

## Créer un jar utilisant une librairie externe

Lorsque votre application utilise une ou plusieurs bibliothèques contenues dans un ou plusieurs fichiers jar, il est nécessaire de prendre en compte ces fichiers lors de la création du jar exécutable de votre propre application.

Deux solutions sont possibles.

**Extraction du jar.** La première solution consiste à extraire le contenu du jar de la bibliothèque, dans votre répertoire classes par exemple. Les paquetages et les fichiers .class inclus dans cette bibliothèque apparaissent alors dans la hiérarchie des classes, avec celles de votre application. Il suffit de les inclure lorsque vous créez votre propre jar.

Par exemple, supposons que vous développiez une application disposant d'un seul paquetage appli et que cette application utilise une bibliothèque bib.jar contenant deux paquetages pack1 et pack2 et que le fichier bib.jar se trouve dans votre répertoire lib. Vous devez alors opérer ainsi :

1. Depuis le répertoire classes, extrayez la bibliothèque (option x de la commande jar) :

```
.../classes> jar xvf ../lib/bib.jar
```

Les dossiers pack1 et pack2 ont été maintenant déployés dans classes avec leurs fichiers .class.

2. Créer votre jar en incluant les paquetages de la bibliothèque :

```
.../classes> jar cvfm ../monappli.jar ../monmanifest.txt appli  
pack1 pack2
```

ou avec les sources et documentations :

```
.../classes> jar cvfm ../monappli.jar ../monmanifest.txt appli  
pack1 pack2 -C .. src -C .. docs
```

**Joindre la bibliothèque et informer le manifeste** La seconde solution consiste à laisser la bibliothèque à part, mais à la placer dans un répertoire bien défini. Ensuite on modifie le manifeste du jar que l'on crée pour y ajouter l'information précisant où se trouve la bibliothèque.

Cette information s'ajoute de manière similaire à l'information Main-Class : . Il s'agit cette fois, assez naturellement, d'une information appelée Class-Path : que l'on fait suivre de la liste des répertoires contenant des classes et/ou fichiers jar avec leurs chemins d'accès séparés par des espaces. Les chemins peuvent être donnés en absolu ou relatif.

Par exemple, supposons que vous disposiez d'une application dont la classe principale est appli.MonMain et qu'elle utilise des classes contenues dans une archive bib.jar placée dans le répertoire lib, ainsi que des .class placés dans le répertoire /opt/java/biblio2/classes.

Vous devez créer le manifest suivant :

```
Main-Class: appli.MonMain  
Class-Path: lib/bib.jar /opt/java/biblio2/classes
```

et vous créez ensuite votre jar comme habituellement en utilisant ce manifeste.

Cependant, lorsque vous utiliserez votre jar (par java -jar ...), il faudra impérativement que l'archive bib.jar se trouve dans un sous-répertoire lib de votre répertoire d'exécution (dans ce cas chemin relatif) et que les autres classes de la seconde bibliothèque se trouvent dans le dossier /opt/java/biblio2/classes (dans ce cas chemin absolu).

## Rapides commentaires

- la première solution a pour grand défaut que la bibliothèque, a priori externe, a été incluse dans le projet. Si elle évolue (correction de bug, amélioration, etc.) on ne peut la changer facilement sans reconstruire le jar.

De plus, certaines bibliothèques sont volumineuses et d'usage répandu. On risque donc de se retrouver avec plusieurs applications utilisant la même bibliothèque, cette dernière étant dupliquée dans chacun des jars...

- la seconde solution a pour défaut éventuel que les emplacements des bibliothèques sont figés par rapport au chemin d'exécution du `jar`. Il est donc nécessaire de faire attention à ce point. Cependant l'installation de bibliothèques à des emplacements précis est largement utilisée lors du déploiement d'applications.

Il est donc très fortement souhaitable d'utiliser la seconde version.