

Tutoriel n°2 – Création et utilisation d'une classe UNZIP

Introduction

Voici la suite du tutoriel n°1 qui va vous permettre de dézipper un fichier avec l'extension .zip. La démonstration qui suivra est bien entendu une méthode parmi tant d'autres pour dézipper un fichier.

Définition de la classe UNZIP

Définition des attributs de la classe:

```
private String fileZip; // Fichier sélectionné qui sera dézippé
private String outDir = "."; // Répertoire de sortie
private String currentOutDir = outDir; // Aide à la décompression des
données. Cette variable sera plus détaillée par la suite.

private FileOutputStream fos; // Utilisé pour écrire les fichiers se
situant dans le fichier zippé
private BufferedOutputStream buffOutput; // Lecture des données sur le
fichier ZIP
private ZipInputStream zin; // Initialise la lecture des données du ZIP
private ByteBuffer buffer;
private FileChannel fOut;
```

Création des constructeurs, des getters et des setters

```
// Les constructeurs
public FileUnzip(String fileZip, String outDir) throws
FileNotFoundException {
    this(new File(fileZip), outDir);
} // FileUnzip

public FileUnzip(File fileZip, String outDir) throws
FileNotFoundException {
    setOutDir(outDir);
    setFileZip(fileZip);
} // FileUnzip
```

Les constructeurs ci-dessus ont juste pour but d'initialiser le dossier de sortie et de mettre en place le fichier sélectionné pour la décompression de celui-ci.

```
public void setOutDir(String outDir) {
    this.outDir = outDir;
} // setOutDir

public String getOutDir() {
    return outDir;
} // getOutDir
```

Tutoriel n°2 – Création et utilisation d'une classe UNZIP

```

public String getFileZip() {
    return fileZip;
} // getFileZip

public void setFileZip(String fileZip) throws FileNotFoundException {
    setFileZip(new File(fileZip));
} // setFileZip

public void setFileZip(File fileZip) throws FileNotFoundException {
    File in = fileZip;
    if(!in.exists())
        throw new FileNotFoundException("Zip file must be exist");
    this.fileZip = fileZip.getPath() ;
    initInStream(in); /* Cette fonction est détaillé dans la prochaine
                       section */
} // setFileZip

```

Initialisation des flux d'entrées et de sorties

Le flux d'entrée ne représente rien d'autres que notre fichier à dézipper.

```

private void initInStream(File in) throws FileNotFoundException {
    if(null != zin)
        zin = null;
    zin = new ZipInputStream(new FileInputStream(in));
} // initInStream

```

Les flux de sorties représentent tout les fichiers, dossiers, sous-dossiers.... que contient notre fichier zip. La fonction ci-après sera donc appelée à chaque fois que l'on aura un fichier à dézipper.

```

private void initOutputStream(File in) throws FileNotFoundException
    if(null != fos)
        fos = null;
    fos = new FileOutputStream(in);
    buffOutput = new BufferedOutputStream(fos, ALLOCATE_DEFAULT);
} // initOutputStream

```

Vérification de l'existence du dossier dézippé sur l'ordinateur

Comme nous le montre l'intitulé, la fonction suivante va vérifiée si le dossier censé représenté le fichier dézippé se trouve ou non à l'adresse que nous lui avons communiquée.

```

private void checkRootDirectory() throws FileNotFoundException {
    outDir = !outDir.equals(".") ? outDir + File.separator : "";
    String dirRoot = outDir + new File(fileZip.replaceAll(FileUtil.getExtension(new
        File(fileZip)), "")).getName() + File.separator ;
    File rootFile = new File(dirRoot);
    if(!rootFile.exists())
        rootFile.mkdirs();
    outDir = dirRoot ;
}

```

Tutoriel n°2 – Création et utilisation d'une classe UNZIP

```
// checkDirectories
```

```
/*
```

Remarque: Ici nous avons la présence de la fonction **getExtension**, celle-ci faisant partie d'une classe utilitaire faisant partie d'un développement parallèle, elle sera expliquée dans un prochain article ou tutoriel. Néanmoins vous la trouverez dans le dossier des sources à télécharger

```
*/
```

Obtiens le chemin d'une entrée d'un ZIP

```
private String getDirectoryZentry (ZipEntry zE) {
    File zEfile = new File(outDir + File.separator + zE.getName());
    if (!zEfile.getParentFile().exists())
        zEfile.getParentFile().mkdirs();
    return zEfile.getAbsolutePath();
} // getDirectoryZentry
```

La fonction précédente à un double rôle, celui de créer le dossier ou fichier parent si celui-ci n'existe pas. Elle est utilisée dans le cas où nous avons des sous répertoires se situant dans le zip, car ils ont besoin d'être créés physiquement sur l'ordinateur

Les fonctions principales de la classe: « UnZipFile » et « Execute »

```
private void unZipFile (File file) throws IOException {
    int octetReaded = -1;
    initOutputStream (file);
    byte [] buff = new byte [ALLOCATE_DEFAULT];
    while (-1 != (octetReaded = zin.read (buff, 0, ALLOCATE_DEFAULT)))
        buffOutput.write (buff, 0, octetReaded);
    buffOutput.flush ();
    buffOutput.close ();
    fos.flush ();
} // unZipFile
```

Cette fonction n'est appelée dans le cas où nous avons **un fichier à dézipper**.

```
public void execute () throws IOException {
    ZipEntry zEntry = null;
    checkRootDirectory ();
    while (null != (zEntry = zin.getNextEntry ())) {
        File file = new File (getDirectoryZentry (zEntry));
        if (!zEntry.isDirectory ()) {
            if (!file.getName ().equals (TEMP_OS_WINDOWS_THUMBS) && !
                file.getName ().equals (TEMP_OS_MAC_STORE)) {
                unZipFile (file);
            }
        }
        zin.closeEntry ();
    } else
        file.mkdirs ();
}
```

Tutoriel n°2 – Création et utilisation d'une classe UNZIP

```
fos.close();  
zin.close();  
} // execute
```

Exemple d'utilisation de la classe UnZip

Voici un exemple d'utilisation de la classe unzip

```
FileUnzip fu = new FileUnzip (« C:\monfichier.zip », « C:\ »)  
fu.execute();
```

Remarque : Comment dire.... deux lignes... c'est pas la mort..