JAR (Java ARchive)

1 Distribuer une application Java

En Java une application se présente comme une pluralité de fichiers et de répertoires (autant de répertoires que de paquetages, alors que pour d'autres langages une application correspond à un fichier exécutable. Une telle organisation rend complexe et risquée la distribution des applications. C'est pourquoi les exécutables Java sont distribués sous la forme d'archives `zip` ou, plutôt, `jar`. Une archive `jar` est la même chose qu'une archive `zip`, sauf qu'une archive `jar` est censée contenir un fichier `manifest`, situé dans le chemin `META-INF/MANIFEST.MF`. Ce fichier permet de paramétrer le JAR :

– Numéro de version, éditeur, etc.

```
Manifest-Version: 1.0
Created-By: 1.6.0 (Sun Microsystems Inc.)
```

– Point d'entrée de l'exécutable (classe contenant le main)

```
Main-Class: MyPackage.MyClass
```

– Référence vers d'autres librairies et d'autres JAR

1.1 Utilisation en ligne de commande

La commande de base pour créer un fichier `Jar` est :
`jar cfv fichier.jar fichier_inclus_1 ... fichier_inlcus_n`
Où :

– l'option `c` indique qu'il faut créer une archive Jar
– l'option `f` indique que le résultat sera redirigé dans un fichier
– l'option `v` (pour verbose) fait afficher les commentaires associés à l'exécution de la commande, en particulier, les noms des éléments ajoutés au fichier d'archive au fur et à mesure qu'ils y sont ajoutés
– `fichier.jar` est le nom du fichier d'archive créé
– les `fichier_inclus_1` ... `fichier_inlcus_n` sont une suite de noms de fichiers qui seront inclus dans l'archive ; ces noms peuvent utiliser des * ; s'ils font référence à des répertoires, leur contenu sera récursivement inclus dans l'archive.

La commande génère un fichier d'archive placé dans le répertoire courant. La commande génère également un fichier `MANIFEST` pour cette archive.
Imaginons que votre projet soit organisé suivant la hiérarchie suivante, dans trois paquetages dont la racine est située en dessous du répertoire projet :

```
projet/paquet1/Util.java
              /Util.class
      /paquet2/Personne.java
              /Personne.class
      /paquet3/Principal.java
```

```
/Principal.class
/Repertoire.java
/Repertoire.class
```

On suppose ici que `Principal` est la classe principale qu'il faut appeler pour exécuter le programme (elle crée un répertoire et des personnes en utilisant des utilitaires).
Afin de créer une archive de ce projet, on peut utiliser des outils comme WinZip (sous Windows, qui crée une archive ZIP) ou comme la commande tar Unix qui permet de créer une archive compressée. Une autre manière, permettant d'être indépendant de la plateforme, consiste à utiliser la commande "jar" qui utilise Java pour créer une Java ARchive.
Voici comment s'utilise cette commande pour créer une archive :
`jar cvf projet2014.jar projet/*`
L'extraction de l'arborescence de cette archive projet2014.jar se fait par la commande :

```
jar xvf projet2014.jar
```

Tous les fichiers placés dans cette archive sont indispensables pour pouvoir compiler ou modifier l'application. En revanche, seuls les fichiers d'extension "`.class`" sont requis pour l'exécution (le bytecode des classes).

1.2 Un jar exécutable

Il est possible de créer un fichier, dit `jar exécutable` qui peut ne contenir que les fichiers de bytecode (`.class`) et qui précise quelle classe doit être appelée en premier en cas d'exécution. Avec un éditeur de textes quelconque, il suffit de composer un fichier nommé `MANIFEST.MF` comportant l'unique ligne

```
Main-Class: paquet3.Principal
```

et puis créer l'archive exécutable, depuis le répertoire projet, par :
`jar -cvfm archiveExecutable.jar MANIFEST.MF paquet*/*.class`
Vous pourrez alors vérifier qu'avec ce seul fichier `archiveExcecutable.jar`, il est possible d'exécuter l'application grâce à la commande :

```
java -jar archiveExcecutable.jar
```

1.3 Produire un JAR depuis Eclipse

a) Sélectionner un projet, puis `Files > Export`... ou clic droit sur le projet, puis sur "`Export`"

b) Choisir "`JAR file`" puis `Next`.

c) Choisir le projet, son contenu (les ressources à mettre dans le JAR), son format et le nom de l'archive, puis entrer l'adresse de destination du JAR et cliquer sur "`Next`".

d) Préciser la classe contenant le main (gestion du `manifest`) comme point d'entrée du programme.

e) Cliquer sur "`Finish`"