Introduction : HTML, XHTML et CSS

Concepts et Technologies XML

Année 2013-14

Un rapide historique

Les langages de balises : *SGML* (*Standard Generalized Markup Language*) date du début des années 70

Le web : création au CERN en 1989 par Tim Berners-Lee.

Objectif : mettre à disposition facilement des documents.
À l'origine, des travaux de recherche en physique qui intéressaient une communauté internationalement dispersée.

Un rapide historique

En utilisant un langage de description de documents : HTML (*HyperText Markup Language*)

- langage de balisage hérité de SGML
- liens entre des documents situés n'importe où sur le réseau
- diffusion de documents contenant du texte, des images, du son
- la mise en forme est interprétée par le navigateur

Basé sur un protocole : HTTP

Architecture client/serveur

- un serveur web : ou serveur HTTP-D processus qui attend des requêtes d'un client web ;
- un client web : ou navigateur, processus qui effectue les requêtes *HTTP* auprès d'un serveur.
- exécution sur le serveurs de traitements (scripts CGI, servlets...)
- ou sur le client
- généralisation : les web services

Uniform Resource Identifier

Localisation d'une ressource sur Internet :

- Uniform Resource Locator (URL)

```
protocole://nomHote[:port]/chemin/nomFichier[#part]
```

- Uniform Resource Name (URN) tous les autres moyens d'identifier une ressource sans préciser son emplacement

Les protocoles

- file accès à un fichier du disque local
- ftp accès à un serveur ftp
- http accès à un serveur http
- https accès à un serveur http avec une liaison sécurisée
- mailto envoi d'un message
- news accès à un forum Usenet
- telnet connexion vers un service telnet
- ...

Nécessité d'établir des normes

Le W3C (World Wide Web Consortium) `www.w3.org` a été fondé en 1994 par Tim Berners-Lee.

Son objectif est de rédiger des recommandations pour la spécification des langages, des services, des protocoles liés au Web.

Dans les années 90, guerre entre les navigateurs : ajouts de spécificités au langage HTML.

Site optimisé pour ...

Avoir du code normalisé :

être indépendant de la plateforme du client !

Les principes du W3C :

- Web for All
- Web on Everything

Le W3C

L'équipe : une soixantaine de chercheurs et d'ingénieurs, pour la plupart dans l'un de ces trois centres

- le MIT (Massachusset's Institute of Technology)
- l'ERCIM (Consortium Européen de Recherche en Informatique et Automatique)
- Keio University (Japon)

Les membres : 383 membres actuellement

- des universités
- des centres de recherche
- des sociétés privées (AT&T, Google, IBM, ILOG, MicroSoft, Mozilla, Nokia, SUN, etc...)

Le W3C

Le W3C édite des recommandations, qui sont des spécifications, i.e. des normes pour tous les protocoles et technologies du Web afin d'assurer leur interopérabilité et de guider l'évolution du web.

Lorsqu'une recommandation est éditée, après un long processus, c'est qu'un consensus a été trouvé entre tous les membres du W3C.

Outre les recommandations, le w3c propose de la documentation, des tutoriaux, des validateurs...

http://www.w3.org

Contenu du cours

- XHTML et les CSS
- Javascript, Ajax, JQuery, ...
- XML et DTD
- XPath
- XSLT
- schémas XML
- web sémantique
- les API de programmation XML (SAX, DOM)
- XQuery

Les limitations du HTML

- à l'origine, HTML décrit la structure du document
- mais dérive vers un langage d'apparence (codage de la mise en forme *en dur*)
- permissivité du HTML
- interprétation différente selon les navigateurs
- pas de possibilité de sorties spécialisées (impression, braille, etc)

XML

XML : eXtensible Markup Language

ensemble de règles qui permettent la création de langages de balisage personnalisés ou l'utilisation de langages de balisage existants.

Nombreuses technologies autour de XML.

Pour l'affichage de documents : les feuilles de style CSS Cascading Style Sheets

XHTML

XHTML : version de HTML qui hérite de XML

- traduction de HTML 4.0 compatible avec XML
- cadre strict du XML : meilleur comportement
- permet d'accéder aux extensions de XML.

XHTML 1.0 existe en trois versions :

- Strict XHTML : élimine de nombreux éléments de mise en forme du HTML. Utilisation d'une CSS pour obtenir la mise en forme voulue ;
- Transitional XHTML : conserve les éléments et attributs de HTML. Garantit la compatibilité avec les anciens navigateurs.
- Frameset XHTML : idem que strict avec la possibilité d'utiliser des cadres.

HTML 5

- en discussion depuis 2004, toujours en statut *draft* ! Certains modules ont un statut *Candidate Recommendation*.
- objectifs :
  - fournir une syntaxe HTML et une syntaxe XHTML
  - améliorer les balises du langage
  - préciser les modèles de traitement des documents HTML pour favoriser l'interopérabilité des applications
  - introduire de nouvelles balises et des APIs pour les nouvelles applications Web (meilleure prise en compte du web dynamique)

Les validateurs

Services offerts par le W3C :

- validateurs XHTML et HTML
- validateur de CSS

XHTML et HTML5

Éléments syntaxiques spécifiques XML :

- balises en minuscules ;
  ce n'est pas obligatoire en HTML, mais c'est plus joli !
- toutes les balises doivent comporter une balise de fin *</p>*, *
*
  ce n'est pas obligatoire en HTML, mais c'est une règle à conserver
- les balises vides doivent se finir par un marqueur
  ce n'est pas le cas en HTML !
- toutes les valeurs d'attributs doivent être quotées XML : *<input name="unNom" required="required" />*
  ce n'est pas toujours le cas en HTML : HTML : `<input name="unNom" required>`

XHTML et HTML 5

- Pour créer un document XHTML 1.1
  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xht
- Pour créer un document HTML 5
  <!DOCTYPE html>

XHTML et HTML 5

- Structure : *html, head, body*
- Éléments de l'entête
  - Metadata : *meta, title*
  - Liaisons : *base, link*
  - Scripts côté client : *script*
  - Styles : *style*

Éléments de l'entête

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <title>Ceci est un exemple de page
    </title>
    <meta name="keywords"
          content="css, cascading style sheets">
    <meta charset="utf-8" />
    <link rel="stylesheet" type="text/css"
          href="./style.css" />
  </head>
  ...
</html>
```

Éléments du corps

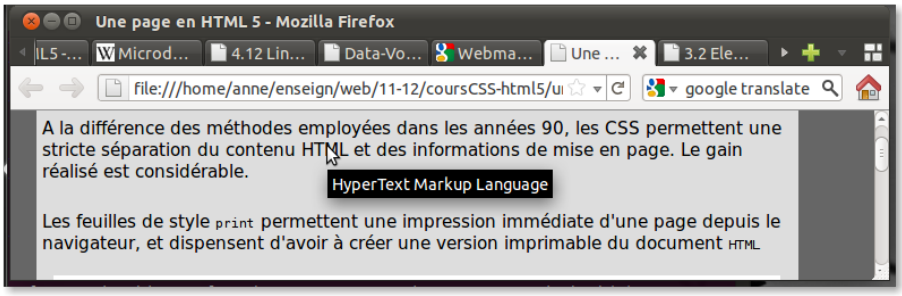
- Éléments de type blocs
  - de structure : *h1-h6, p, div*
  - d'expression : *address, blockquote, pre*
  - de présentation : *center, hr*
- Éléments de type en ligne
  - de structure : *em, q, strong, span*
  - d'expression : *abbr, cite, code, dfn, del, ins, kbd, samp, var, img*

Éléments blocs, Éléments en ligne

```
<p>A la différence des méthodes employées dans
les années 90, les <span title="Cascading StyleSheet">CSS</span>
permettent une stricte séparation du contenu
<span title="HyperText Markup Language">HTML</span>
et des informations de mise en page. Le gain réalisé
est considérable.</p>

<p>
Les feuilles de style <code class="token">print</code>
permettent une impression immédiate d'une page depuis
le navigateur, et dispensent d'avoir à créer une version
imprimable du document
<code class="token">HTML</code> </p>
```

Éléments blocs, Éléments en ligne



XHTML et HTML 5

- Liens : *a*
- Listes : *dl, dt, dd, ol, ul, li*
- Tables : *table, td, th, tr*, ...
- Formulaires : *form, input, select, option, textarea*, ...
- Images : *img, area, map*
- Scripts côté client : *noscript, script*
- Événements : *onclick, ondblclick, onmousedoum, onmouseup, onmouseover, onmousemove, onmouseout, onkeypress, onkeydown, onkeyup*
- ...

Les nouveautés HTML 5

- attention : HTML5 n'est toujours pas un standard du web, beaucoup d'évolutions (et aussi de retours en arrière) diversement reconnues suivant les navigateurs ...
- un gros ajout de HTML5 est le graphique avec l'élément *canvas*
- les balises associées à de la mise en forme ne sont plus autorisées : *big, center, s, tt, u, font*, ...
- mais
  - on a toujours *strong*, *emph*
  - *mark* permet de surligner des morceaux de texte
  - *wbr* permet de suggérer une coupure de mot au navigateur

Les nouveautés HTML 5

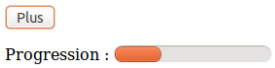
- arrivée de nouveaux petits éléments graphiques
  - une jauge *meter* (pour usage statique)

```
<p><meter id="m1" min="0" max="18" value="13"></meter></p>
<p><meter id="m2" value="0.5"></meter></p>
```

  - et une barre de progression (pour usage dynamique)

```
<form onsubmit="augmente()">
 <input type="button" value="Plus"
        onclick="p.value=p.value+10">
```

```
</form>
<p>Progression :
 <progress id="p" max=100 value=10></progress>
</p>
```



Les nouveautés HTML 5

- nouvelles balises pour la structure du document :
  - *article* et *section* :
    - permettent de regrouper des informations textuelles, telles que des paragraphes, des titres, d'autres *section*
    - *article* doit contenir une partie de texte indépendante du reste de la page ;
    - les *h1*,... *h6* qui s'y trouvent sont interprétés en fonction du contexte ;
  - *aside* permet de rassembler des éléments (paragraphes, titres) qui sont fortement liés entre eux mais restent connectés aux éléments environnants ;
  - *header*, *footer* et *nav* :
    - *header* permet de contenir des éléments introductifs à la page, ou des menus de navigation
    - *footer* contient des éléments de fin de page ou de section. Peut contenir des menus de navigation
    - *nav* contient des menus de navigation

Les nouveautés HTML 5

- *figure* et *figcaption*
  - *figure* pour encapsuler les illustrations (images, vidéos, cartes...)
  - *figcaption* pour ajouter une légende aux illustrations
- de nouveaux éléments permettent une prise en charge du multimédia :
  - éléments *video* et *audio*
    ```
    <video src="jjrBis.webm" controls width="100" height="80">
    ```
  - *svg* pour inclure des dessins vectoriels
    ```
    <svg> <circle r="50" cx="50" cy="50" fill="yellow"/> </svg>
    ```

Les nouveautés HTML 5

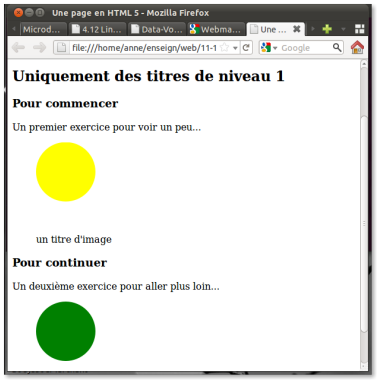
```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Une page en HTML 5</title>
  </head>
  <body>
    <header>
```

```
      <nav id="menuDuHaut">
        <ul> <li>un lien</li> <li>un autre lien</li> </ul>
      </nav>
      <nav id="menuDeGauche">
        <ul> <li>encore un lien</li>
             <li>encore un autre lien</li> </ul>
      </nav>
    </header>
```

Les nouveautés HTML 5

```
<div id="contenuPrincipal">
  <article class="theme">
    <h1>Uniquement des titres de niveau 1</h1>
    <section class="exercice dur">
      <h1>Pour commencer</h1>
      <p>Un premier exercice pour voir un peu...</p>
      <figure>
        <svg> <circle r="50" cx="50" cy="50" fill="yellow"/> </svg>
        <figcaption>un titre d'image</figcaption>
      </figure>
    </section>
    <section class="exercice facile">
      <h1>Pour continuer</h1>
      <p>Un deuxième exercice pour aller plus loin...</p>
    </section>
  </article>
</div>
```

Les nouveautés HTML 5



Les nouveautés HTML 5

Pour les formulaires :

- l'élément *output* : comme un label, une zone non ouverte à la saisie pour l'utilisateur, mais dont le contenu est lié à des valeurs et à des changements dans les *input*
- l'élément *input* a gagné de nouvelles valeurs pour son attribut *type* et de nouveaux attributs : comme *max*, *min*, *step*, *required*, *pattern* ... Pour l'attribut `type`, quelques exemples :

```
<input id="a2" name="a2" type="number"
       step="2" value="2">
```


```
0<input type="range" id="a3" value="40"
       min="0" max="100">100
```

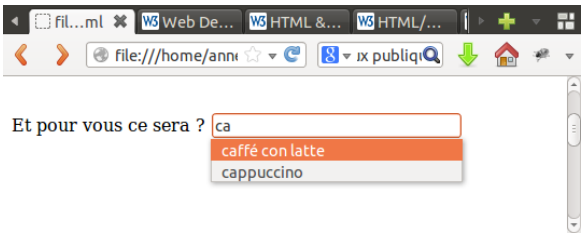
Les nouveautés HTML 5

```
<form>
  <input id="a" name="a" type="number" step="2"> +
  <input id="b" name="b" type="number" step="any"> =
  <output name="x" for="a b"
     oninput="value = parseInt(a.value) + parseInt(b.value)">
  </output>
</form>
```

Les nouveautés HTML 5

- Enfin l'élément *datalist* permet de faire de l'auto-complétion pour un élément *list* de type *list*

```
<form>
Et pour vous ce sera ? <input type="list" list="maListe" />
<datalist id="maListe">
<option value="caffé con latte">
<option value="cappuccino">
<option value="chocolat chaud">
<option value="thé">
</datalist>
</form>
```



CSS : déclaration

Pour accéder à une feuille de style, dans l'entête du document :

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

On peut ajouter un attribut `media` pour choisir la feuille de style selon le média d'accès

```
<link rel="stylesheet" href="maFeuilleDeStyleImpres.css"
      media="print" type="text/css"/>
```

```
<link rel="stylesheet" href="maFeuilleDeStyleEcran.css"
      media="screen" type="text/css" />
```

CSS : déclaration

Valeurs possibles pour l'attribut `media` :

screen : par défaut ; *tty* ; *tv* ; *projection* ; *handheld* ; *print* ; *braille* ; *aural* ; *all*

Le choix du média peut être également effectué à l'intérieur de la feuille de style :

```
@media print {
  body { font-size: 10pt }
}

@media screen {
  body { font-size: 12pt }
}
```

CSS : déclaration

Le choix de la mise en forme peut être laissé à l'utilisateur par la proposition des feuilles de styles alternatives.

```
<link rel="stylesheet" title="normale" href="normaleScreen.css"
      media="screen" type="text/css"/>
<link rel="stylesheet" title="normale" href="normaleImpression.css"
      media="print" type="text/css"/>
<link rel="alternate stylesheet" title="Une css originale"
      href="originaleImpression.css" media="print" type="text/css"/>
<link rel="alternate stylesheet" title="Une css originale"
      href="originaleScreen.css" media="screen" type="text/css"/>
<link rel="alternate stylesheet" title="Une autre css"
      href="differente.css" type="text/css"/>
```

CSS : principe de fonctionnement

Une feuille de style CSS est

- une collection de règles
- une règle : *selecteur* { *propriétés*}
- une propriété : *nom_propriété : valeur ;*

CSS : principe de fonctionnement

```
h4 {
  text-align: left;
  font-weight:bold;
  font-variant: small-caps;
  font-size: 100%;
  margin-top: 1em;
  margin-bottom: 0.5em;
}

h5 {
  text-align: left;
  text-decoration: underline;
}
```

CSS : principe de fonctionnement

- plusieurs règles peuvent s'appliquer sur un même élément : algorithme de résolution de conflits
- héritage de propriétés : *Cascading* style sheets !

Limite de CSS niveau 2

- pas de modification de l'ordre des éléments

– pas de génération de table des matières ...
– pas de tests logiques
– sélecteurs limités aux noms d'éléments, attributs et contexte

CSS : les sélecteurs

`nom-element [attr = "valeur"]`

Le plus simple : le sélecteur `*`

```
* {
    color: blue;
}
```

`monelt` sélectionne tous les éléments de nom `monelt`.

CSS : les sélecteurs

`monelt[monattr]` sélectionne tous les éléments de nom `monelt` qui possèdent un attribut `monattr`.

`monelt[monattr="mavaleur"]` sélectionne tous les éléments de nom `monelt` qui possèdent un attribut `monattr` dont la valeur a été fixée à `"mavaleur"`.

`monelt[monattr1="mavaleur1"][monattr2="mavaleur2"]` sélectionne tous les éléments de nom `monelt` qui vérifient tous les critères précisés.

`monelt[monattr~="mavaleur"]` sélectionne tous les éléments de nom `monelt` qui possèdent un attribut `monattr` dont la valeur contient `"mavaleur"`.

CSS : les sélecteurs

Le plus classique : `p[class="uneClasse"]`
équivalent à `p.uneClasse`
et pour sélectionner tous les éléments d'une certaine classe : `.uneClasse`

document XHTML

```
<h3 class="exemple">Un titre</h3>
<p class="exemple">blablabla</p>
```

et feuille de style

```
p.exemple { color : red;}
.exemple { background-color : #555;}
```

CSS : les sélecteurs

Pour une mise en forme d'un élément unique
`#moneltunique`
s'appliquera à l'élément qui possède un attribut `id` dont la valeur est `"moneltunique"`
Pour appliquer une même règle à des éléments différents
`elt1, elt2, elt3`
Exemple :

```
h1, h2, h3, h4, h5, h6 { font-weight : bold;}
```

CSS : sélection contextuelle

On peut exploiter des informations contextuelles pour appliquer une règle :
- ascendance de l'élément
- voisins de même niveau

CSS : sélection contextuelle

`td > p` : un paragraphe qui est un descendant direct d'une cellule de tableau

```
ul {indent: 3em;}
ul > ul {indent: 6em;}
ul > ul > ul {indent: 9em;}
```

`table p` : un paragraphe qui est un descendant d'un tableau, quelque soit la profondeur

CSS : sélection contextuelle

`li:first-child` : sélectionne le premier élément d'une liste

`p:first-line` : sélectionne la première ligne d'un paragraphe (attention ! zone variable)

`body > p:first-child:first-letter` : pour la première lettre du premier paragraphe...

```
body > p:first-child:first-letter {
   font-size: 300%;
   font-color: red;
}
```

`h1 + p` : sélectionne un paragraphe immédiatement précédé d'un titre de niveau 1

`h1 ~ p` : sélectionne tous les paragraphes précédés à un même niveau d'un titre de niveau 1

CSS : les pseudo-classes

`:before :after` : sélectionne un point juste avant ou juste après l'élément

```
body > *:first-child:before {
     content: "Il était une fois ";
     font-weight: bold;
}
```

mais aussi
a:link, *a:hover*, *a:active*, *a:visited*

CSS : choix des règles

Principe général : toutes les règles s'appliquent. En cas de conflit, les sélecteurs les plus spécifiques ont la priorité. Les conflits sont réglés pour chaque valeur de propriété.

1. les sélecteurs de type *id* ont la priorité la plus haute
2. le nombre de sélecteurs d'attributs et de pseudo-classes est important
   **[class="uneclasse"][attr="unevaleur"]* est plus spécifique que *p:first-child*
3. on privilégie les descriptions généalogiques les plus précises (hors pseudo-éléments)
4. s'il reste un conflit, c'est la dernière règle spécifiée qui l'emporte

CSS : héritage de propriétés

- les éléments héritent les propriétés de leurs ancêtres
- *body* est l'endroit idéal pour placer les valeurs par défaut
- certaines propriétés restent cependant ignorées (*background-image*)

CSS : les principales propriétés

Plus de 120 propriétés dans CSS2 !

Les unités de mesure

- *les mesures absolues* : *mm*, *cm*, *in* et les mesures typographiques *pt* et *pc*
- *les mesures relatives* :
  - *em* : taille de la police courante
  - *ex* : hauteur de *x* dans la police courante
  - les pourcentages

CSS : les principales propriétés

Propriétés de texte

- *famille de police* : *font-family*. Valeur : *serif, sans-serif, monospace, cursive, fantasy* pour les classes génériques, ou noms de police (à indiquer du plus spécifique au moins spécifique)
- *taille de police* : *font-size*. Valeur : mesure ou mots-clés *xx-small, x-small, small, medium, large, x-large, xx-large* mais aussi *smaller* et *larger*
- *style et graisse* : *font-style*. Valeur : *normal, italic, oblique, inherit* ; mais aussi *font-weight*. Valeur : *light, normal, bold, lighter, bolder*
- *couleur* : attributs *color* et *background-color*. Valeur prédéfinie ou hexadécimale.

CSS : les principales propriétés

Propriétés de texte : alignement et indentation

- *alignement* : *text-align*. Valeur : *left, right, center, justify*
- *indentation* : *text-indent* pour la première ligne d'un bloc. Peut avoir une valeur négative.

CSS : les principales propriétés

Les propriétés des blocs

- *les marges* : *margin-left*, *margin-right*, *margin-top*, *margin-bottom*. Valeur : une longueur ou un pourcentage de la largeur de l'élt conteneur ;
- *les bordures* : (à l'intérieur des marges) *border*
  - largeur *thin, medium, thick*
  - style *solid, dashed, dotted, double, ...*
  - couleur *blue, green, #12bc3f, ...*
- *les espacements* : pour séparer le texte de la bordure à l'intérieur *padding*
- *la largeur* : *width*. Elle correspond au contenant **plus les marges, bordures et espacements gauches et droits**.

CSS : positionnement des éléments de type bloc

- *flux normal* : les blocs sont positionnés dans un flux vertical. Les éléments en-ligne sont positionnés dans un flux de gauche à droite et de haut en bas.
- *sortir du flux* : *float* peut prendre les valeurs *left*, *right*, *inherit* ou *none* (valeur par défaut).

CSS : positionnement des éléments de type bloc

- le placement vertical n'obéit pas aux mêmes règles.
- pour éviter des débordements : *clear* empêche une cohabitation horizontale avec un bloc flottant. Valeurs possibles : *left*, *right*, *both*, *none*, *inherit* (par défaut, *none*)

CSS : positionnement des éléments de type bloc

- modifier la position d'un bloc : *position* peut prendre les valeurs *static*, *relative*, *absolute*, *fixed* (par défaut, *static*)
- visibilité d'un élément : *visibility* peut prendre les valeurs *visible* (par défaut) ou *hidden*, ou alors propriété *display* qui peut prendre la valeur *none*.

CSS : les principales propriétés

et d'autres...

- pour gérer les tables
- pour gérer les listes
- pour insérer du texte
- pour obtenir un compteur (de titre, de liste, ...)

```
body {
    counter-reset : exercice;
 }
.exercice:before{
    counter-increment : exercice;
    content : "Exercice " counter(exercice) " : ";
    font-weight : bold;
```

```
    }
- ...
```