Diffusion d'information multimédia
Chapitre 3 - XML
Master 2 SIMI

Sébastien Choplin
IUP MIAGe Amiens

2005

Un langage du web mixte : XML (*eXtensible Markup Language*)

Liens :

- http ://colas.nahaboo.net/cours/
- http ://www.u-picardie.fr/∼ferment/xml/
- http ://www.w3.org/ (W3C)
- http ://xmlfr.org/
- ...

XML : c'est quoi ?

Méta-langage permettant de définir des langages décrivant la structure des documents.

Très à la mode . . .

XML - Présentation - Pourquoi XML ?

Besoin d'échanger des documents entre personnes ou applications

Comment matérialiser ces données ?

avec HTML ?

- Assocation de présentation à un document difficile (CSS)
- Très peu de sens au document (isolation d'information difficile)
- Pas extensible

- Impossible d'échanger des documents ou des données
- Recherches difficiles.

XML - Présentation - avec SGML ?

Méta-langage apparu en 1986

- définir des langages à bases de tags
- beaucoup utilisé par les industriels pour les documentations techniques
- trop complexe pour être implanté dans des navigateurs.

XML - Présentation - Buts de XML

- permettre une utilisation simple sur internet,
- pouvoir s'appliquer à plusieurs domaines,
- être compatible avec SGML pour pouvoir continuer à utiliser les formats existants,
- être facilement manipulable par des programmes ou des scripts,
- avoir le moins d'options possibles
- être lisible par un humain et clair
- être disponible rapidement
- avoir une spécification formelle et concise
- être facile à créer, même à la main
- ne pas être concis mais explicite.

XML - Le format XML

1 - **Prologue**

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
```

2 - **DTD** (optionnelle, si standalone="yes")

3 - **Contenu**

- character data : mélange de texte et d'élements de markup
- elements : start-tag contents end-tag
- start-tag : `<element-name attributs>`
- end-tag : `</element-name>`
- attribut : `attribut-name="valeur d'attribut"` ou `attribut-name='attribute valeur'`

XML - Syntaxe

Un document XML peut être représenté sous la forme d'un **arbre** enraciné et doit donc avoir une et une seule **racine**, par exemple en HTML : `<html>...</html>`.

Syntaxe incorrecte :	Syntaxe correcte :
	<livres>
<livre>Dune</livre>	<livre>Dune</livre>
<livre>Dosadi</livre>	<livre>Dosadi</livre>
	</livres>

NB : lorsqu'il y a plusieurs mêmes éléments, l'ordre peut être important.

XML - Syntaxe

Un document XML doit être **bien parenthésé** :

Syntaxe correcte :

```
<body><p>paragraph</p></body>
```

Syntaxe incorrecte :

```
<body><p>paragraph</body></p>
```

Les **élements vides** peuvent être construits de deux manières :

```
<hr size='4'/> ou <hr size='4'><hr/>
```

XML - Syntaxe

Les **noms d'élément** ne doivent pas :
- commencer par un chiffre
- contenir d'espace
- commencer par `xml`

Les **character data** et les **valeurs d'attributs** peuvent contenir n'importe quoi sauf les caractères < et `&`.

XML - syntaxe

Il faut respecter **la casse** :
Syntaxe correcte :

```
<p>paragraph</p>
```

Syntaxe incorrecte :

```
<p>paragraph</P>
```

XML - Présentation - syntaxe

Tout **attribut** doit avoir une **valeur** (même vide) :
Syntaxe correcte :

```
<element compact=""/>
```

Syntaxe incorrecte :

```
<element compact>
```

Les **commentaires** doivent être délimiter par

```
<!-- ... -->
```

, ne doivent pas contenir - et peuvent être placé n'importe où en dehors des tags.

XML - Syntaxe

Un document XML est dit ***well-formed*** s'il respecte la syntaxe XML.

XML - Exemple de base

```
-----------------------------------------
<?xml version="1.0"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
-----------------------------------------
```

XML - Exemple de catalogue de CD

```
<?xml version="1.0"?>
<catalog>
  <cd>
    <!-- mes commentaires -->
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price currency="dollard" >10.90</price>
    <year>1985</year>
  </cd>
  <cd>
    <title>Hide your heart</title>
    <artist>Bonnie Tyler</artist>
    <country>UK</country>
```

```
    <company>CBS Records</company>
    <price currency="euros" >9.90</price>
    <year>1988</year>
  </cd>
  <cd>
    <title>Greatest Hits</title>
    <artist>Dolly Parton</artist>
    <country>USA</country>
    <company>RCA</company>
    <price currency="dollard" >9.90</price>
    <year>1982</year>
  </cd>
</catalog>
```

Outils

voir du xml : Mozilla, Internet Explorer, ...

éditer du xml : n'importe quel éditeur texte, XML NotePad, VisualXML,...

Structure physique

La structure physique d'un document XML peut (ce n'est pas obligatoire) être décrite dans une **DTD** ou un **Schema**.

DTD (*Document Type Definition*)

Elle décrit les **balises** et leur **contenus** (attributs,données ou autres balises).

Un document XML est dit ***well-formed*** s'il respecte la syntaxe XML.

Il est dit ***valide*** si en plus il respecte la grammaire optionnelle de la DTD.

Intérêts :
- échange de documents XML ayant la même structure,
- validation automatique

Mais : pas de typage des données

DTD - Exemple

```
---------------------------------------
<!ELEMENT note (to+, from,heading,body?) >
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
---------------------------------------
```

L'élément contient (dans l'ordre)

- au moins un destinataire (to+)
- exactement un expéditeur (from)
- exactement un entête (heading)
- 0 ou 1 message (body ?)

Un message est une donnée textuelle : body(#PCDATA)

PCDATA = Parsed Character Data

DTD - interne

La DTD peut être incorporée au document XML ou non (attribut *standalone* de la déclaration xml dans le prologue)

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE note [
<!ELEMENT note (to+, from,heading,body?) >
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
  <to>Bob</to>
  <from>Alice</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

DTD - externe

```
===== note.dtd ==============
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
============================
===== ma note.xml ===========
<?xml version="1.0" standalone="no"?>
<!DOCTYPE note SYSTEM "note.dtd">
  <to>Bob</to>
  <from>Alice</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
============================
```

Définition d'Attribut dans une DTD

```
<!ELEMENT note (to+, from,heading,body?) >
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>

<!ATTLIST note date CDATA #REQUIRED #IMPLIED >
<!ATTLIST note priority (normal|urgent) "normal">
```

L'attribut 'date' de l'élément 'note' est obligatoire et est une chaine de caractères constante (sans transformation XML).

L'attribut 'priority' est facultatif, choisi entre de 'normal' et 'urgent', mais par défaut 'normal'.

Attribut or not Attribut ?

```
<note date="20 janvier 2005" priority="urgent">
<to>Bob</to>
<from>Alice</from>
<heading>Hello</heading>
<body>Don’t forget our meeting</body>
</note>
```

pourrait être remplacé par :

```
<note to="Bob" from="Alice">
<date>20 janvier 2005</date>
<priority>urgent</priority>
<body heading="Hello">Don’t forget our meeting</body>
</note>
```

Entités

Document XML utilisant des entités

```
<?xml version='1.0' encoding='ISO-8859-1' standalone='no'

<!DOCTYPE note SYSTEM "note.dtd"[
<!ENTITY al "Alice">
]>
<note>
 ...
 <from>&al;</from>
 ...
</note>
```

Entités non XML et notations

```
<!NOTATION flash SYSTEM "/usr/bin/flash.exe">
```

déclaration de notation associant le format nommé “flash” l’application spécifiée

```
<!ENTITY animation SYSTEM "../anim.fla" NDATA flash>
```

déclaration de l’entité “animation” dont le contenu est le fichier de format (NDATA : notation data) non XML ...

```
<scene loc=’animation’ />
```

le "parser" ne verra que la valeur ’animation’, mais l’application flash pourra traiter le contenu et donc l’animation

Entités paramètres

utile pour composer une DTD à partir de plusieurs autres, dépasse le cadre de ce (bref) cours sur XML

Doctype

```
<!DOCTYPE élément-racine SYSTEM "URI-de-la-dtd">
```

l'URI Uniform Ressource Identifier est toujours une URL.

```
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG December  1999//EN"
  "http://www.w3.org/Graphics/SVG/SVG-19991203.dtd">
```

déclaration (dans un fichier SVG) de la DTD par FPI Formal Public Identifier (nom public) : si le FPI n'est pas connu ou trouvé, c'est l'URI qui suit qui est utilisée.

```
<!DOCTYPE élément-racine[
  déclaration  d'élément, entités, notations ...]>
```

```
<!DOCTYPE élément-racine SYSTEM "URI-de-la-dtd" [
  déclaration complémentaire d'élément,
  entités, notations ... ]>
```

Doctype pour un fichier HTML :

```
 <!DOCTYPE html public '-//W3C//DTD HTML 4.0//EN'
       'http://www.w3.org/TR/REC-html40/strict.dtd'>

   <html><head> ... </html>
```

Instructions de la DTD - déclaration d'élément : ELEMENT

```
<!ELEMENT element1 (element2, element3)>
```

element1 est composé de element2 suivi de element3

```
<!ELEMENT element1 (element2 | element3)>
```

element1 est composé de element2 ou de element3

```
<!ELEMENT element1 (element2)?>
```

element1 est composé de 0 ou 1 element2

```
<!ELEMENT element1 (element2)*>
```

element1 est composé de 0 ou plusieurs element2

```
<!ELEMENT element1 (element2)+>
```

element1 est composé de 1 ou plusieurs element2

```
<!ELEMENT element1 (#PCDATA)>
```

element1 est composé de caractères "parsable", donc s'il y a des entités, elles sont résolues.

```
<!ELEMENT element1 EMPTY>
```

element1 est toujours vide de contenu

```
<!ELEMENT element1 ANY>
```

element1 contient n'importe quels éléments définis dans la DTD ou (#PCDATA)

Instructions de la DTD - déclaration d'attribut : ATTRIBUTE

triplet nom_d'attribut type valeur_par_défaut

Un attribut est plus qu'un élément "texte" : le type est plus précis, et sa valeur par défaut est mieux précisée.

Syntaxe :

```
<!ATTLIST nom-de-l'élément nom_d'attribut type valeur_par_défaut>
```

Types d'attributs

- (valeur1 | valeur2 | ...) liste de valeurs possibles pour l'attribut
- CDATA texte non "parsé"
- ID un identifiant unique
- IDREF une référence à un identifiant unique du document
- ENTITY un nom d'entité de la DTD.
- NOTATION le nom d'une notation (référençant une entité non XML)
- ...

Valeur par défaut de l'attribut

- #REQUIRED la valeur de l'attribut doit obligatoirement être indiquée
- #IMPLIED la valeur peut être omise, mais les applications traitant le document lui attriburont une valeur par défaut.
- "defaultValue" valeur par défaut, si elle est omise
- #FIXED "fixedValue" valeur fixée !

XML - Sérialisation

voir cours C. Nahaboo

XMLSchema

Pour résoudre les problèmes de la DTD et mieux traiter les données.

Particularités :
- format XML
- support des namespaces
- types prédéfinis
- déclaration de nouveaux types élémentaires
- combinaison de types
- héritage
- modularité

Quelques standards XML

- **Recherche et partage de documents (RDF, WebDAV, RSS, ...)**
- Le **graphique (SVG)** Langage standard permettant de décrire, en XML, des graphiques à 2 dimensions, en vue de les redimensionner.
- Le **multimédia (SMIL)** Langage standard permettant de représenter et d'échanger des représentations multimédias dynamiques.
- Les **documents scientifiques (MathML)** Permet de décrire une équation mathématique en termes de signification et de présentation, pour pouvoir échanger et retraiter ces formules mathématiques dans des documents.

XPath,XPointer,Xlink : les hyperliens XML

XPath est un langage pour décrire des endroits dans un document XML.
XPointer spécifie des cibles via XPath.
Xlink décrit des liens entre des XPointers

XSL : les styles pour XML

XSL est un langage extensible permettant de définir des feuilles de styles. La feuille de styles est composée de règles qui permettent de sélectionner des éléments du document XML et d'y appliquer des attributs de formatage. Ainsi on peut extraire d'un document XML un autre document (XML,HTML,texte,. . .) construit à partir des règles de réecriture de la feuille de style. Les stylesheet XSL sont au format XML.

Exemple : Avec cette feuille de style,

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/
```

```
<xsl:output method="html" omit-xml-declaration="yes" inden
<xsl:template match="/" >
 <html>
   <table>
     <tr><td>Titre</td><td>Artiste</td></tr>
    <xsl:for-each  select="*/cd">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
    </xsl:for-each>
   </table>
 </html>
</xsl:template>
</xsl:stylesheet>
```

le document XML de l'exemple "Catalogue de CD" donnera :

```
<html>
<table>
<tr>
<td>Titre</td><td>Artiste</td>
</tr>
<tr>
<td>Empire Burlesque</td><td>Bob Dylan</td>
</tr>
<tr>
<td>Hide your heart</td><td>Bonnie Tyler</td>
</tr>
<tr>
```

```
<td>Greatest Hits</td><td>Dolly Parton</td>
</tr>
</table>
</html>
```

Utilisation de XML avec le Web

TOC

- XPath,XPointer,Xlink : les hyperliens XML ❖
- XSL : les styles pour XML ❖
- Utilisation de XML avec le Web ❖