IFT 1020
Programmation II

Etienne Bergeron

7 juillet 2005

Travail pratique #4
Technique de recherche, Entrées/Sorties
et Mécanismes d'exception

1 Explications générales

Équipe : Le travail est à faire en binôme ou en monôme, mais pas plus de deux. Vous **devez** changer de partenaire pour faire votre travail.

Remise : La remise électronique et papier doit se faire au plus tard le 28 juillet au début de l'examen final.

Conseil : N'attendez pas la dernière minute pour faire votre travail, on ne sait jamais les surprises qui nous attendent ! Aussi, lisez **attentivement** l'énoncé et prenez bien conscience du problème avant de vous lancer dans la programmation.

2 But

Le but du travail est de réaliser un décodeur de code crypté. Il vous permettra de vous familiariser avec les techniques de recherche, le modèle d'entrées/sorties et les mécanismes d'exception de Java.

3 Énoncé

3.1 Mise en situation

Lors de la première guerre mondiale, de nombreux messages cryptés ont été interceptés et conservés dans les archives du département de la défense nationale. Le gouvernement a décidé de rendre publics tous les messages en les publiant. Le travail étant trop ardu pour être réalisé à la main, vous avez été engagé pour écrire un programme qui décode automatiquement les messages et les formate correctement, prêts pour la publication. Une équipe d'informaticiens ont construit une base de données XML contenant l'ensemble des communications.

3.2 Exécution de l'application

Votre programme doit décoder les messages selon les paramètres donnés à votre application sur la ligne de commande. Si l'élément est un fichier, il est décodé. Si l'élément est un répertoire, votre programme parcours récursivement le répertoire et décode les fichiers. Si aucun élément n'est passé en paramètre, le répertoire courant est supposé comme paramètre, et ce répertoire est parcouru récursivement.

Lorsque le traitement d'un répertoire ou d'un fichier génère une erreur (exception de n'importe quel type) vous devez écrire sur la sortie standard le nom du fichier en erreur et continuer **normalement** le traitement sur le prochain fichier. Un fonctionnaire du gouvernement, après une de ses nombreuses pauses, s'occupera de régler les problèmes avec les messages qui comportent des erreurs. Votre application doit continuer le traitement pour convertir le maximum de fichiers possible et ainsi éviter la lenteur administrative.

```
% java Tp4
% java Tp4 .
% java Tp4 test message1.xml
```

Dans le premier cas, votre application décrypte récursivement tous les fichiers dans le répertoire courant (ainsi que les sous-répertoires). Le deuxième cas est équivalent au premier cas. Dans le troisième cas, votre application décrypte tous ces fichiers contenus dans le répertoire `test` et décrypte le fichier `message1.xml`.

3.3 Extraction

La première phase de traitement consiste à extraire le texte contenu dans la base de données. Les balises XML n'ont pas de sens pour le traitement que vous devez appliquer au contenu du fichier. On vous demande d'extraire tous les tags XML et de ne conserver que le texte entre les balises. De plus, vous devez remplacer les caractères spéciaux de XML.

Pour réaliser cette tâche, on vous demande d'écrire une classe `XMLFilter` qui étend (`extends`) la classe abstraite `Reader`. Ainsi, vous pourrez utiliser votre filtre de la façon suivante :

```
BufferedReader read = new BufferedReader(
                        new XMLFilter(
                         new FileReader( fichier )));
String line = read.readLine();
System.out.println("Ligne:" + line );
```

Vous pouvez supposer que le fichier est du XML valide. Donc, tous les caractères spéciaux sont encodées correctement. Pour enlever les tags, il suffit d'enlever tout ce qui est entre les caractères < et >.

Pour les caractères spéciaux, vous pouvez vous limiter aux caractères accentués français, les caractères < et >, l'espace, et les caractères numérotés.

Entité	Caractère
É	É
é	é
>	>
<	<
A	A
a	a

Exemple d'entités XML.

Le code ci-dessous devrait extraire le texte suivant : "Tourlou, Étienne".

```
BufferedReader read =
  new BufferedReader(
    new StringReader("<TOP><MESSAGE id=\"1\">Tourlou, " +
                     "&Eacute;tienne</MESSAGE></TOP>"));
String line = read.readLine();
System.out.println( line );
```

3.4 Décryption

Un fichier nommé "decode.txt", présent dans chaque répertoire contient les règles de substitution vous permettant de décrypter les messages d'un répertoire. En l'absence de ce fichier, une exception est lancée et le répertoire n'est pas traité (par contre, les sous-répertoires sont tout de même traités).

Vu le grand nombre de messages à traduire, la performance de votre algorithme de conversion est importante. On vous demande donc d'implanter un algorithme de tri et un algorithme de recherche **efficaces** pour trouver rapidement un patron et son résultat équivalent.

Chaque ligne du fichier contient un patron d'encodage suivi de son patron de décodage.
Contenu du fichier "decode.txt" :

```
alpha allo
bravo bonjour
charlie coucou
lucie et
octave final
demo bergeron
prof lokbani
[...]
```

3.5 Processus de conversion

Les messages sont contenus dans des fichiers portant l'extension ".xml". Votre programme devra convertir ce fichier et produire le résultat dans un fichier portant l'extension ".txt" (en convervant le même prefix et le même répertoire).

La conversion de message consiste à découper le contenu du fichier en patron séparé par des espaces ou caractères de ponctuation, et ensuite à trouver le patron correspondant. Il n'y a pas de conversion à faire pour les retours de chariot et les signes de ponctuation, ils sont simplement remis tels quels dans le fichier produit.

Voici le contenu du fichier "Exemple.xml" :

```
<message id=1>alpha prof lucie demo!!!</message>
```

et voici le résultat du fichier "Exemple.txt" produit par votre application :

```
allo lokbani et bergeron!!!
```

Si une erreur survient, la conversion du fichier est interrompue.

4 Rapport

Le présent travail est destiné à un milieu gouvernemental. Donc, il est important de bien documenter votre programme et de fournir un bon rapport. Votre salaire dépend de votre capacité à expliquer ce que vous avez réalisé.

Votre programme **doit** être documenté avec les commentaires à la `javadoc`. La remise de la `javadoc` n'est pas obligatoire. De plus, vous devez fournir un document en deux sections. La première est un manuel de l'utilisateur et la seconde un manuel du programmeur.

Le manuel de l'utilisateur explique aux fonctionnaires qui vont utiliser votre programme comment fonctionne votre programme et ce qu'il fait. Vous ne devez pas prendre pour acquis qu'il a lu l'énoncé. Le manuel du programmeur consiste à expliquer la structure de votre programme. Un programmeur ayant de bonnes connaissances en Java devrait pouvoir comprendre rapidement la structure de votre programme et les algorithmes utilisés. De plus, on vous demande d'analyser la performance de vos algorithmes de tri et de recherche.

5 Remise

Votre programme principal **doit** porter le nom `Tp4` et **doit** être contenu dans le fichier `Tp4.java`. Vous pouvez remettre plusieurs fichiers.

Il est important de noter que votre TP sera exécuté par les démonstrateurs sur une machine Linux avec le compilateur jdk (1.4.2_01). Si par choix vous décidez d'utiliser un autre compilateur, vérifiez que le code que vous avez produit (qui normalement fonctionne correctement chez vous) fonctionne bien sur les ordinateurs du DIRO. Avant de faire cela, assurez vous d'abord que vous avez la bonne version de jdk, en utilisant pour cela la commande : "java −version" devra donner le numéro de version "1.4.2_01".

La remise comprend 3 (TROIS) choses :

1. Envoyez vos fichiers par la procédure de remise électronique habituelle (Pour obtenir de l'aide sur cette commande, lancer la commande : `man remise`).
   Respectez les noms des fichiers. Cette remise est due pour le **28 juillet avant 17h30**

   remise dift1020 Tp4.java AutreFichier.java Foo.java Bidon.java

2. Remettez une copie papier d'un rapport qui devrait décrire votre programme. Cette remise doit être faite le **28 juillet 2005 avant 17h30**,

6 Barème

Ce TP4 est noté sur 12 points. Le barème est le suivant :

1. **Code**
   (a) 2 points sur le fonctionnement de la conversion (filtre, décryptage).
   (b) 2 points sur l'algorithme de tri et de recherche.
   (c) 2 points sur le parcours récursif des fichiers.
2. **Rapport**
   (a) 2 points sur le manuel de l'utilisateur.
   (b) 2 points sur le manuel du programmeur.
   (c) 2 points sur la qualité de présentation.

À noter que :

1. La non remise électronique (volontaire ou par erreur) est sanctionnée par la note 0.
2. Un programme qui ne compile pas : note 0.
3. Un programme qui compile mais ne fait pas les choses prévues dans la spécification : note 0.
4. La non remise papier vous pénalise de 6 point.
5. Les programmes ne contenant pas d'en-tête, -4 points.
6. Le code doit être lisible, bien indenté et commenté.

ATTENTION : Aucun plagiat ne sera toléré.
Citez vos sources correctement.

7 Des questions à propos de ce TP ?

L'adresse email : dift1020@iro.umontreal.ca
Pour faciliter le traitement de votre requête, inclure dans le sujet de votre email, au moins la chaîne :
"[IFT1020]"

8 Mise à jour

07-07-2005 diffusion