

XSLT

(eXtensible Stylesheet Language Transformation) et XPath synthèse en 14 exemples

Christian Soutou
soutou@iut-blagnac.fr



Plan

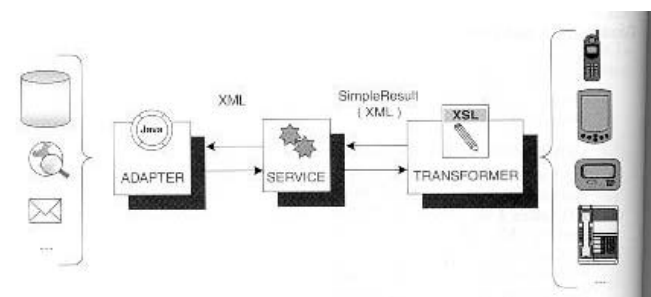
- Généralités
- Concepts de base XSLT
- Xpath (1.0)
- Programmation XSLT (1.0)
- Un point sur XPath 2.0 et XSLT 2.0

Généralités

- Langage de programmation pour extraire et transformer des documents XML dans divers formats
 - HTML, XML, PDF (avec XSL-FO)
 - multimédia (avec le langage SMIL)
 - graphique (avec le langage SVG)
- Deux spécifications :
 - XSLT 1.0 de 1999 (<http://www.w3.org/TR/xslt>) est la plus exploitée
 - XSLT 2.0 de 2007 (<http://www.w3.org/TR/xslt20>)
- Tous les éléments XSLT sont dans l'espace de noms
<http://www.w3.org/1999/XSL/Transform>

Généralités

- Exemple de transformation « classique »
 - XML+XSL = HTML « sobre »
 - Puis HTML+CSS = HTML « soigné » (polices, couleurs, styles)



Exemple pris dans une doc d'Oracle

Généralités

- Traitements possibles :
 - ✓ Définition de modèles (*templates*) sur les éléments ou attributs d'un document XML, pour leur appliquer des traitements
 - ✓ Fonctions (extractions d'éléments ou attributs un document XML, calculer un maximum, compter le nombre de résultats...)
 - ✓ Structures répétitives (« pseudo » *for*) et alternatives (*if* mais sans *else*)

• Concepts de base XSLT

Racine stylesheet

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  ...
</xsl:stylesheet>
```

- Un « programme » XSLT est un document XML
- L'élément du premier niveau : « feuille de style »
- Tous les éléments du langage sont dans l'espace de nom standardisé (préfixés par `xsl`)

Résultat à produire : output

```
<xsl:output method="html" | "xml" | "text"
  encoding = ...
  indent = yes|no media-type=...
/>
```

- `method` indique au processeur le type de document à produire (par défaut `xml`)
- `encoding` précise le jeu de caractères
- `indent` formate la sortie (indentation)
- `media-type` type MIME de sortie

Déclaration de règles avec template

```
<xsl:template {match="expression"}
  {name="nom règle"}
  ...
</xsl:template>
```

- `match` désigne les éléments dans l'arborescence du document XML pour lesquels la règle peut se déclencher
- `name` permet de nommer la règle qui pourra être invoquée par la suite (`xsl:call-template`)
- `expression XPath` (`/`, `nomEtudiant`, `@insee`, `/avion/capacite`, `*`, etc.)

Déclenchement de règles avec apply-templates

```
<xsl:apply-templates select="expression"/>
```

- Utilisé dans une balise `xsl:template`
- Indique au processeur d'appliquer la règle associée pour chaque partie de document XML (identifiée par `select` et désigné par l'`expression XPath`)
- Si l'attribut `select` est absent, le processeur va rechercher les modèles qui s'appliquent à tous les fils de la partie de document XML sélectionnée

Extraction de valeurs par value-of

```
<xsl:value-of select="expression"/>
```

- Utilisé dans une balise `xsl:template`
- Extrait la valeur textuelle de l'expression XPath et l'insère dans le document résultat
- Insertion selon la nature de `expression`
 - Élément (balise) : contenu textuel sans le balisage
 - Texte : texte
 - Attribut : valeur textuelle de l'attribut
 - Expression générale : résultat de l'évaluation

Exemple 1 (template)

- Données XML (`liste_avions1.xml`)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<liste_avions>
  <avi immat="F-GWSD">A320</avi>
  <avi immat="F-GWSS">Concorde</avi>
  <avi immat="F-FGLS">TB20</avi>
  <avi immat="F-GAFO">A380</avi>
  <avi immat="F-GFDG">A320</avi>
</liste_avions>
```

Flotte :

- Résultat HTML recherché :
- F-GWSD : A320
 - F-GWSS : Concorde
 - F-FGLS : TB20
 - F-GAFO : A380
 - F-GFDG : A320

Exemple 1 (template)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" encoding="ISO-8859-1"
    doctype-public="-//W3C//DTD HTML 4.01//EN"
    doctype-system="http://www.w3.org/TR/html4/strict.dtd"
    indent="yes" />

  <xsl:template match="liste_avions"> : Règle appellable pour chaque élément liste_avions
    <html><body>
      <p>Flotte :</p>
      <ul>
        <xsl:apply-templates select="avi" /> : Appel de la règle qui concerne la balise avi
        </ul>
      </body></html>
    </xsl:template>

  <xsl:template match="avi"> ← Expression de la condition pour déclencher la règle
    <li>
      <xsl:value-of select="@immat" /> : Lecture de l'attribut immat
      <xsl:text> : </xsl:text> : Ajout de texte simple
      <xsl:value-of select="." /> : désigne le chemin courant (dans la balise avi)
    </li>
  </xsl:template>
</xsl:stylesheet>
```

ex1.xsl

Associer l'XML à son XSLT

- Dans le document XML (liste_avions1.xml)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="./ex1.xsl"?>
<liste_avions>
  <avi immat="F-GWSD">A320</avi>
  <avi immat="F-GWSS">Concorde</avi>
  ...
</liste_avions>
```

- Par programmation (exemple en PHP)

```
$xh = xslt_create();
xslt_set_base ($xh, 'file:/// . getcwd () . /');
$result = xslt_process($xh, 'liste_avions1.xml', 'ex1.xsl');
if (!$result)
  echo ("Erreur XSLT ...");
else
  echo ($result);
```

Création de documents avec element et attribute

```
<xsl:element name="nomElement">
  ...
</xsl:element>
```

- Permet de produire un nouvel élément
- Des attributs peuvent lui être associés en y imbriquant :

```
<xsl:attribute name="nomAttribut">
  ...
</xsl:attribute>
```

- xsl:attribute peut être utilisé à l'intérieur d'un xsl:element ou après un élément littéral, exemple
- ```
<avion><xsl:attribute name="immat">...</avion>
```

# Exemple 2 (éléments et attributs)

```
<liste_avions>
 <avi nbplaces="140">A320
 <immat>F-GWSD</immat>
 </avi>
 <avi nbplaces="150">A320
 <immat>F-FGTY</immat>
 </avi>
 <avi nbplaces="500">A380
 <immat>F-WTXF</immat>
 </avi>
</liste_avions>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Flotte>
 <Avion immatriculation="F-GWSD">
 <TypeAv>A320</TypeAv>
 <Capacite>140</Capacite>
 </Avion>
 <Avion immatriculation="F-FGTY">
 <TypeAv>A320</TypeAv>
 <Capacite>150</Capacite>
 </Avion>
 <Avion immatriculation="F-WTXF">
 <TypeAv>A380</TypeAv>
 <Capacite>500</Capacite>
 </Avion>
</Flotte>
```