TP N° 01

Objectifs

L'objectif de cette série de TP est de :

- Faire un rappel sur le langage HTML : le langage HTML est un autre langage basé sur les balises, tout comme XML. Ce langage a fait l'objet de plusieurs travaux durant les deux premières années de formation. Dans cette série, vous allez revoir la notion de document HTML en créant un document riche en éléments.
- Faire un rappel sur la notion de DOM : Document Object Model est le modèle principal pour la manipulation des éléments d'une page (ou document HTML). Dans cette série, l'objectif est d'analyser un document HTML et d'extraire sa structure hiérarchique.
- Créer et parser un premier document XML : en se basant sur les deux premiers rappels, vous allez pouvoir créer un document XML. Pour l'exploiter, ce document doit être "parsé". La première manipulation proposée sera un simple affichage.

Travail à réaliser

Créer un document HTML

La premier étape dans cette série consiste à créer un document HTML. Pour pouvoir l'exploiter plus tard, le document à créer doit être riche en éléments :

- Le document doit présenter une structure claire : la strcuture d'une page HTML se compose généralement des composants suivants :
  - Entête,
  - Barre de navigation,
  - Panneau de navigation (droite, gauche, ou bien les deux à la fois),
  - Contenu principal : ce contenu peut être aussi structuré en entête, corps et pieds.
  - Pieds de la page.
- Cette structuration peut utiliser des tableaux (appoche ancienne) ou bien des divs stylées (approche plus moderne)
- Le document doit contenir une table et afficher une image.
- Vous êtes libres à ajouter les éléments de styling (CSS) que vous jugez nécessaires.

Analyse du document HTML

Un dcoument HTML se compose d'un ensemble de noeuds. Ces noeuds sont structurés sous forme hiérarchique. Cette stucture repose sur le modèle DOM. Pour pouvoir manipuler dynamiquement le

contenu d'une page HTML, cette structure hiérarchique peut être utilisée.

Dans cette série, l'objectif est d'analyser le document créé dans la première étape et d'extraire sa structure hiérarchique.

Création d'un premier document XML

Comme son nom l'indique (eXtensible Markup Language), XML repose sur le même principe des balises que HTML. La différence réside dans l'ensemble des balises utilisées : au moment où HTML définit un ensemble précis de balises, XML donne la liberté au développeur pour créer les balises selon son modèle de données ou bien l'objectif de son utilisation. Cette carctéristique permet à XML de trouver des applications plus variées que HTML qui se limite à la création des pages web.

Dans la troisième étape, proposez un modèle de données puis implémenter le en utilisant un fichier XML. Le DOM créé doit être d'une profondeur 2 au minimum.

Parser le document XML créé

Pour pouvoir le manipuler, un document XML peut être "parsé" pour extraire sa structure DOM. Comme les autres langages de programmation, XML doit être analysé avant d'être traité. La différence réside dans le résultat de ce traitement. Dans le cas d'un langage de programmation général, le résultat est un code binaire exécutable. Dans le cas d'un document XML, le résultat sera une strcuture de données qui contiendra l'ensemble des données contenues dans le fichier.

En Java, les objets en relation avec le traitement des fichier XML sont :

- **Node** : l'élément de base,
- **Document** : tous les noeuds du fichier XML,
- **Element** : un élément du fichier XML,
- **Attr** : un attribut d'un élément du fichier XML,
- **Text** : le contenu d'un élément XML.

Avant d'accéder à ces objets, il est nécessaire de créer un parser et de lui passer le fichier XML. En Java, la méthode la plus simple consiste à utiliser "Document Builder" :

```
File fichierXml = new File("exemple.txt");
DocumentBuilderFactory builderFactory =
                        DocumentBuilderFactory.newInstance();
DocumentBuilder builder = builderFactory.newDocumentBuilder();
```

Ensuite, il suffit de lui passer le fichier de la manière suivante :

```
Document document = builder.parse(fichierXml);
```

Il est aussi possible de lui passer une chaîne de carcatères, c'est-à-dire, vous pouvez lire vous même le fichier ou bien lire un flux de données à partir de la ligne de commande et ensuite vous paasez ce contenu au parseur.

Les méthodes les plus courantes pour accéder et manipuer les différents éléments et leurs attributs sont :

- L'objet Document :
  - getDocumentElement() : pour récupérer la racine du document.
- L'objet Node :
  - getFirstChild() : pour récupérer le premier noeud fils,
  - getLastChild() : pour récupérer le dernier neuds fils,
  - getNextSibling() : pour récupérer le noeud suivant du même niveau,
  - getPreviousSibling() : pour récupérer le noeud précédent du même niveau,
  - getAttribute(nom_attribut) : pour récupérer la valeur de l'attribut "nom_attribut",
  - getChildren() : pour récupérer tous les noeuds fils.
- Accès au contenu :
  - getTextContent()
  - getText()

Travail demandé

- Essayez l'exemple ci-dessous,
- Modifier le code pour parcourir le documents autrement (getChildNodes() par exemple)

Exemple de parcour

Soit le fichier XML suivant

```
<?xml version="1.0" ?>
<informatique>
    <licences>
        <licence>Systèmes informatiques</licence>
    </licences>
    <masters>
        <master>SIAD</master>
        <master>R et S</master>
        <master>IA</master>
        <master>ILM</master>
    </masters>
</informatique>
```

Ce fichier peut être traité par le code suivant (Attention : enlevez tous les espaces bancs et les nouvelles lignes pour éviter une erreur) :

```
import java.io.File;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import org.w3c.dom.Document;
import org.w3c.dom.Node;

public class Main {
    public static void main(String[] args) {
        try {
            DocumentBuilderFactory dbf =
                        DocumentBuilderFactory.newInstance();
            DocumentBuilder db = dbf.newDocumentBuilder();
            Document d = db.parse(new File("Exemple.xml"));

            Node racine = d.getDocumentElement();
            Node licence_informatique = racine.getFirstChild();

            System.out.println("Licence disponible : "
                + licence_informatique
                        .GetFirstChild()
                        .getTextContent());

            Node master = racine.getLastChild().getFirstChild();
            while(master != null){
                System.out.println("Master disponible : "
                        + master.getTextContent());
                master = master.getNextSibling();
            }
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    }
}
```

Le résultat d'excution sera :


dist : bash — Konsole
Fichier Édition Affichage Bookmarks Configuration Aide
-$ java -jar FirstXML.jar
Licence disponible : Systèmes informatiques
Master disponible : SIAD
Master disponible : R et S
Master disponible : IA
Master disponible : ILM
-$
dist : bash