Last Updated: January 28, 2019 · paolodt

How to make a JAR file Linux executable

JAVA

Every Java programmer knows - or should known - that it is possible to create a runnable Java package archive (JAR), so that in order to launch an application it is enough to specify the jar file name on the Java interpreter command line along with the `-jar` parameter. For example:

```
$ java -jar helloworld.jar
```

There are plenty of tutorials showing how to implement this feature using Ant, Maven, Eclipse, Netbens, etc.

Anyway in its basic form, it just requires to add a `MANIFEST.MF` file to the jar package. The manifest must contain an entry `Main-Class` that specifies which is the class defining the main method for your application. For example:

```
$ javac HelloWorld.java
$ echo Main-Class: HelloWorld > MANIFEST.MF
$ jar -cvmf MANIFEST.MF helloworld.jar HelloWorld.class
```

But this still requires your users to invoke the Java interpreter with the `-jar` option. There are many reasons why it would be preferable to have your app runnable by simply invoking it on the terminal shell like any other command.

Here comes the protip!

This technique it is based on the ability to append a generic binary payload to a Linux shell script. Read more about this here: http://www.linuxjournal.com/content/add-binary-payload-your-shell-scripts

Taking advantage of this possibility the trick is just to embed a runnable jar file into a Bash script file. The script when executed will launch the Java interpreter specifying itself as the jar to run. Too complex? Much more easier to do in practice, than to explain!

Let's say that you have a runnable jar named `helloworld.jar`

Copy the Bash script below to a file named `stub.sh`

```
#!/bin/sh
MYSELF=`which "$0" 2>/dev/null`
[ $? -gt 0 -a -f "$0" ] && MYSELF="./$0"
java=java
if test -n "$JAVA_HOME"; then
    java="$JAVA_HOME/bin/java"
fi
exec "$java" $java_args -jar $MYSELF "$@"
exit 1
```

Than append the jar file to the saved script and grant the execute permission to the file resulting with the following command:

```
cat stub.sh helloworld.jar > hello.run && chmod +x helloworld.run
```

That's all!

Now you can execute the app just typing `helloworld.run` on your shell terminal.

The script is smart enough to pass any command line parameters to the Java application transparently. Cool! Isn't it ?!

In the case your are a Windows guy, obviously this will not work (except you will run a Linux compatibility layer like Cygwin).

Anyway exist tools that are able to wrap a Java application into a native Windows `.exe` binary file, producing a result similar to the one explained in this tutorial. See for example http://launch4j.sourceforge.net/

Written by Paolo Di Tommaso

♡ **Recommend**

Say Thanks

Update Notifications Off

Respond

10 Responses

athieriot
That's amazing!

over 1 year ago · ♡

md2perpe
During reading this article I got a feeling - rather than a clear memory - that I sometime, more than ten years ago, had seen a way to make Linux itself recognize files as executable.

Here you can see how to do:
http://www.kernel.org/doc/Documentation/binfmt_misc.txt
Specific for Java:
http://www.kernel.org/doc/Documentation/java.txt

over 1 year ago · ♡

paolodt
@md2perpe I known about Linux support for custom binary format, but for many users it cannot be an option because they may not have root permissions. So the above script is the best solution, if you need a transparent runnable Java app!

over 1 year ago · ♡

dongli
This is just what I need! Thank you!

over 1 year ago · ♡

dongli

@paolodt I encounter a problem when I use dynamic compilation in my JAR. The JAR will read in external Java sources that implement an interface in the JAR. When in Eclipse IDE, things work fine, but after creating the final executable file, it will fail with error "cannot read zip file entry". So there is any solution for this problem?

over 1 year ago ·

paolodt

Hi, but what if you try to launch it using "java -jar <your executable jar>"? Does it work or it fails

over 1 year ago ·

hduchesn

I have an issue with your script and my jar :).
my java apps is a GUI launcher in charge to start another program. When I launch the .run from terminal it works fine (I mean, I keep my env). But when I double click the .run, the GUI launcher appears but I lost my env and the start of other program failed because JAVA_HOME is not set...
How to keep my env when I double click the .run?

over 1 year ago ·

zcoklin

All works perfect except the last line. It should go like this:

```
cat stub.sh helloworld.jar > hello.run && chmod +x hello.run
```

over 1 year ago ·

fatso83

While I have tried this technique many years ago with good results, I just saw a weird side effect I cannot explain. There seems to be a difference when executing the jar file directly and executing the embedded jar in the script. I have problems finding JSTL classes in the embedded version. Weird as f*ck.

over 1 year ago ·

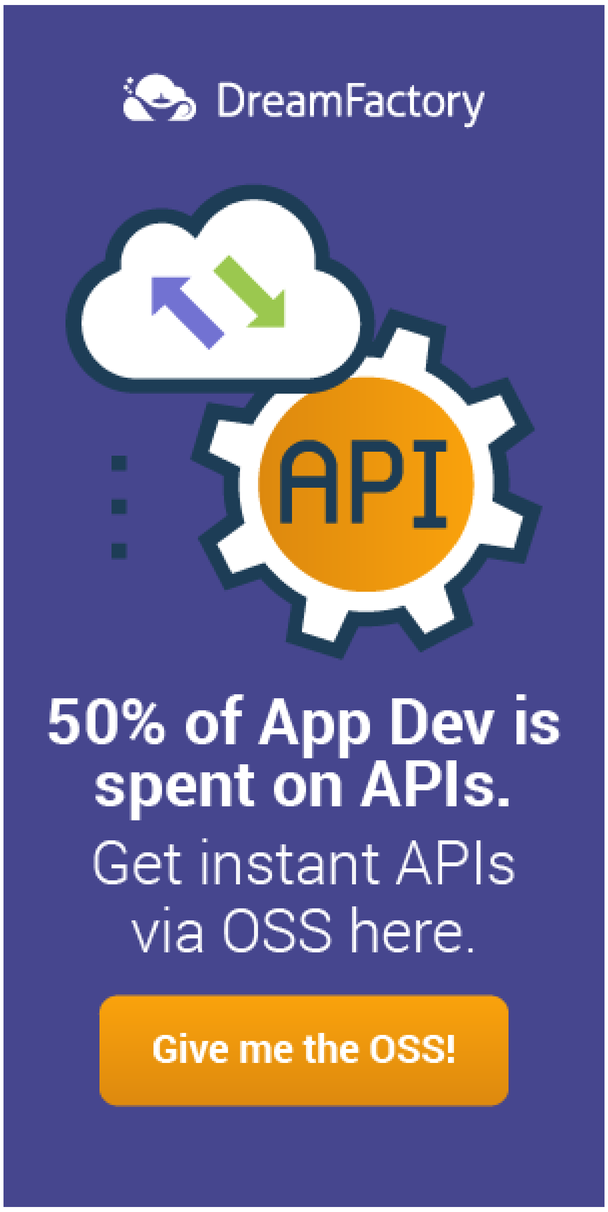
antava07

Could you please explain what each line of the above Bash script is actually doing? Would be really helpful :)

3 months ago ·

Awesome Job See All Jobs

Post a job for only $299


DreamFactory
API
50% of App Dev is spent on APIs.
Get instant APIs via OSS here.
Give me the OSS!