

CHAPTER

8

Packaging and Deploying Java Projects in Forte

This chapter introduces to use Forte's Archive wizard to package project files for deployment. You will also learn how to create shortcut for applications on Windows.

8.1 Java Archiving

Your project may consist of many classes and supporting files, such as image files and audio files. All of these files have to be provided to the end-users if your programs are to run on their side. For convenience, Java supports an archive file that can group all the project files in a compressed file.

The Java archive file format (JAR) is based on the popular ZIP file format. JAR can be used as a general archiving tool, but transporting Java applications, applets, and their requisite components (.class files, images, and sounds) in a single file was the primary motivation for its development.

This single file can be deployed on an end-user's machine as an application. It also can be downloaded to a browser in a single HTTP transaction, rather than opening a new connection for each piece. This greatly simplifies application deployment and improves the speed with which an applet can be loaded onto a Web page and begin functioning. The JAR format also supports compression, which reduces the size of the file and improves download time still further. Additionally, individual entries in a JAR file can be digitally signed by the applet author to authenticate their origin.

You can create an archive file using the JDK **jar** command or using the Archive wizard. The following command creates an archive file named WelcomeApplet.jar for classes WelcomeApplet.class:

```
jar -cf WelcomeApplet.jar WelcomeApplet.class
```

The **-c** option is for creating a new archive file, and the **-f** option specifies the archive file's name.

8.2 Using the Archive Wizard to Package Projects

With the **jar** command, you have to manually identify the dependent files. Forte provides the Archive wizard that

gathers all the classes on which your program depends into one JAR archive that includes image and audio files.

Let us use the `WelcomeApplet` class in Chapter 2, "Creating and Testing Applets," to demonstrate packaging projects. To create a JAR file in Forte, you must first create a JAR contents file. The JAR contents file stores information about your JAR file, which you can later modify and use to create a new JAR file.

8.2.1 Creating JAR Content Files

Here are the steps to create a JAR contents file for the `WelcomeApplet` class.

1. In the context menu of the `c:\tutorial` node in the Explorer, choose *New, Java Packager, Jar Recipe* to display the Archive wizard, as shown in Figure 8.1.

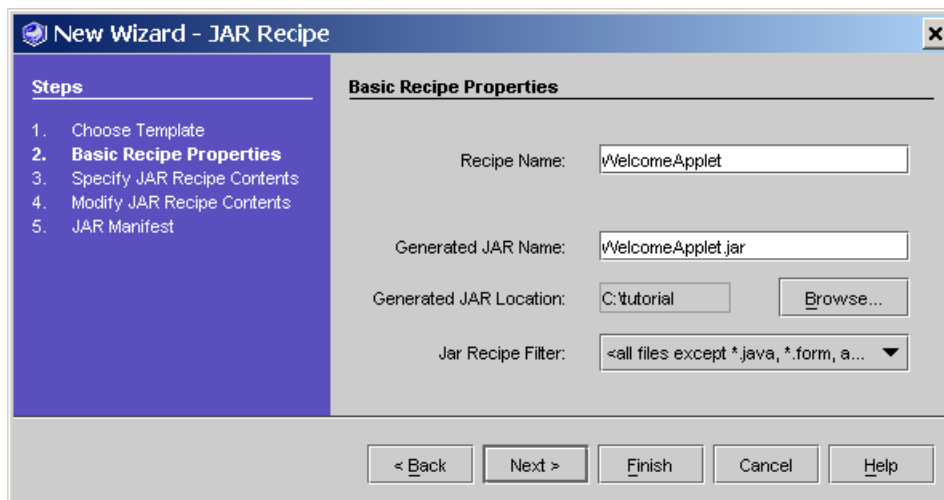


Figure 8.1

The Archive wizard enables to specify recipe for the archive.

To create a JAR file in Forte, you must first create a JAR contents file. The JAR contents file stores information about your JAR file, which you can later modify and use to create a new JAR file.

4.2.1 Creating JAR Content Files

Here are the steps in creating a JAR contents file for the `WelcomeApplet` class.

1. In the context menu of the `c:\tutorial` node in the Explorer, choose *New, Java Packager, Jar Recipe* to display the JAR Recipe wizard, as shown in Figure 8.2.

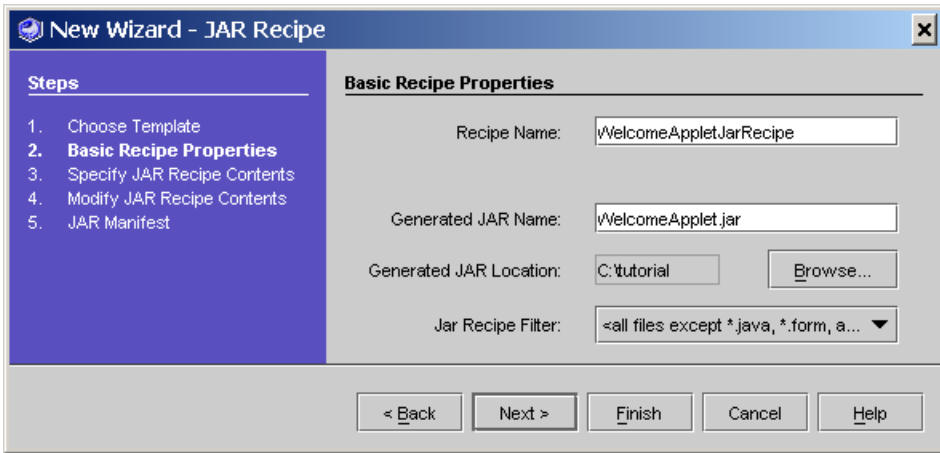


Figure 8.2

You can create a JAR recipe using the Jar Recipe wizard.

2. Type `WelcomeAppletJarRecipe` in the Recipe Name field, `WelcomeApplet.jar` in the Generated JAR Name field, and click `Next` to display the Specify Recipe Contents panel, as shown in Figure 8.3.

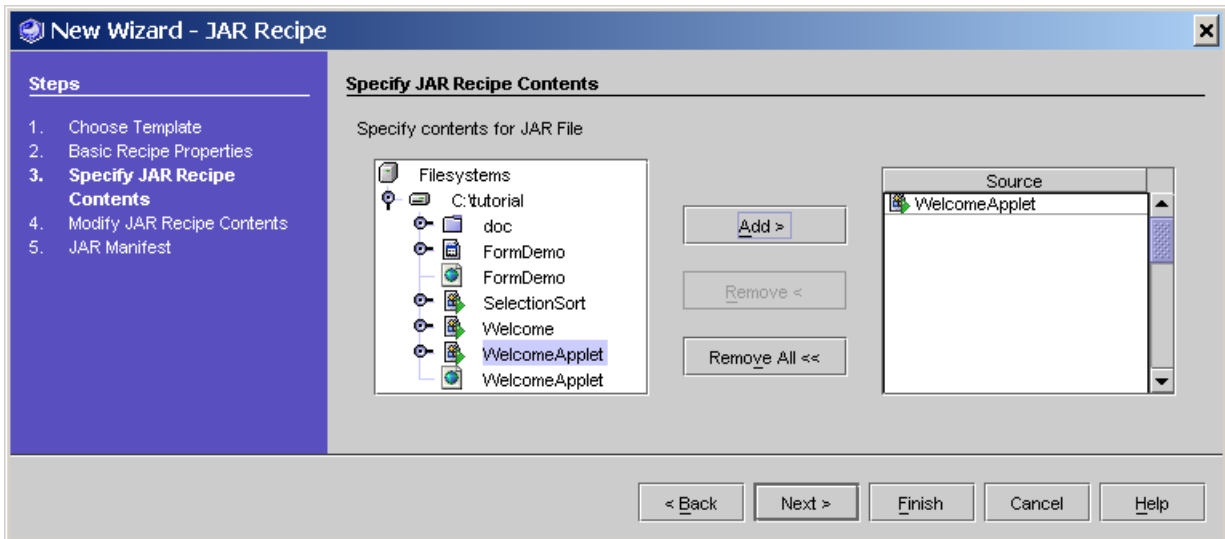


Figure 8.3

You can add class files and other resource files to the JAR content.

3. Choose `WelcomeApplet` and add it to the Source pane. Click `Next` to display the Modify JAR Recipe Contents panel, as shown in Figure 8.4. Click `Next` to display the JAR Manifest panel, as shown in Figure 8.5.

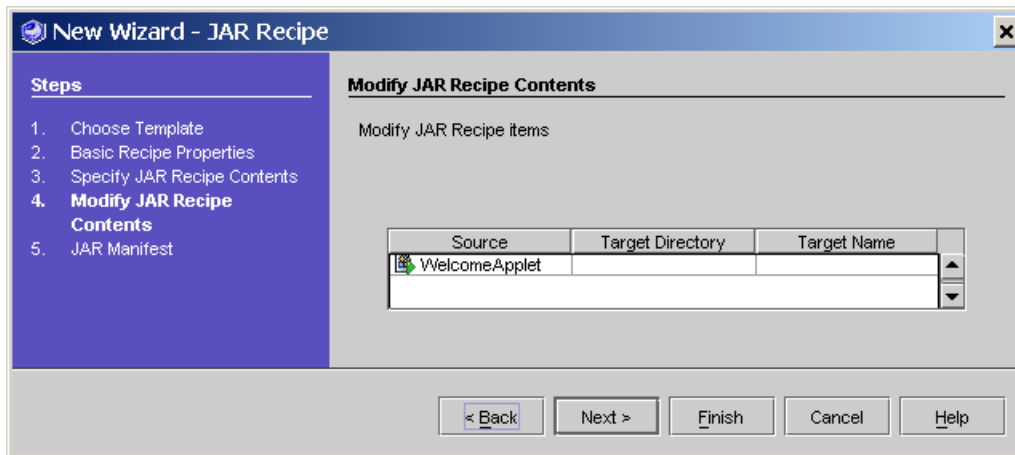


Figure 8.4

You can modify JAR Recipe Contents in the wizard.

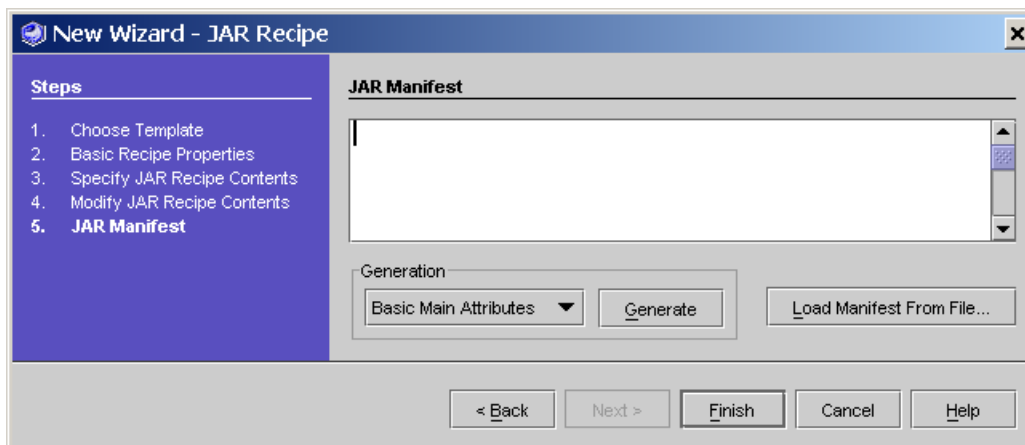


Figure 8.5

You can generate and modify a manifest file in the JAR Manifest panel.

4. In the Generation section of the JAR Manifest panel, choose *Basic Main Attributes* and click the *Generate* button to generate the manifest information. Add the following line to specify a main class so you can run the class from the JAR standalone, as shown in Figure 8.6.

Main-Class: mortgagedemo.ComputeMortgage

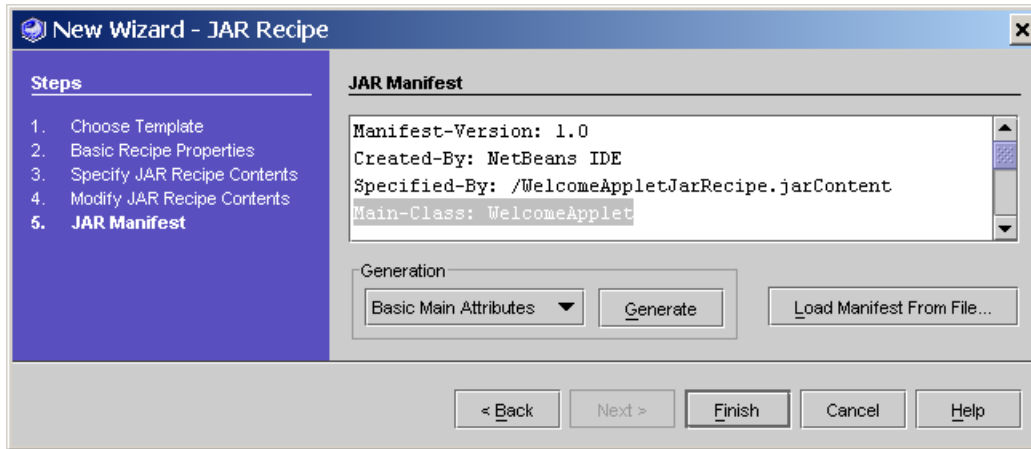



Figure 8.6

The manifest file is generated.

5. Click *Finish* to generate the JAR content file. A filesystem node for the JAR content file with the icon () appears in the Explorer, as shown in Figure 8.7. Since you specified mortgage as the JAR file in Figure 8.2, the label for this node is mortgage.

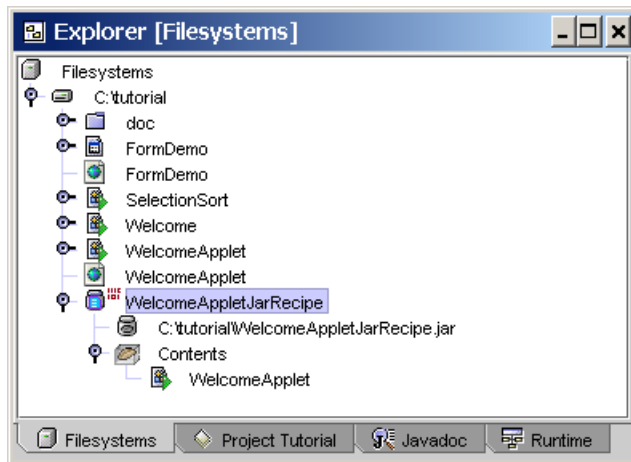
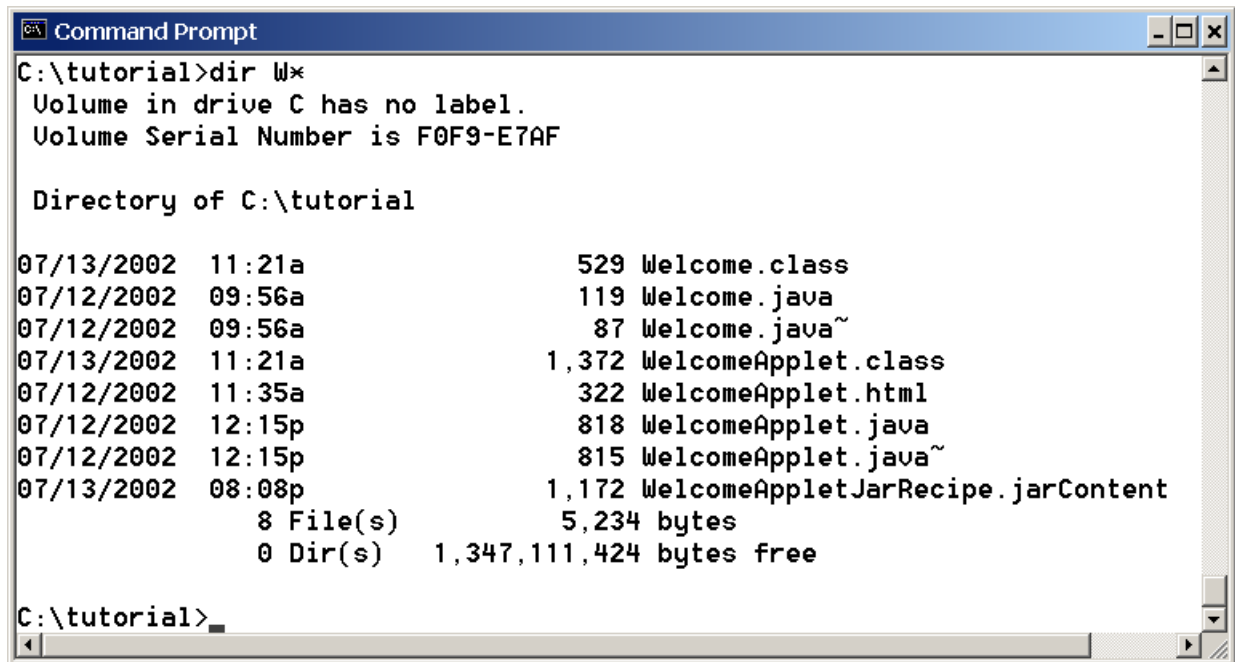


Figure 8.7

The WelcomeAppletJarRecipe node appears in the Explorer.

NOTE: If you show the directory in **c:\tutorial**, you will see the file `WelcomeApplet.jarContent`, as shown in Figure 8.8. This file describes the contents of the JAR file to be created.



```
Command Prompt
C:\tutorial>dir W*
Volume in drive C has no label.
Volume Serial Number is F0F9-E7AF

Directory of C:\tutorial

07/13/2002  11:21a                529 Welcome.class
07/12/2002  09:56a                119 Welcome.java
07/12/2002  09:56a                 87 Welcome.java~
07/13/2002  11:21a            1,372 WelcomeApplet.class
07/12/2002  11:35a                322 WelcomeApplet.html
07/12/2002  12:15p                818 WelcomeApplet.java
07/12/2002  12:15p                815 WelcomeApplet.java~
07/13/2002  08:08p            1,172 WelcomeAppletJarRecipe.jarContent
                8 File(s)                5,234 bytes
                0 Dir(s)      1,347,111,424 bytes free

C:\tutorial>
```

Figure 8.8

JAR content file describes the contents in the archive.

8.2.2 Generating JAR Files

To generate the actual JAR specified by the content file, choose *Compile* from the context menu of the JAR Recipe file node (WelcomeAppletJarRecipe) in the Explorer. A JAR file named WelcomeApplet.jar is now created. To modify the JAR file, you have to modify the JAR content file and recompile the JAR content file. To modify the JAR content file, choose the *JAR content* file node in the Explorer, and choose *Properties* to display the Properties window, as shown in Figure 8.9. Click the ellipsis button in the Manifest field to display the property editor for the manifest file, as shown in Figure 8.10.



Figure 8.9

You can modify the JAR recipe file in the Properties window for the recipe file.

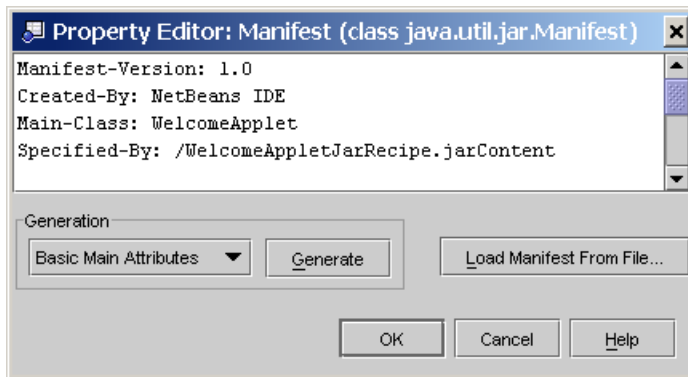


Figure 8.10

You can modify the manifest file in the Manifest property editor.

NOTE: If your Java source code is changed, you need to update the .jar file. You can do this by choosing *Compile* in the context menu of the JAR recipe file node in the Explorer.

8.3 Running Archived Projects

The JAR file packages all the class files and dependent resource files into one compressed file that can be distributed to the end-user. If the project is a Java application, the user should have a Java Runtime Environment already installed. If it is not installed, the user can download Java Runtime Environment (JRE) from JavaSoft at www.javasoft.com/ and install it.

NOTE: The Java Runtime Environment is the minimum standard Java platform for running Java

programs. It contains the Java interpreter, Java core classes, and supporting files. The JRE does not contain any of the development tools (such as Applet Viewer or javac) or classes that pertain only to a development environment. The JRE is a subset of JDK.

8.3.1 Running Archived Files from Java Applications

You can run a JAR file if a main class is specified in the manifest. To run it from Forte, simply choose *Execute* from the context menu of the JAR content node in the Explorer. You can also run it from the DOS prompt using the `java` command with the `-jar` option, as follows:

```
java -jar WelcomeApplet.jar
```

NOTE: You can create an icon on the Windows Desktop to enable the end-user to run the program by double-clicking on it. To learn how to do this, see Section 8.4, "Creating Shortcut for Java Applications on Windows."

8.3.2 Running Archived Files from Java Applets

To run `WelcomeApplet` as an applet from the JAR file, you need to include an `ARCHIVE` attribute in the HTML file. The `ARCHIVE` attribute specifies the archive file that contains the applet. For example, shown below is how you would modify the HTML file for running `WelcomeApplet`:

```
<applet  
  codebase=..  
  code="WelcomeApplet.class"  
  archive = "WelcomeApplet.jar"  
  width   = 400  
  height  = 300  
  >  
</applet>
```

8.4 Create Shortcut for Java Applications on Windows

You can create an icon on Windows Desktop, so that the end-user can use it to start the program by double-clicking on it. Here are the steps to make a shortcut to run a Java program:

1. Right click on the desktop and choose *New, Shortcut* to display the Create Shortcut wizard, as shown in Figure 8.11.
2. Type `java -jar WelcomeApplet.jar` in the text box as shown in Figure 8.11. Click *Next* to display the dialog box as shown in Figure 8.12.

3. Type **WelcomeApplet** as the name for the shortcut in Figure 8.12 and click *Finish* to create the shortcut, as shown in Figure 8.13.
4. Right click the WelcomeApplet shortcut you just created and select Properties to display the WelcomeApplet Properties dialog box as shown in Figure 8.14.
5. Type **c:\liang** in the Start in field. This folder contains WelcomeApplet.jar. Click OK to close the WelcomeApplet Properties dialog box. You can now double click WelcomeApplet on the desktop to run the WelcomeApplet application.
6. (Optional) You can set a custom icon for the application by clicking the Change Icon button in the WelcomeApplet Properties dialog box.

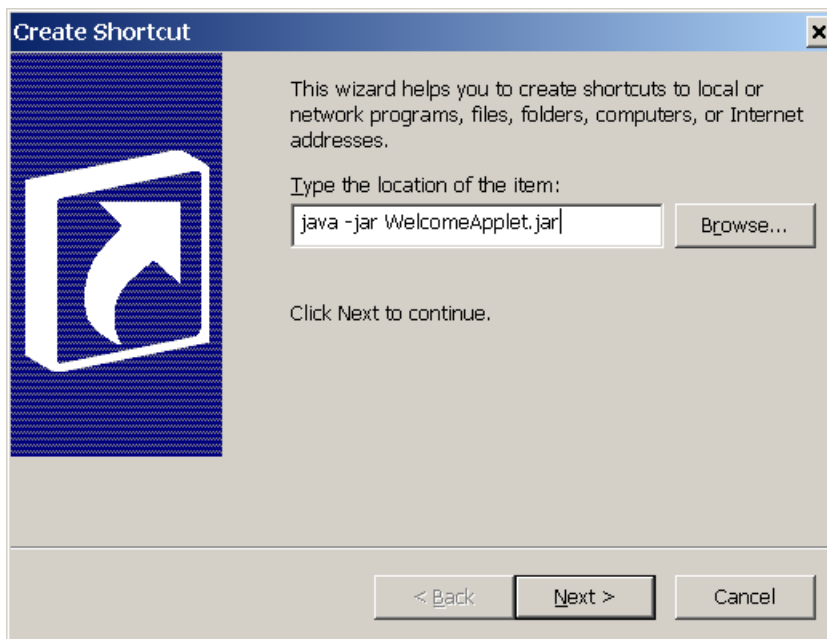


Figure 8.11

The Create Shortcut wizard helps to create a windows shortcut.

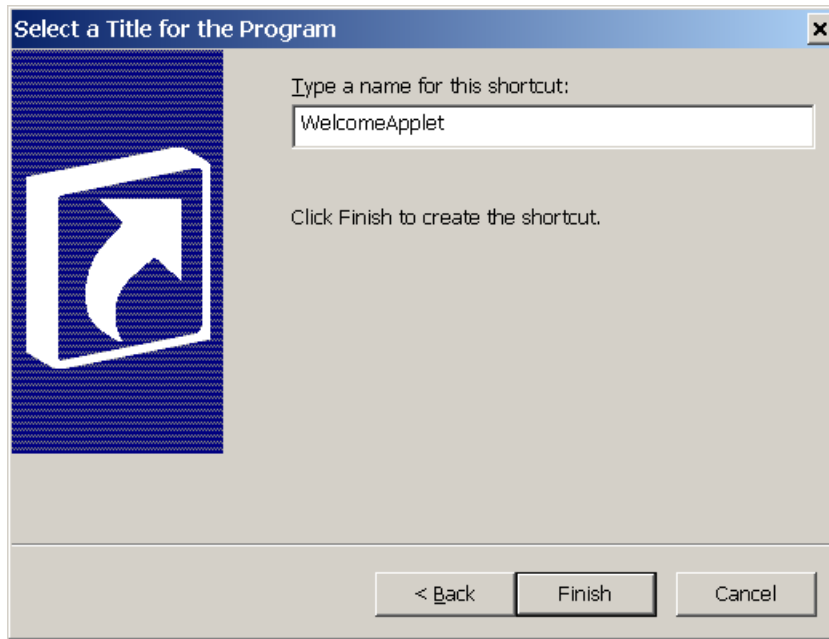


Figure 8.12

Select an appropriate name for the shortcut.



Figure 8.13

The WelcomeApplet shortcut is created on the Windows desktop.

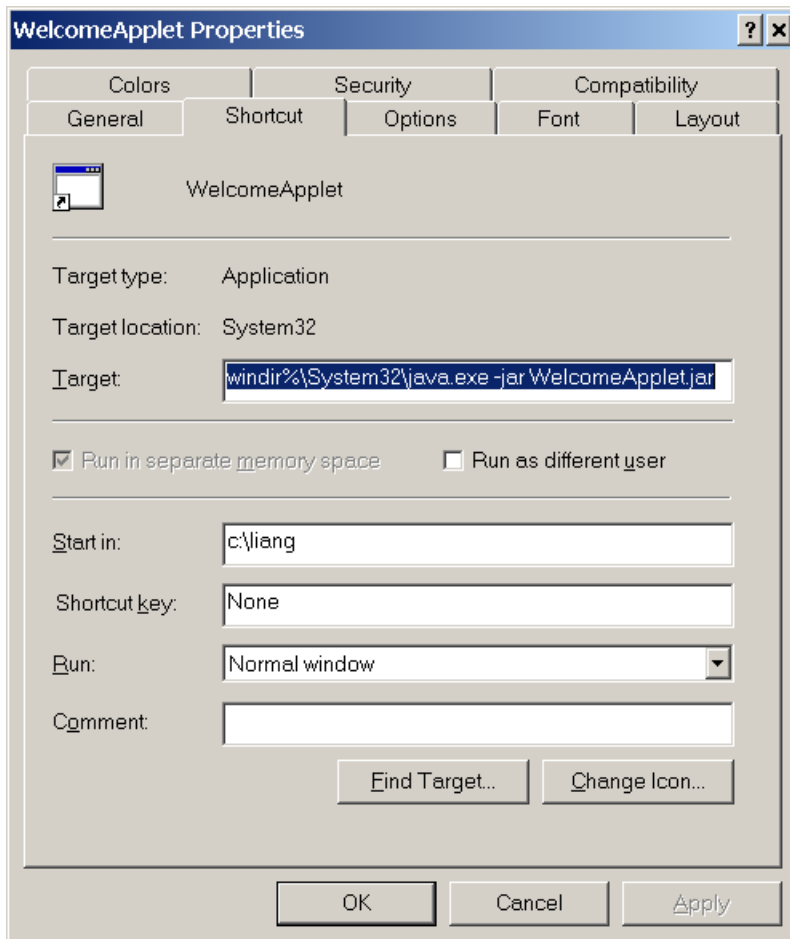


Figure 8.14

Select an appropriate name for the shortcut.