“jar”
(Java Archive Tool)

Sang Shin
Michèle Garoche
www.javapassion.com
“Learn with Passion!”

Topics

- “jar” tool (from JDK)
- Packaging an application into a “jar” file
- “jar” file in the classpath
- “jarscan” utility (Free 3rd-party software)
- “jarsigner” utility (from JDK)

“jar” Tool

jar – Java Archive Tool

- The jar tool combines multiple files into a single JAR archive file.
  - The jar tool is a general-purpose archiving and compression tool, based on ZIP and the ZLIB compression format.
  - However, jar was designed mainly to facilitate the packaging of java applets or applications into a single archive.

Why jar?

- When the components of an applet or application (.class files, images and sounds) are combined into a single archive, they may be downloaded by a java agent (like a browser) in a single HTTP transaction, rather than requiring a new connection for each piece.
  - This dramatically improves download times.
- jar also compresses files and so further improves download time
- A jar archive can be use as a class path entry, whether it is compressed or not

How to use jar command?

- Create myFile.jar from all classes in the current directory
  - C:\Java> jar cf myFile.jar *.class
- A manifest file entry named META-INF/MANIFEST.MF is automatically generated by the jar tool and is always the first entry in the jar file
  - The manifest file is the place where any meta-information about the archive is stored as name: value pairs.

Common JAR file operations

- To create a JAR file
  - *jar cf jar-file input-file(s)*
- To view the contents of a JAR file
  - *jar tf jar-file*
- To extract the contents of a JAR file
  - *jar xf jar-file*
- To extract specific files from a JAR file
  - *jar xf jar-file archived-file(s)*

Packaging an Application into a “jar” file

JAR Files as Applications

- You can run JAR-packaged applications with the Java interpreter
  - *java -jar jar-file*
- The -jar flag tells the interpreter that the application is packaged in the JAR file format
- You can only specify one JAR file, which must contain all the application-specific code
- To indicate which class is the application's entry point, you must add a Main-Class header to the JAR file's manifest. The header takes the form:
  - *Main-Class: main_classname*

Example: JAR Files as Applications

- We want to execute the main method in the class MyClass in the package MyPackage when we run the JAR file.
- We first create a text file named Manifest.txt with the following contents:
  - Main-Class: MyPackage.MyClass
- We then create a JAR file named MyJar.jar by entering the following command:
  - jar cfm MyJar.jar Manifest.txt MyPackage/*.class

Adding Classes to the JAR File's Classpath

jar in the classpath

- The jar files can be specified in the classpath
  - java -classpath mystudentpackage.jar;myanotherpackage.jar;/somedirectory studentpackage.StudentRecordExample

“jarscan” utility

“jarscan” utility

- Open source tool for finding out which jar file contains which class
- java -jar <jarscan-directory>\jarscan.jar -dir . -class MyClass

jarsigner

jarsigner

- The jarsigner tool is used for two purposes:
  - to sign Java ARchive (JAR) file
  - to verify the signatures and integrity of signed JAR files.

Thank you!

Sang Shin
Michèle Garoche
http://www.javapassion.com
“Learn with Passion!”