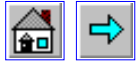


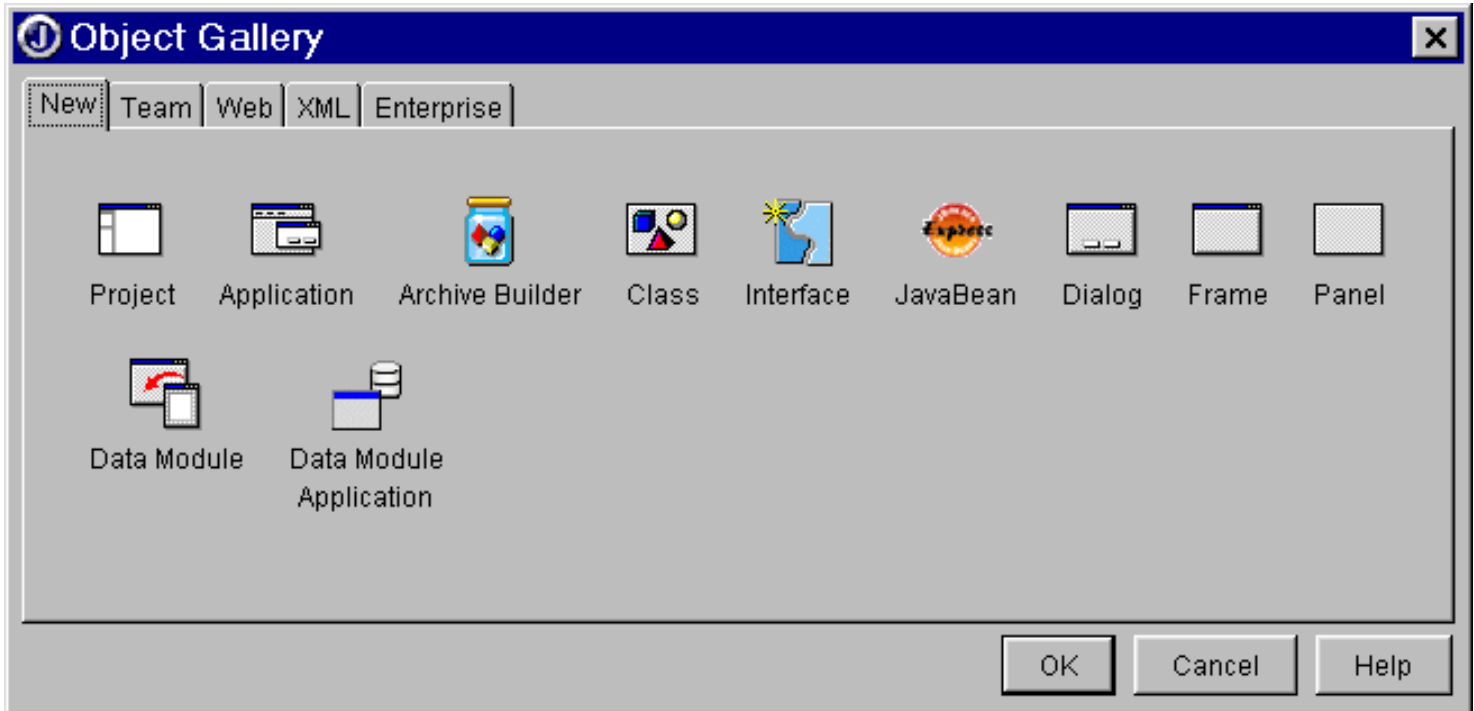
JBuilder 5 Tutorial: Building an application



Step 1: Creating the project

Before creating an application in JBuilder, you must first create a project to work in. JBuilder uses a project file (.jpx or .jpr) to organize the application files and maintain the settings and project properties. The Project wizard can create a project for you.

1. Choose File|New to open the object gallery.



2. Double-click the Project icon to open the Project wizard.
3. Make the following changes to the appropriate fields in Step 1 of the Project wizard:

1. Project Name field: HelloWorld

Note: As you type in the Project Name field, the same name is entered in the Project Directory Name field by default. The project is saved to this HelloWorld project directory.

2. Type (File type) field: .jpr or .jpx

Note: JBuilder uses a .jpr or .jpx extension for project files. The .jpr file type is useful for general purposes. The .jpx file type, an XML project file, is used in a team development environment and version control. Either file type works for this tutorial. The tutorial uses the .jpr extension.

Step 1 of the Project wizard should look similar to this:

Project Wizard - Step 1 of 3

Select name and template for your new JBuilder project

Enter names for your project and its directory. Also select an existing project to provide default settings.

Project name: **Type:**

Select project template and directory names

Project for defaults: ...

Root path: ...

Project directory name:

Source directory name:

Backup directory name:

Documentation directory name:

Output directory name:

Project directory is parent to source and output directories

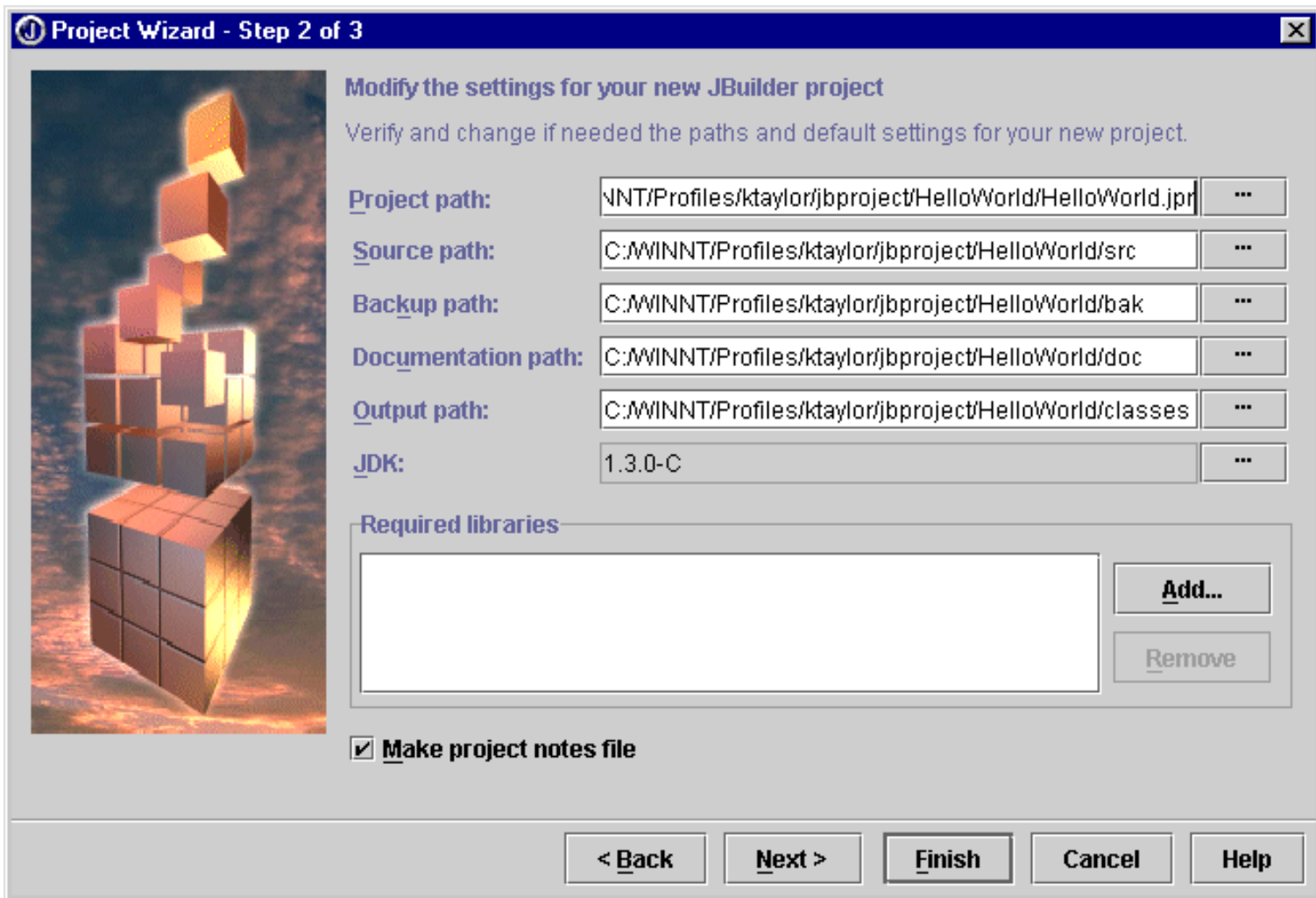
< Back Next > Finish Cancel Help

4. Accept all other defaults in Step 1. Note the root path where the project is saved.

Note: Projects in JBuilder are saved by default in the /<home>/jbproject/ directory. The home directory varies by platform. See "[Documentation conventions.](#)"

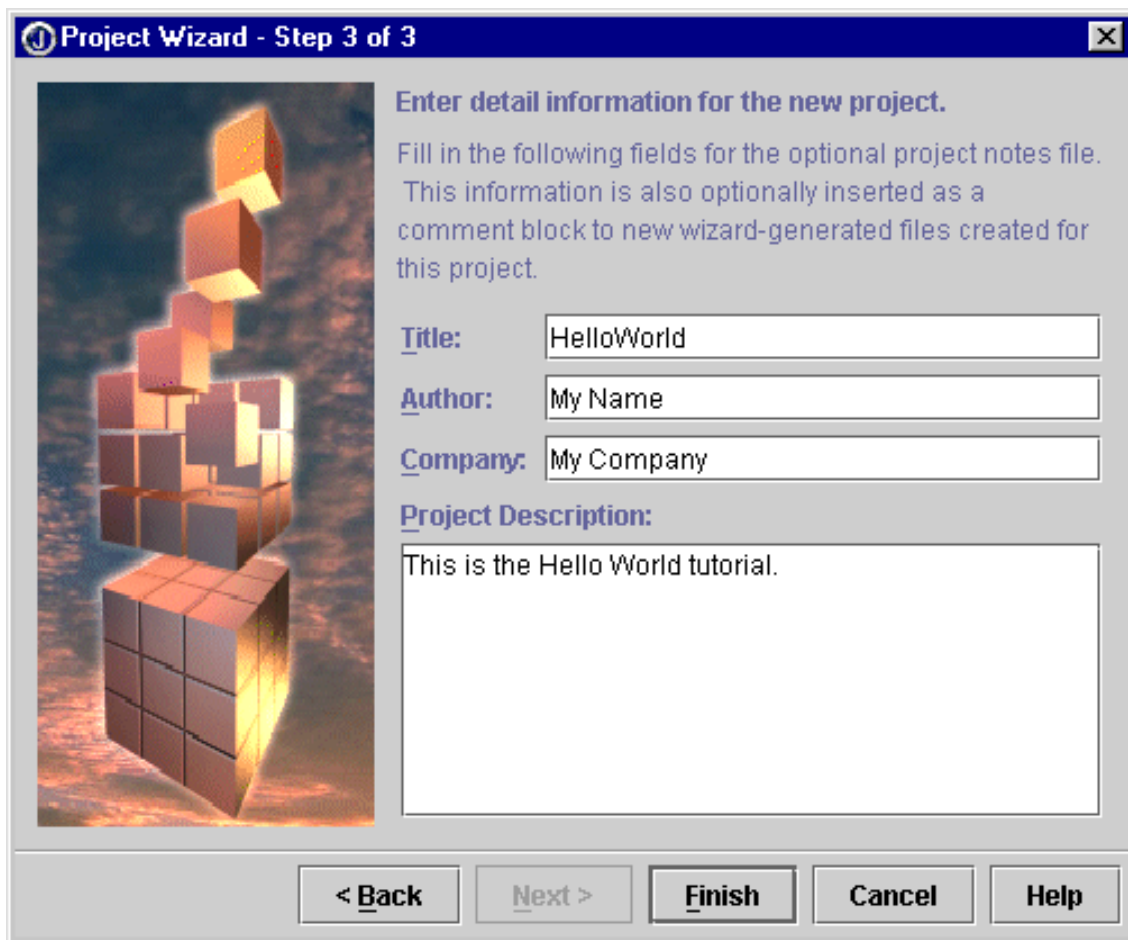
For more information on projects, see "[Creating and managing projects](#)" in Building Applications with JBuilder.

5. Click Next to go to Step 2 of the Project wizard.



6. Accept the defaults in Step 2 for the project, source, backup, documentation, and output paths and the JDK version. Note where the project, class, and source files will be saved. Also note that the option, Make Project Notes File, is checked. This option creates an HTML file that contains the project information entered in the next step of the wizard.
7. Click Next to go to Step 3 of the wizard.
8. Make the following changes in the appropriate fields of Step 3:
 1. Type HelloWorld in the Title field.
 2. Enter your name, company name, and a description of your application in the appropriate optional fields. This information appears in the project HTML file and as optional header comments in the source code.

Step 3 of the Project wizard looks similar to this:



- Click the Finish button. Two files, `HelloWorld.jpr` and `HelloWorld.html`, are generated by the wizard and appear in the project pane of the AppBrowser.

See also: ["How JBuilder constructs paths"](#) and ["Where are my files?"](#) in "Creating and managing projects" in Building Applications with JBuilder.

- Double-click `HelloWorld.html`, the project notes file, to view it in the content pane. Note that it contains the project name, author, company, and description information you just entered in Step 3 of the Project wizard.



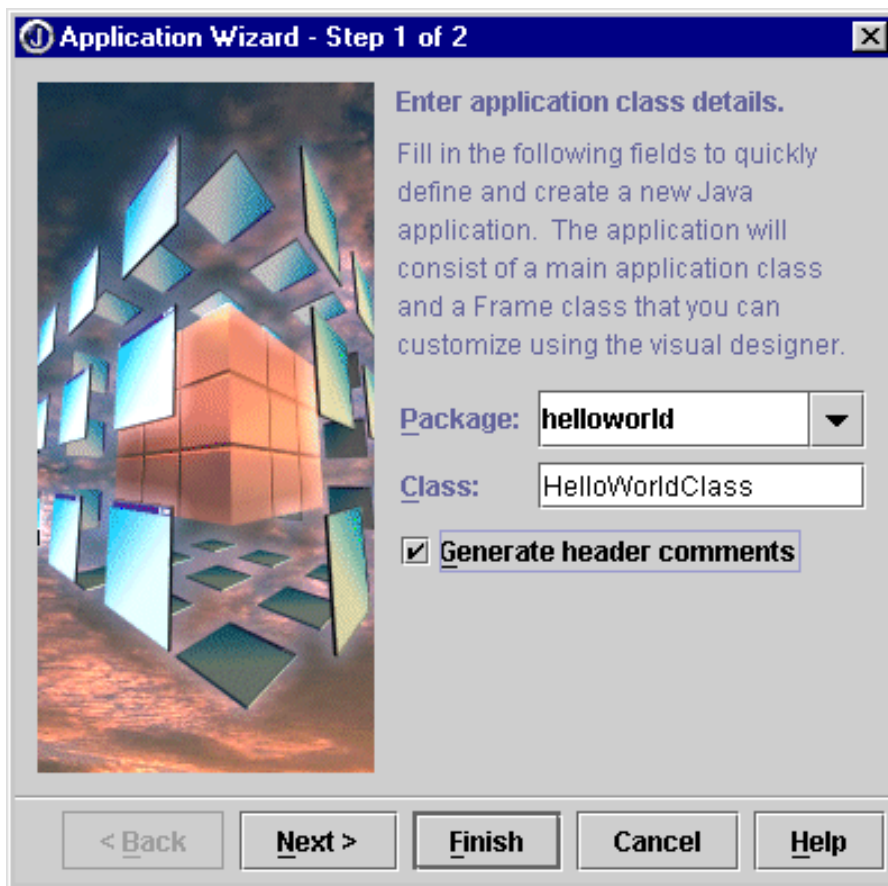
Step 2: Generating your source files

The Application wizard creates `.java` source files that are added to the project you just created.

To generate source files for your application using the Application wizard, follow these steps:

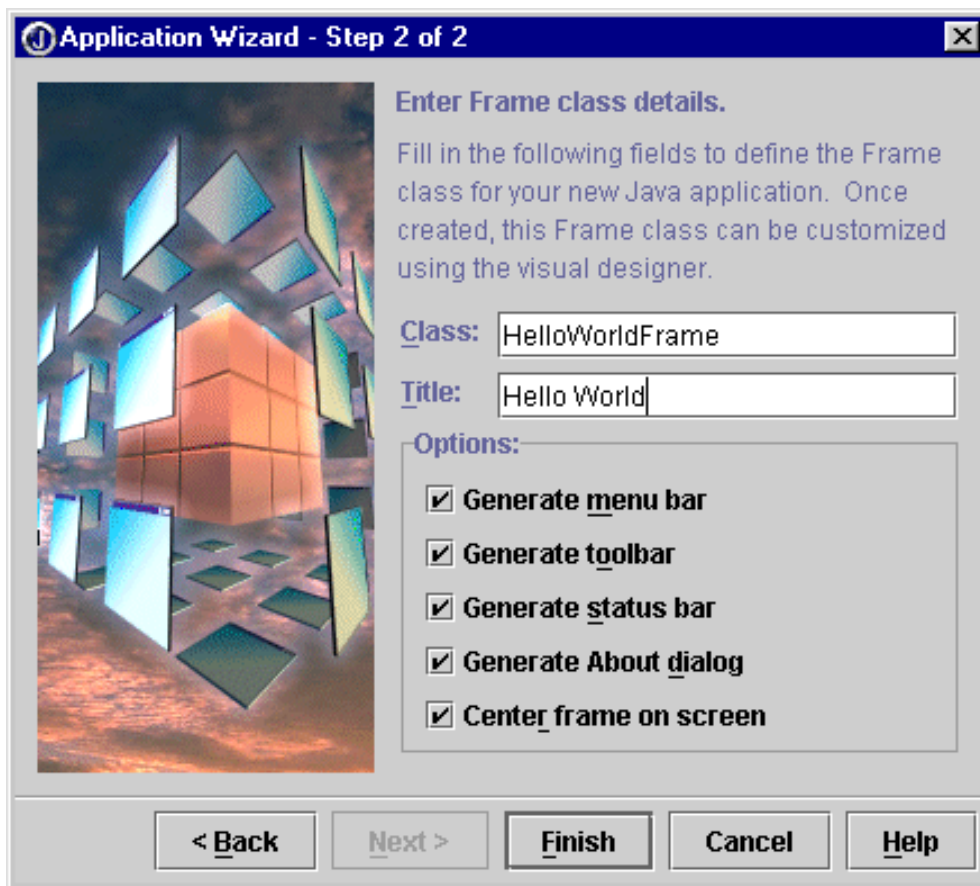
- Select File|New to open the object gallery.
- Double-click the Application icon to open the Application wizard.
- Accept the default package name, `helloworld`, in Step 1 of the Application wizard. By default, the wizard takes the package name from the project file name, `HelloWorld.jpr`.
- Enter `HelloWorldClass` in the Class field. This is a case-sensitive Java class name.
- Check Generate Header Comments. When you select this option, the project notes information you entered in Step 3 of the Project wizard appear at the beginning of each source file that the Application wizard generates.

Step 1 of the Application wizard should look like this:



6. Click the Next button to go to Step 2 of the Application wizard.
7. Type HelloWorldFrame in the Class field to name the Frame class.
8. Type Hello World in the Title field. This text appears in the title bar of the frame in your application.
9. Check all of the options for additional application features: Generate Menu Bar, Generate Toolbar, Generate Status Bar, Generate About Dialog, and Center Frame On Screen. The wizard generates the basic code to support these features.

Step 2 of the Application wizard should look like this:

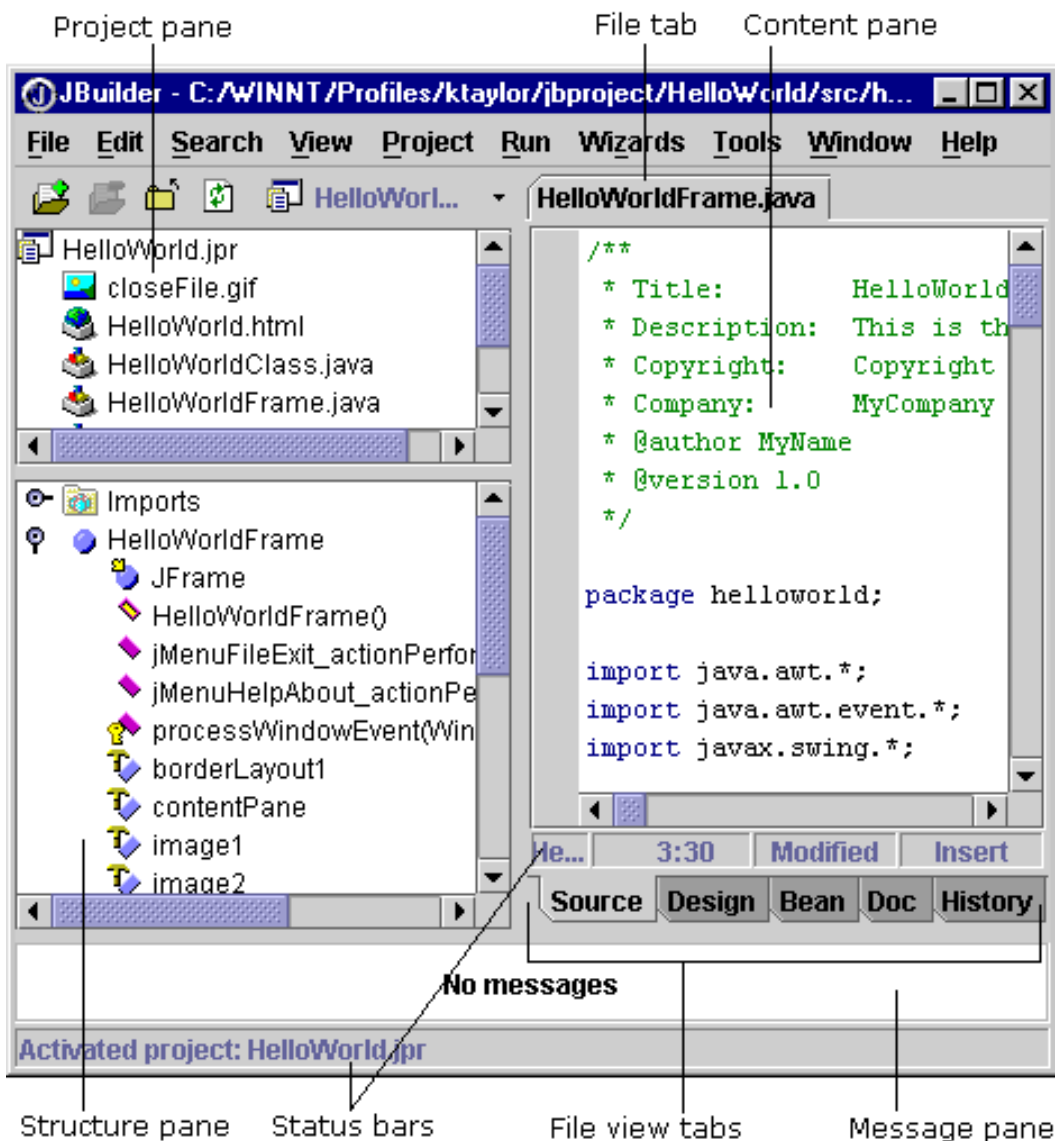


10. Click the Finish button.

The new .java source files and tool bar images are added to your project and displayed as nodes in the project pane. The source code for HelloWorldFrame.java is open in the content pane as shown in the following image.

Note: In JBuilder Professional and Enterprise editions, an automatic source package node named helloworld also appears in the project pane if the Enable Source Package Discovery And Compilation option is enabled on the [General page](#) of the Project Properties dialog box (Project|Project Properties).

AppBrowser elements



11. Choose File|Save All to save the source files and the project file.

Note: The source files are saved to: /<home>/jbproject/HelloWorld/src/helloworld
The class files are saved to: /<home>/jbproject/HelloWorld/classes/helloworld



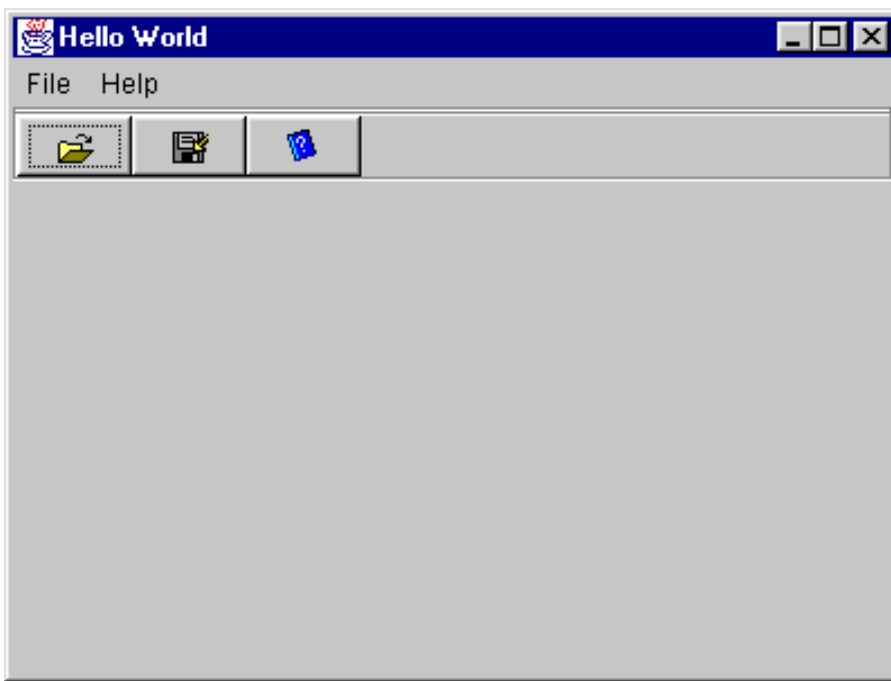
Step 3: Compiling and running your application

Now, compile and run the application.

1. Choose Run|Run Project  or click the Run button to compile and run your application.

Tip: You can also select HelloWorldClass.java in the project pane, right-click, and select Run.

First, the message pane opens displaying the run process. Then, your application is displayed and should look like this:



Note: The running application in this tutorial reflects the Windows Look & Feel.

2. Choose File|Exit in the "Hello World" application to close it.
3. Right-click the HelloWorldClass tab in the message pane at the bottom of the AppBrowser and select Remove "HelloWorldClass" Tab to close any messages.

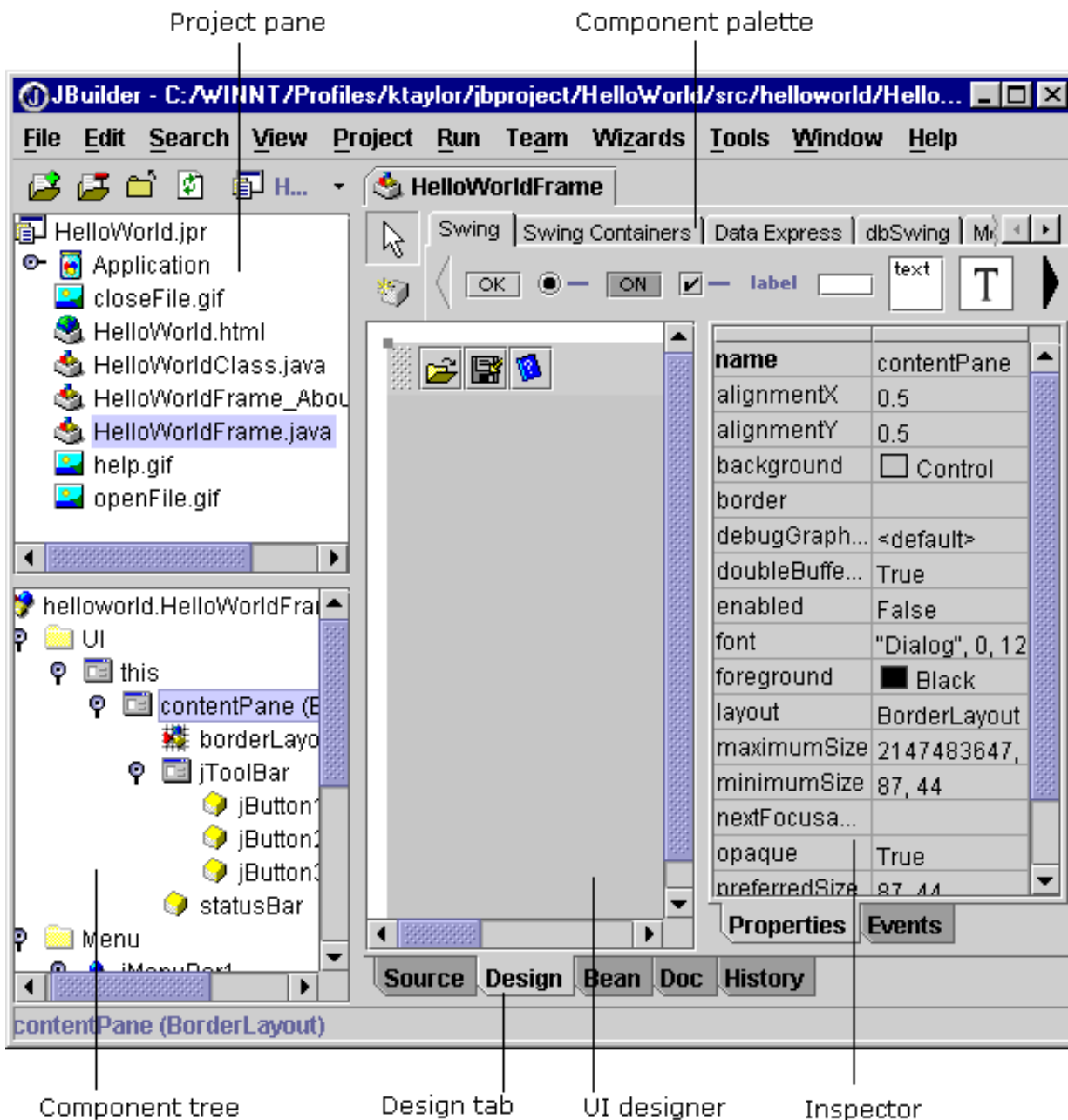


Step 4: Customizing your application's user interface

Follow these steps to customize your application's user interface.

1. Double-click `HelloWorldFrame.java` in the project pane if it's not already open.
2. Click the Design tab to change to design view. The UI designer appears in the content pane with the component palette above it and the Inspector on the right. You'll use the component palette to add components to your UI and the Inspector to modify properties and add events to your code. The structure pane now contains a component tree containing components and such folders as UI, Menu, and Other.

UI designer elements



3. Click the Swing Containers tab on the component palette above the UI designer and choose the JPanel component to add a panel to your design.

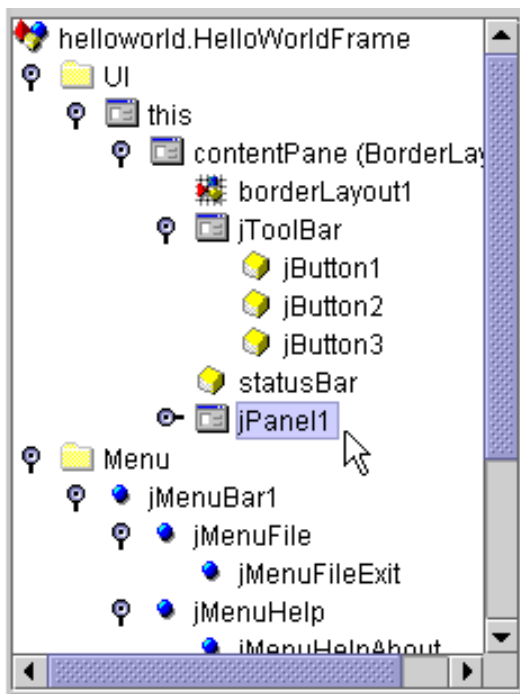
Tip: Place the cursor over a component on the palette to see its name in a tool tip.




4. Click the center of the frame in the UI designer to drop the component into the center of your design's frame.

Note: The constraints property in the Inspector should be Center. If not, click the column to the right of the constraints property, and choose Center from the drop-down list.

jPanel1 is now selected in your application design and in the component tree.



Note: The component selected in the component tree or the UI designer displays in the status bar below the structure pane.

5. Set the background color of `jPanel1` to White.
 1. Click the column to the right of the background property in the Inspector.
 2. Click the Down arrow to open the color drop-down list and select White at the top of the list.
6. Add a line border to `jPanel1` and change the border color to Gray.
 1. Click the column to the right of the border property in the Inspector.
 2. Click the Down arrow to open the border drop-down list and select Line.
 3. Select the ellipsis button  to access the Border dialog box.
 4. Click Black under Options|Color to access the color drop-down list and select Gray.
 5. Click OK to close the Border dialog box.
7. Change the layout manager for `jPanel1` to `null`.
 1. Click the column to the right of the layout property in the Inspector.
 2. Choose `null` from the drop-down list.

`null`, or no layout, is a good choice when prototyping your design. Because `null` does not use a layout manager, you can place components exactly where you want them. However, because `null` uses absolute positioning of components instead of relative positioning, the UI will not resize properly when the user resizes the application window. Therefore, it's recommended that you never leave a container in `null` for deployment. Later in the tutorial, you'll switch to an appropriate portable layout for your design before deploying it.

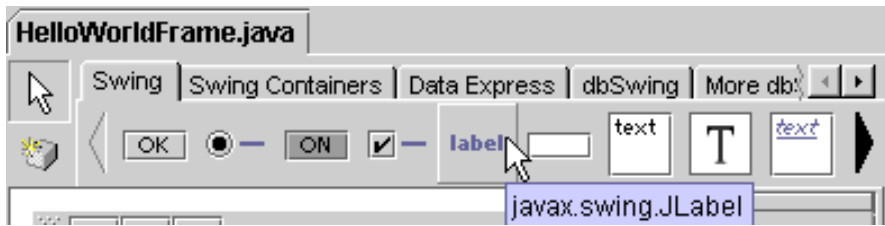
8. Save the project.



Step 5: Adding a component to your application

Now you'll use the component palette to add a `JLabel` component to the `JPanel` component.

1. Select the Swing tab on the component palette and click the `JLabel` component.

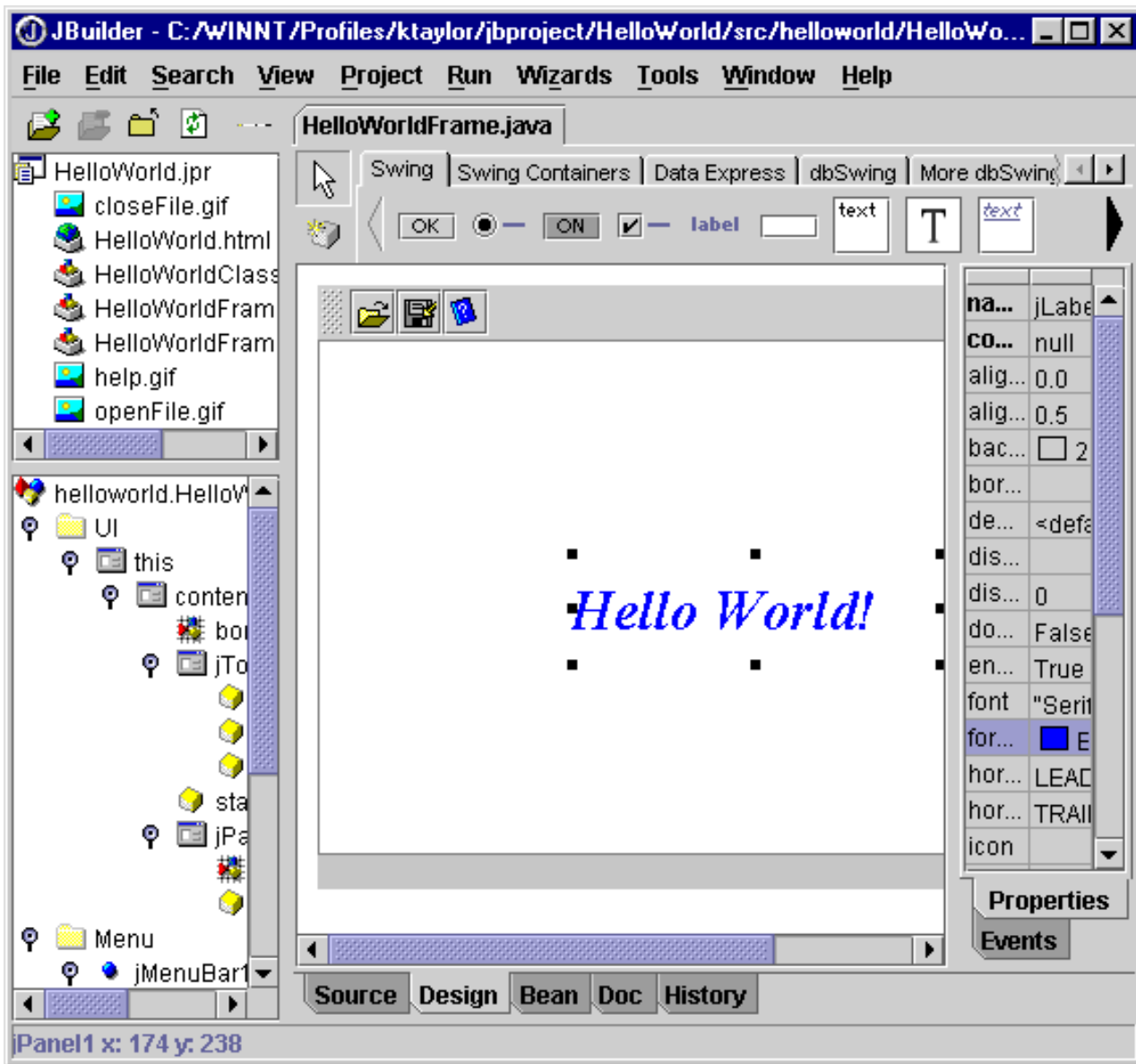


2. Drop the component into your design's JPanel1 using one of the following two methods:
 - Click JPanel1 in the component tree. This places it in the upper-left corner of the panel.
 - Click JPanel1 in the UI designer. The upper-left corner of the component is placed where you click.

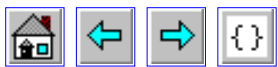
Note that JLabel1 is added below JPanel1 in the component tree. If you drop the component in the wrong place, you can select it in the component tree or in the designer and press the Delete key. Then re-add it.

3. Click the middle of the label component in the designer, and drag it to the center of the panel.
4. Select JLabel1 in the component tree and complete the following steps:
 1. Double-click the column to the right of the text property in the Inspector and type Hello World! Press Enter. "Hello World!" partially appears in the label. Don't worry if it doesn't completely show in the label. You'll fix that after changing the font.
 2. Click the column to the right of the font property to set the font. Click the ellipsis button to open the Font dialog box.
 3. Choose Serif from the list of fonts and check Bold and Italic. Enter 28 in the Size box, then click OK.
 4. Resize JLabel1 by dragging the black handles until "Hello World!" is completely visible.
 5. Click the column to the right of the foreground property in the Inspector to set the color of the "Hello World!" text. Click the Down arrow and select Blue from the drop-down list of colors.

Your design now looks similar to this:



5. Choose File|Save All.



Step 6: Editing your source code

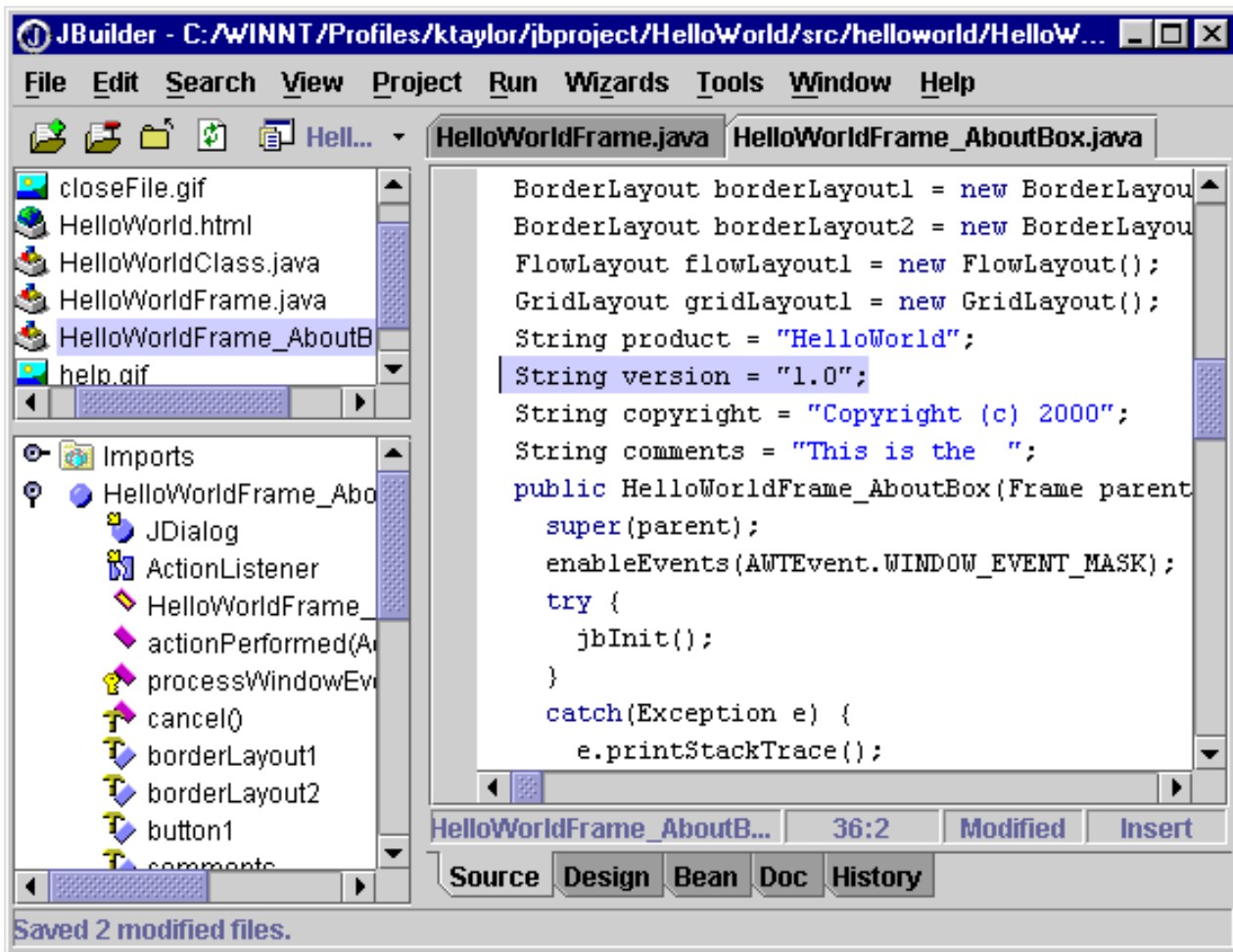
You can change information in the About box by directly editing the code. The default application version created by the Application wizard is version 1.0.

1. Double-click `HelloWorldFrame_AboutBox.java` in the project pane to open the file. The content pane changes to the source view where you can edit the code in the editor.
2. Choose Search|Find. Type the following line of code in the Find/Replace Text dialog box:

```
String version = "1.0";
```

3. Click Find.

The editor finds the selected text.



4. Select 1.0 and enter 2.0 inside the quotes.
5. Choose File|Save All.



Step 7: Compiling and running your application

Now you can compile and run the application.

1. Choose Project|Make Project.
2. Choose Run|Run Project.

The "Hello World" application is displayed:



3. Choose Help|About. The version data you changed is now displayed in the About dialog box.



4. Click OK to close the dialog box.
5. Choose File|Exit in your "Hello World" application to close it.



Step 8: Running your application from the command line

You can also run the application outside the JBuilder environment from the command line.

Note: The `jdk/bin/` directory which contains the **java** command must be on your path. If **java** is on your path, when you type `java` at the command line, information explaining the command should display. If it's not on your path, you need to run the application from within the `jdk/bin/` directory.

To run the application,

1. Open the command-line window.
2. Run the application with the following command on one line at the command prompt:

```
java -classpath
/<home>/jbproject/HelloWorld/classes helloworld.HelloWorldClass
```

Note: For Windows, use a backslash (\).

This command should be in the following form:

```
java -classpath package-name.main-class-name
```

In this example,

- java = the launcher for the Java application
- -classpath = an option that sets the search path for application classes and resources.
- classpath = /<home>/jbproject/HelloWorld/classes
The classpath should point to the directory containing the compiled classes. In this example, the classes directory was set as the output path for compiled classes on Step 1 of the Project wizard.
- <home> = your home directory, for example, c:\winnt\profiles\
- package-name = helloworld
- main-class-name = HelloWorldClass


3. Close the "Hello World" application.

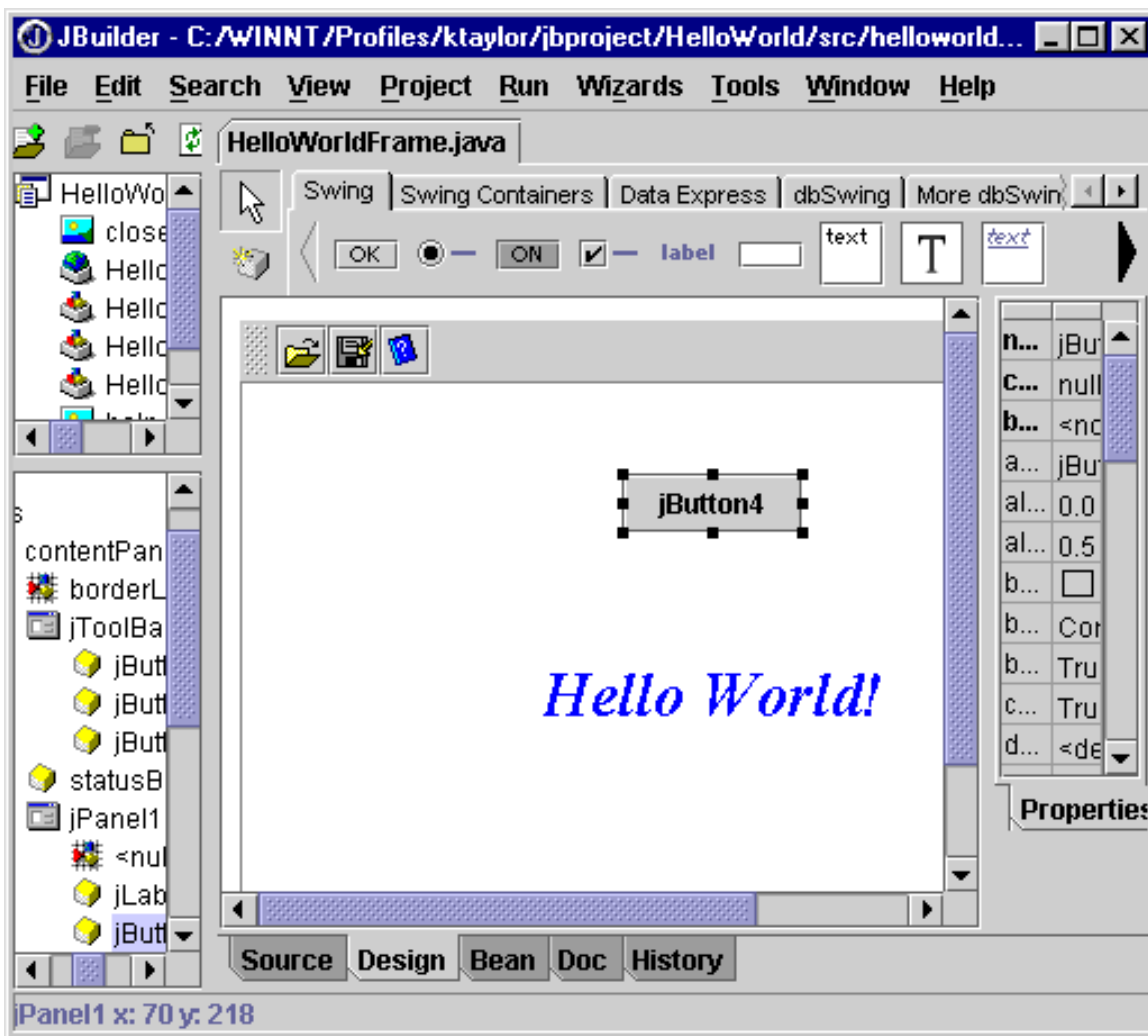
See also: "[Setting the classpath](http://java.sun.com/j2se/1.3/docs/tooldocs/tools.html)" and "[Basic tools](http://java.sun.com/j2se/1.3/docs/tooldocs/tools.html)" at <http://java.sun.com/j2se/1.3/docs/tooldocs/tools.html>



Step 9: Adding an event to a button

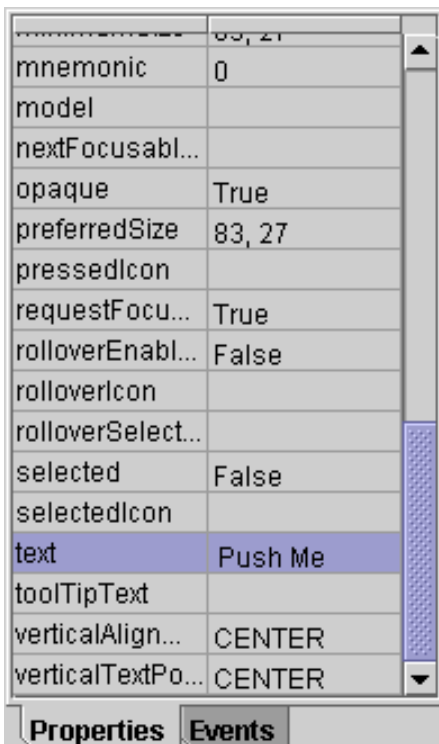
Next, you'll add another Swing component to your application.

1. Open HelloWorldFrame.java and click the Design tab to switch to the UI designer.
2. Click the JButton component  on the Swing tab of the component palette and drop it on either JPanel1 in the component tree or in the panel in your design. jButton4 is added below JPanel1 in the component tree.
3. Click jButton4 in the design and drag it to the top center of the design as shown in the image below.



4. Change the Text property in the Inspector from jButtonon4 to Push Me. Press Enter. Enlarge the button by dragging the black handles until "Push Me" shows completely.

The Inspector should look like this:



5. Click the Inspector's Events tab to define what happens when jButton4 is pressed.
6. Double-click the column to the right of the ActionPerformed event.

JBuilder switches to the editor where the following skeleton code has been added for the ActionPerformed event.

```
void jButton4_actionPerformed(ActionEvent e) {  
  
}
```

You can now enter code that defines the event.

7. Type the following code indicated in bold:

```
void jButton4_actionPerformed(ActionEvent e) {  
    jLabel1.setForeground(new Color(255,0,0));  
}
```

Tip: Use CodeInsight to complete the code for you. Type `jLabel1.` and wait for the pop-up window or press `Ctrl+spacebar` to invoke it. Type `setfor` to highlight `setForeground(Color)` in the pop-up window or use the arrow keys. Press Enter. You can configure CodeInsight in the Editor Options dialog box (Tools|Editor Options|CodeInsight).

Now, when you run the application and push the "Push Me" button, "Hello World!" should turn red.

8. Choose File|Save All.
9. Choose Project|Make Project
10. Choose Run|Run Project.
11. Click the "Push Me" button. The color of the "Hello World!" text turns red.



12. Choose File|Exit to close the "Hello World" application.



Step 10: Completing your UI

As the final step in completing your application, you need to change your layout to a portable layout. Remember, if you leave `JPanel1` as null or no layout, the components in your UI will not reposition when the user resizes the application window at runtime. Because null uses absolute positioning, the components remain in the same location, no matter what size the window. Therefore, null is not a good layout manager for final deployment of your application.

To demonstrate this window resizing problem with null, run the application as follows:

1. Right-click `HelloWorldClass.java`, which contains the `main()` method, and choose Run.
2. Resize the "Hello World" application window, making it larger and smaller, and observe the behavior of the components. Notice that the label and button components do not move as you resize the window.



3. Close the "Hello World" application.

Portable layouts, unlike null, use relative positioning which is dynamic. Components in portable layouts reposition correctly when the user resizes the application window. In this example, you'll change the layout to `GridBagLayout`. For more information on layouts, see ["Using layout managers"](#) in Building Applications with JBuilder.

1. Return to the `HelloWorldFrame.java` file in the content pane and click the Design tab to switch to the designer.
2. Choose `JPanel1` in the component tree.
3. Choose the Inspector's Properties tab.
4. Change `JPanel1`'s layout to `GridBagLayout` in the Inspector. Select each component in the designer and notice the grid area each component fills.

`GridBagLayout` is a good choice for laying out components of varying sizes in a grid. See ["GridBagLayout"](#) in Building Applications with JBuilder.

5. Save and run the application.
6. Resize the application window and notice how the components reposition as the window changes in size.



7. Close the "Hello World" application.