TP3 XML DOM

Objectif : DOM

Exo 1: Parcours et transformation de documents XML en utilisant l'API DOM

Le but de cet exercice est de parcourir un document XML avec l'API DOM en utilisant le langage JAVA et de générer en sortie un document HTML. Le fichier XML en question est une liste bibliographique (bib.xml).

Pour cela vous disposez d'un prototype de programme Java permettant de lire un fichier XML, de construire une instance de parseur DOM et de générer l'arbre DOM associé.

Tâches à réaliser :

- copier le fichier bib.xml (voir dans le zip fournis), ainsi que les fichiers Java. Attention, n'ouvrez pas le fichier bib.xml avec un éditeur de texte, cela le rendra inutilisable pour l'exercice et vous aurez à le re-télécharger.
- étudier le programme Java
- modifier les chemins d'accès aux fichiers (si besoin).
- développer la partie manquante du code et compiler.
- exécuter le programme Java et afficher le résultat dans un navigateur Web.

Vous devez obtenir en sortie un fichier html dont le format est celui du fichier output.html.
PS : vous pouvez consultez l'API DOM de java sur l'url :
http://java.sun.com/j2se/1.4.2/docs/api/index.html

Exo 2 :

```
<?xml version="1.0"?>
<E-mail>
<To>Rohan</To>
<From>Amit</From>
<Subject>Surprise....</Subject>
<Body>Be ready for a cruise...</Body>
</E-mail>
```

Ecrire le code Java-DOM qui permettrait de lire sur la console le nom d'un élément (nom de balise) et d'afficher son contenu.

Écrire le programme Java-DOM qui permettrait de lire sur la console le nom d'un élément (nom de balise) et de l'effacer du fichier xml donné ci-dessus.

Exo 3:

Soit les fichiers XML gender.xml et gender-sordted.xml ci-dessous.

On prend en entrée un document XML du type de gender.xml, qui contient une liste de personnes (avec leur sexe) et pour chaque personne la liste de ses enfants. On veut obtenir en sortie un document du type gender-sorted.xml dans lequel on sépare clairement les garçons des filles.

- Écrivez un programme Java utilisant DOM pour réaliser cette transformation.

gender.xml (en entrée)

```
<?xml version="1.0"?>
<!DOCTYPE list [
  <!ELEMENT list (person)*>
  <!ELEMENT person (name, children)>
  <!ATTLIST person gender (M | F) #REQUIRED>
  <!ELEMENT name (#PCDATA)>
  <!ELEMENT children (person)*>
]>

<list>
<person gender="M">
   <name>Julien</name>
   <children>
     <person gender="F">
       <name>Sophie</name>
       <children></children>
     </person>
     <person gender="F">
       <name>Ursule</name>
       <children></children>
     </person>
   </children>
</person>

<person gender="F">
   <name>Marie</name>
   <children>
     <person gender="M">
       <name>Joseph</name>
       <children>
         <person gender="F">
           <name>Marie</name>
           <children>
           </children>
         </person>
      </children>
     </person>
   </children>
</person>
</list>
```

gender-sorted.xml (en sortie)

```
<?xml version="1.0"?>
<!DOCTYPE list [
  <!ELEMENT list (man | woman)*>
  <!ELEMENT man (sons, daughters)>
  <!ATTLIST man name CDATA #REQUIRED>
  <!ELEMENT woman (sons, daughters)>
  <!ATTLIST woman name CDATA #REQUIRED>
  <!ELEMENT sons (man)*>
  <!ELEMENT daughters (woman)*>
]>

<list>
   <man name="Julien">
      <sons> </sons>
      <daughters>
         <woman name="Sophie">
            <sons></sons>
            <daughters></daughters>
         </woman>
         <woman name="Ursule">
            <sons></sons>
            <daughters></daughters>
         </woman>
      </daughters>
   </man>

<woman name="Marie">
   <sons>
      <man name="Joseph">
         <sons></sons>
         <daughters>
            <woman name="Marie">
               <sons></sons>
               <daughters></daughters>
            </woman>
         </daughters>
      </man>
   </sons>
   <daughters></daughters>
</woman>

</list>
```