ECM1410
Summative Coursework

Jar file creation example walkthrough and checking

This document details a short walkthrough outlining the generation of a jar file from the ECM1410 coursework beanbags package (as initially released) and uses a simple Java application available on ELE to check the jar file has been set up correctly.

Lines in *italics* are comments to explain what is going on, and any expectations on directory structure at the start. '>>' is the terminal prompt in Linux, and lines starting with this detail commands to enter. Lines in **bold** detail the printout expected in the terminal window in response to the command (note in some cases there will be no printout expected after a command, so the subsequent line is empty).

In the example, the current directory has two folders, src and bin, which you need to have created already, and the file ProofOfLifeTest.java from ELE, you can check this with the 'ls' command

>> ls
ProofOfLifeTest.java bin src

In the src directory you need to have put the beanbags directory and .java files download from ELE. The bin directory should be empty.

>> ls src
beanbags
>> ls bin

We now compile the package and put the resultant .class files in the bin directory. As such, after this operation, src still contains the source files of the package, and bin now contains the compiled bytecode files

>> javac -d bin/ src/beanbags/*.java

We can check that the bytecode has been created and the package directory created by listing the bin contents.

>> ls bin
beanbags

Now we put all the compiled .class files into a newly created jar file, called beanbags.jar, in the current directory. In terms of the arguments, 'cvf' - create, verbose, file. '-C bin .' from bin, get all the files

>>jar cvf beanbags.jar -C bin .
added manifest
adding: beanbags/(in = 0) (out= 0)(stored 0%)
adding: beanbags/BeanBagStore.class(in = 1804) (out= 672)(deflated 62%)
adding: beanbags/IllegalNumberOfBeanBagsSoldException.class(in = 340) (out= 233)(deflated 31%)
adding: beanbags/ReservationNumberNotRecognisedException.class(in = 346) (out= 234)(deflated 32%)
adding: beanbags/IllegalIDException.class(in = 304) (out= 220)(deflated 27%)
adding: beanbags/InvalidMonthException.class(in = 310) (out= 223)(deflated 28%)
adding: beanbags/IllegalNumberOfBeanBagsAddedException.class(in = 342) (out= 234)(deflated 31%)
adding: beanbags/BeanBagMismatchException.class(in = 316) (out= 225)(deflated 28%)
adding: beanbags/BadStore.class(in = 2612) (out= 855)(deflated 67%)
adding: beanbags/PriceNotSetException.class(in = 308) (out= 222)(deflated 27%)
adding: beanbags/BeanBagIDNotRecognisedException.class(in = 330) (out= 233)(deflated 29%)
adding: beanbags/BeanBagNotInStockException.class(in = 320) (out= 227)(deflated 29%)
adding: beanbags/IllegalNumberOfBeanBagsReservedException.class(in = 348) (out= 237)(deflated 31%)
adding: beanbags/ObjectArrayList.class(in = 1429) (out= 810)(deflated 43%)
adding: beanbags/InsufficientStockException.class(in = 320) (out= 228)(deflated 28%)
adding: beanbags/InvalidPriceException.class(in = 310) (out= 223)(deflated 28%)

For the coursework submission, I want all the source files in the jar file too, so we now add these by updating the jar file. In terms of the arguments: 'uvf' - update, verbose, file. '-C bin .' from src, get all the files.

>> jar uvf beanbags.jar -C src .
adding: beanbags/(in = 0) (out= 0)(stored 0%)
adding: beanbags/InvalidPriceException.java(in = 545) (out= 245)(deflated 55%)
adding: beanbags/ObjectArrayList.java(in = 4445) (out= 1289)(deflated 71%)
adding: beanbags/IllegalNumberOfBeanBagsAddedException.java(in = 613) (out= 260)(deflated 57%)
adding: beanbags/IllegalIDException.java(in = 534) (out= 243)(deflated 54%)
adding: beanbags/InvalidMonthException.java(in = 545) (out= 245)(deflated 55%)
adding: beanbags/BeanBagIDNotRecognisedException.java(in = 585) (out= 253)(deflated 56%)
adding: beanbags/BadStore.java(in = 2874) (out= 656)(deflated 77%)
adding: beanbags/IllegalNumberOfBeanBagsSoldException.java(in = 605) (out= 257)(deflated 57%)
adding: beanbags/InsufficientStockException.java(in = 565) (out= 248)(deflated 56%)
adding: beanbags/BeanBagMismatchException.java(in = 560) (out= 248)(deflated 55%)

adding: beanbags/IllegalNumberOfBeanBagsReservedException.java(in = 622) (out= 266)(deflated 57%)
adding: beanbags/BeanBagNotInStockException.java(in = 565) (out= 250)(deflated 55%)
adding: beanbags/BeanBagStore.java(in = 16418) (out= 2730)(deflated 83%)
adding: beanbags/PriceNotSetException.java(in = 541) (out= 245)(deflated 54%)
adding: beanbags/ReservationNumberNotRecognisedException.java(in = 617) (out= 257)(deflated 58%)

We can check the jar file contains what we want using

>> jar -tf beanbags.jar
META-INF/
META-INF/MANIFEST.MF
beanbags/
beanbags/BeanBagStore.class
beanbags/IllegalNumberOfBeanBagsSoldException.class
beanbags/ReservationNumberNotRecognisedException.class
beanbags/IllegalIDException.class
beanbags/InvalidMonthException.class
beanbags/IllegalNumberOfBeanBagsAddedException.class
beanbags/BeanBagMismatchException.class
beanbags/BadStore.class
beanbags/PriceNotSetException.class
beanbags/BeanBagIDNotRecognisedException.class
beanbags/BeanBagNotInStockException.class
beanbags/IllegalNumberOfBeanBagsReservedException.class
beanbags/ObjectArrayList.class
beanbags/InsufficientStockException.class
beanbags/InvalidPriceException.class
beanbags/InvalidPriceException.java
beanbags/ObjectArrayList.java
beanbags/IllegalNumberOfBeanBagsAddedException.java
beanbags/IllegalIDException.java
beanbags/InvalidMonthException.java
beanbags/BeanBagIDNotRecognisedException.java
beanbags/BadStore.java
beanbags/IllegalNumberOfBeanBagsSoldException.java
beanbags/InsufficientStockException.java
beanbags/BeanBagMismatchException.java
beanbags/IllegalNumberOfBeanBagsReservedException.java
beanbags/BeanBagNotInStockException.java
beanbags/BeanBagStore.java
beanbags/PriceNotSetException.java
beanbags/ReservationNumberNotRecognisedException.java

Success! The jar file contains the beanbags directory, which holds the .class and the .java files needed. Note: the final version of the package you submit will need to also contain Store and all the other classes you have written (in both .class and .java).

Using the jar file we can compile the simple application which makes an instance of BadStore

>> javac -cp .:beanbags.jar JarProcessTestApp.java
>> java -cp .:beanbags.jar JarProcessTestApp
BadStore instance successfully made, with 0 beanbags in stock.

Again, success! Note, as the beanbag directories are not in the current path (i.e. '.') – only src and bin are – we can assure ourselves that the package is only being accessed through the beanbags.jar file, so it must have been set up correctly for JarProcessTestApp *to compile and run.*