PPRO0203 – Projet CMI

Etape 1 : recherches documentaires

Quelles bibliothèques utiliser ?

Le problème de base était de trouver le moyen de parser un document xml (ici contenu dans une archive ODF, semblable à un ZIP) pour le rendre utilisable selon un modèle document-objet (DOM).

La première piste fut donc de comparer les library les plus courantes sur internet : https://www.baeldung.com/java-xml .

Après réflexion, il a semblé judicieux de privilégier un (ou plusieurs) d'entre elles qui ne serait pas trop complexe pour une première prise en main, et qui serait inclue dans le JDK. De ce fait, nous avons choisit le DOM parseur du package org.w3c.dom, contenant tous les objets du DOM tels que Document, Element, Node, etc. Pour le traitement des informations et le parcours du DOM ainsi obtenu nous utiliserons ensuite XPath du package javax.xml.xpath (voir javadoc https://docs.oracle.com/en/java/javase/15/docs/api/jdk.javadoc/module-summary.html).

Documentation complémentaire :

https://www.baeldung.com/java-xpath

https://mkyong.com/java/how-to-read-xml-file-in-java-dom-parser/

https://www.w3.org/TR/2017/REC-xpath-31-20170321/#id-named-function-ref

Enfin, pour analyser le flux de l'archive (ou éventuellement la décompressé), a été choisi une bibliothèque indépendante : Zip4J https://github.com/srikanth-lingala/zip4j

Premier essai

Avant d'aller plus loin, voyons déjà si l'utilisation des bibliothèques choisies est opérationnelle.

Voici l'exemple de référence sur lequel nous allons nous appuyer pour réaliser l'architecture commune de notre application :

```java
    try
    {
        System.out.println("Loading...");
        zip = new ZipFile(odfFilePath);
        var header = zip.getFileHeader("content.xml");
        documentBuilder = DocumentBuilderFactory.newInstance().newDocumentBuilder();
        System.out.println("Parsing...");
        var is = zip.getInputStream(header);
        document = documentBuilder.parse(is);
        is.close();
        System.out.println("Document created with success.");
    }
    catch(Exception e)
    {
        System.out.println("\nFailure");
        System.out.println("------------------------");
        e.printStackTrace();
        return;
    }
    try
    {
        System.out.println("XPath expression generating...");
        xpath = XPathFactory.newInstance().newXPath();
        String exp = "/document-content/*";
        var res = xpath.compile(exp).evaluate(document, javax.xml.xpath.XPathConstants.NODESET);
        System.out.println("\nResults :");
        System.out.println("------------------------");
        if(res instanceof NodeList)
        {
            NodeList nodes = (NodeList)res;
            for(int i=0; i<nodes.getLength(); i++)
            {
                System.out.println(nodes.item(i).getNodeName());
            }
        }else{
            System.out.println("Extraction from NodeList failed.");
        }
    }
    catch(Exception e)
    {
        System.out.println("\nFailure");
        System.out.println("------------------------");
        e.printStackTrace();
        return;
    }
    System.out.println("\nProgram run succesfully");
}
```

Arborescence

Ci-contre, l'arborescence générale des fichiers :

```
FICHIER.ODF
    |
    +-- META-INF/
    |       |
    |       +-- manifest.xml
    |
    +-- Thumbnails/
    |       |
    |       +-- thumbnails.png
    |
    +-- Configuration2/
    |       |
    |       +-- [...]
    |
    +-- Pictures/
    |
    +-- Basics/
    |
    +-- meta.xml
    |
    +-- content.xml
    |
    +-- settings.xml
    |
    +-- styles.xml
    |
    +-- mimetype
    |
    +-- manifest.rdf
    |
    +-- layout-cache
```

Architecture XML dans les fichiers ODT

Content.xml

Le fichier content.xml contient les informations de base du document : il regroupe le contenu de la page et sa structure, sous la forme d'un arbre XML.

Meta.xml

Le fichier meta.xml contient les métadonnées liées au document et à l'utilisateur : date de création, dernière modification, auteur initial, auteur, langue, etc...

Styles.xml

Le fichier styles.xml contient les informations liées à la mise en page du document, à son apparence. Il permet notamment de déclarer les « Styles » de paragraphes : taille de police, couleur, etc...

Settings.xml

Le ficher settings.xml contient le reste des informations liées au logiciel, enregistrées localement pour tel ou tel document : par exemple, le zoom de la page, le mode de lecture, etc...

Les actions de l'utilisateur ?

Base commune

Pour le fichier ODT