STI 4e année – Java avancée
TD: SAX

Ce TD vous est proposé par Cyril M. J. Dejonghe et J.Francois Lalande.

1 SAX

Dans ce TD, nous proposons d'étudier le parseur SAX. Pour cela, il faut se réferer à l'interface ContentHandler dont une implémentation DefaultHandler. Dans DefaultHandler, il y a notamment :

- `void startDocument()` : recoit la notification du début de document
- `void endDocument()` : recoit la notification de la fin de document
- `void startElement(String uri, String localName, String qName, Attributes atts)` : recoit la notification d'un début d'élément (balise).
- `void endElement(String uri, String localName, String qName)` : recoit la notification de la fin d'un élément (balise)
- `void characters(char[] ch, int start, int length)` : recoit la notification de caractéres (il donne l'endroit ou il est rendu avec l'entier start, et la longueur de la chaine avec length)

Cette interface permet de faire des actions quand un événement se produit pendant le parsing du document XML.

Question 1 Créer un petit document XML file.xml avec plusieurs balises imbriquées dont la balise "<truc>".

Question 2 Réalisez le handler MyHandler qui hérite de DefaultHandler et qui affiche "Bonjour" quand le début du document est rencontré.

Question 3 Récupérez Xerces-J-bin.2.9.0.zip sur le site de Xerces. Incluez dans votre projet xercesImpl.jar. A ce stade, vous pouvez créer un parser SAX en utilisant l'implémentation de Xerces :

```
XMLReader parser = XMLReaderFactory.createXMLReader("org.apache.xerces.parsers.SAXParser");
```

Cette ligne, ne doit plus faire d'exception si xerces est correctement inclu dans votre projet.

Question 4 Changez le handler de votre parser en utilisant la méthode setContentHandler().

Question 5 Appelez la méthode parse() sur FileInputStream utilisant un InputStream. Cela va parser votre fichier file.xml.

Question 6 Réalisez enfin le parser qui permet d'afficher le contenu de toutes les balises "<truc> ... </truc>" mais pas des autres balises, en modifiant votre classe MyHandler.

2 Log4j

Pour comprendre log4j, se référer au tutoriel situé à : `http://supportweb.cs.bham.ac.uk/docs/tutorials/docsystem/build/tutorials/log4j/log4j.html`

Question 7 Allez lire le début de la FAQ de log4j. Pourquoi n'est ce pas dangereux d'utiliser log4j dans votre programme si vous vous référez à la question 2 de la FAQ ?

Question 8 Ajoutez log4j dans votre projet de parseur XML (log4j-1.2.15.jar).

Question 9 Créez un logger et ajoutez lui un Appender qui écrit dans un fichier avec un layout de type SimpleLayout.

Question 10 Loggez un message de niveau INFO au début de votre main.

Question 11 Provoquez une exception de type "fichier XML mal formé". Récupérez la et utilisez et loggez là au niveau FATAL.

Question 12 Vérifiez que les deux types de logs apparaissent dans votre fichier. Ajoutez alors un filtre sur le logger pour ne plus afficher que le niveau supérieur à FATAL.

Question 13 Récupérez JavaMail 1.4.1 (Sun) et ajoutez mail.jar. Ajoutez un Appender de type SMTPAppender pour le niveau Fatal (méthode setThreshold()). Configurez le pour avoir le bon From :, To :, Subject :, et SMTPHost. N'oubliez pas d'appeler la méthode activateOptions() pour qu'il prenne en compte vos modifications. Vous devez maintenant recevoir des mails en cas d'erreur FATAL !

Remarque final : au lieu de configurer log4j par le code, il est plus malin de faire un fichier de configuration XML de votre logger (plus simple à écrire et modifiable sans retoucher le code java (et sans recompiler !)). Il suffit alors d'appeller la méthode configure() sur un configurateur de logger pour qu'il lui attribue les options que vous avez défini dans le fichier de conf.

Question 14 Essayez de le faire !