TP4 Les API d'entrée sortie

OBJECTIF DU TP

- Enregistrer les données dans un fichier.

Objectifs spécifiques de ce TP

- Lire un flux de données d'entrée à partir d'un fichier
- Créer un répertoire
- Créer un fichier

Travail à faire

Malgré que les bases de données sont devenus le moyen le plus utilisé pour stocker les données sur un support secondaire, dans certaines applications informatiques nous utilisons parfois les fichiers, comme solution pour enregistrer les informations. En effet, la plus part des échanges informatisés entre les applications se font sous forme de fichiers (import-export). En plus, certains formats normalisés de fichier permettent la publication et l'impression de ces données plus facilement :pdf, html, xml,….

1. Créez le projet TP7

   - Créez un nouveau package : **importation**
   - **Dans le package importation, créez la classe Etudiant suivante :**

Etudiant
- nom : chaine de caractères - prenom : chaine de caractères + constructeur(…..) +toString() :String

   - Dans le package **importation**, créez une nouvelle classe java : **FichierEtudiant**

FichierEtudiant
- Repertoire : chaine de caractères - nomFichier : chaine de caractères
+ importer () :TreeSet<Etudiant>

   - Proposez un constructeur ayant 2 paramètres (tous les attributs)
   - Codez la méthode **importer** :
     - Choisissez pour le type de retour un TreeSet d'Etudiant

- Déclarez un objet r du même type de retour que la méthode importer et n'oubliez pas de le construire.

```
/**
 * extrait la liste des étudiants du fichier à importer
 *
 * @return
 */
public TreeSet<Etudiant> importer() {
    TreeSet<Etudiant> r = new TreeSet<>();

    return r;
}
```

- Construisez un objet File avec le nom du répertoire (attribut)

```
//construire un objet file avec le nom du repertoire
File leRepertoire = new File(Repertoire);
```

- Vérifiez si l'objet File ainsi construit référencie bien un répertoire alors construisez un objet File dans ce répertoire, sinon dans le répertoire courant.

```
//construire un objet file avec le nom du repertoire
File leRepertoire = new File(Repertoire);
File leFichier;
//vérifier si le File construit est bien un repertoire
if (leRepertoire.isDirectory()) {
    //construire un objet File avec nomRepertoire et nomFichier
    leFichier = new File(leRepertoire, nomFichier);
} else {
    //sinon construire un objet File dans le repertoire courant
    //et avec le nom du fichier
    leFichier = new File(nomFichier);
}
```

- Construisez un objet FileReader sur l'objet File leFichier. Pour pouvoir lire à partir de ce fichier. Attention le constructeur lance une exception qu'il faut gérer.

```
//construire un objet FileReader sur File
FileReader fr = new FileReader(leFichier);
```

- Construisez un objet flux d'entrée (BufferedReader) dans lequel nous allons lire les données à partir du FileReader : fr.

```
//construire un objet flux d'entrée (bufferedReader)
//à partir du FileReader
BufferedReader entree = new BufferedReader(fr);
```

- Parcourrez la totalité du fichier en lisant les lignes avec la méthode readLine du BufferedReader. Le résultat sera retourné sous forme de chaine de caractères.

```
String ligne = null;

//parcours total de tous les enregistrements (les lignes dans le fichier)
do {
    //lecture d'une chaine avant un saut de ligne
    ligne = entree.readLine();
} while (ligne != null);
```

- Nous allons utiliser un objet de la classe StringTokenizer, pour pouvoir extraire des sous chaines séparées par des tokens. dans notre cas le token est \t : tabulation.

```
String ligne = null;
StringTokenizer st;
String nom, prenom;

//parcours total de tous les enregistrements (les lignes dans le fichier)
do {
    //lecture d'une chaine avant un saut de ligne
    ligne = entree.readLine();
    if (ligne != null) {
        //construire avec cette chaine : ligne
        //un objet StringTokenizer : une chaine de caractères avec des séparateurs
        //par exemple Ben salem Ali,01234588
        // le premier séparateur c'est \t : tabulation entre le nom et le prénom
        //le deuxième séparateur c'est , entre le prénom et le cin
        st = new StringTokenizer(ligne);
        //le premier token: sous chaine séparé par \t
        nom = st.nextToken("\t");
        //le deuxième token: sous chaine séparé par \t
        prenom = st.nextToken("\t");
    }
} while (ligne != null);
```

- Construisez un objet Etudiant avec le nom et prénom ainsi obtenus et l'ajoutez à la liste r.

```
//construire un objet etudiant et l'ajouter au TreeSet r
e = new Etudiant(nom, prenom);
r.add(e);
```

- N'oubliez pas de fermer le buffer de lecture une fois le parcours est terminé avec la méthode close.
- Testez la classe FichierEtudiant dans une autre classe main : **TestFichierEtudiant**. Le fichier à utiliser est ***etud.txt*** fournit avec l'atelier. Le répertoire c'est là où vous l'avez placé.

La classe FichierEtudiant :

```
package Importation;

import Inscription.Etudiant;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.StringTokenizer;
import java.util.TreeSet;

/**
 *
 * @author MARIEM
 */
public class FichierEtudiant {

    //nom du fichier à importer
    private String nomFichier;
    //nom du repertoire
    private String Repertoire;

    /**
     * @param nomFichier nom du fichier à importer avec extention
     * @param Repertoire nom du repertoire dans lequel se trouve le fichier à
     * importer chemin absolu exemple d:/listes/
     */
    public FichierEtudiant(String nomFichier, String Repertoire) {
        this.nomFichier = nomFichier;
        this.Repertoire = Repertoire;
    }
```

```java
/**
 * extrait la liste des étudiants du fichier à importer
 * @return
 */
public TreeSet<Etudiant> importer() {
    TreeSet<Etudiant> r = new TreeSet<>();
    //construire un objet file avec le nom du repertoire
    File leRepertoire = new File(Repertoire);
    File leFichier;
    //vérifier si le File construit est bien un repertoire
    if (leRepertoire.isDirectory()) {
        //construire un objet File avec nomRepertoire et nomFichier
        leFichier = new File(leRepertoire, nomFichier);
    } else {
        //sinon construire un objet File dans le repertoire courant
        //et avec le nom du fichier
        leFichier = new File(nomFichier);
    }

    try {
        //construire un objet FileReader sur File
        FileReader fr = new FileReader(leFichier);
        //construire un objet flux d'entrée (bufferedReader)
        //à partir du FileReader
        BufferedReader entree = new BufferedReader(fr);

        String ligne = null;
        StringTokenizer st;
        String nom, prenom;
         Etudiant e;
```

```
            //parcours total de tous les enregistrements (les lignes dans le fichier)
            do {
                //lecture d'une chaine avant un saut de ligne
                ligne = entree.readLine();
                if (ligne != null) {
                    //construire avec cette chaine : ligne
                    //un objet StringTokenizer : une chaine de caractères avec des séparateurs
                    //par exemple Ben salem Ali,01234588
                    // le premier séparateur c'est \t : tabulation entre le nom et le prénom
                    //le deuxième séparateur c'est , entre le prénom et le cin
                    st = new StringTokenizer(ligne);
                    //le premier token: sous chaine séparé par \t
                    nom = st.nextToken("\t");
                    //le deuxième token: sous chaine séparé par \t
                    prenom = st.nextToken("\t");
                    //construire un objet etudiant et l'ajouter au TreeSet r
                    e = new Etudiant(nom, prenom);
                    r.add(e);
                }
            } while (ligne != null);

            //le fichier doit être fermé sinon
            //il sera considéré comme étant ouvert par une autre application
            entree.close();
        } catch (FileNotFoundException ex) {
            System.err.println("fichier non trouvé " + ex.getMessage());
        } catch (IOException ex) {
            System.err.println("fermeture impossible " + ex.getMessage());
        }

        return r;
    }
```

La classe TestFichierEtudiant :

```
import Importation.FichierEtudiant;


/**
 *
 * @author MARIEM
 */
public class TestFichierEtudiant {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        //le fichier "etud.txt" est placé sur d:/
        FichierEtudiant f=new FichierEtudiant("etud.txt", "d:/");
        System.out.println(f.importer());
    }

}
```