Développement en JAVA

Système de fichier

Agenda

- Notion de fichier
- La classe FileInputStream
- La classe FileOutputStream
- L'interface Serializable

Notion de fichier

- Un fichier en Java est représenté par la classe java.io.File
- Un fichier se caractérise par son chemin sur le système de fichier dont le format peut varier selon l'OS
- La classe Fichier offre un certains nombre de méthode pour inspecter le fichier cible:
  - isDirectory: le fichier est-il un répertoire ?
  - exists: le fichier existe-t-il ?
  - isHidden: le fichier est-il caché ?
  - etc…

Lecture du contenu d'un fichier

- La classe FileReader permet de lire le contenu d'un fichier en mode texte
- Elle hérite de la classe InputStreamReader ce qui lui permet de posséder les méthodes de lecture suivantes:
  - Caractère par caractère avec la méthode int read()
  - Bloc par bloc avec la méthode int read(char[] buff, int offset, int lenght)

La classe InputStream

- La classe InputStream permet de lire du contenu en mode binaire
- C'est une classe abstraite dont hérite notamment la classe FileInputStream qui permet donc de lire dans des fichiers en mode binaire

Ecriture dans un fichier

- La classe FileWriter permet d'écrire dans un fichier en mode texte
- Elle hérite de la classe InputStreamReader ce qui permet de lire le fichier
  - Caractère par caractère avec la méthode int read()
  - Bloc par bloc avec la méthode int read(char[] buff, int offset, int lenght)

La classe OutputStream

- La classe OutputStream permet de lire du contenu en mode binaire
- C’est une classe abstraite dont hérite notamment la classe FileOutputStream qui permet donc d’écrire dans des fichiers en mode binaire

Bufferisation

- La bufferisation est un mécanisme permettant d'améliorer les performances des I/O en créant un tampon entre la mémoire et la disque qui ne travaillent pas là la même vitesse
- La classe java.io.BufferedReader implémente une lecture bufferisée et propose notamment une fonction String readLine() qui permet de lire la prochaine ligne de texte
- La classe BufferedWriter permet de faire l'opération inverse

L’interface serializable

- L’interface serializable est une interface qui ne comporte aucune méthode ni aucun attribut
- Elle permet à la JVM de s’assurer que le programmeur sérialize bien du contenu qui peut l’être
- L’ensemble des attributs de la classe doivent être sérializable
  - Si jamais ce n’est pas le cas, une exception NotSerializableExpcetion sera levée
- Le mot clé transient permet de marquer un attribut comme non sérializable

L’interface serializable

- La sérialisation transforme un objet et un tableau d’octets
- Notion de serialVersionUID

Serialization d'un object

- La classe ObjectOutputStream permet de sérialiser un objet, elle se construit à partir d'un objet OutputStream

```
File fichier = new File("tmp/marin.ser") ;
// ouverture d'un flux sur un fichier
ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(fichier)) ;

// création d'un objet à sérializer
Marin m = new Marin("Surcouf", "Robert") ;

// sérialization de l'objet
oos.writeObject(m) ;
```

Déserialization d'un object

- La classe ObjectInputStream permet de désérialiser un objet. Elle se construit à partir d'un objet de type InputStream

```
// dans une méthode main
File fichier = new File("tmp/marin.ser") ;
// ouverture d'un flux sur un fichier
ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(fichier)) ;
// désérialization de l'objet
Marin m = (Marin)ois.readObject() ;
System.out.println(m) ;
// fermeture du flux dans le bloc finally
```