G. Hunault
Licence 2 Informatique

Angers, juin 2018

Technologie XML

1. Structuration en XHTML

Ecrire juste le fragment de texte XHTML 1.0 valide au sens de la grammaire *Strict* qui correspond aux indications suivantes (donc pas de balises html, head ou body) :

> *L'élément X1 contient deux éléments, X2 et X3. L'élément X2 contient deux éléments, X4 et X5. L'élément X3 contient deux éléments, X6 et X7. X4 a pour contenu le texte T1. X5 a pour contenu le texte T2. X6 a pour contenu le texte T3. X7 a pour contenu le texte T4.*

Pour mémoire, les noms des éléments et des attributs doivent être écrits en minuscules, mais pas leur valeur.

Si on remplace X1 par livres, X2 et X3 par livre, X4 et X6 par titre, X5 et X7 par auteur, T1 par "1984", T2 par "George ORWELL", T3 par "Fondation", T4 par "Isaac ASIMOV", quel est le code XHTML correspondant ?

2. Compréhension du vocabulaire

Dans une grammaire de type DTD, quel est la différence entre CDATA et NMTOKEN pour un attribut ?

Dans une grammaire de type DTD, quel est la différence entre NMTOKEN et NMTOKENS pour un attribut ?

3. Ecriture de DTD externe

Les statistiques traitent des colonnes de chiffres nommées VQT (variables quantitatives) et VQL (variables qualitatives). Une VQT est définie par un et un seul nom, une et une seule unité et une plage de variation facultative (valeurs min et max). Une VQL est définie par un et un seul nom, une liste de codes et de labels en nombre égaux et regroupés en modalités. De plus toute variable statistique a un numéro de variable.

Voici un exemple de fichier XML comportant deux VQT et une VQL, dans le fichier varstats.xml :

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

  <variable numero="1">
     <vqt>
        <nom>AGE</nom>
        <unité>ans</unité>
        <min>18</min>
        <max>120</max>
     </vqt>
  </variable>

  <variable numero="2">
     <vqt>
        <nom>POIDS</nom>
        <unité>kilos</unité>
     </vqt>
  </variable>

  <variable numero="3">
     <vql>
        <nom>CODE-SEXE</nom>
        <modalité code="1" label="Homme" />
        <modalité code="2" label="Femme" />
     </vql>
  </variable>

</variables>
```

Donner pour ce fichier varstats.xml une grammaire DTD « minimale et raisonnable » qui permet de décrire les variables statistiques.

4. Un peu de XPATH *via* XMLLINT

Donner les instructions de xmllint exécutées en mode *shell* qui permettent de répondre aux questions suivantes que l'on se pose sur un fichier de variables statistiques structuré comme dans la section précédente.

Attention : il ne s'agit évidemment pas du fichier précédent mais d'un fichier structuré de la même façon. On ne doit donc pas fournir les réponses aux questions mais donner le code xmllint qui fournirait les réponses.

- quel est le nom de la deuxième variable statistique du fichier ?

- combien y a-t-il modalités pour la première variable qualitative du fichier ?

- quel est le label associée au code "3" pour la variable qualitative de nom "PROFESSION" ?

- combien y a-t-il de variables quantitatives dont l'unité fait plus de 3 caractères ?

- combien y a-t-il de variables qualitatives avec plus de deux modalités ?

5. Transformations XSL (1)

Donner le contenu d'un fichier XSL qui effectue une transformation du fichier varstats.xml de la question 3 et qui produit le fichier texte suivant.

On essaiera de respecter le formatage des lignes, mais ce n'est pas grave s'il manque des espaces ou s'il y en a en trop.

```
3 Variables statistiques vues :
   2 variable(s) quantitative(s)
   1 variable(s) qualitative(s)
```

Si vous n'arrivez pas à tout reproduire, ce n'est pas grave. Essayez de produire ce que vous pouvez. Il vaut mieux fournr un code XSL plus court mais correct et fonctionnel qu'un code long mais incorrect.

6. Transformations XSL (2)

Donner le contenu d'un fichier XSL qui effectue une transformation du fichier varstats.xml de la question 3 et qui produit le fichier XHTML suivant. On pourra utiliser les *templates* et autres sous-programmes de stdWeb2.xsl vus en cours et en TP.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" [...]
<html xmlns="http://www.w3.org/1999/xhtml" lang="fr"
[...]
<body class="beige_jpg">
<blockquote>

  <form action="anastat.php">
    Sélectionner la quantitative à analyser :
     <select name="nomvar" id="nomvar">
       <option value="AGE">AGE</option>
       <option value="POIDS">POIDS</option>
     </select>
     <input type="submit"/>
  </form>

</blockquote>
</body>
</html>
```

7. Discussion

Essayez de répondre à la question suivante :

> *A-t-on vraiment besoin du typage très détaillé des nombres et des chaines de caractères fourni par les schémas XSD ?*

Votre réponse devra mettre en évidence votre culture naissante ainsi que votre recul et votre esprit de synthèse en matière de modélisation et de traitement d'informations structurées.

Votre réponse devra faire 10 lignes au minimum, sans limite de maximum. On utilisera au moins 3 mots de 4 syllabes ou plus pour transmettre « un contenu rédactionnel fort ».

ESQUISSE DE SOLUTION

1. Structuration en XHTML

Voici la structuration formelle du document. Attention, le nom des éléments et des attributs doit être en minuscule (mais pas leur contenu).

```
<x1>
  <x2>
    <x4>T1</x4>
    <x5>T2</x5>
  </x2>
  <x3>
    <x6>T3</x6>
    <x7>T4</x7>
  </x3>
</x1>
```

On obtient alors, en XHTML :

```
<livres>
  <livre>
     <titre>1984</titre>
     <auteur>George ORWELL</auteur>
  </livre>
  <livre>
     <titre>Fondation</titre>
     <auteur>Isaac ASIMOV</auteur>
  </livre>
</livres>
```

2. Compréhension du vocabulaire

CDATA est une chaine de caractères générales (sauf marqueurs), NMTOKENS est une chaine normalisée qui ressemble plutôt à un identifiant donc avec un seul mot et sans caractères spéciaux, qui commence par une lettre ou le symbole _.

NMTOKENS est une suite de NMTOKEN séparés par des espaces ou tabulations ou caractères équivalents.

3. Ecriture de DTD externe

Une grammaire DTD, concise, cohérente et minimale pour l'exemple fourni doit ressembler à :

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT variables (variable)+>
<!ELEMENT variable  (vqt|vql)>
<!ELEMENT vqt       (nom,unité,(min,max)?)>
<!ELEMENT vql       (nom,modalité+)>
<!ELEMENT nom       (#PCDATA)>
<!ELEMENT modalité  EMPTY>
<!ELEMENT unité     (#PCDATA)>
<!ELEMENT min       (#PCDATA)>
<!ELEMENT max       (#PCDATA)>
<!ATTLIST variable  numero CDATA   #REQUIRED>
<!ATTLIST modalité  code   CDATA   #REQUIRED>
<!ATTLIST modalité  label  NMTOKEN #REQUIRED>
```

4. Un peu de XPATH

Voici les expressions XPATH demandées.

```
ls //variable[2]//nom  # ok aussi pour  ls //variable[@numero="2"]//nom
xpath count(//vql[1]/modalité)
ls //vql[nom="PROFESSION"]/modalité[@code="3"]/@label
xpath count(//vqt[string-length(unité)>3])
xpath count(//vql[modalité[3]])
```

5. Transformations XSL (1)

```
<?xml version="1.0" ?>
<xsl:stylesheet xmlns:xsl='http://www.w3.org/1999/XSL/Transform' version="1.0">
<xsl:output method="text"/>
<xsl:template match="/">
<xsl:value-of select="count(//variable)" /> Variables statistiques vues :
   <xsl:value-of select="count(//vqt)" /> variable(s) quantitative(s)
   <xsl:value-of select="count(//vql)" /> variable(s) qualitative(s)
</xsl:template>
</xsl:stylesheet>
```

6. Transformations XSL (2)

Voici une transformation possible qui profite des règles de `stdWeb2.xsl` :

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet xmlns:xsl='http://www.w3.org/1999/XSL/Transform' version="1.0">
<xsl:import href="stdWeb2.xsl" />
<xsl:output method="xml"/>
<xsl:template match="/">

<xsl:call-template name="debutPage">
  <xsl:with-param name="leTitre">variables statistiques</xsl:with-param>
</xsl:call-template>

<form action="anastat.php">
Sélectionner la quantitative à analyser :
  <select name="nomvar" id="nomvar">
     <xsl:apply-templates select="//vqt//nom" /><xsl:text>&#xa;   </xsl:text>
  </select>
  <xsl:text>&#xa;   </xsl:text>
  <input type="submit" />
<xsl:text>&#xa;</xsl:text>
</form>

<xsl:call-template name="finPage"   />

</xsl:template>
```

```
<xsl:template match="nom">
<xsl:text>&#xa;   </xsl:text>
<xsl:variable name="nomvar"><xsl:value-of select='.' /></xsl:variable>
<xsl:element name="option">
  <xsl:attribute name="value"><xsl:value-of select="$nomvar" /></xsl:attribute>
  <xsl:value-of select="$nomvar" />
</xsl:element>
</xsl:template>

</xsl:stylesheet>
```