How to create a standalone program

Introduction

Eclipse can produce standalone Java programs called JAR files (for Java ARchive). As long as your computer has a Java Runtine Environment (JRE) installed, you should be able to run a .jar as a program by double-clicking it. Normally, this is very easy to do with a few clicks, but because we're using the non-standard ACM libraries and the Stanford plugin in the ConsolePrograms and GraphicsPrograms we have been writing throughout the course, we have to go through a fairly long process instead. Here are the steps required to produce a standalone program.

1 Adding a `main()` method

You may have wondered why we needed a `main()` method in the LeJOS programs but not in the ConsolePrograms and GraphicsPrograms. This is because the Stanford Eclipse plugin is automatically adding a `main()` method. Before we can export to a JAR, we need to add this method back in. That is, inside your main class (the one that extends ConsoleProgram or GraphicsProgram), you shoudld add the code:

```
public static void main(String[] args) {
   new MyClass().start(args);
}
```

In this code snippet, you should replace "MyClass" with whatever the name of your class is. So for instance if at the top of your file you have the line

```
public class RandomCircles extends GraphicsProgram {
```

then you should write RandomCircles in place of MyClass in the above code.

2 Using Eclipse's Export Wizard

The next series of steps will be to use Eclipse's Export Wizard feature to produce a JAR and a manifest file. We will later need to modify the manifest file to specify that the ACM libraries should be included, but first perform the following steps:

1. Select the project you want to export in the left-hand pane (the "Package Explorer"). At the top of the screen, go to the **File** menu and select **Export**. This should make an export window appear in the middle of the screen.

2. On the "choose export destination" screen, select the **Java** folder, and then select the option **JAR file**. Click the **Next** button.

3. On the "JAR File Specification" screen, check the box next to the name of your project in the left-hand box, and then select the location you want the JAR file saved to in the box labeled "Select the export destination." Then click **Next**.

4. The next screen is "JAR Packaging Options." This screen is not important for us, and you can simply click **Next** without changing anything.

5. The next screen is "JAR Manifest Specification," and requires several steps.

- First, near the bottom of the window, select the main class using the **Browse** button. The main class is the one to which we earlier added the `main()` method in the first step above.
- Then click the radio button labelled "Generate the manifest file."
- Then check the box for "Save the manifest in the workspace."
- Then click **Browse** next to the box for "Manifest file," navigate to the folder where your project exists, and enter "manifest" as the file name. If you do this correctly, the box will likely be filled with something like "/ProjectName/manifest".
- Finally, click **Next**.

After all of these steps, you should see the manifest file show up in the package explorer under the current project.

3 Editing the manifest file

We need to edit the manifest file that was just created to tell it to use the ACM libraries. Open up the manifest file which should have just appeared in your project folder, and add the line

Class-Path: acm.jar

After doing so, the manifest file should appear something like

Manifest-Version: 1.0
Main-Class: MyClass
Class-Path: acm.jar

Make sure you save the manifest file after changing it.

4 Export Wizard again

Now repeat the process described in the secion titled "Using Eclipse's Export Wizard" up until just before step 5. This time, on the screen labelled "JAR Manifest Specification," you should check the radio button for "Use existing manifest from workspace," and use the **Browse** button to select the manifest file you just created.

Click **Finish**. If Eclipse asks you if you want to overwrite the old .jar file, saw yes. We're almost there!

5 Distributing the .jar file

Now you have a .jar file containing your code, but you can't just send it to your friends because they won't have the ACM library. To run your program, you will need to have your .jar plus `acm.jar` in the same directory, and then you can just double-click on your .jar to start your application.

Note that if your application requires any auxiliary files, such as images, you will also need to put them in the same directory with your .jar and `acm.jar`.

If you want to send your application to friends, the easiest way is probably to put all the required files together into a folder, and then compress that folder into a .zip and send it.