**Les flux et les fichiers (suite)**

**5. Création d’un fichier texte**

Pour créer un fichier texte, Il faut instancier un objet de la classe ***FileWriter*** qui hérite de la classe ***OuputStreamWriter*** et qui possède plusieurs constructeurs :

|  |  |
| --- | --- |
| Constructeur | Rôle |
| FileWriter(String) | Si le nom du fichier précisé n'existe pas alors le fichier sera créé. Si il existe et qu'il contient des données celles-ci seront écrasées. |
| FileWriter(File) | Idem mais le fichier est précisé avec un objet de la classe File. |
| FileWriter(String, boolean) | Le booléen permet de préciser si les données seront ajoutées au fichier (valeur true) ou écraseront les données existantes (valeur false) |

Dans certains cas, ***FileWriter*** se révèle suffisante. Mais, si l’on souhaite disposer de possibilité de formatage, on peut recourir à la classe ***PrintWriter*** qui dispose d’un constructeur recevant en argument un objet de type ***FileWriter****.*

***PrintWriter*** sortie = ***new*** ***PrintWriter*** (***new FileWriter*** (nomfich)) ;

**Exemple:** Il s'agit d'un programme qui demande un entier et qui écrit (cet entier a pour carré.....) dans un fichier texte.

**import** java.io.\* ;

**public** **class** Crftxt1

{ **public** **static** **void** main (String args[]) **throws** IOException

{

String nomfich ;

**int** n ;

System.*out*.print ("Donnez le nom du fichier à créer : ") ;

nomfich = clavier.*lireString*() ;

PrintWriter sortie = **new** PrintWriter (**new** FileWriter (nomfich)) ;

**do**

{ System.*out*.print ("donnez un entier : ") ;

n = clavier.*lireInt*() ;

**if** (n != 0)

{ sortie.println (n + " a pour carré " + n\*n) ;

}

}

**while** (n != 0) ;

sortie.close () ;

System.*out*.println ("\*\*\* fin création fichier \*\*\*");

}

}

**6. Lecture d’un fichier texte**

Pour lire un fichier texte, il faut instancier un objet de la classe ***FileReader*** qui hérite de la classe ***InputStreamReader*** et qui possède plusieurs constructeurs pouvant tous lever une exception de type ***FileNotFoundException*** et la classe ***BufferedReader*** comme suit :

***BufferedReader*** entree = ***new BufferedReader*** (***new FileReader*** (nomfich)) ;

Une fois les traitements sur le flux terminés, il faut libérer les ressources qui lui sont allouées en utilisant la méthode *close*().

Exemple :

**import** java.io.\* ;

**public** **class** Lecftxt1

{ **public** **static** **void** main (String args[]) **throws** IOException

{

String nomfich ;

String ligne ;

System.*out*.print ("Donnez le nom du fichier a lister : ") ;

nomfich = clavier.*lireString*() ;

BufferedReader entree = **new** BufferedReader (**new** FileReader (nomfich)) ;

**do**

{ ligne = entree.readLine() ;

**if** (ligne != **null**) System.*out*.println (ligne) ;

}

**while** (ligne != **null**) ;

entree.close () ;

System.*out*.println ("\*\*\* fin liste fichier \*\*\*");

}

}

**7. La gestion des fichiers**

Les fichiers et les répertoires sont encapsulés dans la classe ***File*** du package ***java.io***. Il n'existe pas de classe pour traiter les répertoires car ils sont considérés comme des fichiers. Une instance de la classe ***File*** est une représentation logique d'un fichier ou d'un répertoire qui peut ne pas exister physiquement sur le disque.

Si le fichier ou le répertoire existe, de nombreuses méthodes de la classe ***File*** permettent d'obtenir des informations sur le fichier. Sinon plusieurs méthodes permettent de créer des fichiers ou des répertoires. Voici une liste des principales méthodes :

|  |  |
| --- | --- |
| **Méthode** | **Rôle** |
| boolean canRead() | indique si le fichier peut être lu |
| boolean canWrite() | indique si le fichier peut être modifié |
| boolean createNewFile() | création d'un nouveau fichier vide |
| File createTempFile(String, String) | création d'un nouveau fichier dans le répertoire par défaut des fichiers temporaires. Les deux arguments sont le nom et le suffixe du fichier. |
| File createTempFile(String, String, File) | création d'un nouveau fichier temporaire. Les trois arguments sont le nom et le suffixe du fichier et le répertoire. |
| boolean delete() | détruire le fichier ou le répertoire. Le booléen indique le succès de l'opération |
| boolean exists() | indique si le fichier existe physiquement |
| String getAbsolutePath() | renvoie le chemin absolu du fichier |
| String getPath | renvoie le chemin du fichier |
| boolean isAbsolute() | indique si le chemin est absolu |
| boolean isDirectory() | indique si le fichier est un répertoire |
| boolean isFile() | indique si l'objet représente un fichier |
| long length() | renvoie la longueur du fichier |
| String[] list() | renvoie la liste des fichiers et répertoires contenus dans le répertoire |
| boolean mkdir() | création du répertoire |
| boolean mkdirs() | création du répertoire avec création des répertoires manquants dans l'arborescence du chemin |
| boolean renameTo() | renommer le fichier |

**Exemple :**

**import** java.io.\*;

**public** **class** TestFile {

**public** **static** **void** main(String args[]){

TestFile tf = **new** TestFile(); tf.exec();

}

**void** exec(){

File f = **new** File("source.txt ");

**if** (f.exists()){

System.*out*.println("Information sur le fichier : " + f.getName());

System.*out*.println("l'objet représente un fichier ? "+f.isFile());

System.*out*.println("Nom absolu = "+f.getAbsolutePath());

System.*out*.println("le fichier peut être lu ? "+f.canRead());

System.*out*.println("Est un répertoire = "+f.isDirectory());

System.*out*.println("Longueur du fichier = "+f.length());

}

}

}

Information sur le fichier : source.txt

l'objet représente un fichier ? true

Nom absolu = G:\CEGEP LIMOILOU\workspace\420-EAB-LI\source.txt

le fichier peut être lu ? true

Est un répertoire = false

Longueur du fichier = 63