HTML/XHTML/XML

Olivier Aubert

Liens

- http://www.brics.dk/~amoeller/XML/overview.html
- http://www.w3.org/TR/xhtml1/#guidelines : annexe C, compatibility guidelines
- http://www710.univ-lyon1.fr/~exco/COURS/COURS/HTML_D

Qu'est-ce que HTML ?

- HyperText Markup Language
- Qu'est ce que l'hypertexte ?
  - un document qui contient des liens vers d'autres documents (texte, sons, images, etc)
  - les liens peuvent être activés automatiquement ou à la demande
- Qu'est-ce qu'un langage de *markup* ?
  - une notation pour écrire du texte avec des balises (*tags*)
  - les balises indiquent la structure du texte
  - les balises ont des noms et des attributs
  - les balises peuvent contenir une partie de texte

Motivation originale de HTML

- Échanger des données sur Internet
  - Les données sont publiées par des serveurs
  - Les données sont présentées par les clients (navigateurs)
- HTML a été créé par Tim Berners-Lee et Robert Caillau au CERN en 1991
- Objectif : diffuser les résultats des expériences menées
- HTML décrit la structure des documents
  - Les navigateurs peuvent interpréter les balises de différentes manières
  - Le document reste compréhensible même si certaines balises sont ignorées
- HTML a combiné plusieurs idées
  - Le concept d'hypertexte est connu depuis 1945 (Vannevar Bush)
  - Le concept de langage à balises depuis 1970 (SGML)

Un format compact et lisible

- La plupart des formats de documents sont gourmands en taille
  - l'auteur contrôle précisément la mise en page
  - tous les détails, y compris les descriptions des caractères, sont inclus
- En comparaison, HTML est économe
  - l'auteur sacrifie le contrôle pour la taille
  - on ne représente que le contenu et la structure logique
- Conséquences : économies de stockage, économie de coût de transfert
- Les documents HTML peuvent être édités et modifiés par n'importe quel éditeur texte

Structure logique vs physique

- À l'origine, HTML ne décrivait que la structure
  - `h2` : ceci est un titre de niveau 2
  - `em` : ce texte doit être mis en évidence
  - `ul` : ceci est une liste non ordonnées d'éléments
- Rapidement, les utilisateurs (surtout non-scientifiques) ont voulu plus de contrôle
  - Le titre doit être centré et écrit en Times-Roman en 28pt
  - Le texte doit être mis en italique
- Ceci a conduit à l'intégration de balises décrivant la structure physique du document

Feuilles de style

- *Cascading Style Sheets* (CSS)
  - spécifient les propriétés physiques des balises HTML
  - sont généralement stockées dans un fichier séparé
  - peuvent être utilisées par plusieurs documents
- Avantages
  - Les propriétés logiques et physiques sont séparées
  - Des groupes de documents peuvent partager une apparence commune
  - L'apparence des documents peut facilement être modifiée

Feuilles de style (II)

- Principes
  - Permet de définir plus de 50 propriétés pour chaque balise
  - Les définitions pour une balise peuvent varier suivant son contexte
  - Les propriétés non définies sont héritées des balises qui contiennent la balise courante
- Les feuilles de style permettent de conserver le caractère logique des balises.
- Cependant, seules des propriétés superficielles sont définissables

Exemple

Feuille de style : ensemble de sélecteurs et de propriétés

```
B {color:red;}
B B {color:blue;}
B.foo {color:green;}
B B.foo {color:yellow;}
B.bar {color:maroon;}
```

Définition de sélecteurs spécifiques dans HTML

```
<b class=foo>Hey!</b> <b>Wow!!
   <b>Amazing!!!</b> <b class=foo>Impressive!!!!</b>
<b class=bar>k00l!!!!!</b>
<i>Fantastic!!!!!!</i>
</b>
```

Hey! Wow!! Amazing!!! Impressive!!!! k00l!!!!! *Fantastic!!!!!!*

Les versions de HTML

- 1992 : Première définition de HTML
- 1993 : HTML+ (quelques éléments physiques, formulaires, tableaux)
- 1994 : HTML2.0 (standard pour les éléments principaux)
  HTML3.0 (extension de HTML+ soumise en tant que *draft*)
- 1995 : balises non-standard spécifiques à Netscape
- 1996 : guerre des navigateurs Netscape/Internet Explorer
  HTML3.2 : standard basé sur les pratiques existantes
- 1997 : HTML4.0 (séparation de la structure et de la présentation avec les feuilles de style)
- 1999 : HTML4.01 (modifications mineures)
- 2000 : XHTML1.0 (version XML de HTML4.01)

Syntaxe et validation

- HTML4.01 définit une syntaxe précise et formelle.
- Tout document HTML devrait respecter cette syntaxe
- Elle peut être validée automatiquement
- Dans les faits, la plupart des documents HTML ne sont pas valides
  - Les auteurs ne font pas attention
  - La seule validation des documents est leur test dans un navigateur
  - Le HTML généré par les éditeurs automatiques est souvent invalide

Syntaxe et validation (II)

- Le résultat est toutefois acceptable pour les utilisateurs
  - Les navigateurs font de leur mieux pour tolérer les erreurs
  - Les erreurs de syntaxe ne sont jamais signalées
- Problèmes
  - Promotion du HTML incorrect
  - Différences de rendu entre différents navigateurs
  - Il est difficile d'exploiter les documents de manière automatique

Problèmes de HTML

- Le langage est conçu pour représenter de l'hypertexte (ensemble limité de balises pas forcément adaptées)
- La syntaxe et la sémantique sont confondues
  - La structuration des données impose une certaine représentation
  - Les feuilles de style ne présentent qu'une solution limitée
  - Des vues différentes ne sont pas supportées
- Les standards ne sont pas respectés
  - La plupart des documents HTML sont invalides
  - Les navigateurs ont défini leurs propres extensions non-standard

XML

- XML = *eXtensible Markup Language*
- XML est un *framework* permettant de définir des langages à balises
  - Pas d'ensemble fixé de balises, elles sont adaptées au type d'information à représenter
  - Chaque langage XML est destiné à une utilisation particulière, mais tous les langages partagent de nombreuses fonctionnalités
  - Un seul ensemble d'outils génériques pour traiter les documents
- XML n'est pas un remplacement de HTML
  - HTML (XHTML) n'est qu'un langage défini dans la syntaxe XML
  - XHTML est un langage XML (très populaire) dédié aux documents hypertexte

Objectifs d'XML

XML est conçu pour

- Séparer la syntaxe de la sémantique, afin de proposer un cadre unique de structuration de l'information (le rendu est défini par des feuilles de style)
- Permettre de spécifier des balises spécifiques pour tout domaine d'application
- Permettre l'internationalisation des documents (Unicode)
- Être indépendant des plate-formes utilisées

Outils associés

- *XML Schema*
- *XSLT* permet de transformer un document XML en XHTML (ou HTML), avec éventuellement construction automatiques d'un sommaire, d'un index, ...
- *XLink*, *XPointer* et *XPath* permettent de définir des références croisées
- *XQuery* permet d'exprimer des requêtes

Vue conceptuelle de XML

- Un document XML est un arbre ordonné et labellisé
- Les feuilles de l'arbre (données) contiennent les données effectives (chaînes de caractères). Les nœuds de données doivent être non-vides et non-adjacents à d'autres nœuds de données.
- Les nœuds éléments sont nommés par
  - un nom (souvent appelé *type*)
  - un ensemble d'attributs (couples nom/valeur)
  - Il peuvent avoir des nœuds fils
- Un arbre XML peut contenir d'autres types de feuilles
  - Des instructions de traitement (spécifiques à certains outils ou formats)
  - Des commentaires
  - Des déclarations de type de document

Vue concrète de XML

- Document XML = texte Unicode avec des balises et d'autres méta-informations
- Format des balises (sensibles à la casse) :

```
...<foo attr="val" ...>...</foo>
...<foo attr="val" .../> (raccourci pour les lments
```

- Un document XML doit être conforme (*well-formed*)
  - Les balises de début et de fin doivent correspondre
  - Les éléments doivent être convenablement imbriqués
  - Les valeurs des attributs doivent être entre guillemets
- Autres méta-informations
  - `<?target data...?>` Instruction de traitement spécifique à un outil.*target* identifie l'outil concerné.
  - `<!-- commentaire -->` Commentaire ignoré par les outils de traitement
  - `<!DOCTYPE ...>` Déclaration du type du document

De SGML à XML

- SGML (Standard Generalized Markup Language)
  - Standard ISO (1985)
  - De nombreuses applications d'archivage dans les domaines gouvernementaux, industriels, militaires, académiques...
  - Une application connue : HTML
- XML (eXtensible Markup Language)
  - Recommandation du W3C (1998)
  - Un sous-ensemble de SGML, orienté vers les applications Web
- Canonical XML
  - Recommandation du W3C (2001)
  - Simplification des documents XML généraux (pas de la norme XML)
  - Représentation *canonique* des documents
  - Supprime les déclarations de type, impose un ordre sur les attributs, etc

Fonctionnalités

- Mécanismes d'extensions communes au noyau de la spécification XML : espaces de nommage, inclusion de documents, etc
- Schémas : grammaires définissant des classes de documents
- Liens entre documents : généralisation du système d'ancres et de liens de HTML
- Adressage de parties de documents existants : pointeurs robustes et flexibles permettant de spécifier des emplacements
- Transformations : Conversion d'un format de document vers un autre
- Interrogation : extraction d'information, généralisation des bases de données relationnelles

Technologies associées

- *XML Information Set* : définition d'une terminologie commune aux concepts de XML
- *XML Signature* : signature numérique des ressources
- *XML Encryption* : chiffrement des ressources
- *XML Fragment Interchange* : manipulation de fragments de documents XML
- *XML Protocol*, SOAP (*Simple Object Access Protocol*) : protocoles d'échange d'informations
- *XForms* : un sous-langage commun pour les formulaires (les formulaires XHTML en sont un cas particulier)
- RDF *Resource Description Framework* : un cadre de description des métadonnées (définition de propriétés et de leurs relations)

Compatibilité HTML

- Quelques règles de base pour écrire du XHTML qui sera lisible par les anciens navigateurs HTML.
- Source : http://www.w3.org/TR/xhtml1/#guidelines

Compatibilité HTML (II)

- *Instructions de traitement* (<?target ...?>). Elles peuvent être visualisées sur certains navigateurs. De plus, si la déclaration XML n'est pas précisée dans le document, on ne peut utiliser que le jeu de caractères par défaut UTF-8 ou UTF-16
- *Éléments vides* Inclure un espace avant le *slash* final : <hr />,
, . Ne pas utiliser la syntaxe non minimisée (
</br>).
- *Minimisation des éléments vides*. Si un élément dont le modèle n'est pas EMPTY est vide, ne pas utiliser la forme minimisée : on utilisera <p> </p> et non <p />.
- *Feuilles de style et scripts intégrés*. Utilisez des feuilles de style ou des scripts externes s'ils utilisent les caractères <, &, > ou --. Ne pas utiliser la méthode historique de cacher les feuilles de style ou les scripts à l'intérieur de commentaires, qui peuvent être supprimés par les parsers XML.

Compatibilité HTML (III)

- *Sauts de lignes dans les valeurs d'attributs*. Ne pas utiliser de saut de ligne ou d'espaces multiples dans les valeurs des attributs. Tous les navigateurs ne les gèrent pas de la même façon.
- *Attributs de langage*. Utiliser à la fois les attributs lang et xml:lang pour spécifier le langage d'un élément. La valeur de xml:lang est prise en compte en priorité.
- *Identificateurs de fragments*. En XML, les URI qui se terminent par des identificateurs tels que #foo ne se rapportent pas aux éléments dont l'attribut name="foo", mais à ceux dont l'attribut id="foo". Pour assurer la compatibilité, préciser les deux attributs.
- *Encodage des caractères*. Pour spécifier l'encodage des caractères, utiliser les deux spécifications XML et HTML :

```
<?xml version="1.0" encoding="EUC-JP"?>
<meta http-equiv="Content-type"
     content='text/html; charset="EUC-JP"' />
```

Compatibilité HTML (IV)

- *Utilisation du* et-commercial *dans une valeur d'attribut*. Il faut l'exprimer comme une entité `&` (par exemple un attribut `href` vers un script avec des arguments)

Accessibilité `www.w3.org/WAI`

- *Images et animations*. Utilisez l'attribut `alt` pour décrire la fonction de chaque graphique.
- *Images cliquables*. Utilisez l'élément `map` et décrivez les zones actives.
- *Multimédia*. Fournissez légendes et transcriptions pour l'audio, et des descriptions pour les vidéos.
- *Liens hypertextes*. Utilisez des énoncés pertinents hors contexte. Par exemple, évitez ńcliquer iciż.
- *Organisation*. Utilisez des têtes de sections, des listes et une structure cohérente. Utilisez CSS si possible.
- *Figures et diagrammes*. Décrivez-les dans la page ou avec l'attribut `longdesc`.
- *Scripts, applets et plug-ins*. Fournissez une alternative quand le contenu actif est inaccessible ou non traité.
- *Cadres*. Utilisez l'élément `noframes` et des intitulés utiles.
- *Tableaux*. Facilitez la lecture ligne par ligne. Résumez.
- *Vérifiez votre travail*. Validez, vérifiez (`http://www.w3.org/TR/WAI-WEBCONTENT/full-checklist.`