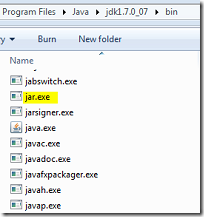
**Java Jar**

**Executable Java Jar File and Windows Exe File Creation**

**How to Create an Executable JAR File?**

JDK gives us a tool to create *jar* files.



Before creating a java jar file, we need to create a java **manifest** file.

Command to create an executable java jar file:

*jar -cvfm BundleExample.jar manifest.mf com /corejava/\*.class*

**How to create a java manifest file?**

Generally we will have the [manifest file](http://docs.oracle.com/javase/tutorial/deployment/jar/manifestindex.html) located in,

* META-INF/manifest.mf

This manifest file contains information about the jar file. Information like which is the main java class to be executed, who created the application, version of the application, etc.

It is a regular text file. Remember to press enter at the end of each line and importantly in the last line.

So with respect to creating an executable java jar file, we need to add the main class of the application using which it can be launched. Our example manifest file is like below,

*Main-Class: com.corejava.JavaBundleExample*

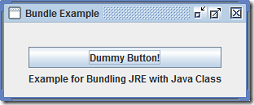
This is the only line I have and that is sufficient for our purpose.

**Sample Application**

We need a sample java application to enjoy this. Lets create a simple hello world type swing application.

|  |
| --- |
| package com.corejava;    import java.awt.GridLayout;    import javax.swing.BorderFactory;  import javax.swing.JButton;  import javax.swing.JFrame;  import javax.swing.JLabel;  import javax.swing.JPanel;    public class JavaBundleExample {      private static void createAndShowGUI() {        JFrame.setDefaultLookAndFeelDecorated(true);      JFrame frame = new JFrame("Bundle Example");      frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);      JPanel pane = new JPanel(new GridLayout(0, 1));      JButton button = new JButton("Dummy Button!");      pane.add(button);      JLabel label = new JLabel("Example for Bundling JRE with Java Class");      pane.add(label);      pane.setBorder(BorderFactory.createEmptyBorder(30, 30, 10, 30));      frame.getContentPane().add(pane);      frame.pack();      frame.setVisible(true);    }      public static void main(String[] args) {      javax.swing.SwingUtilities.invokeLater(new Runnable() {        public void run() {          createAndShowGUI();        }      });    }  } |

Output:



**To run the file type**

Java –jar BundleExample.jar

Just execute the following line (command) to create the jar file, you should have added JDK to PATH.

*jar -cvfm BundleExample.jar manifes   
t.txt com/javapapers/corejava/\*.class*

Now we can double click the jar file and the application will launch and run.

**How to Create a Windows Exe file?**

If we create a windows native exe file, then we loose the platform independent-ness.

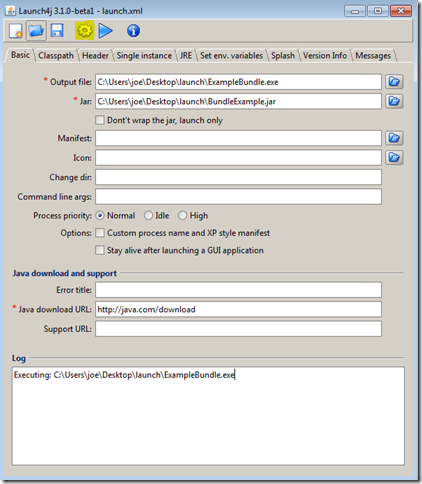


Let us use a tool called [Launch4J](http://sourceforge.net/projects/launch4j/" \t "_blank). It’s a nice tool, works good and easy to use.

Its core features are,

* creating lightweight Windows native EXE
* JRE version control
* app icon
* splash screen
* process name
* Java download page

Download Launch4J and use the GUI to create the windows/Java exe file. Launc4J creates an XML file to save the configuration.

[](http://javapapers.com/wp-content/uploads/2013/04/launch4jconfig.png)

* I have specified the output file, this is the name of the exe file to be created.
* Jar is the input file which we want to bundle as an exe file. Lets use the Java JAR which we created in the previous step.
* In JRE tab, we can specify the Min JRE version required. So this will detect the JRE available in the system and verify if the version is sufficient.
* Click the cog wheel icon from the menu to generate the exe file.