Université Sidi Mohamed Ben Abdallah, Fès

Faculté des Sciences Dhar El Mahraz, Fès

XML
Exercices Corrigés

Prof. El Habib NFAOUI

(elhabib.nfaoui@usmba.ac.ma)

Version 1 : 2014

Contenu

Série d'exercices 1 : Structuration des données en XML

Exercice 1:

On se propose d'écrire une version structurée en XML du document suivant (les valeurs présentées ici sont à titre d'exemple. Elles ne sont pas réelles) :

Une bouteille d'eau AinImouzer de 150 cl contient par litre 71 mg d'ions positifs calcium, et 5,5 mg d'ions positifs magnésium. On y trouve également des ions négatifs comme des chlorures à 20 mg par litre et des nitrates avec 1 mg par litre. Elle est recueillie à Imouzer la Source, dans le département de Fes-Boulemane. Son code barre est 3274080005006 et son pH est de 7,45. Comme la bouteille est sale, quelques autres matériaux comme du fer s'y trouvent en suspension.

Une seconde bouteille d'eau EauHammat a été, elle, recueillie à la source d'Ain Hammat dans les régions de Tafilalet. La concentration en ions calcium est de 98 mg/l, et en ions magnésium de 4 mg/l. Il y a 3,6 mg/l d'ions chlorure et 2 mg/l de nitrates, pour un pH de 7,4. Le code barre de cette bouteille de 50 cl est 3268840001003.

Une bouteille de même contenance est de marque SidiSami, et a été puisée à la source MoulaySami de SoussMassaDaraa, bien connu pour ses sources donnant un pH neutre de 7. Elle comprend 11,5 mg/l d'ions calcium, 8,0 mg/l d'ions magnésium, 13,5 mg/l d'ions chlorures et 6,3 mg/l d'ions nitrates. Elle contient également des particules de silice. Son code barre est 3057640117008.

a. Proposer une représentation graphique d'un arbre XML pour ce document.
b. Proposez une structuration XML de ce document.
c. Vérifiez, à l'aide de l'éditeur, que votre document est bien formé.

Exercice 2: Structuration du contenu d'un livre en XML

On se propose de **structurer** le contenu d'un livre en utilisant le formalisme XML. Le livre est structuré en sections (au moins 2), en chapitres (au moins 2) et en paragraphes (au moins 2).
Le livre doit contenir la liste des auteurs (avec nom et prénom).
Tous les éléments doivent posséder un titre, sauf le paragraphe qui contient du texte.
Remarque :

- Dans cette première version, nous ne voulons pas utiliser d'attributs ;
- L'encodage utilisé est ISO-8859-1

a. Proposer une structuration XML de ce document (avec 2 auteurs, 2 sections, 2 chapitres par section et 2 paragraphes par chapitre). Vous nommez le document « *livre1.xml* ».
b. Vérifier, à l'aide de l'éditeur, que votre document est bien formé.

Exercice 3: Utilisation des attributs

On souhaite compléter la structure du document XML de l'exercice précédent par les attributs *nom* et *prenom* pour les auteurs et *titre* pour le livre, les sections et les chapitres.

a. Analysez la structure du nouveau document. Y a-t-il des simplifications possibles ?
b. Placez dans 2 paragraphes un bloc de texte contenant l'extrait suivant :
<element id="10">></element>

Pour le premier paragraphe, employez les entités prédéfinies.
Pour le deuxième paragraphe, employez une section CDATA.

c. Vérifiez, à l'aide de l'éditeur, que votre document est bien formé.

Exercice 4:

On se propose de développer un jeu de culture générale en informatique nommé **JeuInfo**. L'idée consiste à employer des documents XML pour procurer les questions aux versions Web et mobiles du jeu. Ainsi, nous avons besoin de coder la base de données des questions et des réponses en XML. Le recours à XML permet de coder le contenu d'une façon cohérente et de fournir facilement et régulièrement des mises à jour et de nouveaux contenus.

Le jeu implique un seul type de question à choix multiples. Chaque question possède plusieurs choix de réponses possibles, un ou plusieurs choix à la fois peuvent être corrects (valides).

a. Proposer une représentation graphique d'un arbre XML pour le contenu de ce jeu.
b. Proposer un document XML **JeuInfo.xml** comprenant trois questions.
c. Associer une feuille de style CSS "**JeuInfo.css"** au document **JeuInfo.xml** pour visualiser les données sur un navigateur Web.
d. Vérifier, à l'aide de l'éditeur, que le document est bien formé.

Exercice 5: Le format MathML

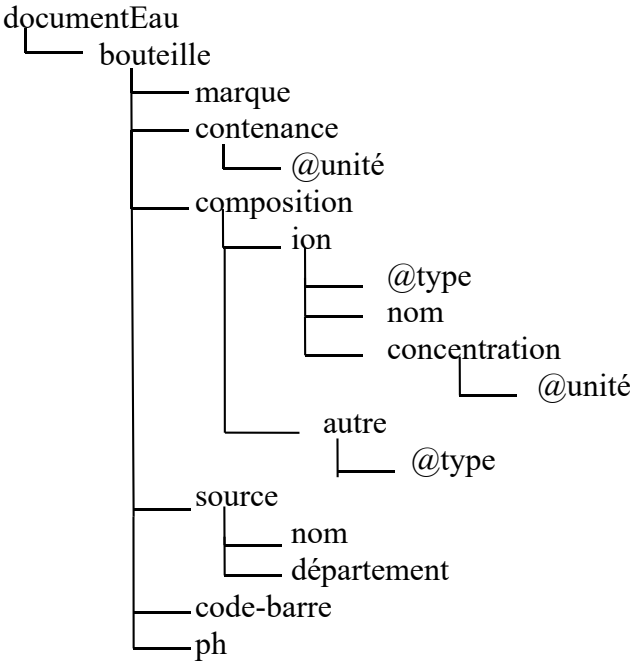
MathML est une recommandation du W3C (*http://www.w3.org/Math/*) servant à décrire des expressions mathématiques. Quelle est l'expression décrite par le document MathML suivant ?

```
<?xml version="1.0" encoding="UTF-8"?>
<math >
    <mrow>
        <mrow>
            <msup> <mi>x</mi> <mn>2</mn> </msup> <mo>+</mo>
            <mrow>
                <mn>4</mn>
                <mo>&InvisibleTimes;</mo>
                <mi>x</mi>
            </mrow>
            <mo>+</mo>
            <mn>4</mn>
        </mrow>
        <mo>=</mo>
        <mn>0</mn>
    </mrow>
</math>
```

Série d'exercices 1 : Corrigé

Exercice 1 :

a. Représentation graphique de l'arbre XML pour le document



N.B. Le PH est une caractéristique de l'eau et non pas de la bouteille. Il est préférable de le prendre comme élément.

b. Structuration XML du document

```
<?xml version="1.0" encoding="UTF-8"?>
<documentEau>
   <bouteille>
      <marque> AinImouzer</marque>
      <contenance unite="cl"> 150</contenance>
      <composition>
         <ion type="positif" >
            <nom> calcium</nom>
            <concentration unite="mg/l">71</concentration>
         </ion>
         <ion type="positif">
            <nom> magnesium</nom>
            <concentration unite="mg/l"> 5,5</concentration>
         </ion>
```

```
        <ion type="negatif">
          <nom>chlorures </nom>
          <concentration unite="mg/l"> 20</concentration>
        </ion>
        <ion type="negatif">
          <nom>nitrates </nom>
          <concentration unite="mg/l"> 1</concentration>
        </ion>
        <autre type="metal">fer</autre>
      </composition>
      <source>
        <nom>Imouzer la Source</nom>
        <departement> Fes-Boulemane</departement>
      </source>
      <ph>7,45</ph>
      <code-bare>3274080005003</code-bare>
    </bouteille>

    <bouteille>
      <marque> EauHammat</marque>
      <contenance unite="cl"> 50</contenance>
      <composition>
        <ion type="positif">
          <nom> calcium</nom>
          <concentration unite="mg/l">98</concentration>
        </ion>
        <ion type="positif">
          <nom> magnesium</nom>
          <concentration unite="mg/l"> 4</concentration>
        </ion>
        <ion type="negatif">
          <nom>chlorures </nom>
          <concentration unite="mg/l">3,6 </concentration>
        </ion>
        <ion type="negatif">
          <nom>nitrates </nom>
          <concentration unite="mg/l"> 2</concentration>
        </ion>
      </composition>
      <source>
        <nom>Ain Hammat</nom>
        <departement>Tafilalt</departement>
      </source>
      <ph>7,4</ph>
      <code-bare>3268840001008</code-bare>
    </bouteille>
```

```
<bouteille>
  <marque> SidiSami</marque>
  <contenance unite="cl"> 150</contenance>
  <composition>
    <ion type="positif">
      <nom> calcium</nom>
      <concentration unite="mg/l">11,5</concentration>
    </ion>
    <ion type="positif">
      <nom> magnesium</nom>
      <concentration unite="mg/l"> 8,0</concentration>
    </ion>
    <ion type="negatif">
      <nom>chlorures </nom>
      <concentration unite="mg/l"> 13,5</concentration>
    </ion>
    <ion type="negatif">
      <nom>nitrates </nom>
      <concentration unite="mg/l">6,3</concentration>
    </ion>
    <autre type="">silice</autre>
  </composition>
  <source>
    <nom>MoulaySami</nom>
    <departement> SoussMassaDaraa</departement>
  </source>
  <ph>7</ph>
  <code-bare>3057640117008</code-bare>
</bouteille>
</documentEau>
```

Exercice 2 :

a. Structuration XML du document

```
<?xml version="1.0" encoding="ISO-8859-1"? >
<livre>
  <titre>XML</titre>

  <auteurs>
    <auteur>
      <nom>NomAuteur</nom>
      <prenom>PrénomAuteur</prenom>
    </auteur>
    <auteur>
      <nom>NomAuteur</nom>
```

```
            <prenom>PrénomAuteur</prenom>
        </auteur>
    </auteurs>

    <sections>
        <section>
            <titre>Section 1</titre>
            <chapitres>
                <chapitre>
                    <titre>Chapitre 1</titre>
                    <paragraphes>
                        <paragraphe>Premier paragraphe</paragraphe>
                        <paragraphe>Deuxième paragraphe</paragraphe>
                    </paragraphes>
                </chapitre>
                <chapitre>
                    <titre>Chapitre 2</titre>
                    <paragraphes>
                        <paragraphe>Premier paragraphe</paragraphe>
                        <paragraphe>Deuxième paragraphe</paragraphe>
                    </paragraphes>
                </chapitre>
            </chapitres>
        </section>
        <section>
            <titre>Section 2</titre>
            <chapitres>
                <chapitre>
                    <titre>Chapitre 1</titre>
                    <paragraphes>
                        <paragraphe>Premier paragraphe</paragraphe>
                        <paragraphe>Deuxième paragraphe</paragraphe>
                    </paragraphes>
                </chapitre>
                <chapitre>
                    <titre>Chapitre 2</titre>
                    <paragraphes>
                        <paragraphe>Premier paragraphe</paragraphe>
                        <paragraphe>Deuxième paragraphe</paragraphe>
                    </paragraphes>
                </chapitre>
            </chapitres>
        </section>
    </sections>
</livre>
```

Exercice 3 :

a. Structuration XML du document

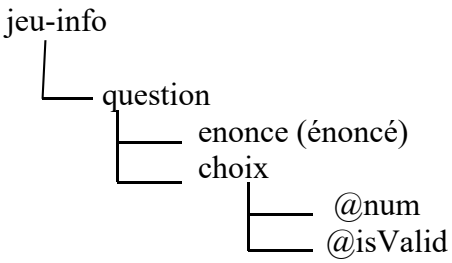
```
<?xml version="1.0" encoding=" ISO-8859-1"?>
<livre titre="XML et ses applications">
   <auteurs>
      <auteur nom="nom_auteur" prenom="pre_auteur"/>
      <auteur nom=" nom_auteur " prenom=" pre_auteur "/>
   </auteurs>
   <sections>
      <section titre="Section 1">
         <chapitre titre="Chapitre 1">
            <paragraphe>Premier paragraphe</paragraphe>
            <paragraphe>Deuxième paragraphe</paragraphe>
         </chapitre>
         <chapitre titre="Chapitre 2">
            <paragraphe>Premier paragraphe</paragraphe>
            <paragraphe>Deuxième paragraphe</paragraphe>
         </chapitre>
      </section>
      <section titre="Section 2">
         <chapitre titre="Chapitre 1">
            <paragraphe>Premier paragraphe</paragraphe>
            <paragraphe>Deuxième paragraphe</paragraphe>
         </chapitre>
         <chapitre titre="Chapitre 2">
            <paragraphe>Premier paragraphe</paragraphe>
            <paragraphe>Deuxième paragraphe</paragraphe>
         </chapitre>
      </section>
   </sections>
</livre>
```

b.

```
<?xml version="1.0" encoding="UTF-8"?>
<livre>
   ....
   <chapitre titre="Chapitre1">
      <paragraphe>&lt;element id="10"&gt;&amp;gt;&lt;/element&gt;</paragraphe>
      <paragraphe><![CDATA[<element id="10">&gt;</element>]]></paragraphe>
   </chapitre>
   .....
</livre>
```

Exercice 4 :

a. Représentation graphique de l'arbre XML pour le document



b. Structuration XML du document

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="JeuInfo2.css"?>
<jeu-info>
  <question>
    <enonce> Quelle est la date de la  première coupe du monde en foot ? </enonce>
    <choix num="1" isValid="true">1930</choix>
    <choix num="2" isValid="false">1920</choix>
    <choix num="3" isValid="false">1953</choix>
  </question>


  <question>
    <enonce>Quelle est la compagnie propiétaire de Windows ? </enonce>
    <choix num="1" isValid="false">InfoSoft</choix>
    <choix num="2" isValid="true">MicroSot</choix>
    <choix num="3" isValid="false">Oracle</choix>
  </question>>

  <question>
    <enonce>Quelle est la compagnie propriétaire de Dreamweaver ? </enonce>
    <choix num="1" isValid="false">Macromedia</choix>
    <choix num="2" isValid="true">Adobe</choix>
    <choix num="3" isValid="false">Oracle</choix>
  </question>
</jeu-info>
```

c. Feuille de style CSS : JeuInfo_version2.css

```
question {
  display: block;
  width: 750px;
  padding: 10px;
  margin-bottom: 10px;
  border: 4px double black;
  background-color: silver;
}

enonce {
  display: block;
  color: black;
  font-family: Times, serif;
  font-size: 16pt;
  text-align: left;
}

choix {
  display: block;
  color:maroon;
  font-family: Times, serif;
  font-size: 12pt;
  text-indent: 15px;
  text-align: left;
}
```

Exercice 5:

Ce document MathML décrit l'expression suivante : x^2 + 4x + 4 = 0
Très brièvement, mo désigne un opérateur, mrow une expression, mi et mn respectivement des opérandes variable (par exemple x) et nombre.

Série d'exercices 2 : DTD (Document Type Definition)

Exercice 1:

On se propose de concevoir une DTD pour un langage de balisage d'entraînement sportif. Ce langage de balisage qu'on va appeler ETML (Endurance Training Markup Languge), pourrait être pratique si on envisage de participer à des compétitions de marathon ou de triathlon. Il modèle en effet les données d'entraînement relatives à des sports d'endurance comme la course à pied, la natation et le cyclisme. Voici les pièces d'informations principales associées à chaque séance individuelle d'entraînement:

- Date, la date et l'heure de la séance d'entraînement;
- Type, le type de séance d'entraînement (course, nage, vélo, etc.);
- Rythme cardiaque, le rythme cardiaque moyen enregistré au cours de la séance d'entraînement;
- Durée, la durée de la séance d'entraînement;
- Distance, la distance couverte pendant la séance d'entraînement (mesurée en Kilomètres ou en miles);
- Lieu, le lieu de la séance d'entraînement;
- Commentaires, remarques générales sur la séance d'entraînement.

a. Sachant que toutes ces informations doivent être stockées dans un élément **séance** d'entraînement, vous déterminez lesquelles sont mieux adaptées comme éléments enfants et lesquelles seraient mieux sous la forme d'attributs?
b. Donner ensuite le code de la DTD ETML (etml.dtd)
c. Créer un document XML (document exemple codé en ETML) comprenant trois séances d'entraînement.

Exercice 2:

On se propose de définir un format XML de stockage des contacts. Un contact contient le nom, le prénom, le numéro de téléphone, l'e-mail et l'adresse d'une personne. L'adresse se compose du numéro, du nom de la rue et de la ville

a. Proposer une DTD.
b. Créer un document XML (document exemple) comprenant deux contacts.
c. Vérifier, à l'aide de l'éditeur que votre document est valide.

Série d'exercices 2 : Corrigé

Exercice 1 :

a. Conception théorique:

Voici quelques raisons logiques qui peuvent expliquer une répartition de ces informations en éléments et en attributs:

- Attributs: Date, Type, Rythme cardiaque. Il s'agit toujours de valeurs courtes et simples. L'attribut type peut être une liste énumérée de valeurs prédéfinies (course, nage, vélo, etc.).
- Eléments enfants: Durée, Distance, Lieu, Commentaires. La durée et la distance d'une séance pourraient parfaitement, en terme de modélisation, être stockées comme éléments ou comme attributs. En retenant l'option éléments on se laisse pourtant la possibilité de leur affecter des attributs supplémentaires, comme l'unité de mesure exacte.
- Le lieu et les commentaires contiennent potentiellement du texte descriptif et sont donc mieux en tant qu'éléments enfants.

b. DTD ETML (etml.dtd)

```
<!ELEMENT journalseance (seance)+>
<!ELEMENT seance (duree, distance, lieu, commentaires)>
<!ATTLIST seance
  date CDATA #IMPLIED
  type (course | natation | cyclisme) "course"
  rythmcard CDATA #IMPLIED
>

<!ELEMENT duree (#PCDATA)>
<!ATTLIST duree
  unites (secondes | minutes | heures) "minutes"
>

<!ELEMENT distance (#PCDATA)>
<!ATTLIST distance
  unites (miles | kilometres | tours) "miles"
>

<!ELEMENT lieu (#PCDATA)>

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

c. Exemple de document ETML (documentETML.xml)

```
<?xml version="1.0"?>
<!DOCTYPE journalseance SYSTEM "etml.dtd">

<journalseance>
  <seance date="11/19/05" type="course" rythmcard="158">
    <duree unites="minutes">50</duree>
```

```
    <distance unites="miles">5.5</distance>
    <lieu>Warner Park</lieu>
    <commentaires>Milieu de matinée, un peu venteux.</commentaires>
  </seance>

  <seance date="11/21/05" type="cyclisme" rythmcard="153">
    <duree unites="heures">1.5</duree>
    <distance unites="miles">26.4</distance>
    <lieu>Natchez Trace Parkway</lieu>
    <commentaires>Parcours en colline. Je me sens fort comme un
                  Bœuf.</commentaires>
  </seance>

  <seance date="11/24/05" type="course" rythmcard="156">
    <duree units="heures">2.5</duree>
    <distance unites="miles">16.8</distance>
    <lieu>Warner Park</lieu>
    <commentaires>Apres midi. Sensation de force honnête.</commentaires>
  </seance>
</journalseance>
```

Exercice 2:

a. DTD (annuaire.dtd)

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT annuaire (contact*)>
<!ELEMENT contact (nom, prenom+, tel+,email*,adresse) >
<!ELEMENT nom (#PCDATA) >
<!ELEMENT prenom (#PCDATA) >
<!ELEMENT tel (#PCDATA) >
<!ELEMENT email (#PCDATA) >
<!ELEMENT adresse (numero, rue,ville) >
<!ELEMENT numero (#PCDATA) >
<!ELEMENT rue (#PCDATA) >
<!ELEMENT ville (#PCDATA) >
```

b. Document XML contenant des contacts

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE annuaire SYSTEM "annuaire.dtd">
<annuaire>
  <contact>
    <nom>AZER</nom>
    <prenom>Azer</prenom>
    <tel>0678905044</tel>
    <email>aa@gmail.com</email>
    <email>aa@yahoo.fr</email>
```

```
        <adresse>
            <numero>2</numero>
            <rue>Ouahrane</rue>
            <ville>Fes</ville>
        </adresse>
    </contact>

    <contact>
        <nom>AAIM</nom>
        <prenom>Moumen</prenom>
        <tel>0678900033</tel>
        <adresse>
            <numero>2</numero>
            <rue>Tanger</rue>
            <ville>Rabat</ville>
        </adresse>
    </contact>
</annuaire>
```

Série d'exercices 3 : XSD (XML Schema Language)

Exercice 1 :

On se propose de concevoir un schéma XML (XSD) pour un langage de balisage d'entraînement sportif. Ce langage de balisage qu'on va appeler ETML (Endurance Training Markup Languge), pourrait être pratique si on envisage de participer à des compétitions de marathon ou de triathlon. Il modèle en effet les données d'entraînement relatives à des sports d'endurance comme la course à pied, la natation et le cyclisme. Voici les pièces d'informations principales associées à chaque séance individuelle d'entraînement:

- Date, la date et l'heure de la séance d'entraînement;
- Type, le type de séance d'entraînement (course, nage, vélo, etc.);
- Rythme cardiaque, le rythme cardiaque moyen enregistré au cours de la séance d'entraînement;
- Durée, la durée de la séance d'entraînement;
- Distance, la distance couverte pendant la séance d'entraînement (mesurée en Kilomètres ou en miles);
- Lieu, le lieu de la séance d'entraînement;
- Commentaires, remarques générales sur la séance d'entraînement.

a. Proposer un code pour le XSD ETML (etml.xsd)
b. Créer un document XML (document exemple codé en ETML) comprenant trois séances d'entraînement.

Exercice 2 :

Un annuaire regroupe un ensemble d'informations sur des personnes. Chaque personne est qualifiée par un code et est définie par :

- Un nom qui est décomposé en nom de famille et prénom
- Un type qui peut être soit « ami », « client », ou « fournisseur »
- Une date de naissance
- De 1 à 3 adresses email qui doivent respecter la forme d'une adresse email.
- Plusieurs numéros de téléphone. Chaque numéro est écrit sous forme d'une **lise de nombres de type byte.** Par exemple : 33 55 44 4.

a. Proposer un schéma XML.
b. Ecrire un document XML comprenant les informations de trois personnes.

Exercice 3 :

Soit un document XML contenant un nombre indéterminé d'éléments sous la forme :

```
<numeros>
	<contact titre="..." techno="...">
		<nom>...</nom>
		<prenom>...</prenom>
		<telephone> ...</telephone>
		<email>...</email>
		<email>...</email>
		...
	</contact>
</numeros>
```

L'élément *telephone* et l'attribut *techno* sont en option. Les textes seront des chaînes simples **xs:string**.

• l'attribut « techno » ne pourra prendre qu'une valeur parmi : XML, Java, Autre.
• le numéro de téléphone est écrit sous forme d'une liste de 5 entiers.
• La valeur de l'email doit respecter le pattern (modèle ou format standard) d'une adresse e-mail.

a. Créer un schéma « contact.xsd » en définissant et utilisant les types simples suivants :
   - technoType : énumération dont les valeurs possibles sont XML, Java, Autre.
   - telType : permet de restreindre le numéro de téléphone à une liste de 5 **nombres de type entier.** Créez d'abord un type nommé « listIntType » pour la liste d'entiers.
   - emailType : pattern `[a-z]+@[a-z]+\.[a-z]{2,3}`

Série d'exercices 3 : Corrigé

Exercice 1 :

a. XSD (etml.xsd)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

 <!-- Le type d'un élémnet peut être global et donc associé à un nom ou bien être local à
  cet élément : les lignes ci-dessous  montrent un exemple
 -->

 <xsd:element name="journalseance">
  <xsd:complexType>
   <xsd:sequence>
     <xsd:element name="seance" type="seanceType" minOccurs="0"
       maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
 </xsd:element>


 <!-- Voici une autre solution utilisant un type déclaré global -->
  <!--
  <xsd:element name="journalseance" type="journalseanceType" />

  <xsd:complexType name="journalseanceType">
      <xsd:sequence>
        <xsd:element name="seance" type="seanceType" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
  </xsd:complexType>
 -->

 <xsd:complexType name="seanceType">
  <xsd:sequence>
   <xsd:element name="duree" type="xsd:duration"/>
   <xsd:element name="distance" type="distanceType"/>
   <xsd:element name="lieu" type="xsd:string"/>
   <xsd:element name="commentaires" type="xsd:string" />
  </xsd:sequence>

  <xsd:attribute name="date" type="xsd:date" use="required"/>
  <xsd:attribute name="type" type="typeType" use="required"/>
  <xsd:attribute name="rythmcard" type="xsd:positiveInteger"/>
 </xsd:complexType>

<!-- Les auteurs de XSD ont considéré que le passage d'un contenu simple
à un type complexe pouvait se faire par dérivation (extension)-->
 <xsd:complexType name="distanceType">
  <xsd:simpleContent>
   <xsd:extension base="xsd:decimal">
    <xsd:attribute name="unites" type="unitesType"
     use="required"/>
```

```
    </xsd:extension>
   </xsd:simpleContent>
  </xsd:complexType>

  <xsd:simpleType name="unitesType">
   <xsd:restriction base="xsd:string">
    <xsd:enumeration value="miles"/>
    <xsd:enumeration value="kilometres"/>
    <xsd:enumeration value="tours"/>
   </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="typeType">
   <xsd:restriction base="xsd:string">
    <xsd:enumeration value="course"/>
    <xsd:enumeration value="natation"/>
    <xsd:enumeration value="cyclisme"/>
   </xsd:restriction>
  </xsd:simpleType>

</xsd:schema>
```

b. Exemple de document ETML (documentETML.xml)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<journalseance
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="etml.xsd">

 <seance  date="2005-11-19" type="course" rythmcard="158">
  <duree>PT45M</duree>
  <distance unites="kilometres">5.5</distance>
  <lieu>Route Sefrou</lieu>
  <commentaires>Milieu de matinée, un peu venteux.</commentaires>
 </seance>

 <seance date="2005-11-21" type="cyclisme" rythmcard="153">
  <duree>PT2H30M</duree>
  <distance unites="miles">37.0</distance>
  <lieu>Forêt Ain chkaf</lieu>
  <commentaires>Parcours en colline. Je me sens fort comme un boeuf.</commentaires>
 </seance>

 <seance date="2005-11-24" type="course" rythmcard="156">
  <duree>PT1H30M</duree>
  <distance unites="miles">8.5</distance>
  <lieu>Forêt Traat</lieu>
  <commentaires>Après midi. Sensation de force honnête.</commentaires>
 </seance>
</journalseance>
```

Exercice 2 :

a. XSD (annuaire.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="anuaire">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="personne" type="personneType" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:complexType name="personneType">
    <xsd:sequence>
      <xsd:element name="nom">
        <xsd:complexType>
          <xsd:attribute name="nomFamille" type="xsd:string"/>
          <xsd:attribute name="prenom" type="xsd:string"/>
        </xsd:complexType>
      </xsd:element>

      <xsd:element name="email" type="emailType" minOccurs="1" maxOccurs="3"/>
      <xsd:element name="tel" type="telType" maxOccurs="unbounded"/>

    </xsd:sequence>

    <xsd:attribute name="code" type="xsd:int"/>
    <xsd:attribute name="dateNaissance" type="xsd:date"/>
    <xsd:attribute name="type">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="ami"/>
          <xsd:enumeration value="client"/>
          <xsd:enumeration value="fournisseur"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>
  </xsd:complexType>

  <xsd:simpleType name="emailType">
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[a-z]+@[a-z]+\.[a-z]{2,3}"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="telType">
    <xsd:list itemType="xsd:unsignedByte">
    </xsd:list>
  </xsd:simpleType>
</xsd:schema>
```

b. Exemple de document XML (annuaire.xml)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<anuaire xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="annuaire.xsd">

   <personne code="1" dateNaissance="2008-02-29" type="client">
      <nom nomFamille="Dadi" prenom="Hassan"/>
      <email>dzt@aze.com</email>
      <tel>33 55 44 4</tel>
   </personne>

   <personne code="2" dateNaissance="2008-02-29" type="ami">
      <nom nomFamille="Rannabi" prenom="Latif"/>
      <email>dzt@aze.com</email>
      <tel>33 55 44 4</tel>
   </personne>

</anuaire>
```

Exercice 3 :

a. XSD (contact.xsd)

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema">

   <xs:element name="numeros" type="numerosType"/>

   <xs:complexType name="numerosType">
      <xs:sequence>
         <xs:element name="contact" type="contactType" maxOccurs="unbounded"/>
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="contactType">
      <xs:sequence>
         <xs:element name="nom" type="xs:string"/>
         <xs:element name="prenom" type="xs:string"/>
         <xs:element name="telephone" type="telType" minOccurs="0"/>
         <xs:element name="email" type="emailType" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="titre" type="xs:string" use="required"/>
      <xs:attribute name="techno" type="technoType" use="optional"/>
   </xs:complexType>

   <xs:simpleType name="technoType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="XML"/>
         <xs:enumeration value="Java"/>
         <xs:enumeration value="Autre"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="listIntType">
```

```
        <xs:list itemType="xs:int">
        </xs:list>
    </xs:simpleType>

    <xs:simpleType name="telType">
        <xs:restriction base="listIntType">
            <xs:length value="5"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="emailType">
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-z]+@[a-z]+\.[a-z]{2,3}"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

Série d'exercices 4 : Publication de documents XML (XSL/XSLT/Xpath)

Exercice 1 :

Considérons le document XML ci-dessous qui contient des données sur des livres.

Fichier livre.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE bookList SYSTEM "livre.dtd">
<?xml-stylesheet type="text/xsl" href="livre.xsl"?>
<bookList>
    <book title="JEE" isbn="1-1123-1234-5">
        <publisher>Editions Toubkal</publisher>
        <language>Français</language>
        <nbPage>200</nbPage>
        <price>300</price>
        <authorList>
            <author>SOULA</author>
        </authorList>
    </book>

    <book title="Oracle" isbn="2-1123-1234-5">
        <publisher>Editions ENI</publisher>
        <language>Français</language>
        <nbPage>300</nbPage>
        <price>100</price>
        <authorList>
            <author>SAMOU</author>
        </authorList>
    </book>

    <book title="Linux" isbn="3-1123-1234-5">
        <publisher>Editions InfoPrint</publisher>
        <language>Anglais</language>
        <nbPage>400</nbPage>
        <price>250</price>
        <authorList>
            <author>SOULA</author>
            <author>DADIO</author>
        </authorList>
    </book>

</bookList>
```

a. Proposer un XSD (ou une DTD) pour valider ce document.
b. Ecrire un document XSL qui transforme le document livre.xml en un document HTML résultat. La présentation de ce dernier dans un navigateur est montrée sur la figure 1 ci-dessous.

c. Modifier le document XSL précédent pour ajouter un hyperlien de recherche autour du numéro ISBN comme le montre la figure 2 ci-dessous.
d. Modifier le document XSL de la question 3 pour trier la liste des livres en ordre croissant des titres.
e. Dans le document XSL de la question 4, ajouter une clé de tri secondaire en ordre décroissant du prix. Préciser le type de données du prix avec l'attribut data-type = "number" de la commande <xsl:sort>, le type de données par défaut est data-type = "text".

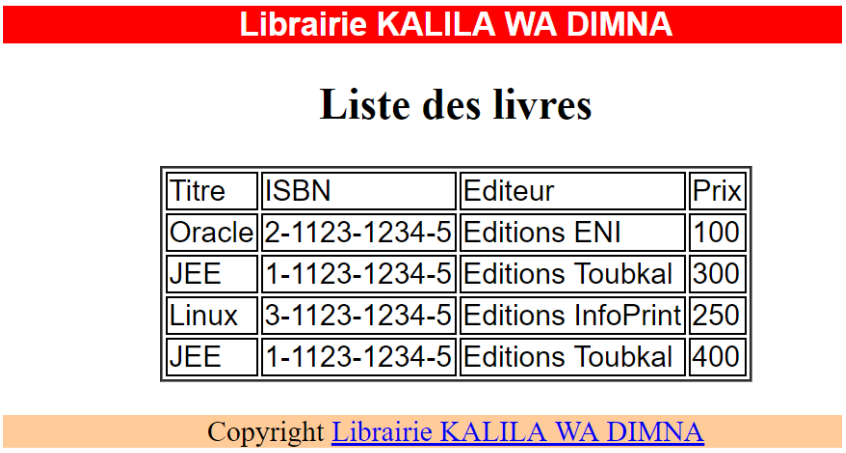


Figure 1

Figure 2

Exercice 2 :

Ecrire un document XSL qui transforme le document XML de l'exercice 1 en un **autre document XML** basé entièrement sur des attributs comme le montre le tableau ci-dessous :

```
<?xml version="1.0" encoding="utf-8"?>
<bookList>
    <book title="Oracle" isbn="2-1123-1234-5" language="Français" nbPage="300" price="100"/>
    <book title="JEE" isbn="1-1123-1234-5" language="Français" nbPage="200" price="300"/>
    <book title="Linux" isbn="3-1123-1234-5" language="Anglais" nbPage="400" price="250"/>
</bookList>
```

Exercice 3 :

Considérons le document XML ci-dessous qui contient des données sur des abonnements.

Fichier operateur.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE operateur SYSTEM "operateur.dtd">
<?xml-stylesheet type="text/xsl" href="operateur.xsl"?>
<operateur>
   <client code="1" nom="HASSOUNI">
      <abonnement num="123" type="GSM" dateAb="2006-1-11">
         <facture numFact="432" dateFact="2006-2-11" montant="350.44" reglee="oui"/>
         <facture numFact="5342" dateFact="2006-3-11" montant="450" reglee="oui"/>
         <facture numFact="5362" dateFact="2006-4-13" montant="800.54" reglee="non"/>
      </abonnement>
      <abonnement num="2345" type="FIXE" dateAb="2006-12-1">
         <facture numFact="5643" dateFact="2007-1-12" montant="299" reglee="oui"/>
         <facture numFact="6432" dateFact="2007-2-12" montant="555" reglee="non"/>
      </abonnement>
   </client>

   <client code="2" nom="ABBASSI">
      <abonnement num="7543" dateAb="2006-12-1" type="GSM">
         <facture numFact="7658" dateFact="2007-2-12" montant="350.44" reglee="oui"/>
         <facture numFact="7846" dateFact="2007-3-12" montant="770" reglee="non"/>
      </abonnement>
   </client>
</operateur>
```

a. Proposer un XSD (ou une DTD et) pour valider ce document.
b. Ecrire un document XSL qui transforme le document operateur.xml en un document HTML résultat. La présentation de ce dernier dans un navigateur est montrée sur la figure 1 ci-dessous.
c. Ecrire un document XSL qui transforme le document operateur.xml en un autre document HTML résultat. Ce dernier affiche seulement les abonnements dont le total des factures est

supérieur strictement à 1000. La présentation de ce dernier dans un navigateur est montrée sur la figure 2 ci-dessous.

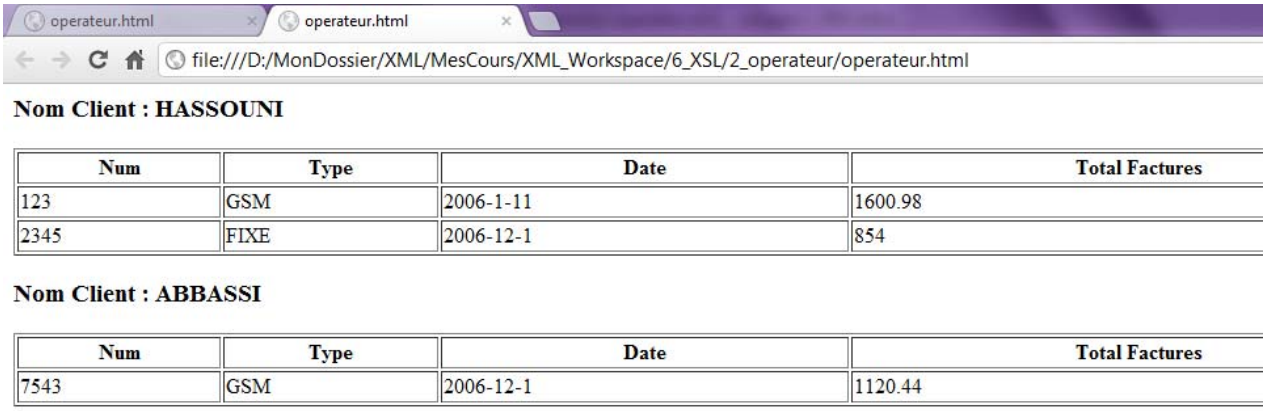


Figure 1

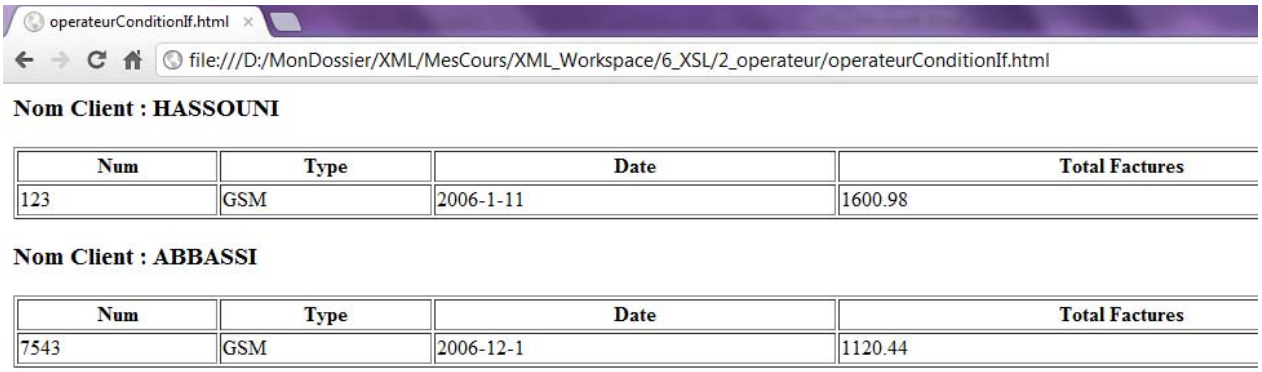


Figure 2

Série d'exercices 4 : Corrigé

Exercice 1 :

a. DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT bookList (book*)>
<!ELEMENT book (publisher, language, nbPage?, price?, authorList)>
<!ATTLIST book
   title CDATA #REQUIRED
   isbn CDATA #REQUIRED
>
<!ELEMENT authorList (author*) >
<!ELEMENT author (#PCDATA) >
<!ELEMENT language (#PCDATA) >
<!ELEMENT nbPage (#PCDATA) >
<!ELEMENT price (#PCDATA) >
<!ELEMENT publisher (#PCDATA) >
```

b. XSL

- En plus du code XSLT, nous avons séparé la mise en forme du document HTML résultat dans une feuille de style CSS externe nommée « cssOutDocument.css ». Le code de cette feuille est indiqué ci-dessous.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

  <xsl:output method="html"/>

  <xsl:template match="/">
    <html>
      <head>
        <title>Transformation XML-XHTML</title>
        <link href="cssOutDocument.css" rel="stylesheet" type="text/css"/>
      </head>
      <body>
        <header>Librairie KALILA WA DIMNA</header>
        <section>
          <h2>Liste des livres</h2>
          <xsl:apply-templates select="bookList"/>
        </section>
        <br/>
        <footer>
          Copyright <a href="http://www.kalilawadimna.ma">Librairie KALILA WA DIMNA</a>
        </footer>
      </body>
```

```
      </html>

   </xsl:template>

   <xsl:template match="bookList">
      <table>
         <tr>
            <td>Titre</td>
            <td>ISBN</td>
            <td>Editeur</td>
            <td>Prix</td>
         </tr>
         <xsl:apply-templates select="book"/>
      </table>
   </xsl:template>

   <xsl:template match="book">
      <tr>
         <td>
            <xsl:value-of select="@title"/>
         </td>
         <td>
            <xsl:value-of select="@isbn"/>
         </td>
         <td>
            <xsl:value-of select="publisher"/>
         </td>
         <td>
            <xsl:value-of select="price"/>
         </td>
      </tr>
   </xsl:template>
</xsl:stylesheet>
```

- cssOutDocument.css

```
header{
   background-color:red;
   font-family:arial, helvetica, sans-serif;
   color:white;
   font-size:large;
   font-weight:bold;
   text-align:center;
}
 footer{
        background-color: #FFCC99;
        text-align:center;
}
table{
   border-style:groove;
```

```
    margin-left: auto;
    margin-right: auto;

}

td, th {
    border: solid 1px black;
}

td{
    font-family:arial, helvetica, sans-serif;
}

h2{
    text-align:center;
}
```

c. XSL ajoutant un hyperlien de recherche autour du numéro ISBN

Nous gardons le code de la **question b** et nous modifions uniquement le code de la règle template concernant l'élément **book** comme le suivant :

```
.....
<xsl:template match="book">
  <tr>
    <td>
      <xsl:value-of select="@title"/>
    </td>
    <td>
      <a>
        <xsl:attribute name="href">
          <xsl:text>searchBookByIsbn? isbn=</xsl:text>
          <xsl:value-of select="@isbn"/>
        </xsl:attribute>
        <xsl:value-of select="@isbn"/>
      </a>
    </td>
    <td>
      <xsl:value-of select="publisher"/>
    </td>
    <td>
      <xsl:value-of select="price"/>
    </td>
  </tr>
</xsl:template>
```

d. XSL : trier la liste des livres en ordre croissant des titres

Nous gardons le code de la **question c** et nous modifions uniquement le code de la règle template concernant l'élément **bookList** comme le suivant :

```
……
<xsl:template match="bookList">
  <table>
    <tr>
      <td>Titre</td>
      <td>ISBN</td>
      <td>Editeur</td>
      <td>Prix</td>
    </tr>
    <xsl:apply-templates select="book">
      <xsl:sort select="@title" order="ascending"/>
    </xsl:apply-templates>
  </table>
</xsl:template>
……..
```

e. XSL : trier la liste des livres en ordre croissant des titres

```
……..
<xsl:apply-templates select="book">
  <xsl:sort select="@title" order="ascending" data-type="text"/>
  <xsl:sort select="price" order="descending" data-type="number"/>
</xsl:apply-templates>
……..
```

Exercice 2:

Voici une proposition d'un code XSLT permettant de transformer l'arbre XML du document XML de l'exercice 1 en un **autre arbre** basé entièrement sur des attributs.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    exclude-result-prefixes="xs"
    version="2.0">

  <xsl:output method="xml"/><!-- Valeur par défaut -->

  <xsl:template match="/">
    <xsl:apply-templates select="bookList"/>
```

```
    </xsl:template>

    <xsl:template match="bookList">
      <bookList> <!-- is equal to: <xsl:element name="bookList"> : quasiment jamais utilisé -->
        <xsl:apply-templates select="book"/>
      </bookList>
    </xsl:template>

    <xsl:template match="book">
      <book>
        <xsl:attribute name="title">
          <xsl:value-of select="@title"/>
        </xsl:attribute>

        <xsl:attribute name="isbn">
          <xsl:value-of select="@isbn"/>
        </xsl:attribute>

        <xsl:attribute name="language">
          <xsl:value-of select="language"/>
        </xsl:attribute>

        <xsl:attribute name="nbPage">
          <xsl:value-of select="nbPage"/>
        </xsl:attribute>

        <xsl:attribute name="price">
          <xsl:value-of select="price"/>
        </xsl:attribute>
      </book>
    </xsl:template>
</xsl:stylesheet>
```

Exercice 3 :

a. DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT operateur (client+) >
<!ELEMENT client (abonnement+) >
<!ELEMENT abonnement (facture+) >
<!ELEMENT facture EMPTY >
<!ATTLIST client
        code NMTOKEN #REQUIRED
        nom CDATA #REQUIRED>
<!ATTLIST abonnement
        num NMTOKEN #REQUIRED
        type (GSM|FIXE) 'FIXE'
        dateAb CDATA #REQUIRED>
<!ATTLIST facture
        numFact NMTOKEN #REQUIRED
        dateFact CDATA #REQUIRED
        montant CDATA #REQUIRED
        reglee (oui|non) 'non'>
```

b. XSL (présentation de la figure 1)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:template match="/">
    <html>
      <head></head>
      <body>
        <xsl:apply-templates select="operateur/client"/>

      </body>
    </html>
  </xsl:template>

  <xsl:template match="client">
    <h3>
      Nom Client : <xsl:value-of select="@nom"/>
    </h3>
    <table border="1" width="80%">
      <tr>
        <th>Num</th><th>Type</th><th>Date</th><th>Total Factures</th>
      </tr>
      <xsl:apply-templates select="abonnement"/>
    </table>

  </xsl:template>

  <xsl:template match="abonnement">
    <tr>
      <td><xsl:value-of select="@num"/></td>
      <td><xsl:value-of select="@type"/></td>
      <td><xsl:value-of select="@dateAb"/></td>
      <td><xsl:value-of select="sum(facture/@montant)"/></td>
    </tr>
  </xsl:template>
</xsl:stylesheet>
```

c. XSL (présentation de la figure 2)

Nous gardons le code de la **question b** et nous modifions uniquement le code de la règle template concernant l'élément **abonnement** comme le suivant :

```
……
<xsl:template match="abonnement">
    <xsl:if test="sum(facture/@montant) > 1000">
      <tr>
        <td><xsl:value-of select="@num"/></td>
        <td><xsl:value-of select="@type"/></td>
        <td><xsl:value-of select="@dateAb"/></td>
        <td><xsl:value-of select="sum(facture/@montant)"/></td>
      </tr>
    </xsl:if>
  </xsl:template>
……
```