîtriser l'apparence des pages pour *tout* navigateur utilisé. Néanmoins, dans le cadre de ce cours, on utilisera exclusivement Mozilla Firefox, qui fait partie des navigateurs les plus conformes aux normes de développement W3C.

1.4 Par la suite ...

Ce poly se concentre sur le langage XHTML, c'est-à-dire qu'il apprend à décrire la *structure* d'un document, et non sa *mise en forme*. Il introduit néanmoins avec les consignes de style "*au fil de l'eau*" quelques propriétés qui seront utiles au moment d'aborder CSS.

2 XHTML, un langage à balises

Les langages à balises permettent de *structurer* la description d'un document par le biais d'éléments logiques que l'on peut spécifier à l'aide de certains attributs. Dans un document (et pour tout support d'édition de ce document), on voudra notamment savoir dissocier un titre de niveau 1 d'un titre de niveau 2, du texte d'une image, une figure d'un tableau (plus spécifiquement sur une ressource web, fichier audio *vs.* vidéo, texte statique *vs.* dynamique) ; on pourra aussi vouloir indiquer des dépendances, *ex.*, tel paragraphe est un sous-paragraphe de tel autre, tel paragraphe arrive à la suite de tel titre : c'est le rôle des *éléments*. On voudra enfin pouvoir spécifier les *paramètres* de telle ou telle image, de tel ou tel paragraphe : c'est le rôle des *attributs*. Par exemple, quelle URL pour tel lien dynamique, quel identifiant pour telle section, quel nature d'information – théorème, définition, preuve – pour tel paragraphe, *etc.*

2.1 Éléments

En XHTML, les éléments logiques (*ex.* : titre de niveau 1, paragraphe, image) sont déclarés par des balises de la forme :

<élément> (balise ouvrante) </élément> (balise fermante)

Tout contenu de la page doit être introduit à l'aide d'un élément logique. Quelques exemples :

Titre de niveau 1 <h1>Ma première page</h1>
on écrit le contenu “Ma première page” en le déclarant comme élément de titre “h1”
Paragraphe blablabla
on écrit le contenu “Ma première page” en le déclarant comme élément de paragraphe “p”

Certains éléments n'utilisent qu'une balise, conjointement ouvrante et fermante :

<élément />

C'est par exemple le cas de l'élément “img”.

Image
on écrit le contenu “maphoto.jpg” en le déclarant comme élément “img”

2.2 Attributs

Les éléments disposent de certains paramètres dont on peut (ou doit, selon les cas !) spécifier la valeur. Ces paramètres sont appelés *attributs* en XHTML, et la valeur des attributs d'un élément est spécifiée dans la *balise ouvrante* de l'élément (la balise ouvrante et fermante si l'élément ne dispose pas de balise fermante). La syntaxe consiste, dans la balise ouvrante, à lister un ensemble de couples *attribut="val"* après la déclaration l'élément :

<élément attribut$_1$="val$_1$" attribut$_2$="val$_2$"... attribut$_\ell$="val$_\ell$"> ... </élément>

Pour certains attributs (notamment, pour ceux définissant des propriétés de style), des valeurs par défaut sont définies.

2.3 Illustration

L'élément “img” dispose de l'attribut “src” qui doit être obligatoirement renseigné, puisqu'il indique l'image à insérer. Il dispose également de l'attribut “alt”, lui aussi obligatoire : cet attribut permet de définir, par une chaîne de caractères, une altenative à l'image (au cas où celle-ci ne pourrait pas s'afficher). Autre exemple : (quasi) tous les éléments disposent de l'attribut optionnel “id” qui permet (notamment) de définir une référence dans la page XHTML :

```
<h1 id="titre-principal">Ma première page</h1>
<p id="paragraphe-blabla">blablabla</p>
```

Si l'URL du fichier est par exemple “http://toto.free.fr/index.html”, alors les URL :

“http://toto.free.fr/index.html#titre-principal”
“http://toto.free.fr/index.html#paragraphe-blabla”

sont également valides, et permettent d'accéder à la page “index.html” aux niveaux du titre “Ma première page” et du paragraphe “blablabla”, respectivement.

3 Structure et syntaxe d'un document xhtml

3.1 Structure et syntaxe d'un document xhtml

Pour écrire un document xhtml, il faut :

1. Déclarer que l'on est en train d'écrire en XHTML ; cela se fait en spécifiant la DTD à laquelle le fichier se conforme (en l'occurrence, on prend la DTD W3C pour XHTML 1.0 “stricte”) :
   ```
   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "DTD/xhtml1-strict.dtd">
   ```
   La balise DOCTYPE est une balise SGML (et XML) : on n'est pas encore entré dans le XHTML (il en est d'ailleurs de même des marques de commentaire).

2. Disposer d'un élément racine "html" (unique).
   <html> ... </html>
   Attention : *tout contenu XHTML doit être contenu dans un élément "html" !*

3. Dans l'élément racine, disposer d'un élément d'entête "head", ainsi que d'un élément "body" (tous deux obligatoires et uniques) :

   (a) La zone d'entête définit certaines propriétés du document ; par exemple, le titre (qui apparaît dans la barre de titre du navigateur), la langue, les mots clés, l'encodage des caractères, l'auteur, les feuilles de style, l'URL de base, ...
   <head> ... </head>
   Attention : *aucun contenu ne doit apparaître dans la zone d'entête !*

   (b) Le corps décrit le contenu même du document.
   <body> ... </body>

4. Dans l'élément "head", disposer d'un élément "title" (obligatoire et unique)
   <title> ... </title>

Ainsi, un document XHTML devra au minimum contenir les éléments suivants :

```
<!-- langage utilisé -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "DTD/xhtml1-strict.dtd">
<html>                                          <!-- début du document xhtml -->
   <head>                                          <!-- début de l'entête -->
      <title>La page de Toto</title>                <!-- titre de la page -->
   </head>                                         <!-- fin de l'entête -->
   <body>                                          <!-- début du contenu -->
   </body>                                         <!-- fin du contenu -->
</html>                                         <!-- fin du document xhtml -->
```

3.2 Pour mieux comprendre... un peu de XML

La DTD est une spécification de langage, soit, la description de sa syntaxe. Indiquer la DTD utilisée permet de vérifier que l'on est conforme à la syntaxe qui y est décrite. On commente simplement ici quelques extraits de la DTD XHTML.

1. <!ELEMENT html (head, body)>
   Déclaration d'un élément "html", qui contient *obligatoirement* un élément "head" suivi d'un élément "body".

2. <!ATTLIST head ... id ID #IMPLIED ...>
   Déclaration des attributs (liste incomplète ici) de lélément "head". On voit que "head" dispose de l'attribut "id", de type "ID", et qui est optionnel (valeur #IMPLIED).

3. <!ELEMENT img EMPTY>
   Déclaration d'un élément "img", qui ne contient pas d'élément (valeur EMPTY).

4. <!ATTLIST img ...
   src %URI; #REQUIRED alt %Text; #REQUIRED ...
   height %Length; #IMPLIED width %Length; #IMPLIED
   ...>
   Déclaration des attributs (liste incomplète ici) de lélément "img" : "src" de type URI et "alt" de type Text, toux deux obligatoires (valeur #REQUIRED), "width" et "length", de type Length, à caractère optionnel (valeur #IMPLIED). Ces 4 attributs attendent tous des chaînes de caractères, les entités URI, Text et Length étant toutes trois synonymes de CDATA.

5. <!ENTITY % URI "CDATA">
   Définition d'une entité (équivalent à une constante symbolique en C) URI, de valeur "CDATA" (CDATA désigne le type "chaîne de caractères"). Pour consulter la valeur de la constante URI, il faut écrire : %URI;

3.3 Syntaxe élémentaire

3.3.1 Marque de commentaires

Tout langage a sa marque de commentaire. En XHTML, il s'agit de la balise "<!-- -->" ; exemple :

1. <!-- texte commenté -->

3.3.2 Quelques règles

1. Tout élément *ouvert* doit obligatoirement être *fermé* (ce n'était pas toujours le cas en HTML) :

<élément> ... </élément>	si "élément" dispose de deux balises ouvrante et fermante (ex. : "p")
<élément />	si "élément" dispose d'une seule balise ouvrante/fermante (ex. : "img")

2. Les éléments comme leurs attributs doivent être écrits en *caractères minuscules* (ce n'était pas le cas en HTML) :
   <élément attribut="val"> *et non pas* <ÉLÉMENT attribut="val"> *ni* <élément ATTRIBUT="val">

3. La valeur d'un attribut doit être spécifiée *entre guillemets* :
   <élément attribut="val"> *et non pas* <élément attribut=val>

4. Tout contenu doit être compris dans *au moins* un élément (ne serait-ce que l'élément de contenu principal "body").

5. Les éléments doivent être refermés dans l'ordre inverse de leur ordre d'ouverture (*i.e.*, gestion "LIFO, Last In First Out"). On nécrira donc pas :

   Coucou c'est moi

   Mais plutôt (où l'élément "em" est un élément de texte indiquant une mise en exergue) :

   Coucou c'est moi

Attention à la souplesse syntaxique ! Un document XHTML sera toujours lu par un navigateur, que le document comporte des erreurs (*ex.* : absence de balise fermante pour un élément ouvert, mauvaise valeur d'un attribut, nom d'attribut incorrect pour un élément, ...), ou qu'il ne soit pas conforme aux normes W3C (*ex.* : élément écrit en caractères majuscules, utilisation d'un attribut obsolète, ...). En effet... les navigateurs n'affichent pas de message d'erreur ! C'est pour cela qu'il est d'autant important de s'auto-discipliner et de se conformer aux normes de développement édictées par le W3C. Il existe un moyen de vérifier en ligne la validité de ses pages (puisqu'il s'agit simplement d'être syntaxiquement conforme à la DTD !) : `http://validator.w3.org/`

3.3.3 Caractères spéciaux

La spécification d'un caractère spécial commence par le caractère "&" et finit par le caractère ";" ; entre les deux : il faut spécifier, soit le code ASCII du caractère à afficher (précédé du caractère #), soit (lorsque celui-ci est défini) son code XHTML. Voici quelques exemples :

caractère	**désignation par code HTML**	**désignation par code ASCII**
&	&amp;	&#38;
€	&euro;	&#128;
<	&lt;	&#60;
>	&gt;	&#62;
"	&quot;	&#34;
#	-	&#35;

Il existe en particulier des codes XHTML pour spécifier les accents :

accent	aigu	grave	circonflexe	tréma	cédille
code HTML	acute	grave	circ	uml	cedil

Ainsi, un caractère accentué peut être désigné, soit par spécification du caractère et de l'accent (syntaxe : &LettreAccent;), soit par spécification de son code ASCII (syntaxe : &#CodeASCII;). Par exemple, "Ó", "Ò", "Ô" et "Ö" écriront respectivement les caractères "Ó", "Ò", "Ô" et "Ö". Il en sera de même si l'on utilise les codes ASCII des caractères : "Ó", "Ò", "Ô" et "Ö", respectivement.

Pour finir, un caractère spécial fort utile, l'espace insécable " " (*non blank space* en anglais) : un espace insécable est un caractére blanc de séparation qui interdit le passage à la ligne (indispensable, notamment, en français, avant les caractères de ponctuation "!", "?", ":" et ";").

4 Quelques éléments pour commencer

4.1 Éléments de niveau texte

4.1.1 Éléments de mise en exergue (ou emphase)

balise ouvrante	**rôle**	**affichage usuel**
	exergue forte	caractères gras
	exergue faible	caractères italiques
ex. :	ceci est crucial	
	ceci est important	

4.1.2 Premier pas vers le dynamisme : les liens

Un lien est une adresse ou, plus exactement, une URL généralisée (adresse web, document, e-mail etc.). L'élément de lien est ouvert par la balise <a> et fermé par la balise . Son attribut essentiel est l'attribut "href" qui prend la valeur de l'URL visée. Exemples de liens :

Vers un site quelconque :
cliquez ici pour effectuer une recherche google
Vers une page locale :
cliquez ici pour visualiser mon CV
Vers un document :
cliquez ici pour télécharger mon CV (format pdf)
Vers une adresse e-mail :
cliquez ici pour m'écrire !

L'élément "a" peut également faire référence aux ancres qui sont définies sur les éléments dont l'attribut "id" est affecté. Si l'on écrit sur le fichier "cv.html" :

<h3 id="1999">depuis 1999</h3>

Alors l'URL suivante devient une URL valide :

cv.html#1999

À laquelle on peut se référer, de manière à se placer *exactement* sur l'élément "h3" d'identifiant 1999 :

année1999
lien sur le fichier cv.html

ce que je faisais en1999
lien sur tout fichier situé dans le même répertoire que cv.html

ce que je faisais en1999
lien sur tout fichier

4.2 Éléments de niveau bloc

balise(s)	signification	...en français	exemple / remarque
	heading 1	titre de niveau 1	**ex. :** Ma page
	heading 2	titre de niveau 2	**ex. :** CV
	paragraph	nouveau paragraphe	**ex. :** 1999-2000
<pre></pre>	*preformated*	paragraphe de texte préformaté	**ex. :** <pre>1999 - 2000</pre>
 	line break	saut de ligne	**rq. :** une seule balise !
<hr />	*horizontal rule*	barre horizontale	**rq. :** une seule balise !

1. "h1, h2, ..., h6" : ces éléments définissent 6 niveaux de titres hiérarchiques, de "h1" (titre de plus haut niveau) à "h6" (titre de plus petit niveau).
2. "p" et "pre" : ces deux éléments définissent des paragraphes. La particularité de "pre" est qu'il considère le texte *tel qu'il est écrit* : notamment, les sauts de lignes, les espaces, les tabulations et autres caractères d'espacement sont *tous* considérés (tandis qu'en xhtml de manière générale, toute suite non vide de caractères d'espacement est considéré comme un seul caractère d'espacement).
3. "br" et "hr" : "br" opère un saut de ligne, tandis que "hr" trace une ligne horizontale.

4.3 Éléments génériques de regroupement "div" et "span"

Ces deux éléments permettent de regrouper des éléments et des bouts de texte (*ex.* : une section est un ensemble de paragraphes). En l'absence de définition de valeurs d'attributs spécifiques, l'utilisation des éléments "div" (regroupement de niveau bloc) et "span" (regroupement de niveau texte) est parfaitement neutre.

4.3.1 Regroupement de niveau bloc : "div"

Exemple : définir (logiquement) une section regroupant différents paragraphes. Dans cet exemple, on donne un identifiant aux éléments "div" (ce n'est pas obligatoire évidemment, mais cela permet d'y faire référence, par exemple, pour la navigation).

```
<h2 style="text-align:left;">Formation du groupe</h2>
<div id="bio-formation" style="text-align:justify;">
   <p>blablabla</p>
   <p>blablabla</p>
</div>

<h2 style="text-align:left;">Premier album</h2>
<div id="bio-album-1" style="text-align:justify;">
   <p>blablabla</p>
</div>
```

4.3.2 Regroupement de niveau texte : "span"

Exemple : identifier les expressions exprimées dans une langue étrangère, les acronymes.

```
<h2 style="text-align:left;">Premier album</h2>
<div id="bio-formation" style="text-align:justify;">
   <p>Les <span class="acronyme">RHCP</span> sortent alors leur
   album <span class="gb">OOne Hot Minute</span> qui fut bla bla bla...</p>
</div>
```

L'attribut "class" permet d'attribuer une classe à un élément. Contrairement aux identifiants (attribut "id") qui doivent être uniques, il n'y a aucune restriction sur l'utilisation des classes.

5 Attribut générique style

L'attribut "style" permet de spécifier la valeur de toute propriété de mise en forme et est défini pour (quasi) tous les éléments de contenu xhtml. On s'intéresse pour l'instant à la taille de police, à la couleur de police, à la couleur de fond, et à l'alignement horizontal.

```
<p style= "color:#FF0000;">je vois rouge</p>
<p style= "font-size:6; text-align:center;">je vois grand et au centre</p>
<p style= "color:#FF0000; background-color:#0000FF; font-size:6;">je vois rouge et grand sur fond bleu</p>
```

La valeur que l'on affecte à l'attribut "style" est un peu particulière, puisque l'on peut lister un ensemble de valeurs pour différentes propriétés de style. La syntaxe est la suivante :

<élément style="propriété$_1$:valeur$_1$; propriété$_2$:valeur$_2$; ... propriété$_\ell$:valeur$_\ell$; ">...</élément>

Ainsi, la valeur de l'attribut est une chaîne de caractères qui liste un ensemble de couples *propriété:valeur;*. Pour l'instant, on se restrendra aux propriétés "color" (couleur de police), "font-size" (taille de police), "background-color" (couleur de fond) et "text-align" (alignement du texte).

5.1 Taille de police

On peut définir des tailles absolues (7 tailles sont prédéfinies) ou relatives (par rapport à *l'élément contenant*) de police :

type de valeur	**valeurs admises**	**exemple**
relative	smaller, larger	plus petit
absolue	xx-small, x-small, small, medium large, x-large, xx-large	petit

5.2 Couleurs

En XHTML, les couleurs sont définies en hexadécimal selon le codage RRVVBB "Rouge Vert Bleu" : les deux premier caractères donnent la composante rouge (de RR=00, c'est-à-dire pas de rouge à RR=FF=$15\times16^1+15\times16^0$=255, c'est-à-dire un maximum de rouge), les deux suivants la composante verte, les deux derniers la composante bleue. Par exemple, FF0000 désigne le rouge pur, 00FF00 le vert pur et 0000FF le bleu pur. Le gris est un mélange équitable des trois couleurs ; ainsi, pour obtenir du gris clair, on pourra par exemple prendre la valeur C0C0C0 (C0=$12\times16^1+0\times16^0$=192), et pour du gris foncé, la valeur 303030 (30=$3\times16^1+0\times16^0$=48). Pour le jaune, il faut mélanger du vert et du rouge : FFFF00 ; enfin, le blanc étant la combinaison des trois couleurs primaires, il est donné par FFFFFF. Pour affecter une valeur aux propriétés "color" et "background-color" de l'attribut "style", on peut, soit donner explicitement le nom de la couleur, soit en donner le codage RRVVBB (précédé alors par la caractère #) :

```
<p style="color:#FF0000;">Ce</p>          <p style="color:red;">Ce</p>
<p style="color:#00FF00;">n'est</p>       <p style="color:green;">n'est</p>
<p style="color:#0000FF;">pas</p>         <p style="color:blue;">pas</p>
<p style="color:#C0C0C0;">un</p>
<p style="color:#303030;">arc-en-ciel</p>
<p style="color:#FFFF00;">mais</p>        <p style="color:yellow;">mais</p>
<p style="color:#FFFFFF;">presque</p>     <p style="color:white;">presque</p>
```

5.3 Alignement horizontal

Pour tous les éléments de niveau bloc, on peut définir le type d'alignement de son contenu, via l'attribut "style" et la propriété "text-align" ; les valeurs possibles sont "left" (alignement à gauche), "right"(alignement à droite), "center" (alignement au centre) et "justify" (alignement justifié) :

```
<h1 style="text-align:center;">Biographie du groupe X</h1>
<h2 style="text-align:left;">Formation du groupe</h2>
<p style="text-align:justify;">blablabla</p>
<h2 style="text-align:left;">Premier album</h2>
<p style="text-align:justify;">blablabla</p>
<h2 style="text-align:left;">Second album</h2>
<p style="text-align:justify;">blablabla</p>
<p style="text-align:right;"><!-- barre de liens --></h2>
```