

Packaging and Deploying Java Projects in JBuilder

NOTE: This supplemental material may be used at the end of Chapter 12, "Applets and Advanced Graphics." Archive builder wizard is available in JBuilder Professional and JBuilder Enterprise.

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 JBuilder Archive Builder Wizard. The following command creates an archive file named `chapter12.jar` for classes `TicTacToe.class` and `TicTacToe$Cell.class`:

```
jar -cf chapter12.jar chapter12/TicTacToe.class chapter12/TicTacToe$Cell.class
```

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

Using the Archive Builder to Package Projects

With the **jar** command, you have to manually identify the dependent files. JBuilder provides the Archive Builder that gathers all the classes on which your program depends into one JAR archive that includes image and audio files.

NOTE: Archive builder is a feature in JBuilder Professional and Enterprise.

Let us use the `TicTacToe` class in Example 12.7, "The TicTacToe Game," to demonstrate packaging projects. Here are the steps to follow in generating the archive file:

1. Open the `chapter12.jpr` project. Choose Wizard, Archive Builder to start Archive Builder, as shown in Figure 1.

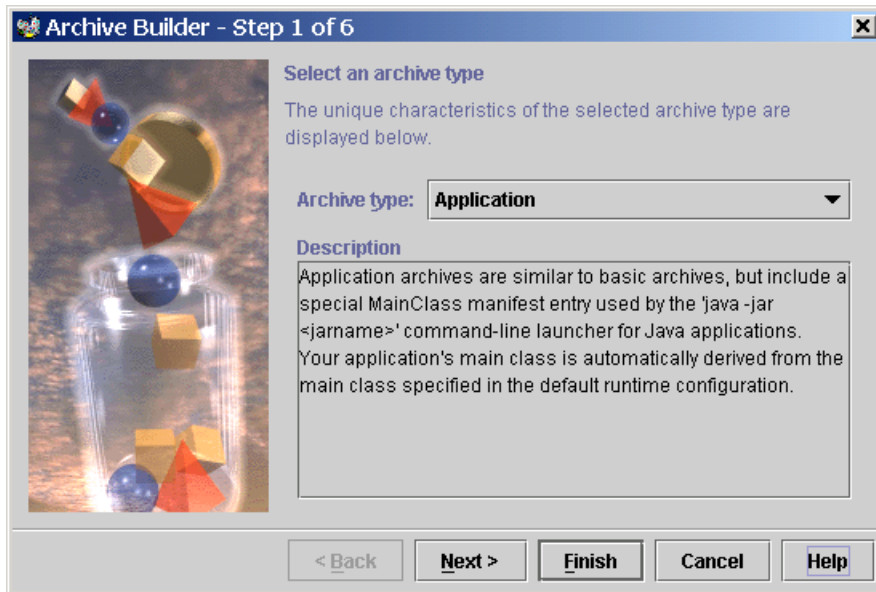


Figure 1

You can use the Archive Builder Wizard to create an archive file for the project.

2. Since `TicTacToe` can run as an application, choose `Application` as `Archive type`. Click `Next` to display `Archive Builder - Step 2 of 6`, as shown in Figure 2. Check `Compress the contents of the archive`. Click the ellipses to display a `File` dialog to enter a new file name `chapter12.jar` in the `File` field.

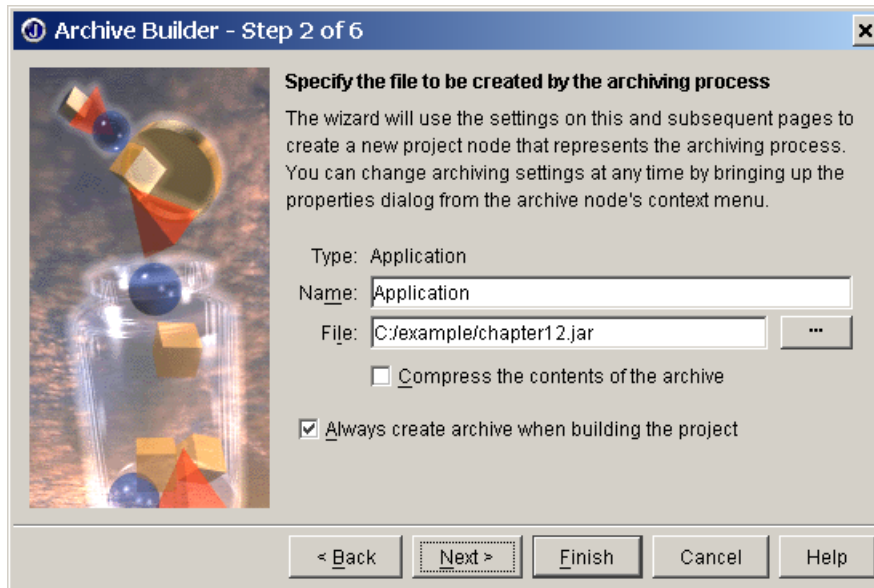


Figure 2

You specify an archive file in Step 2 of Archive Builder.

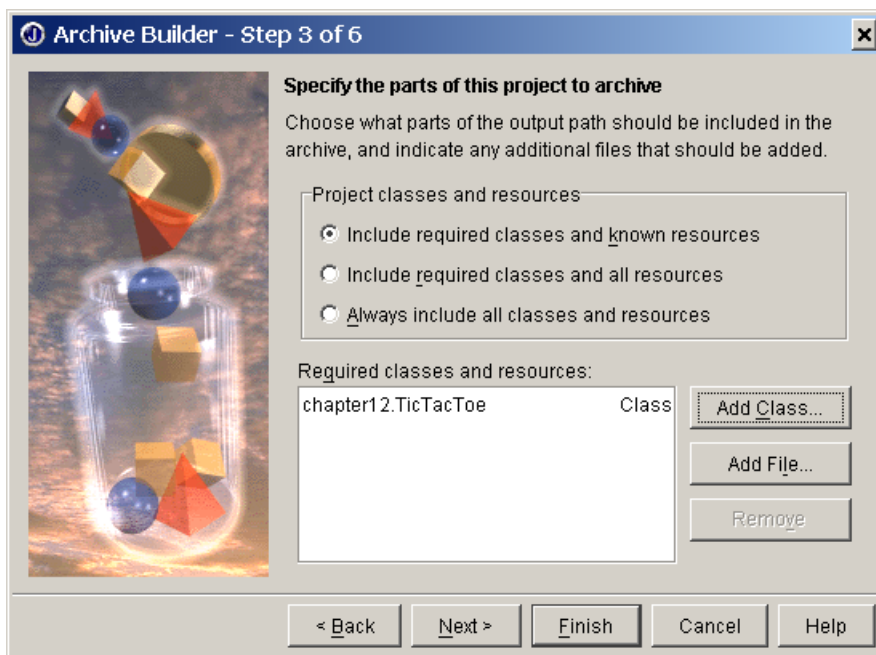


Figure 3

You specify the parts of the project to archive in Step 3 of Archive Builder.

3. Click Next to display Step 3 of 6 of Archive Builder, as shown in Figure 3. Check "Include required classes and known resources." Click Add Class to display the Select a Class dialog box to select chapter12.TicTacToe. Click

Next continually to skip Step 4, and 5. In Step 6 of 6 of Archive Builder, as shown in Figure 4, check Use the class specified below. Click the ellipses to display the Select Main Class for Archive dialog box to select chapter12.TicTacToe.

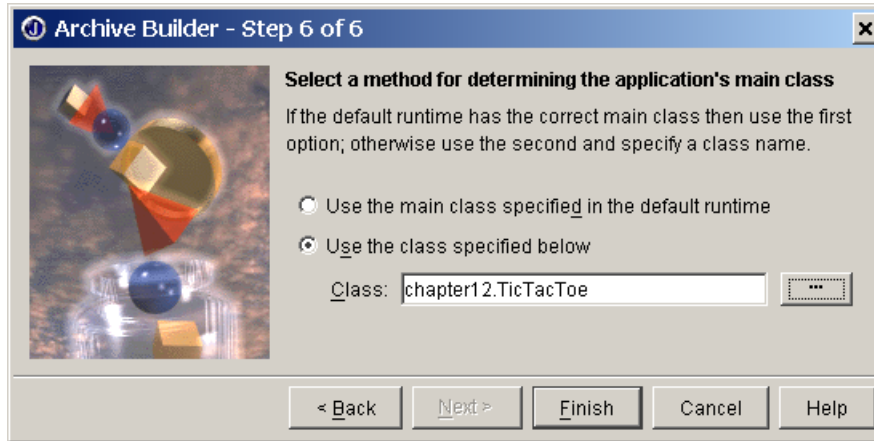


Figure 4

You specify the main method for the archive file in Step 6 of Archive Builder.

4. Click Finish to close the Archive Builder. Rebuild the project to create the archive file.

NOTE: You can view the contents of a .JAR file using WinZip, a popular compression utility for Windows, as shown in Figure 5.

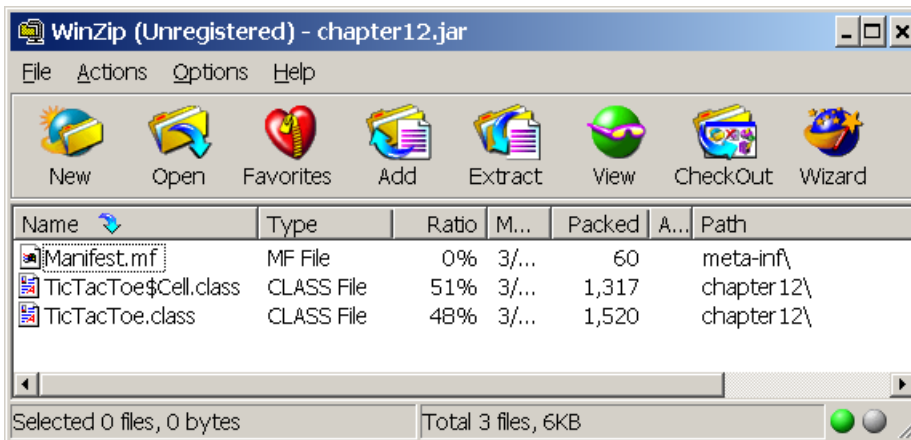


Figure 5

You can view the files contained in the archive file using the WinZip utility.

The Manifest File

As shown in Figure 5, a manifest file was created with the path name `meta-inf\`. The manifest is a special file that contains information about the files packaged in a JAR file. For instance, the manifest file in Figure 5 contains the following information:

```
Manifest-Version: 1.0  
Main-Class: chapter12.TicTacToe
```

The Main-Class line specifies that the jar file contains a Java application whose main class is `chapter12.TicTacToe`.

Running Archived Projects

The Archive Builder packages all the class files and dependent resource files into an archive 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.

Running Archived Files from Java Applications

To run `TicTacToe` as an application, simply type the following command:

```
java -jar chapter12.jar
```

Running Archived Files from Java Applets

To run `TicTacToe` file to include an ARCHIVE attribute. The ARCHIVE attribute specifies the archive file that contains the applet. For example, the HTML file for running the `TicTacToe` can be modified as follows:

```
<APPLET  
  CODE = "chapter12.TicTacToe.class"  
  ARCHIVE = "chapter12.jar"  
  WIDTH = 400  
  HEIGHT = 300  
  HSPACE = 0  
  VSPACE = 0  
  ALIGN = Middle  
>  
</APPLET>
```

NOTE: If your Java source code is changed, you need to update the .jar file. To update the .jar file, right click the project file in the project pane to display the context menu, and choose Rebuild.