XML : Eléments syntaxiques

GMIN206 (information biologique)

Structure d'un document XML

- En-tête
  - XML, version et codage des caractères

  Exemple :

```
<?xml version="1.0" encoding="UTF-8"?>
```

  seule balise notée par `<? ?>`
- Corps
  - Une balise (ou élément) racine qui contient le reste du document
  - Exemple :

```
<protein> … </protein>
```

Eléments d'un document XML

- Une balise est nommée :
  - Par convention avec des minuscules, éventuellement avec des caractères étendus et des chiffres
- Ecriture des balises :

Balises ouvrante `<sequence> et` fermante `</sequence>`

`Balise vide <sequence/>`

- Document bien formé

Exemple :

`<proteine> <sequence> </sequence></proteine>`

Attributs d'éléments

- Eléments vides ou non vides
  - Les attributs sont décrits au sein de la balise
  - Syntaxe : att="val" ou att='val'
  - L'ordre des attributs est sans importance
  - La balise fermante reste une balise simple
  - Exemple :

```
<protein name="Integrin">
    ….
  </protein>
```

Commentaires et caractères spéciaux

- Commenter une partie d'un document en l'entourant de `<!--` et de `-->`
  - Le contenu sera ignoré pour tout traitement
- Insérer dans du texte les caractères suivants : < > &
  - notations :

`<`	(less than)	<
`>`	(greater than)	>
`&`	(ampersand)	&

- Insérer un caractère étendu si on en connaît le code :

`&#233;`	(é)	(décimal)
`&#xE9;`	(é)	(hexadécimal)

Les DTD (grammaires XML)

- Principes
- Syntaxe
- Association document-DTD
- Validation d'un document : la grammaire est un modèle pour une famille de documents

Syntaxe d'une DTD

- Définie dans un fichier à part (extension .dtd)
- Une ligne d'en-tête : idem que pour un document XML

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

  - Note : le codage de caractères est indépendant des fichiers XML associés…
- Puis, dans un ordre quelconque, la déclaration des éléments du document

Déclaration d'un élément

<!ELEMENT nomElt typeElt>

Avec typeElt

<!ELEMENT nomElt (#PCDATA)> : l'élément ne contient que du texte

<!ELEMENT nomElt EMPTY> : l'élément est terminal

<!ELEMENT nomElt (elt1[,elt2,...])> : l'élément contient d'autres éléments, avec possibilité de précision des cardinalités

Structure d'un élément imbriqué

- Quels éléments, combien, et dans quel ordre, composent l'élément en cours de définition

(e1, e2) : un élément e1 puis un élément e2

(e1 | e2) : l'élément e1 ou l'élément e2

(e1+) : 1 ou plusieurs fois l'élément e1

(e1*) : 0 ou plusieurs fois l'élément e1

(e?) : 0 ou 1 fois l'élément e1

  - Les éléments e1 et e2 sont ensuite décrits de la même manière

Exemples

- Eléments complexes :

```
<!ELEMENT protein (categorieProt+, geneCodant?,
  geneRégulé*)>
```

Zéro à plusieurs gènes régulés, un gène codant optionnel

- Eléments `textuels`

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

- `Eléments terminaux`

```
<!ELEMENT geneCodant EMPTY>
<!ELEMENT geneRégulé EMPTY>
```

Exemples

- Document initial :

```
<protein>
<categProt>adhesion</categProt>
</protein>
```

Attributs

- Les attributs : propriétés intrinsèques de l'élément
- Syntaxe :

```
<!ATTLIST elt
          attribut1    type1      option1
          attribut2    type2      option2
          …>
```

Attributs

- Type d'un attribut
  - CDATA : texte non contraint
  - ID : identifiant unique
  - (valeur1 | valeur2 | …) : choix parmi une liste
- Options d'un attribut
  - #REQUIRED : attribut obligatoire
  - #IMPLIED : attribut optionnel
- Exemple :

```
<!ATTLIST geneCodant
   name  CDATA #REQUIRED>
```

Instanciation XML possible :

```
<geneCodant name="ITGB1" />
```

Exemple DTD complète

```
<!ELEMENT protein (categProt+, geneCodant?,
  geneRégulé*)>
<!ATTLIST protein name CDATA #REQUIRED
                  AltName CDATA #IMPLIED>
<!ELEMENT categProt (#PCDATA) >
<!ELEMENT geneCodant EMPTY>
<!ATTLIST geneCodant name CDATA #REQUIRED >
<!ELEMENT geneRégulé EMPTY>
<!ATTLIST geneRégulé name CDATA #REQUIRED >
```

Exemples

- Document valide (respecte la DTD) :

```
<protein name = "integrin beta 1" altName
  = "glycoprotein IIa" >
<categProt>adhesion</categProt>
<geneCodant name="ITGB1" />
</protein>
```

Association DTD-document

- Rien à faire dans la DTD
- Dans l'en-tête du document XML :

```
<!DOCTYPE racine SYSTEM "fichier.dtd">
```

Où racine est le nom de l'élément de plus haut niveau défini dans la DTD

- Il est possible d'utiliser une DTD à distance

```
<!DOCTYPE racine SYSTEM "http://www..dtd">
```

Sequence Tiny DTD (NCBI)

```
<!ELEMENT TSeq (
        TSeq_seqtype,
        TSeq_gi?,
        TSeq_accver?,
        TSeq_sid?,
        TSeq_local?,
        TSeq_taxid?,
        TSeq_orgname?,
        TSeq_defline,
        TSeq_length,
        TSeq_sequence)>
<!ELEMENT TSeq_seqtype %ENUM;>
<!ATTLIST TSeq_seqtype value (
        Nucleotide | protein
        ) #REQUIRED >
```

sequence au format Tiny Seq

```
<TSeqSet>
<TSeq>
<TSeq_seqtype value="nucleotide"/>
<TSeq_gi>182519230</TSeq_gi>
<TSeq_accver>NM_002211.3</TSeq_accver>
<TSeq_taxid>9606</TSeq_taxid>
<TSeq_orgname>Homo sapiens</TSeq_orgname>
<TSeq_defline>Homo sapiens integrin, beta 1 ... mRNA</TSeq_defline>
<TSeq_length>3879</TSeq_length>
….
</TSeq>
</TSeqSet>
```

En complémentarité

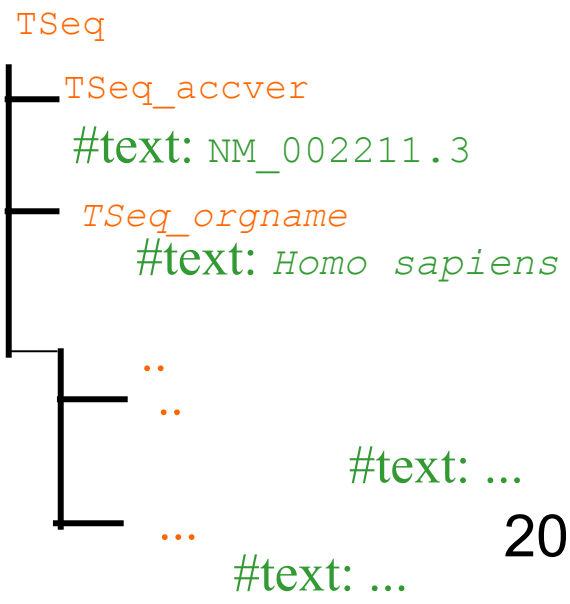
- Echange, documentation
- Langages de programmation : Java, PHP, Python, Perl, etc
- Principes de programmation associés à XML
  - DOM & SAX

Document XML – Flux SAX – Arbre DOM

```
<TSeq>
 <TSeq_accver>
  NM_002211.3
 </TSeq_accver>
 <TSeq_orgname>
  Homo sapiens
 </TSeq_orgname>

</TSeq>
```

startDocument ()
startElement (TSeq)
startElement (TSeq_accver)
characters (NM_002211.3)
endElement (TSeq_accver)
startElement (TSeq_orgname)
characters (Homo sapiens)
endElement (TSeq_orgname)
...



Java et JDOM

- Simplicité de JDOM
  - Utilise les collections SAX pour l'analyse (parsing) des fichiers XML
  - Utilise DOM pour manipuler les éléments d'un Document XML
  - Voir http://cynober.developpez.com/tutoriel/java/xml/jdom/

Exemple fichier xml

- <?xml version="1.0" encoding="UTF-8"?>
- <SeqSet>
- <sequence accNum="BC074923" provenance="GENBANK">
- <organism>Homo sapiens</organism>
- <geneName>SRY</geneName>
- <biomol>MRNA</biomol>
- <seqNuc>caccttcaattttgtcgca</seqNuc>
- </sequence>
- </SeqSet>

Ex. création document XML

```
public class Ex4_C {
   public static void main(String[] args) {
 Namespace ns = Namespace.getNamespace("seqBio","http://www.fdsUm.org");
 Element racine = new Element("biologicalSequence", ns);
 Document document = new Document(racine);
  Element s1 = new Element("sequence1", "seqBio","http://www.fdsUm.org");
  racine.addContent(s1);
  s1.addContent(new Element("mnemonic", ns).setText("HIGTB1"));
  Attribute provenance = new Attribute("provenance","GENBANK");
      s1.setAttribute(provenance);
    // sauvegarde dans un fichier et affichage ecran
    Try { FileOutputStream out = new FileOutputStream("sequencesAvecNS.xml");
      XMLOutputter serializer = new XMLOutputter(Format.getPrettyFormat());
      serializer.output(document, out);
      serializer.output(document, System.out);  out.flush();   out.close();  }
    catch (IOException e) {
      System.err.println(e);}}}
```

Ex. ouverture document XML

```
import java.io.*;
import java.util.*;
import org.jdom2.*;
import org.jdom2.input.*;

public class Ex3_O {
public static void main(String[] args) throws IOException, JDOMException {
SAXBuilder builder = new SAXBuilder();
Document doc = builder.build(new FileInputStream("Exercice2.xml"));
Element root = doc.getRootElement();
System.out.println(root.getName()); // renvoie SeqSet

List<Element> list = root.getChildren();
for (Element x : list) {
System.out.println(x.getAttributeValue("accNum"));
System.out.println(x.getAttributeValue("provenance"));
}}}
```

TP à rendre

- Collection de séquences ou publications au format XML depuis le portail Entrez
- Extraire des informations pertinentes qui viennent alimenter des classes descriptives JAVA (POJO)
- TP suivant : alimenter une BD relationnelle

Ex. classe POJ0

```
public class Sequence
{
        private String  numAcc ;
        private String  organism ;
        private Set lesFeatures = new HashSet();
        public Sequence(String numAcc, String organism ){
                this.numAcc = numAcc;
                this.organism = organism;      }
        public String getNumAcc(){return numAcc;}
        public Set getLesFeatures(){return lesFeatures;}
        public void setNumAcc(String numAcc) {this.numAcc = numAcc;}
        public void setLesFeatures(Set lesFeatures ){
                this.lesFeatures = lesFeatures ;
        }
}
```