TP 7 : Les entrées/sorties en Java

christina.boura@prism.uvsq.fr, stephane.lopes@prism.uvsq.fr

20 mars 2015

Exercice 1 *Ajouter des données à la fin d'un fichier texte*

Le but de cet exercice est d'écrire un programme qui ajoute la phrase "C'est la fin du fichier." à la fin d'un fichier texte. Vous devez réaliser cela de deux manières différentes :

- En utilisant la classe `BufferedWriter` du package `java.io`.
- En utilisant la classe `RandomAccessFile` du package `java.io`, qui permet de faire des lectures ou écritures à une position aléatoire dans un fichier.

Étudiez la javadoc pour choisir les méthodes et les constructeurs qui vous seront utiles. Pour tester vos programmes, créez un fichier texte (extension `.txt`) et écrivez quelque chose dedans. N'oubliez pas d'attraper les exceptions du type `IOException` et de vérifier que les flux ouverts sont bien fermés à la fin de leur utilisation, même en cas de problème.

Exercice 2 *Palindrome*

Un *palindrome* est un texte ou un mot dont l'ordre des lettres reste le même qu'on le lise de gauche à droite ou de droite à gauche, comme dans le mot *kayak* ou dans la phrase *La mariée ira mal*. Pour cet exercice, vous aurez besoin du fichier `palindrome.txt`. Ce fichier contient un nombre de phrases, dont certaines sont des palindromes. Le but est d'afficher à l'écran toutes les phrases du fichier qui représentent un palindrome. Utilisez pour cela les méthodes du package `java.nio.file` (`Paths.get()`, `Files.readAllLines()`, ...).

Exercice 3 *Lire et traiter des informations dans un fichier*

Le but de cet exercice est d'écrire un programme qui lit les informations contenus dans le fichier `NotesEtudiants.txt` et qui applique une analyse aux données. Ce fichier contient sur chaque ligne le nom d'un étudiant ainsi que des notes que cet étudiant a obtenu pendant ces examens. Ces informations sont organisés de la façon suivante :

```
Marie 17.5 12 13.5 14
Damien 9 14 11 10.5
Lisa 7 9.5 13 17.5 13.5
```

Le programme doit afficher à l'écran le nom de l'étudiant ayant la moyenne la plus élevée. Utilisez pour cela la classe `BufferedReader`.

Exercice 4 *Sérialisation*

On veut écrire un programme pour noter dans un fichier les noms et les scores sous la forme d'une suite de lignes ; chaque ligne contient un nom et un entier ; l'entier représente un score obtenu par la personne dont le nom est indiqué. On aura par exemple :

```
Aline 13
Steven 9
Dris 15
```

Le programme va permette de prendre à partir de l'entrée standard des lignes constituées d'un nom et d'un score pour actualiser le fichier.

— Si le nom n'existait pas, une nouvelle ligne est créée pour mettre ce nouveau nom et son score.
— Si le nom existait déjà, le score est actualisé.

Après chaque ajout ou modification de score, le fichier sera actualisé et son contenu sera affiché. L'utilisateur indiquera sur la ligne de commande le nom du fichier utilisé. Si le fichier n'existait pas, il sera alors créé. Vous devez créer une classe, nommée par exemple Joueur contenant deux attributs : un pour un nom (chaîne de caractères), un pour un score (entier) et d'écrire dans le fichier directement les instances de Joueur, comme des Object(s) ; bien entendu, on lira alors dans le fichier des Object(s). Dans ce cas, on ne pourra pas lire le fichier directement, visuellement, en l'ouvrant.

Les étapes à réaliser sont donc :

— Créez une classe `public class Joueur implements Serializable`, avec deux attributs : une chaîne de caractères pour le nom du jouer et un entier pour le score.
  — La classe `Joueur` aura un constructeur qui doit prendre en paramètre une chaîne de caractères. Vous pouvez utiliser une instance de la classe `StringTokenizer` pour séparer la chaîne de caractères en mots (la javadoc vous sera encore utile).
— Créez une classe `public class Enregistreur` avec un seul attribut de type `File` :
  — Cette classe aura un constructeur prenant comme paramètre une chaîne de caractères et qui va correspondre au nom d'un fichier.
  — Créez une méthode

    `void ecrire(Joueur nouveauJoueur) throws Exception`,

    qui va enregistrer un nouveau joueur et son score dans le fichier ou qui mettra à jour le score d'un joueur déjà présent. Pour cette méthode, créez une instance `entree` de la classe `ObjectInputStream`, qui sera initialisée de la façon suivante :

    ```
    entree = new ObjectInputStream (new FileInputStream(fichier));
    ```

    où `fichier` est l'attribut de la classe. Créez également une instance `sortie` de la classe `ObjectOutputStream`, qui sera initialisée de la même manière, mais avec un fichier temporaire. Vous pouvez utiliser les méthodes `readObject()` et `writeObject()` pour lire et écrire sur les fichiers. À la fin de la méthode, copiez le contenu du fichier temporaire dans le fichier principal et effacez le fichier temporaire. Pensez à attraper les exceptions, notamment celles de type `FileNotFoundException` et `EOFException`.
  — Créez une méthode

    `void afficher() throws Exception`,

    qui affichera le contenu du fichier à la sortie standard. L'affichage pourrait avoir la forme suivante :

    ```
    Contenu du fichier
    Nom : Maria, score : 5
    Nom : Antoine, score : 15
    ```

— Créer la methode `main` :
  — Créer une instance de la classe `Enregistreur`, en utilisant le nom d'un fichier contenu dans la case `args[0]` du tableau passé en paramètre à la méthode `main`. Lisez ensuite les noms et les scores tapés sur l'entrée standard. Vous pouvez utiliser les méthodes de la classe `Scanner`, notamment, la méthode `nextLine()`.