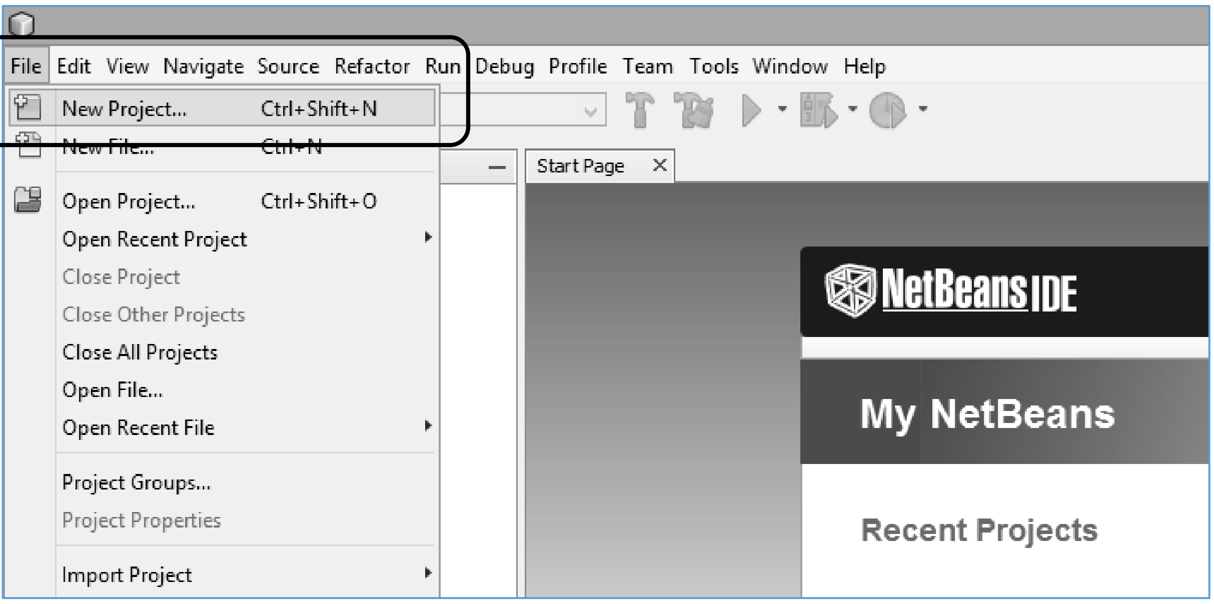
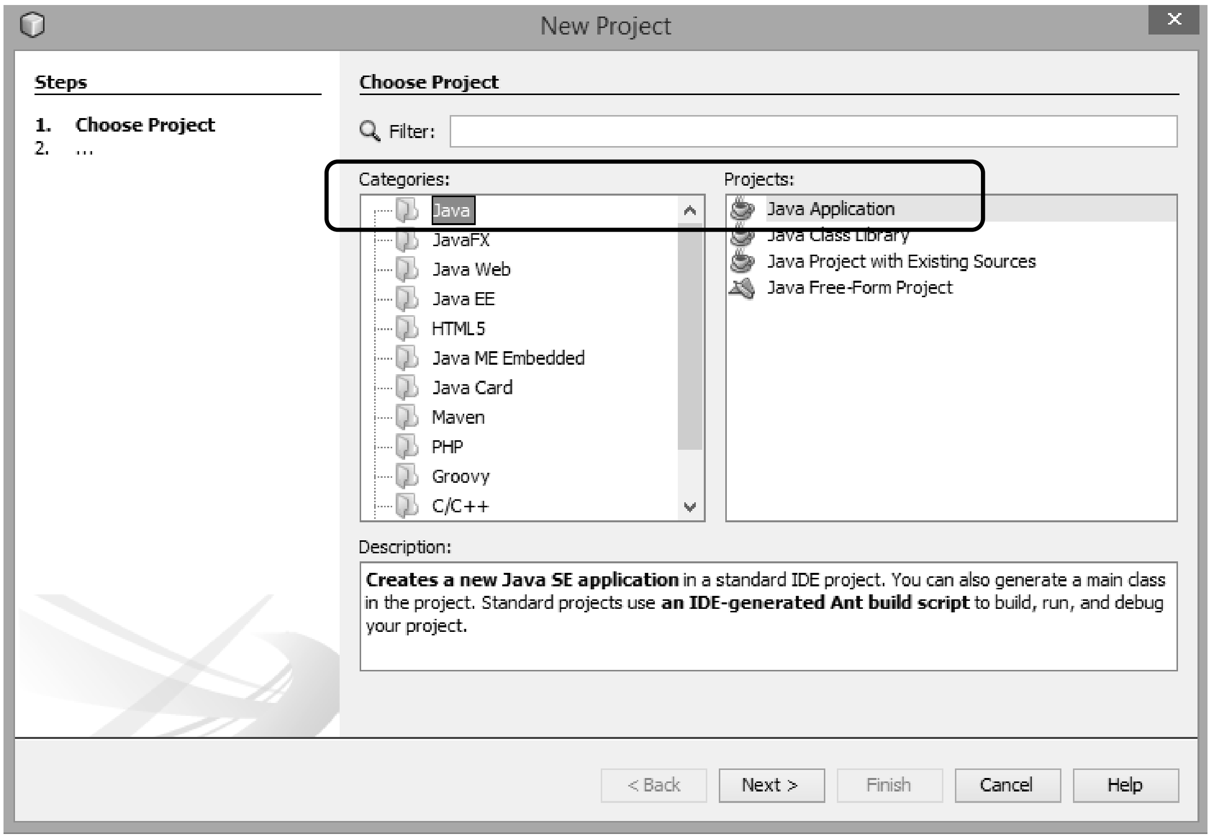
1 Using the NetBeans IDE

In this chapter we will examine how to set up a new application program using the ***NetBeans Integrated Development Environment*** with the language Java. We will create a simple program to display a picture, then allow the picture to be turned on or off by means of buttons.

Start the NetBeans IDE software and select ***File*** from the main menu at the top of the screen. Select ***New Project*** from the drop down list:



Check that ***Java / Java Application*** is highlighted, then click the ***Next*** button:

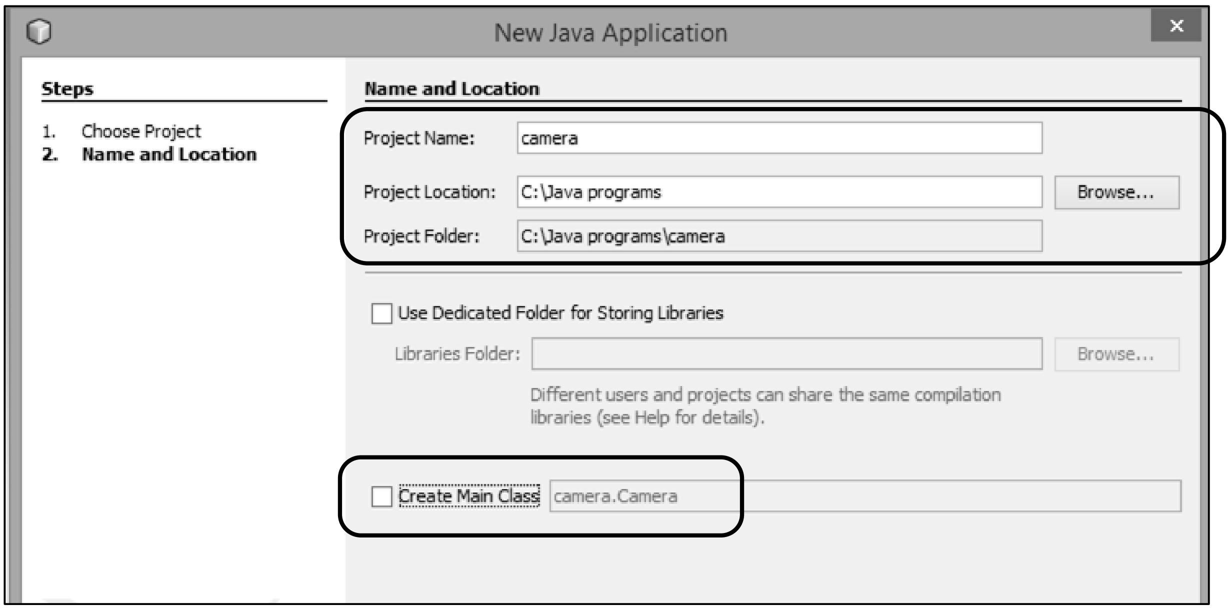


Creates a new Java SE application in a standard IDE project. You can also generate a main class in the project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug your project.

In the window which now opens, you will be asked to give a name for the project you are creating. Use the project name: ***camera***.

You also need to choose a location where the project will be saved. This can be on the computer hard disc or on a USB memory stick. Use the Browse button to find the location.

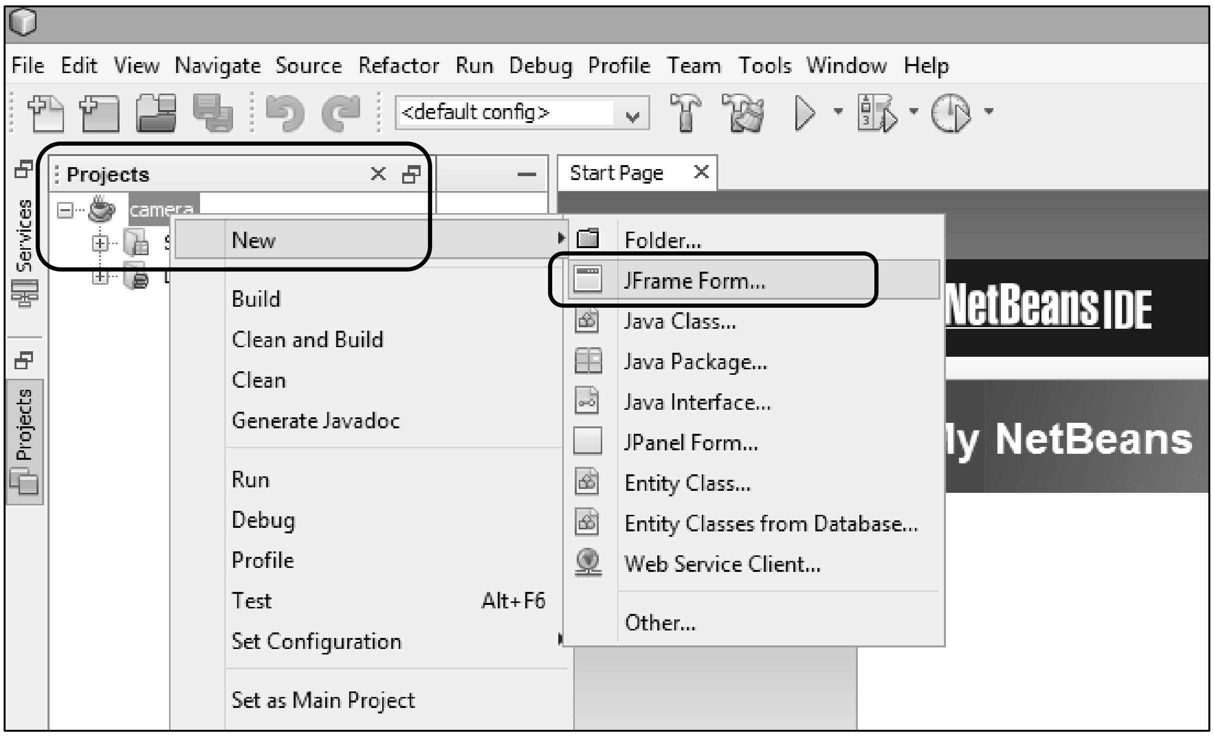
Finally, un-tick the box labelled ***Create Main Class***. We will not require this option.


New Java Application
Steps
1. Choose Project
2. Name and Location
Name and Location
Browse...
Use Dedicated Folder for Storing Libraries
Libraries Folder:
Browse...
Different users and projects can share the same compilation libraries (see Help for details).
Create Main Class

Project Name: camera
Project Location: C:\Java programs
Project Folder: C:\Java programs\camera
Create Main Class: camera.Camera

Click the Finish button to return to the main NetBeans page.

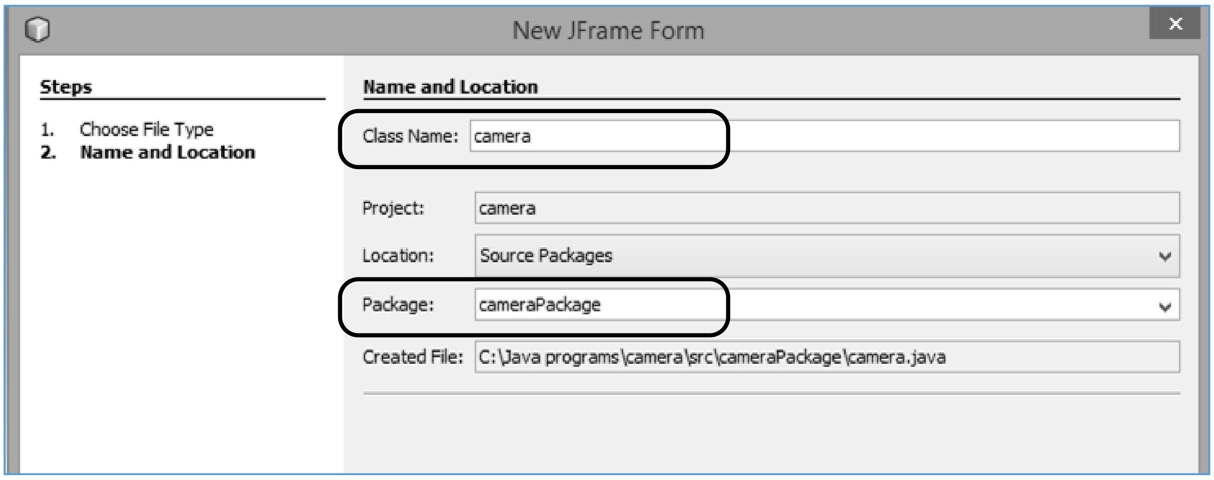
Click the ***Projects*** tab at the left of the page, then right click on the ***Camera*** project icon to open a drop-down menu. Select ***New***, then ***JFrame Form***:



We are now going to create a page which will appear when the program runs.

In the section for **Class Name**, give the same name as the project. This is **camera**.

In the section for ***Package***, give the project name followed immediately by the word Package. In this case, the name will be ***cameraPackage***.



Class Name: camera

Project: camera

Location: Source Packages

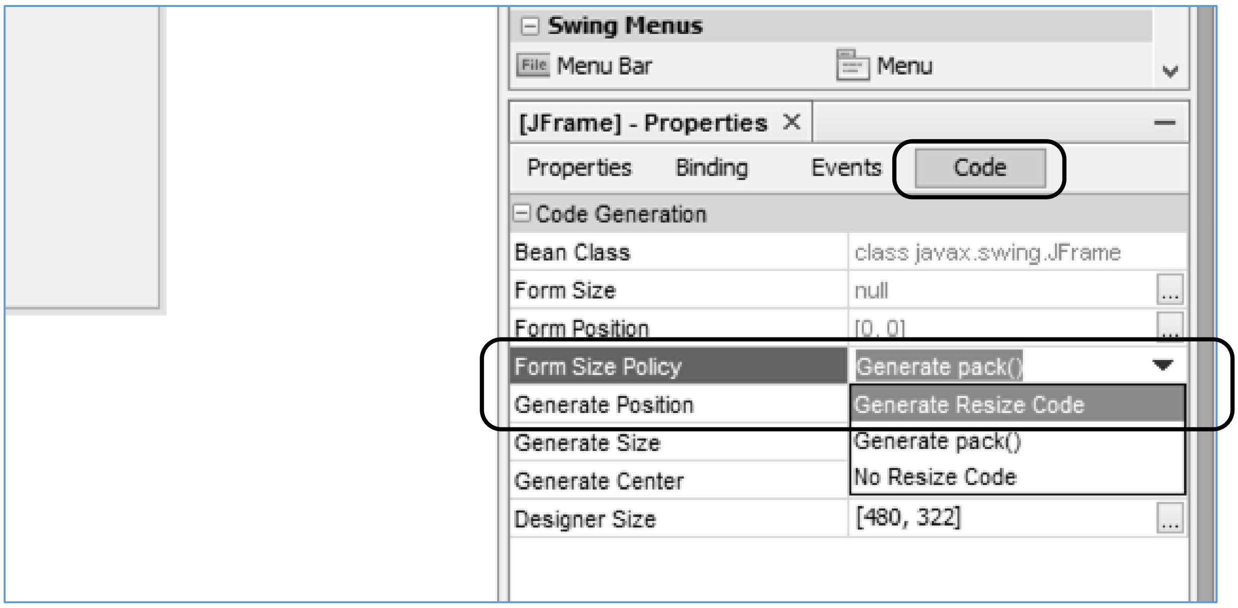
Package: cameraPackage

Created File: C:\Java programs\camera\src\cameraPackage\camera.java

Click the ***Finish*** button. An empty rectangular ***form*** will be displayed, ready to create your page design.

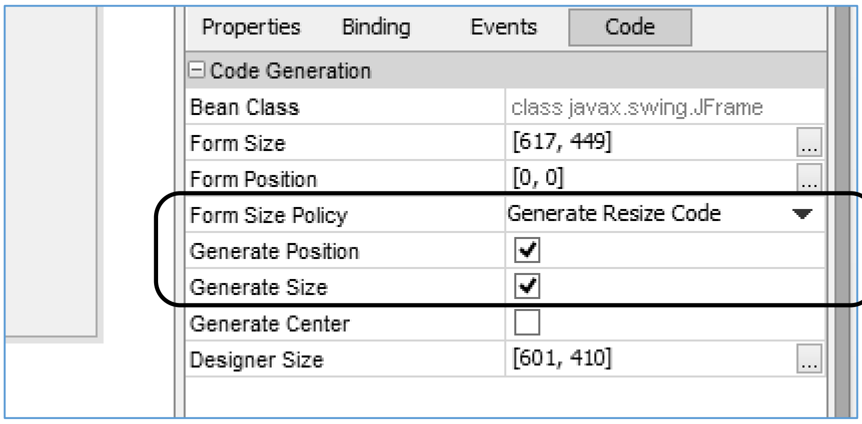
Click on the form and go to the ***Properties*** window at the bottom right of the screen. Select the ***Code*** tab. We are going to create some program code which will allow us to set the size of the window which will appear when the program runs.

Select ***Generate pack()***, then click the ***Generate Resize Code*** option:



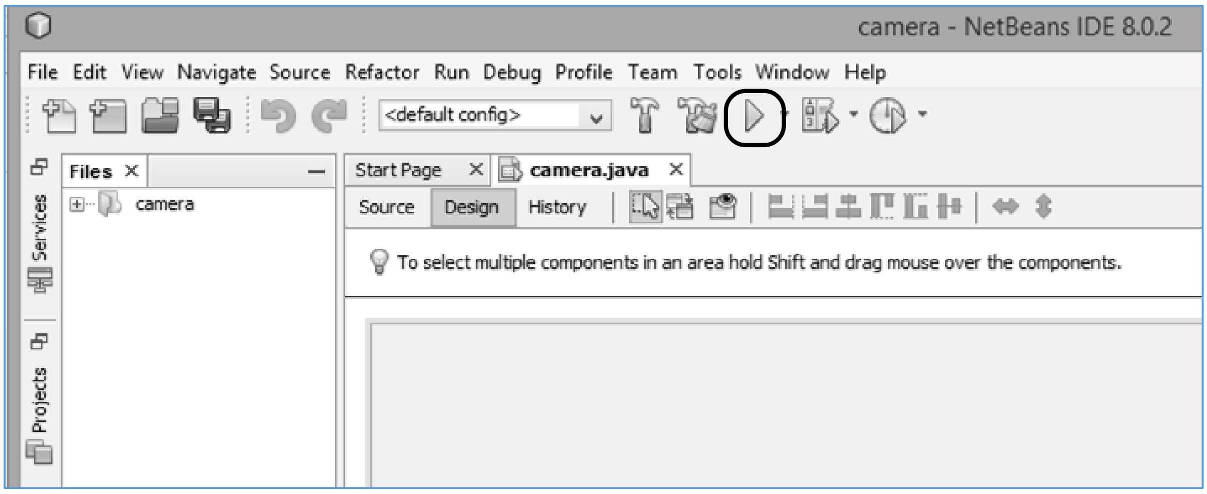
Bean Class	class javax.swing.JFrame
Form Size	null
Form Position	[0, 0]
Form Size Policy	Generate pack()
Generate Position	
Generate Size	
Generate Center	
Designer Size	[480, 322]

After doing this, ticks should appear in the boxes labelled ***Generate Position*** and ***Generate Size***:



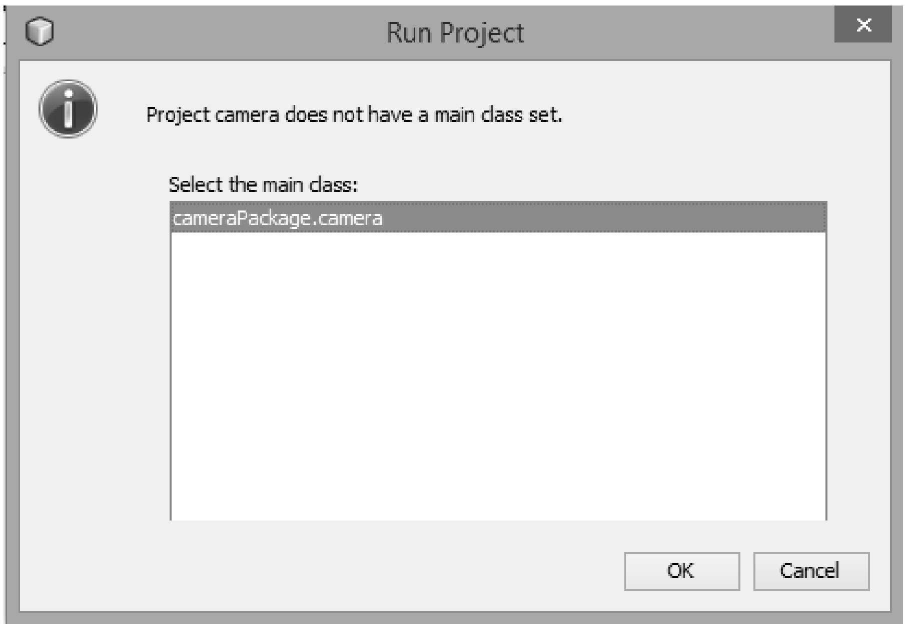
Bean Class	class javax.swing.JFrame
Form Size	[617, 449]
Form Position	[0, 0]
Form Size Policy	Generate Resize Code
Generate Position	✓
Generate Size	✓
Generate Center	
Designer Size	[601, 410]

Test that the empty program has been created correctly by clicking the green ***Run*** arrow on the row of icons at the top of the NetBeans screen:


camera - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Files
camera
Services
Projects
Start Page
camera.java
Source
Design
History

To select multiple components in an area hold Shift and drag mouse over the components.

A window will appear, asking you to confirm that you wish ***cameraPackage.camera*** to be the ***Main*** class.   Click the ***OK*** button to confirm this:



Project camera does not have a main class set.

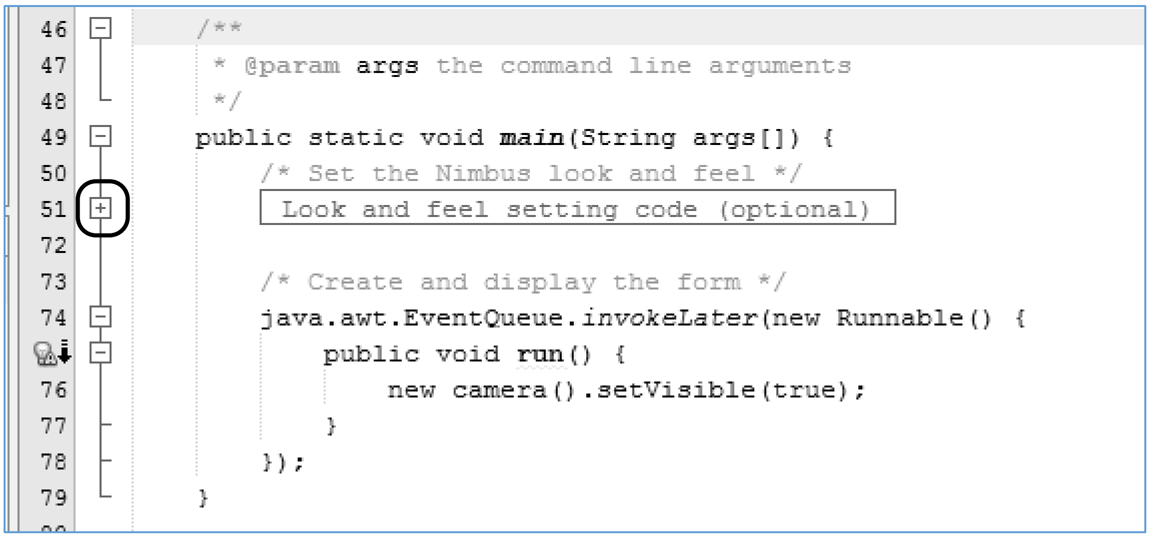
Select the main class:

cameraPackage.camera

An empty program window should now appear. However, you will see that this is not displayed in the normal colour scheme used by other Windows applications. We will now correct this.

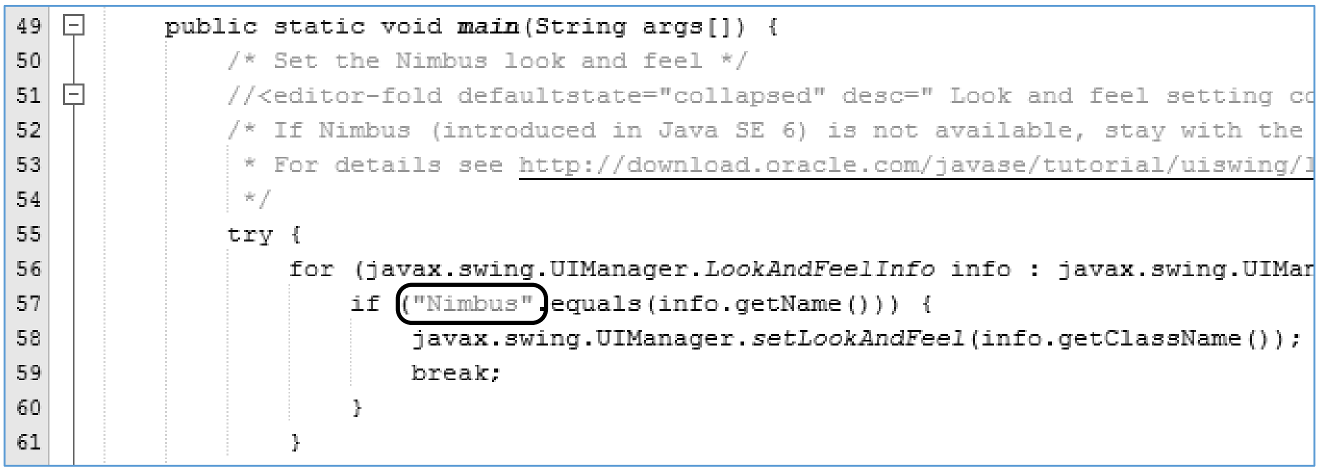
Close the program window and return to the NetBeans editing page. Click the ***Source*** tab at the top left of the design window to open the program code.

Look down through the program listing to find a section called ***main***. Inside this is a label saying '***Look and feel setting code***'. Click the ***+ icon*** to open this option:



```
    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
         Look and feel setting code (optional)

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new camera().setVisible(true);
            }
        });
    }
```

Look for the line of code containing the word "Nimbus", and replace this with "Windows".



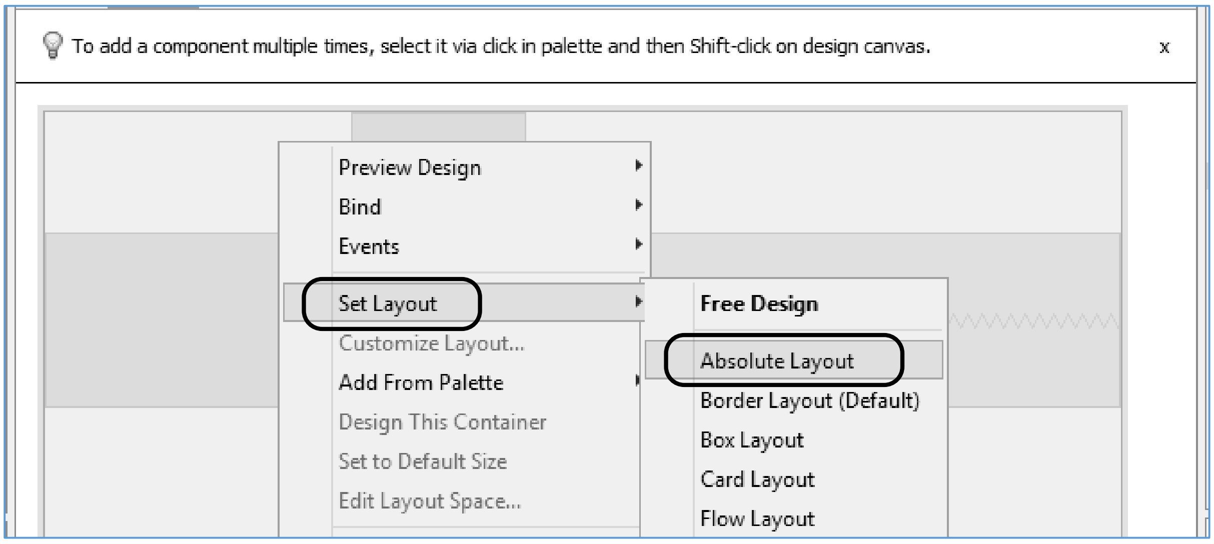
```
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting co
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the
         * For details see http://download.oracle.com/javase/tutorial/uiswing/l
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIMar
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
```

The line should now read:

if ("Windows".equals(info.getName()))

Run the program again, and the form should appear in the correct Windows colour scheme.

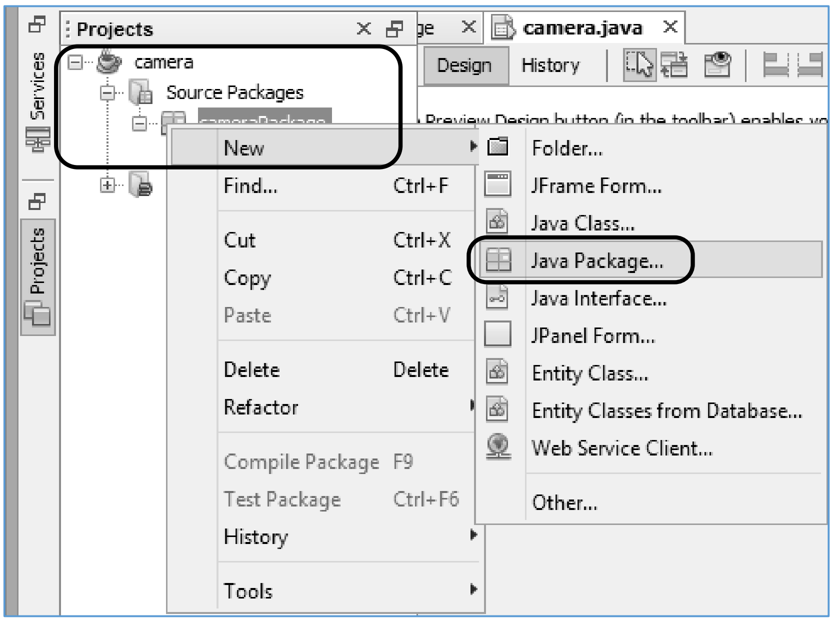
We have one final parameter to set before we begin to design the program. Close the program form and return to the NetBeans design page. Right-click on the form, then select ***Set Layout*** / ***Absolute Layout***. This will make it easy to drag and drop components into any position when creating the program screen.


To add a component multiple times, select it via click in palette and then Shift-click on design canvas.

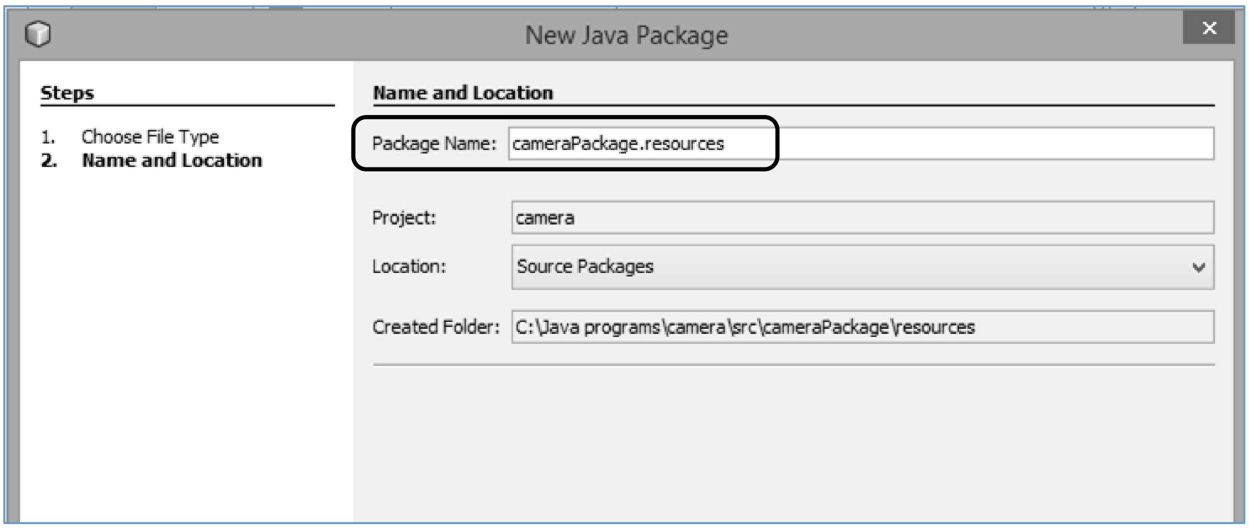
The steps which you carried out above are the standard sequence for setting up a new Windows graphical user interface project in NetBeans. We will follow this sequence each time we create a new program, just changing the project name as required.

We can now move on to create the actual program. This will be show a picture of a camera, with buttons to allow the picture to be visible or hidden.

We will begin by setting up another Package within the project which will provide a storage area for the picture. Use the ***+ icons*** to open the ***camera*** project folders until you reach ***cameraPackage***. Right-click on ***cameraPackage***, and select ***New*** / ***Java Package***:

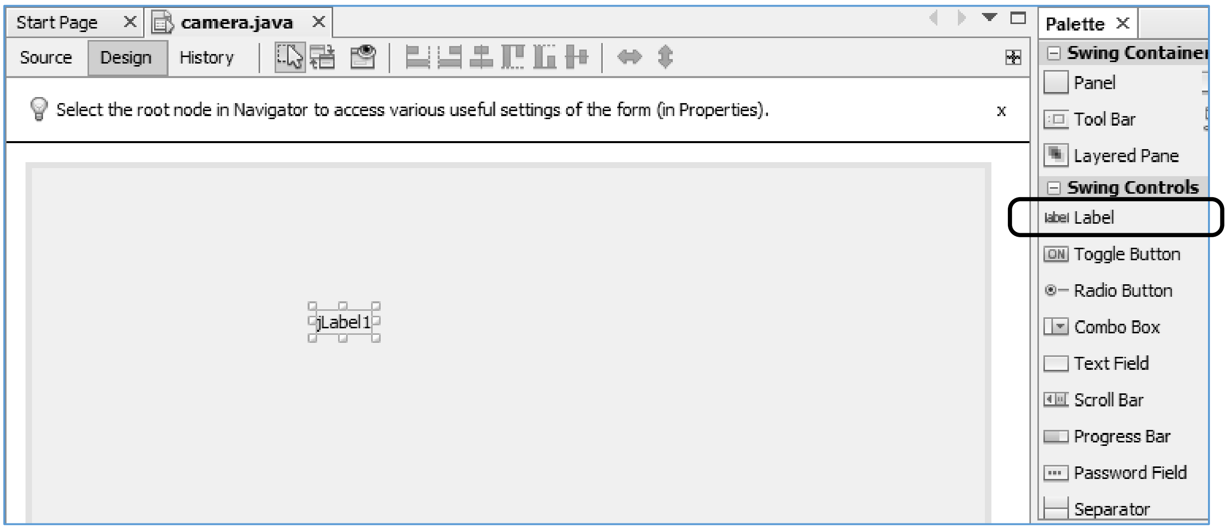


Whenever we use images in a project, we will create a package called ***resources*** into which they can be stored. Set the ***Package Name*** to be ***cameraPackage.resources*** and click the ***Finish*** button.



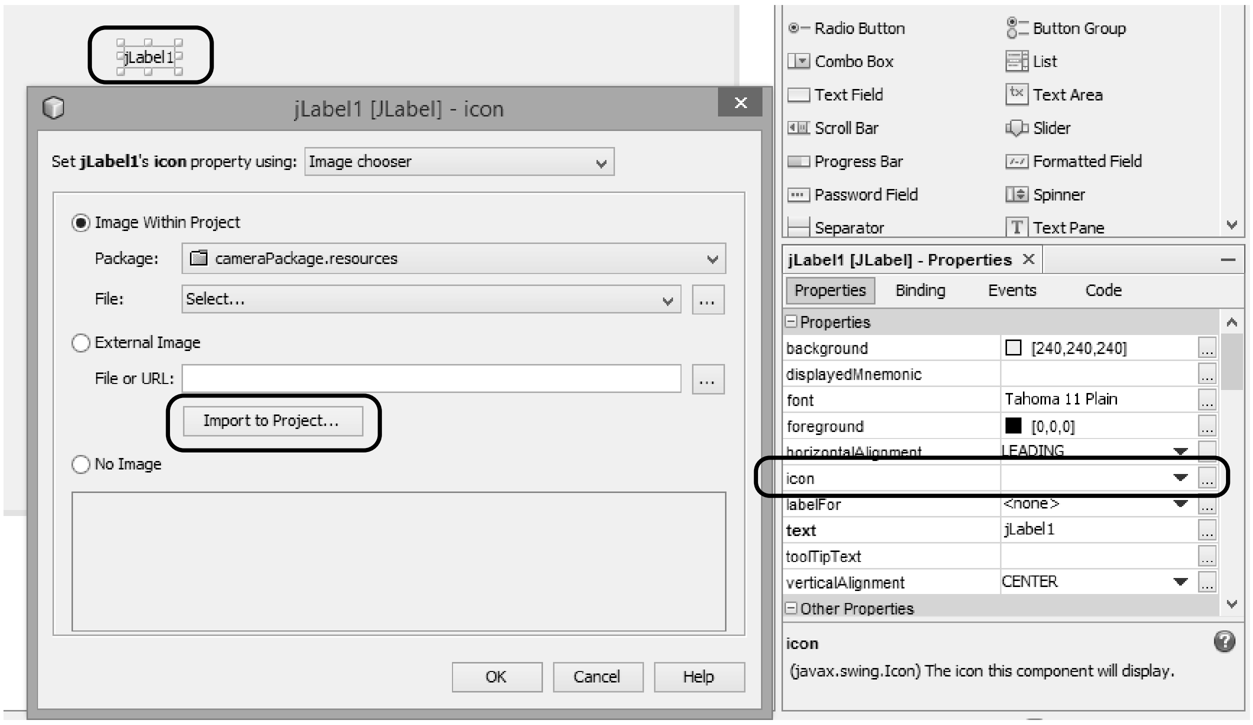
Package Name: cameraPackage.resources
Project: camera
Location: Source Packages
Created Folder: C:\Java programs\camera\src\cameraPackage\resources

Return to the NetBeans design screen. Locate the ***Label*** component in the ***Swing Controls*** section of the ***Palette*** on the right of the screen. Drag and drop a label onto the form:



Before continuing, obtain a JPG image of a camera. Use a graphics application such as Paintshop or Paint to adjust the width of the image to approximately 250 pixels.

Click to select the ***label*** component on the form. Select the ***Properties*** tab above the window at the bottom right of the screen, then locate the ***icon*** property. Click the ***'…'*** (ellipsis) button. A window opens, which will allow us to upload the camera picture to the project:



Set **jLabel1**'s **icon** property using: Image chooser

Package: cameraPackage.resources

File: Select...

File or URL:

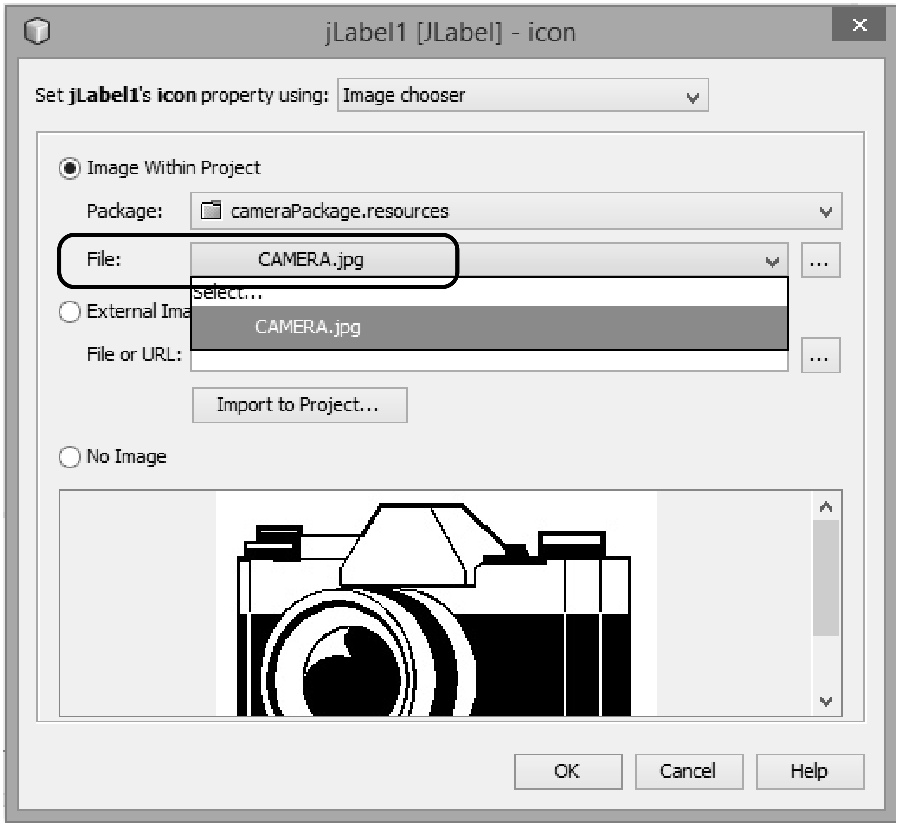
background	[240,240,240]
displayedMnemonic	
font	Tahoma 11 Plain
foreground	[0,0,0]
horizontalAlignment	LEADING
icon	
labelFor	<none>
text	jLabel1
toolTipText	
verticalAlignment	CENTER

Other Properties

icon

(javax.swing.Icon) The icon this component will display.

Click the ***Import to Project*** button, then use the file dialog to locate the picture. Click ***Next*** and select ***resources*** as the target folder, then click ***Finish***. After uploading, the file name should be displayed in the ***File*** text box:



Set **jLabel1**'s **icon** property using: Image chooser

Package: cameraPackage.resources

File: CAMERA.jpg

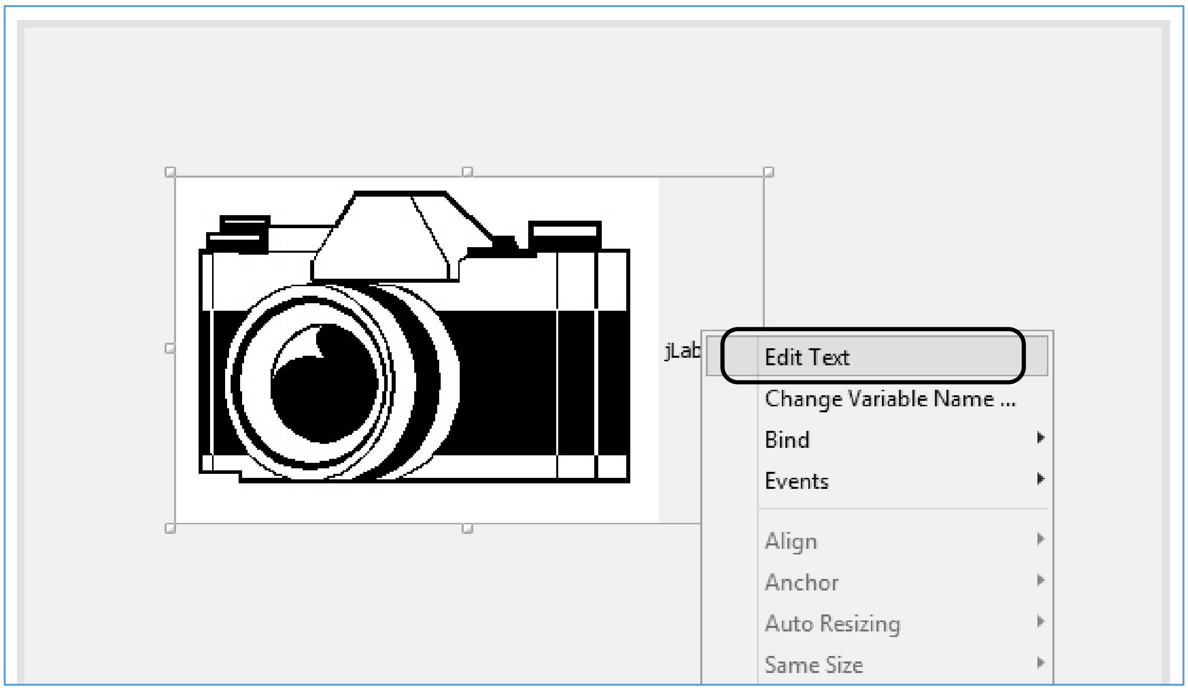
Select...

CAMERA.jpg

File or URL:

Click the ***OK*** button to return to the NetBeans design screen. The camera image should now appear, with label text shown alongside. You may need to drag the image completely onto the form.

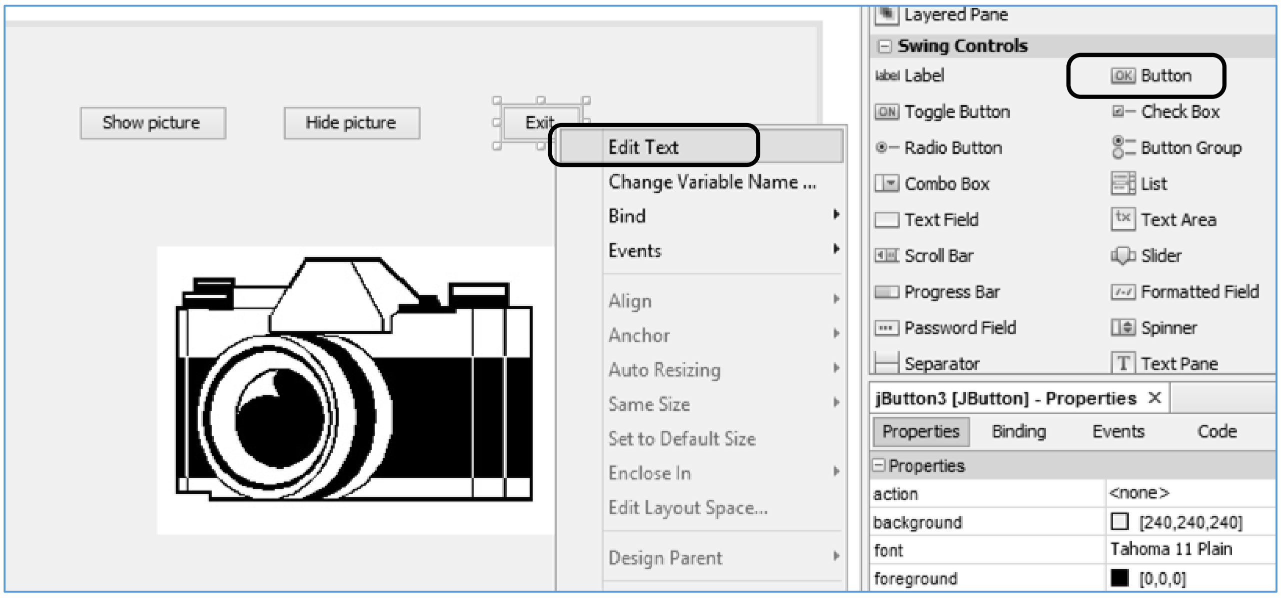
Right-click on the image to open a drop-down menu. Select ***Edit Text***, then delete the label caption.



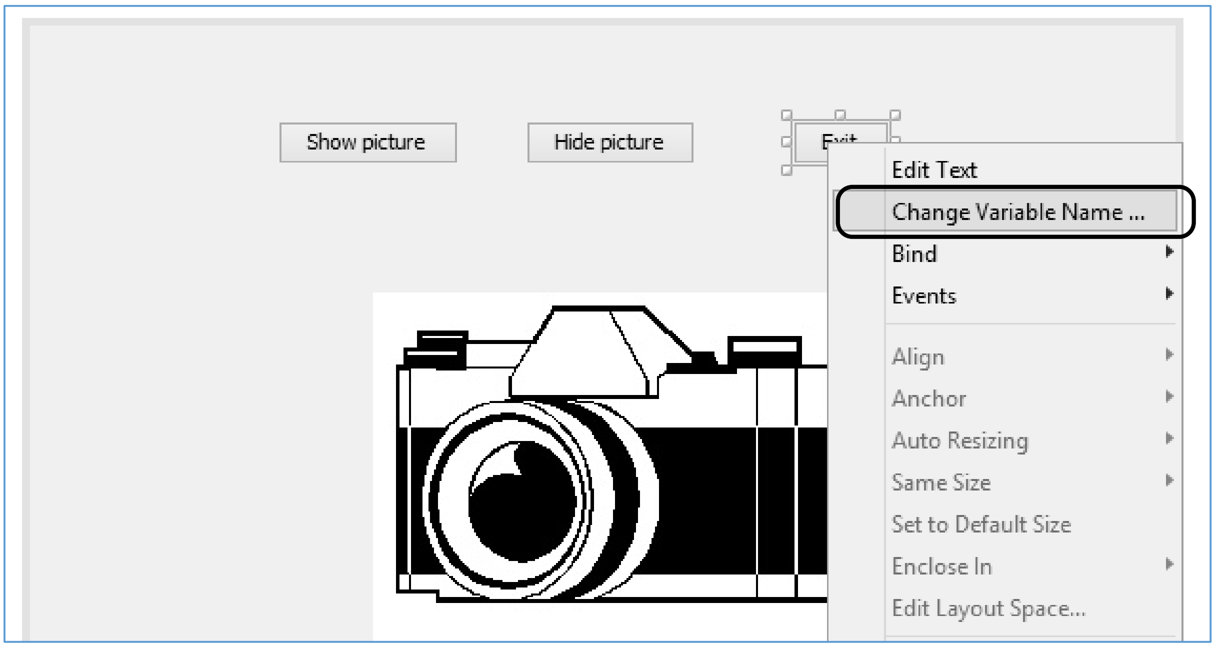
We will now add a set of buttons. Select the Button component from the ***Swing Controls*** section of the Palette, and drag and drop three buttons onto the form and arrange them above the picture.

Right-click on each of the buttons in turn. Select the ***Edit Text*** option from the drop-down menu, and give the buttons the captions:

Show picture
Hide picture
Exit



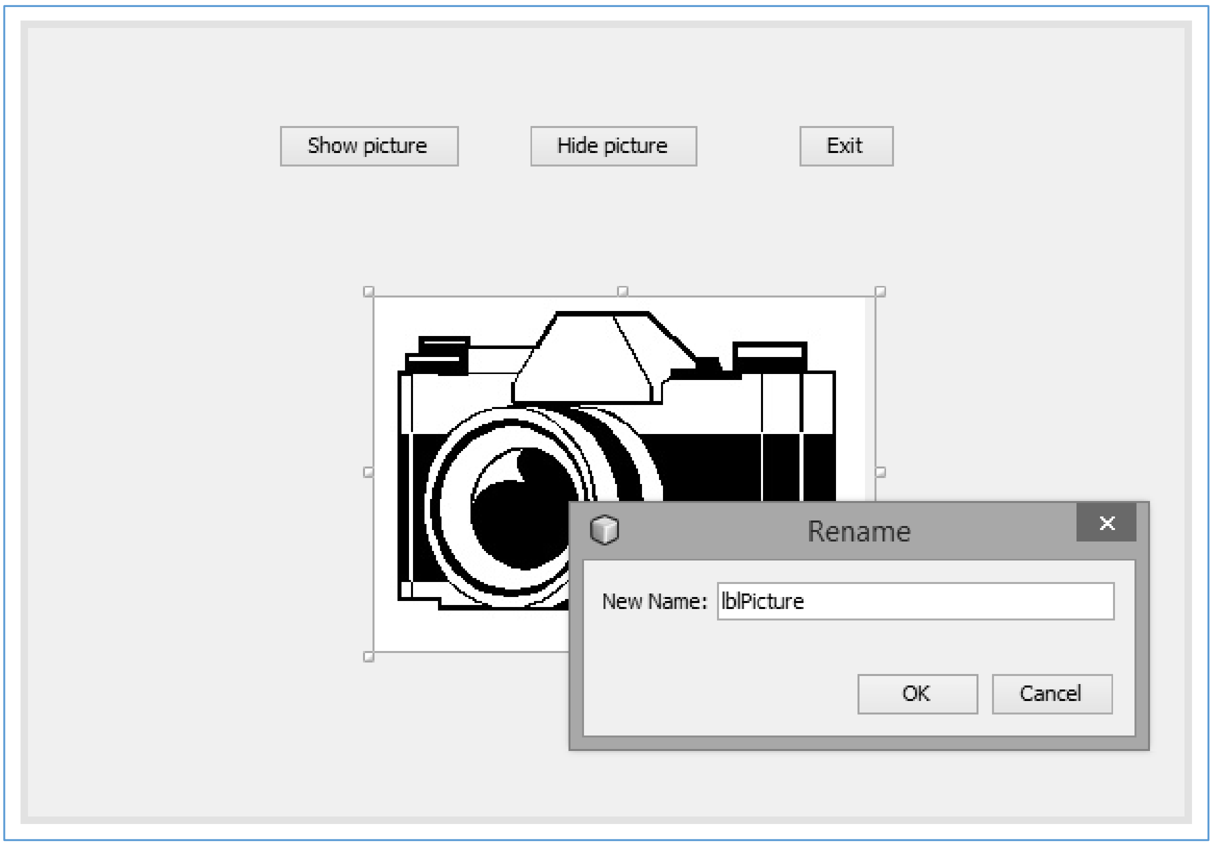
action	<none>
background	[240,240,240]
font	Tahoma 11 Plain
foreground	[0,0,0]



Right-click again on each of the buttons, and this time select ***Change Variable Name.*** We will give the buttons names which clearly identify their purpose in the program. This will make the program code easier to understand, with less chances of making mistakes. Give the buttons the names:

btnShow
btnHide
btnExit

Right-click on the camera picture and select ***Change Variable Name***. Give the name:
lblPicture



New Name: lblPicture

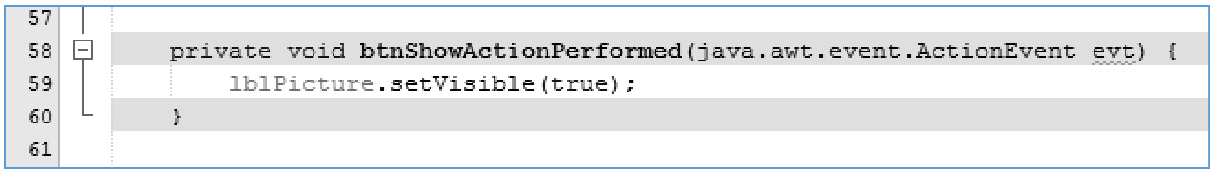
We can now add program code to make the picture appear and disappear.

Double-click the ***Show picture*** button. The program code window will open, and you will notice that an empty ***method*** has been created. This contains a comment in pale grey:

```
// TO DO add your handling code here:
```

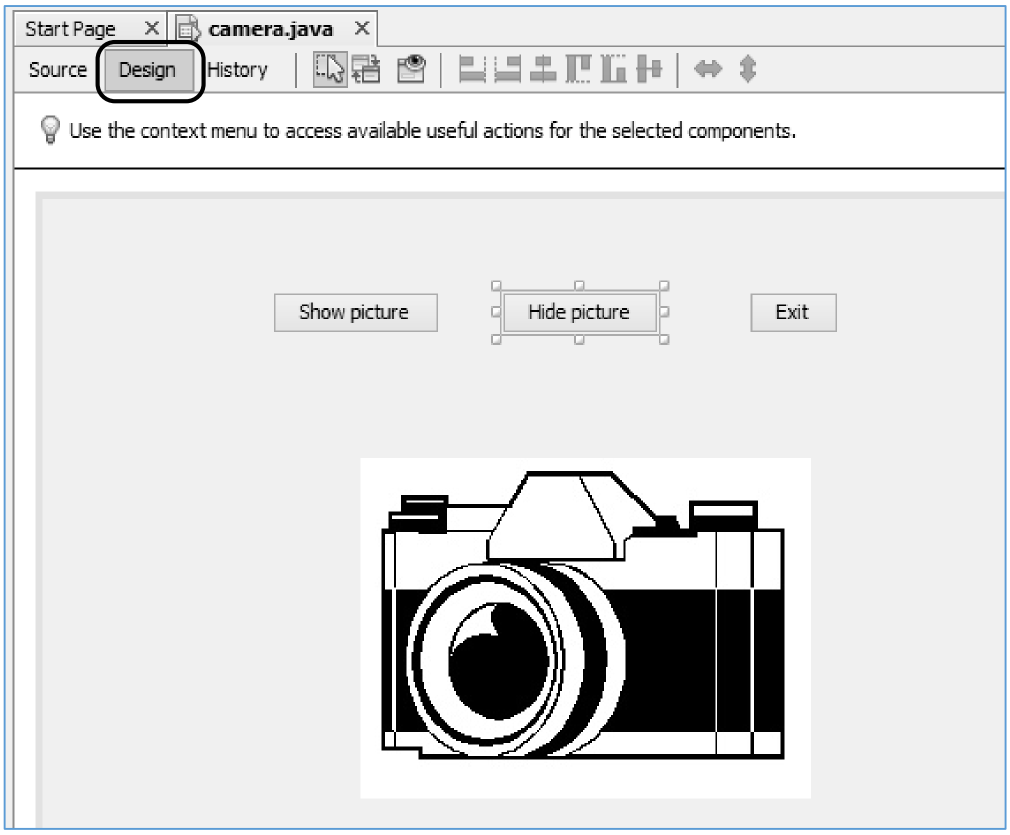
This comment line can be deleted and replaced by your own program code. Any commands which you write inside this method will be carried out when the program is running and the button is clicked. Add the line of code:

```
lblPicture.setVisible(true);
```



```
    private void btnShowActionPerformed(java.awt.event.ActionEvent evt) {
        lblPicture.setVisible(true);
    }
```

Click the Design tab to return to the program form, then double click the ***Hide*** picture button:



Use the context menu to access available useful actions for the selected components.

An empty method will again be created. This time, add the line of program code:

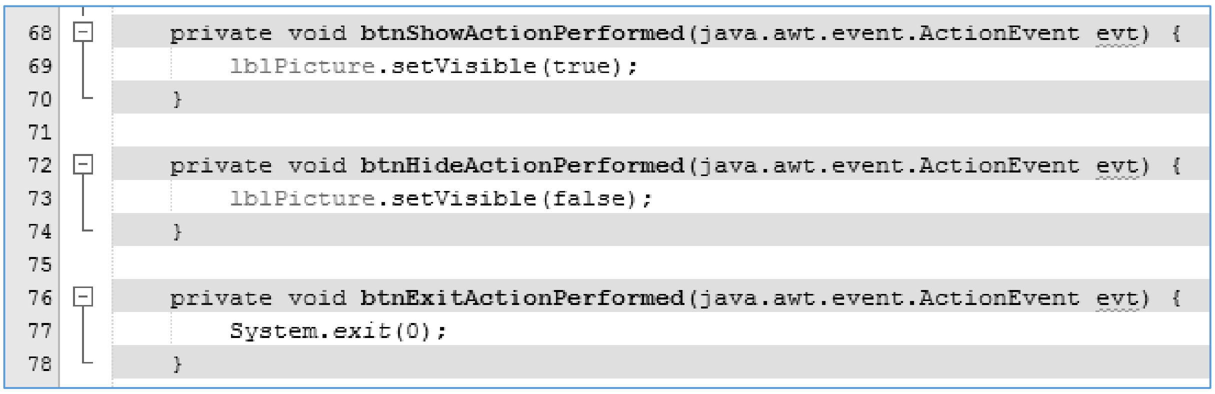
```
lblPicture.setVisible(false);
```

The two buttons will be able to change the ***visible*** property of the picture while the program is running.

Finally, double click the ***Exit*** button and add a line of code to close the program:

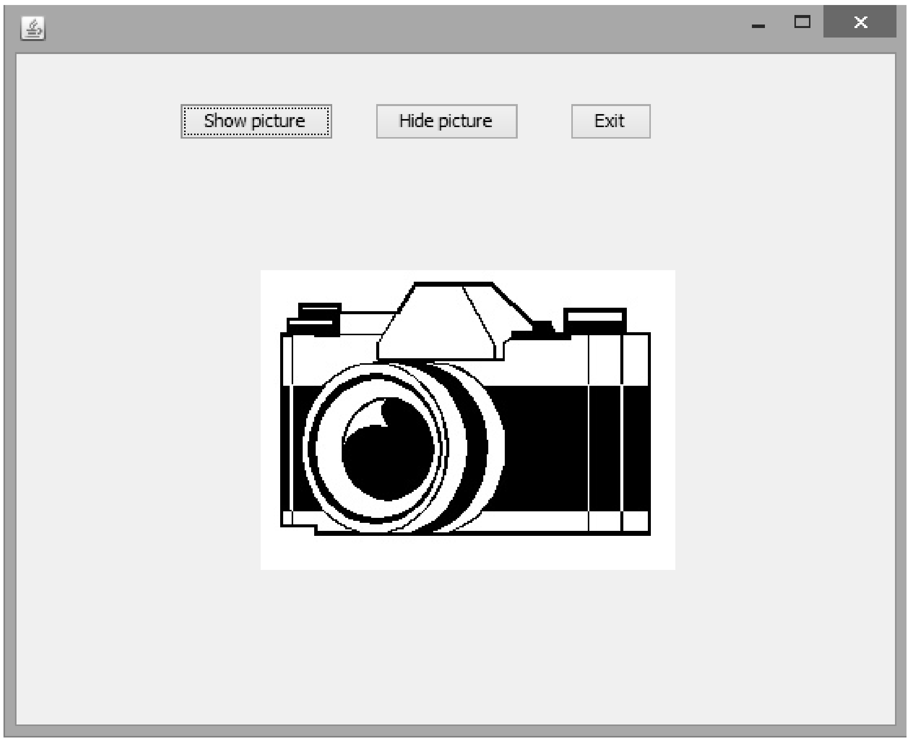
```
System.exit(0);
```

The program code should now contain the three completed button-click methods:



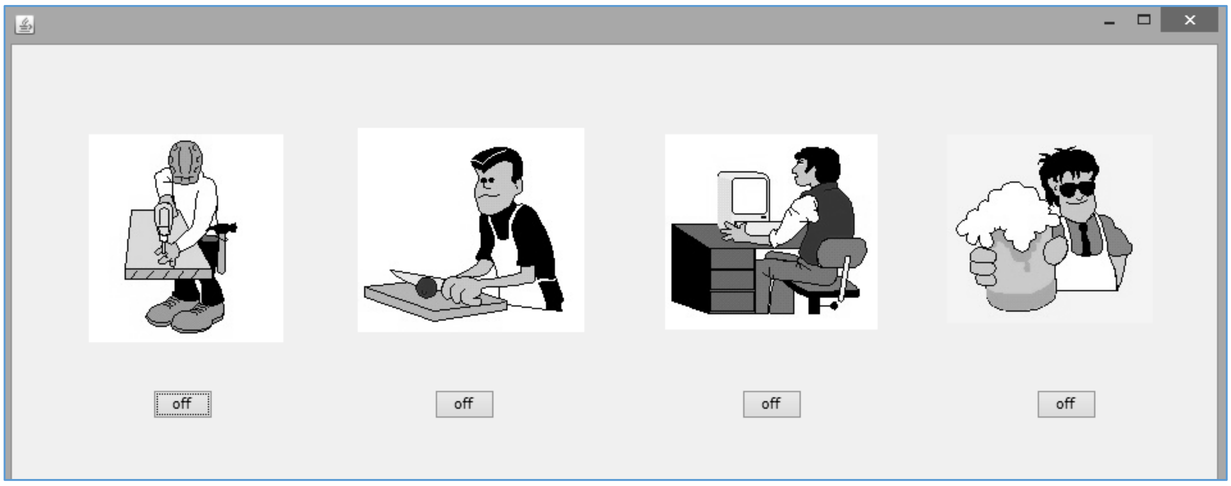
```
    private void btnShowActionPerformed(java.awt.event.ActionEvent evt) {
        lblPicture.setVisible(true);
    }

    private void btnHideActionPerformed(java.awt.event.ActionEvent evt) {
        lblPicture.setVisible(false);
    }

    private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }
```

Run the program by clicking the green ***Run*** arrow on the row of icons at the top of the NetBeans screen. The camera picture can be hidden or displayed by means of the buttons.

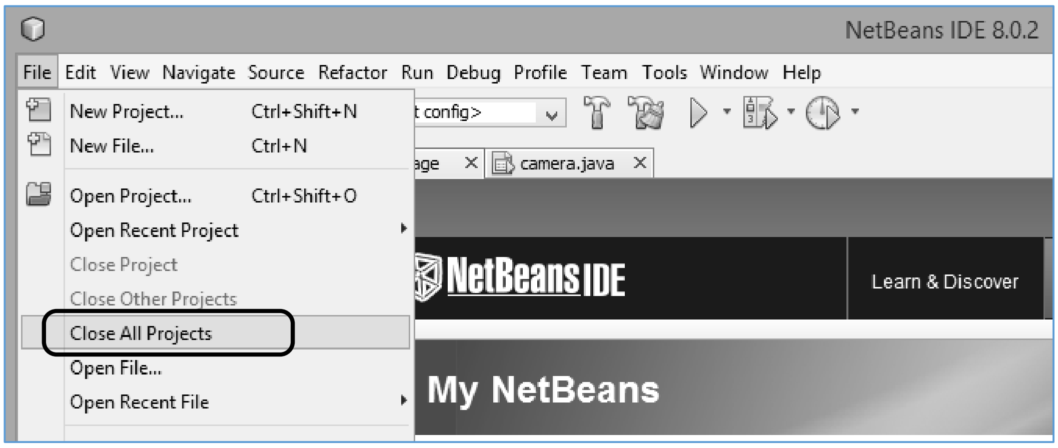


Check that the ***Exit*** button closes the program window correctly.

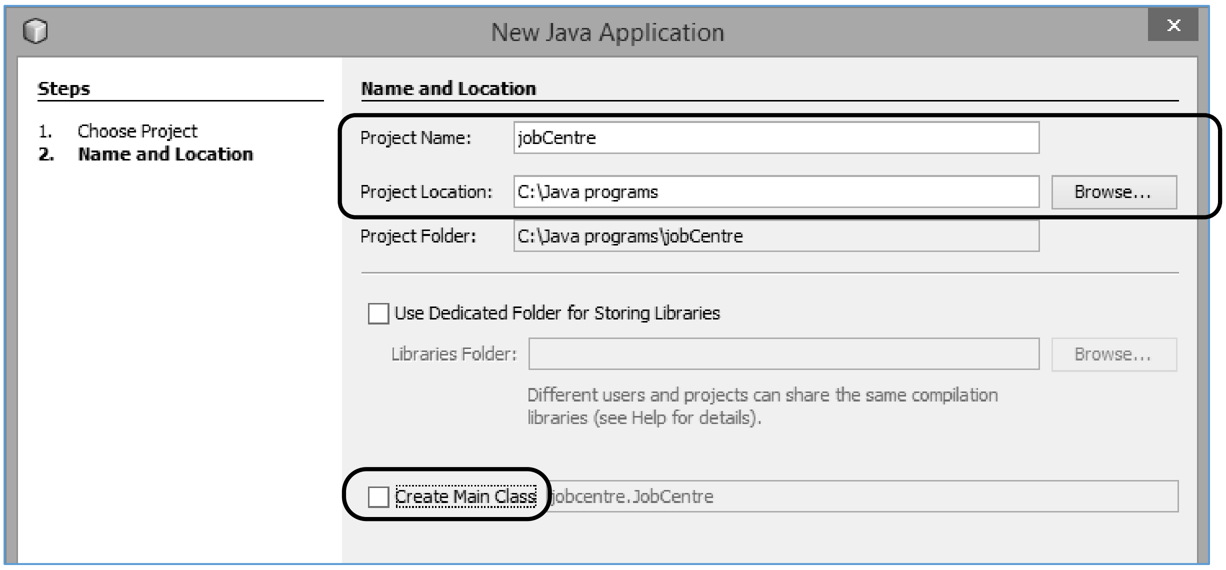
For the next program, we will create a screen for display in a Job Centre which will illustrate the types of work currently available. The program will provide buttons to allow the staff to switch pictures on or off, according to the current job vacancies.



Select the NetBeans ***File*** menu and click the option to ***Close All Projects***:



From the ***File*** menu, select ***New Project***, and check that ***Java*** / ***Java Application*** is highlighted. Give the ***Project Name*** as ***JobCentre***, and un-tick the ***Create Main Class*** option:



New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name: jobCentre

Project Location: C:\Java programs

Project Folder: C:\Java programs\jobCentre

☐ Use Dedicated Folder for Storing Libraries

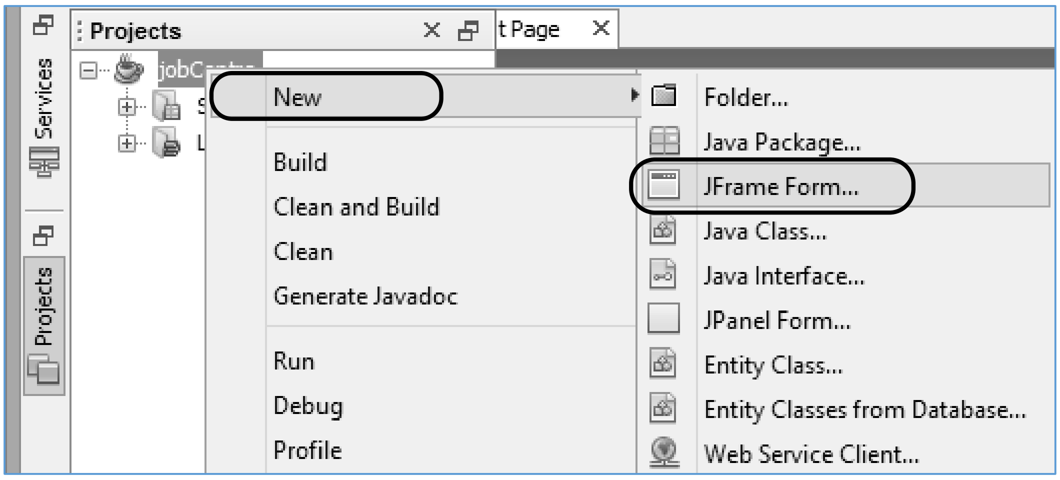
Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

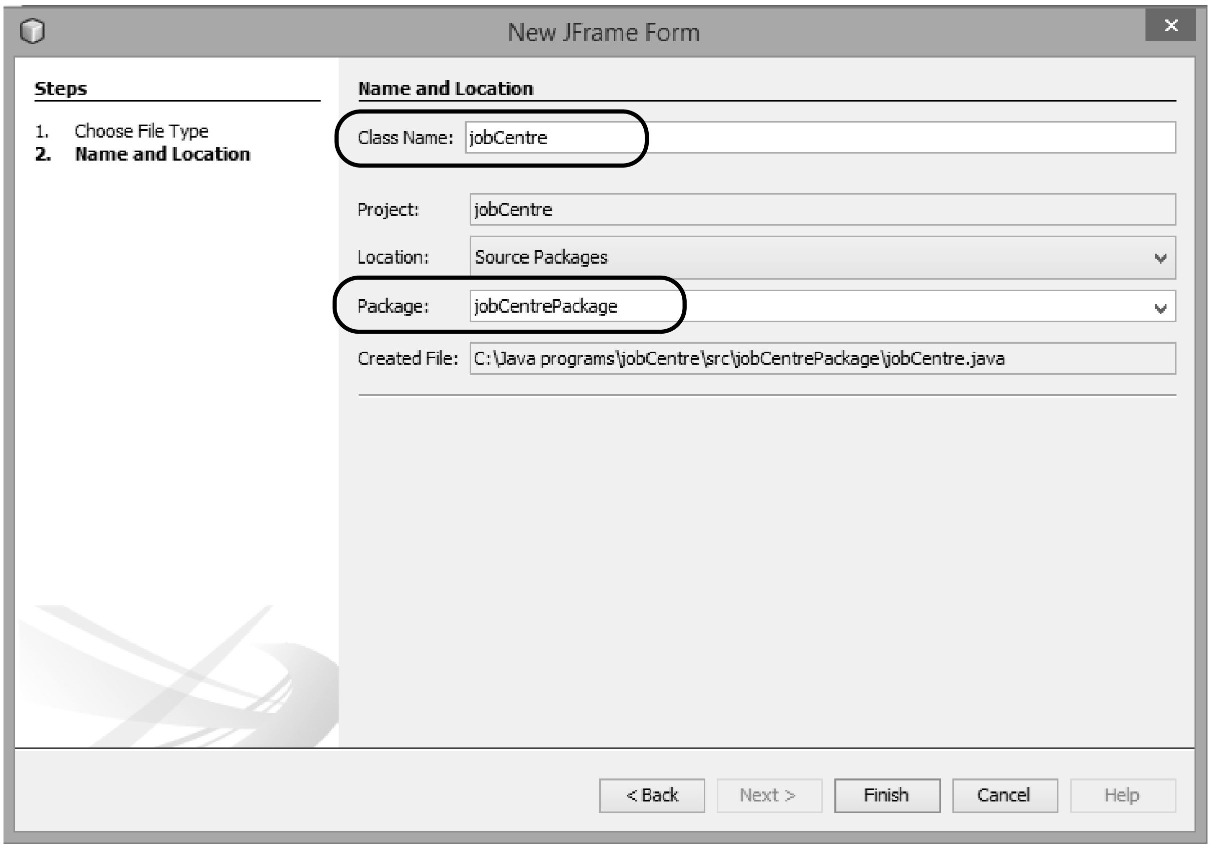
☐ Create Main Class jobcentre.JobCentre

Click ***Finish*** to return to the main NetBeans editor.

Right click the ***jobCentre*** project icon, then select ***New*** / ***JFrame form*** from the drop-down menus:



Enter ***jobcentre*** as the ***Class Name***, and ***jobCentrePackage*** as the ***Package***:


New JFrame Form

Steps
1. Choose File Type
2. Name and Location

Name and Location

Class Name: jobCentre

Project: jobCentre

Location: Source Packages

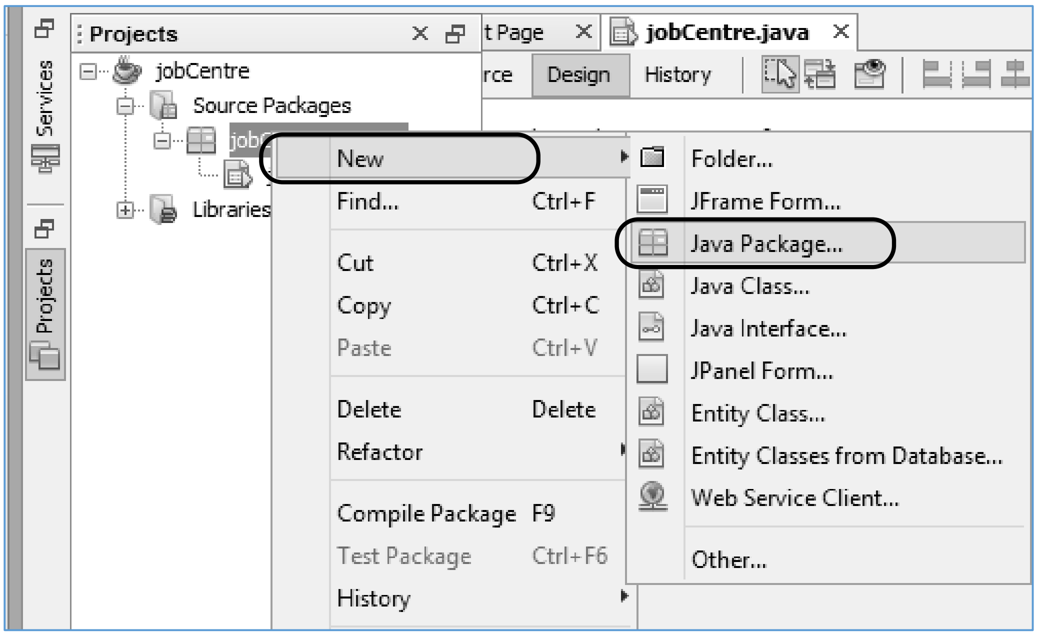
Package: jobCentrePackage

Created File: C:\Java programs\jobCentre\src\jobCentrePackage\jobCentre.java

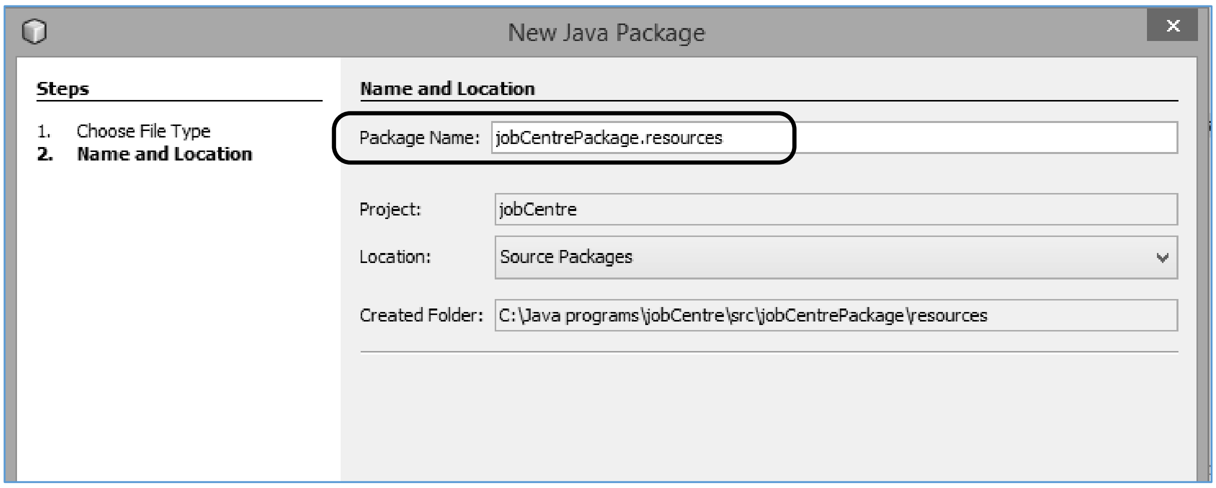
Click the ***Finish*** button to return to the editor screen.

This program will again make use of picture images, so we will create a resources folder into which these can be stored.

Use the + icons to open the jobcentre project folders, until ***jobCentrePackage*** is reached. Right-click ***jobCentrePackage*** and select ***New*** / ***Java Package*** from the drop-down menus:



Give the package name as ***jobCentrePackage.resources***



Package Name: jobCentrePackage.resources

Project: jobCentre

Location: Source Packages

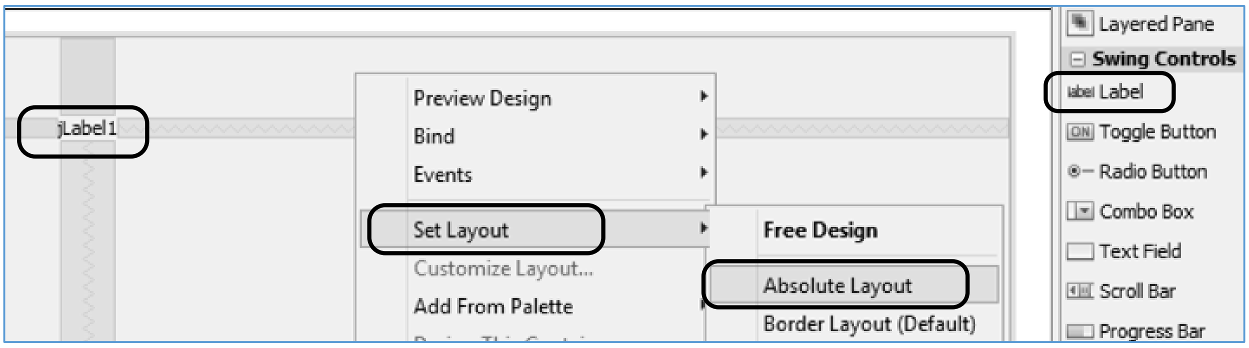
Created Folder: C:\Java programs\jobCentre\src\jobCentrePackage\resources

Click the ***Finish*** button to return to the editing screen.

A blank ***form*** should be displayed, ready to begin the program design.

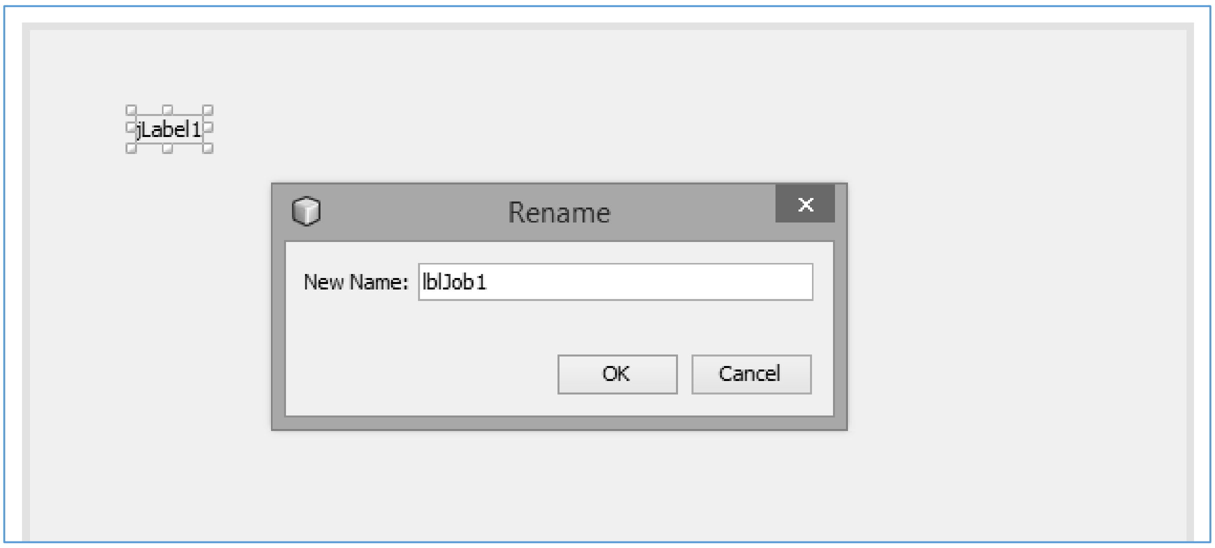
Drag a ***label*** onto the form. You will see grey bars attached to the label component. These are part of a system for aligning components to create input forms, and will not be required in our program. We will turn these off now.

Right-click on the ***form***, then select ***Set Layout*** / ***Absolute Layout*** from the drop-down menus.



Right-click on the ***label***, then use the Change Variable Name option to reset the label name to:

lblJob1



New Name: lblJob1

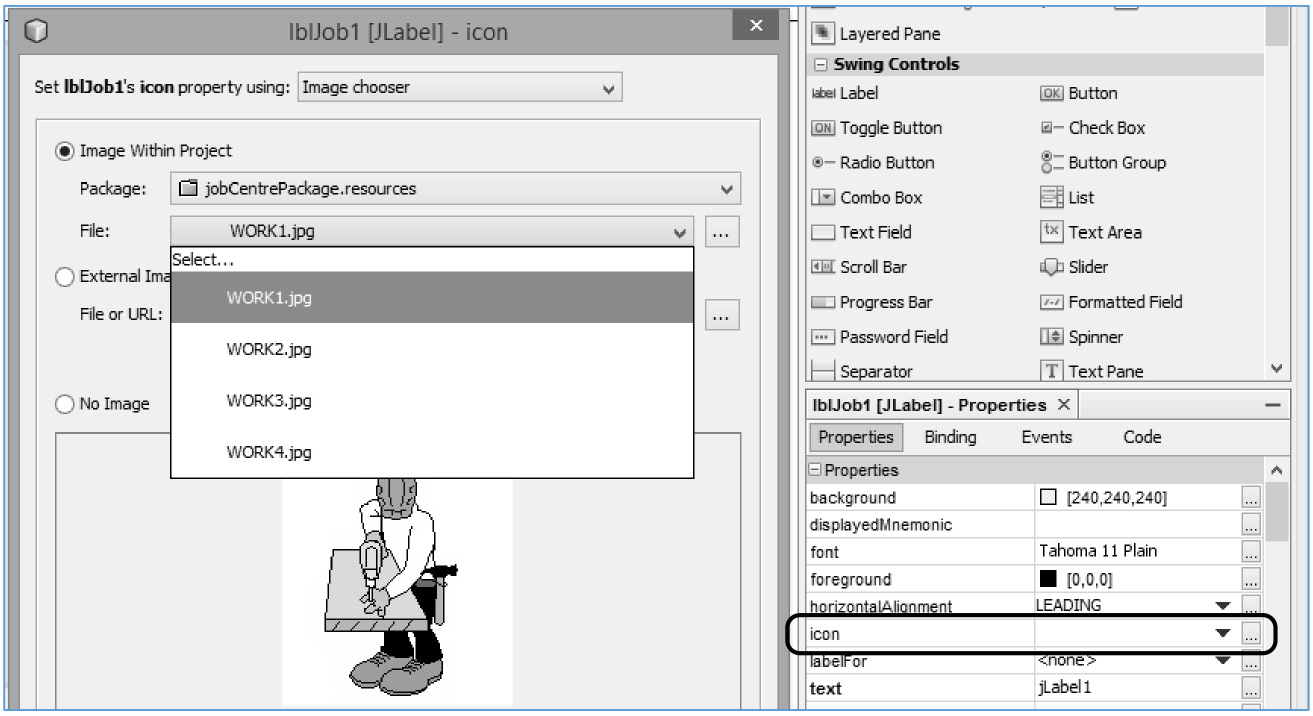
Before continuing with the program, obtain JPG images representing four different jobs, for example:

Carpentry
Catering
Office work
Bar work

Use a graphics editor such as Photoshop or Paint to adjust the width of each image to approximately 150 pixels.

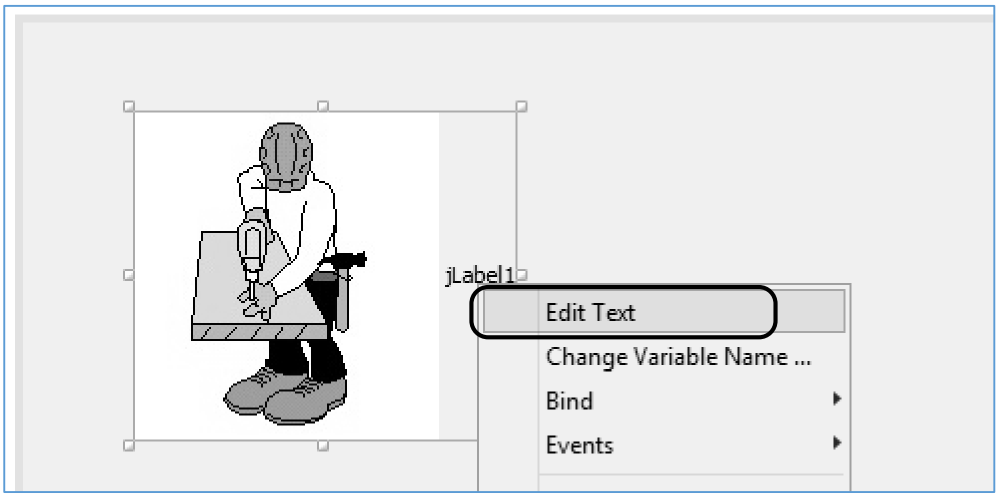
Select the ***label*** which you placed on the form, and go to the ***Properties*** window at the bottom right of the screen. Find the ***icon*** property, and click the ***"..."*** ellipsis symbol at the end of the line:

Use the ***Import to Project*** option to upload the four images to the ***resources*** folder.



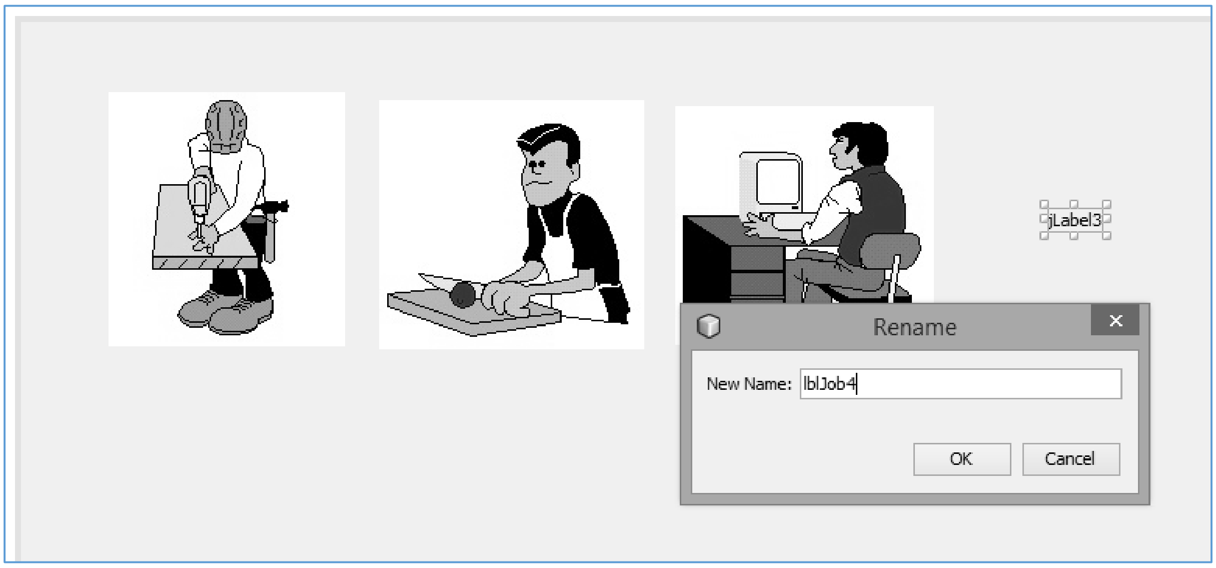
Select one of the images as the ***File*** to be displayed, then click the '***OK***' button to return to the NetBeans editing screen.

Right-click on the image and use the ***Edit Text*** option to delete the label text.



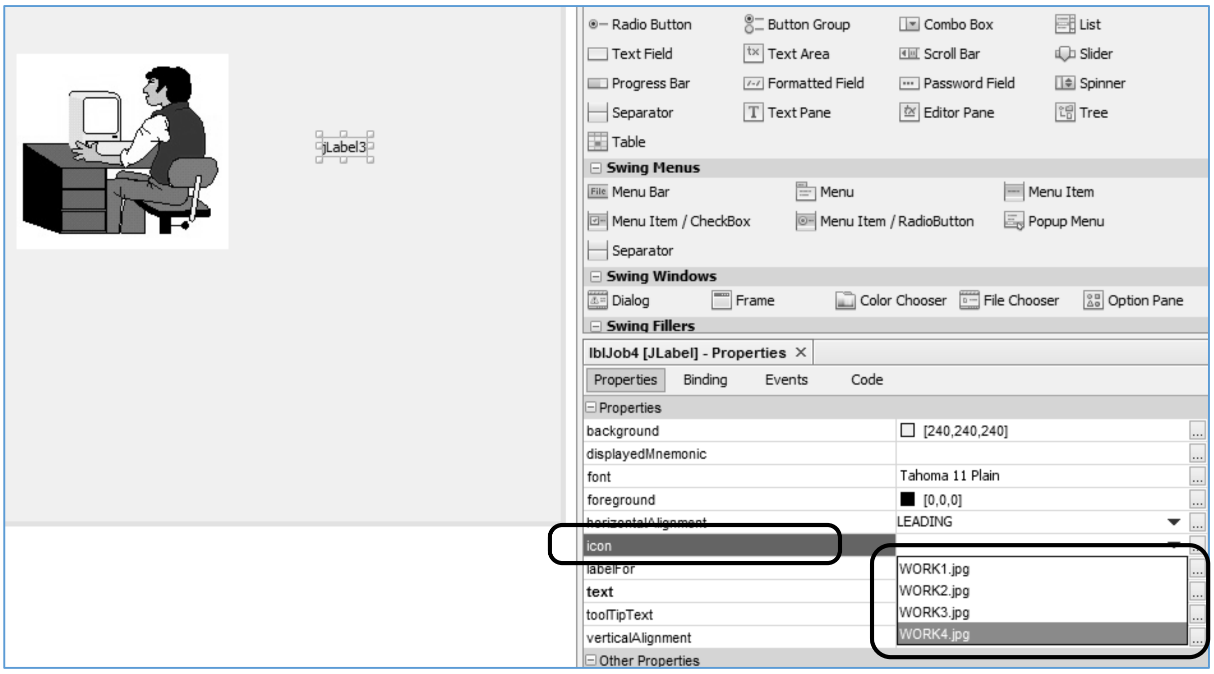
Add three more labels to the form, dragging the form wider if necessary. Change the names of these labels to:

lblJob2
lblJob3
lblJob4



New Name: lblJob4

Set the ***icon*** property of each label to display a picture image. Notice that the JPG images which you uploaded earlier are now available from a drop-down list on the icon property line:

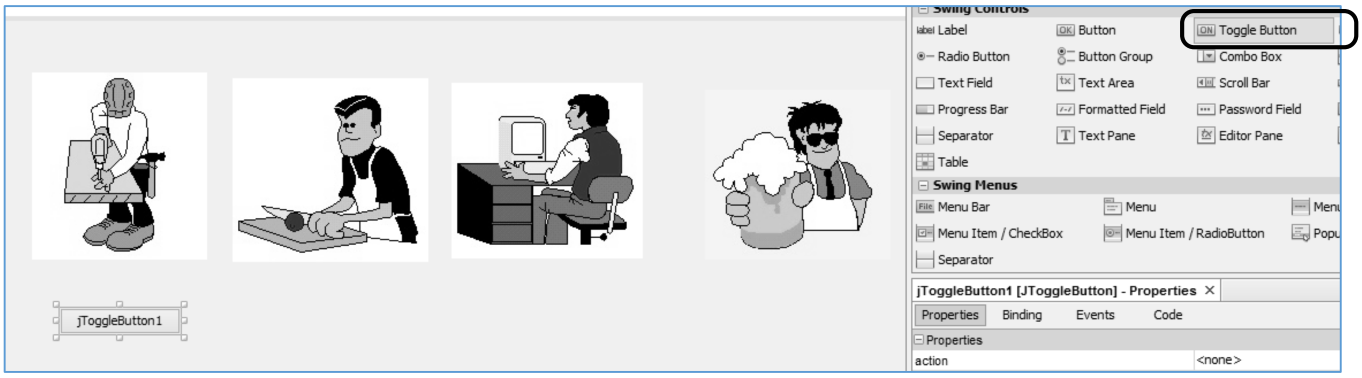


background	[240,240,240]
displayedMnemonic	
font	Tahoma 11 Plain
foreground	[0,0,0]
horizontalAlignment	LEADING
icon	
labelFor	WORK1.jpg
text	WORK2.jpg
toolTipText	WORK3.jpg
verticalAlignment	WORK4.jpg

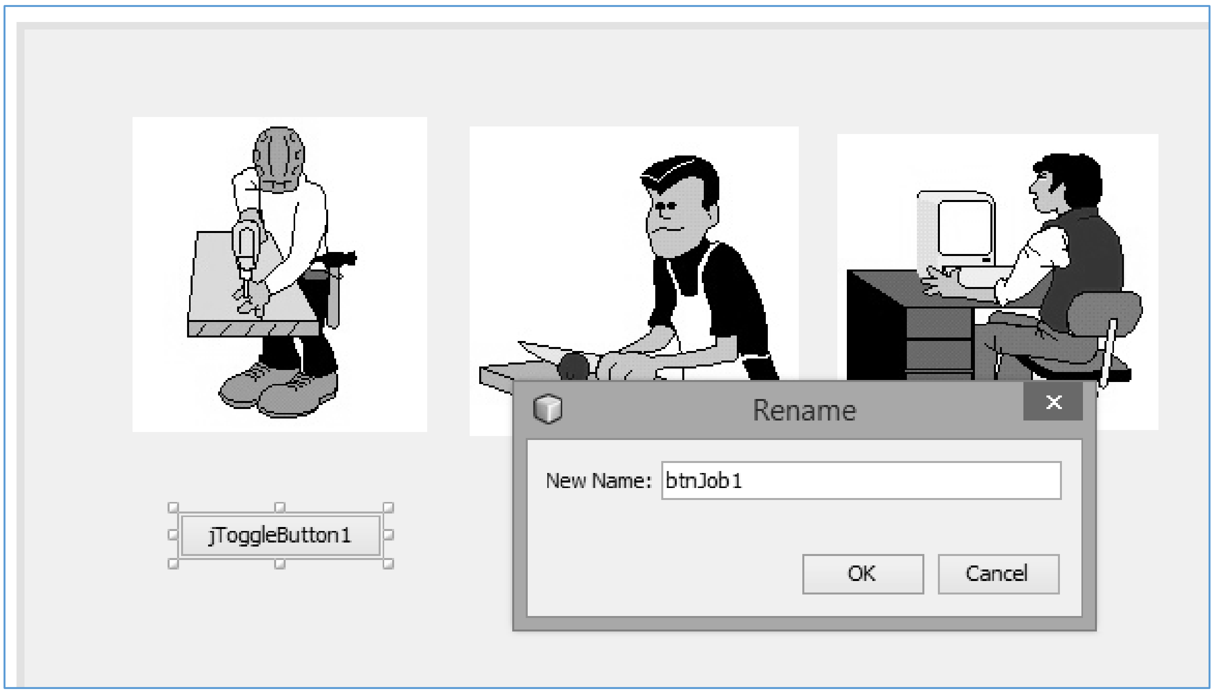
Right-click on each of the images and use the ***Edit Text*** option to delete the label text.

The final stage of the program is to provide buttons to show or hide the picture images. Rather than use two separate buttons for each picture, as we did with the camera program earlier, it is neater to use ***Toggle Buttons***. A toggle button has two states – ***selected*** and ***unselected***. It changes between states each time it is clicked.

Drag a ***Toggle Button*** from the ***Palette*** and position it below the first image:



Change the name of the toggle button to:
btnJob1



New Name: btnJob1

In order to use the toggle button, the program will need an additional code module. Use the ***Source*** tab above the design form to change to the program code page. Add a line of code:

```
import javax.swing.JToggleButton;
```

just after the start of the program, in the position shown below:

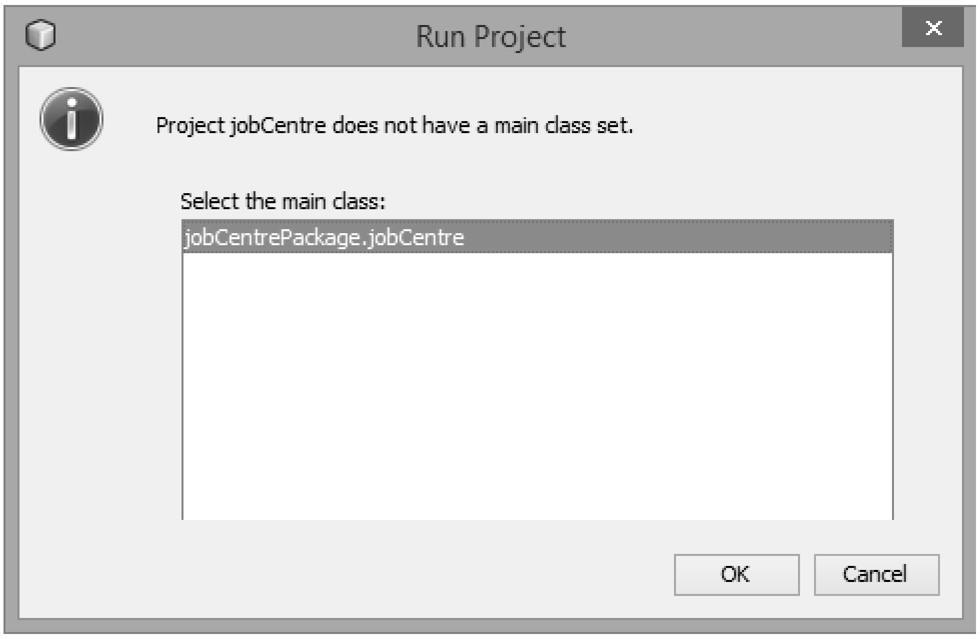
```
package jobCentrePackage;

import javax.swing.JToggleButton;

public class jobCentre extends javax.swing.JFrame {

    public jobCentre() {
        initComponents();
    }
```

Click the ***Design*** tab to return to the design view, then double click the toggle button which you placed on the form. The program will change to the code page and you will see that an empty button click ***method*** has been created. Add the lines of program code shown below:

```
private void btnJob1ActionPerformed(java.awt.event.ActionEvent evt) {
        JToggleButton btn = (JToggleButton) evt.getSource();
        if (btn.isSelected())
        {
            lblJob1.setVisible(true);
            btnJob1.setText("off");
        }
        else
        {
            lblJob1.setVisible(false);
            btnJob1.setText("on");
        }
    }
```

The purpose of this code is to check whether the toggle button is currently in its ***selected*** or ***unselected*** state. Depending on the result, it will display or hide the picture image by changing the ***visible*** property of the label component. The caption on the toggle button will also be changed to either "***on***" or "***off***".

Click the green Run arrow. Click OK to confirm that you want ***jobCentrePackage.jobCentre*** to provide the main class for the project:


Run Project

Project jobCentre does not have a main class set.

Select the main class:

jobCentrePackage.jobCentre

The program should display the images, but the normal Windows colour scheme is not used and the form is not set to the correct window size. We will correct these problems now.

Close the program window to return to the NetBeans editing screen. Click the ***Source*** tab to select the program code, then locate the ***main*** method.

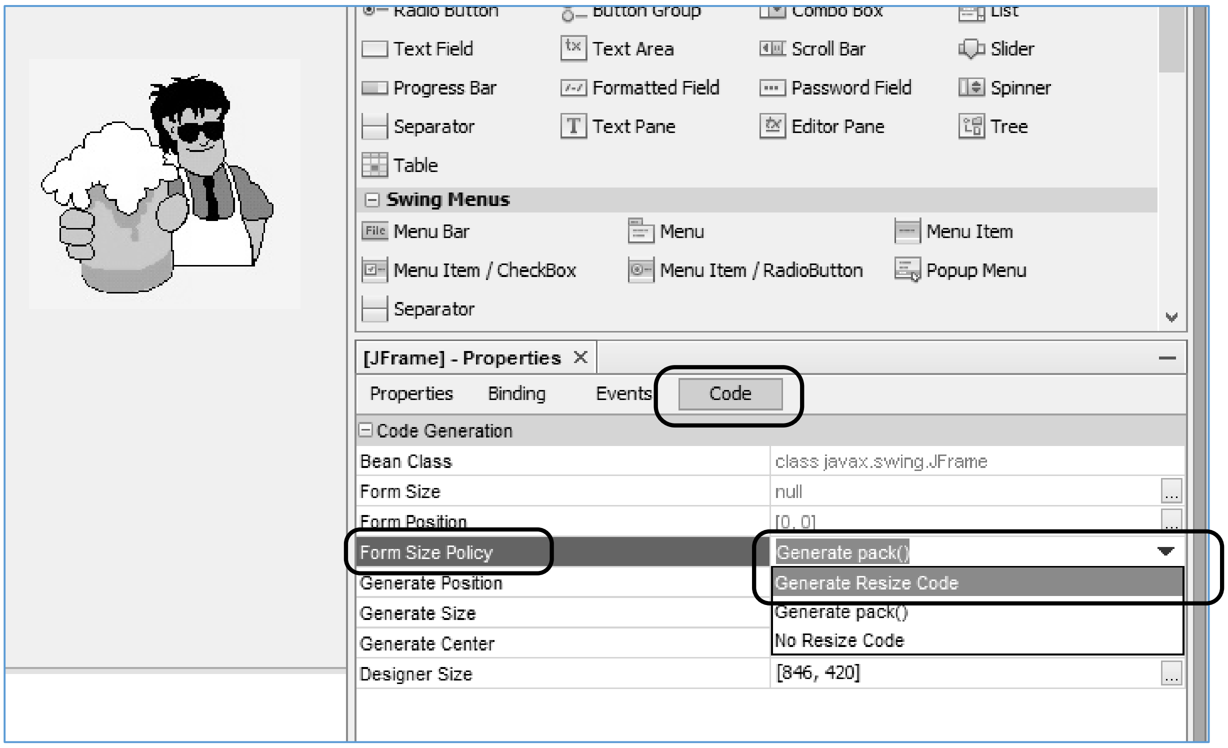
Open the method code by clicking the ***+ icon***, then change the word "***Nimbus***" to "***Windows***" on the line shown below:

```
public static void main(String args[]) {

     try {
         for (javax.swing.UIManager.LookAndFeelInfo info :
           javax.swing.UIManager.getInstalledLookAndFeels()) {

             if ("Windows".equals(info.getName())) {

                 javax.swing.UIManager.setLookAndFeel(info.getClassName());
                 break;
             }
         }
     } catch (ClassNotFoundException ex) {
```

Return to the ***Design*** view and click on an empty area of the ***form*** to select this component.

Click the ***Code*** tab in the ***Properties*** window, then go to the ***Form Size Policy*** line.

Click the ***Generate pack()*** option, and select ***Generate Resize Code*** from the drop-down list:



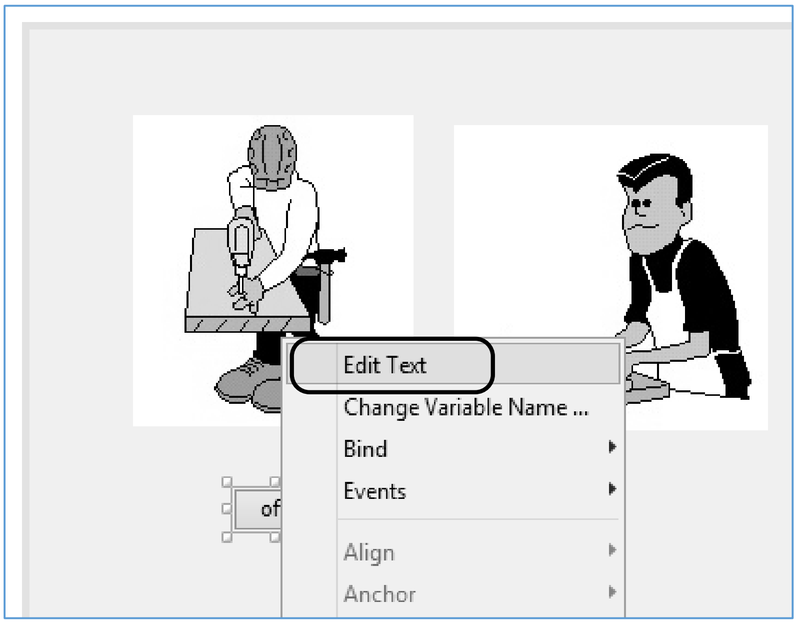
[JFrame] - Properties

Properties | Binding | Events | Code

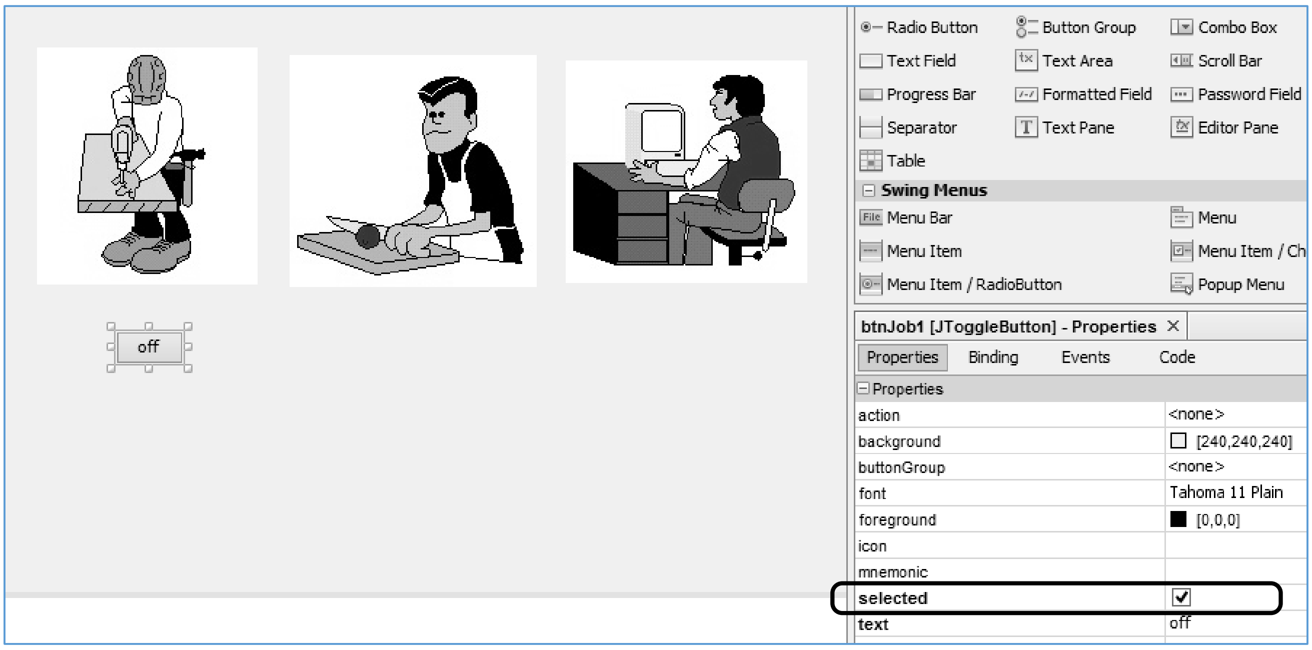
Code Generation

Bean Class	class javax.swing.JFrame
Form Size	null
Form Position	[0, 0]
Form Size Policy	Generate pack()
Generate Position	Generate Resize Code
Generate Size	Generate pack()
Generate Center	No Resize Code
Designer Size	[846, 420]

Return to the ***toggle button*** and right click to open the drop-down list. Use the Edit Text option to set the button caption to the word '***off***' :



Go to the ***Properties*** window and find the ***selected*** property for the ***toggle button***. Put a tick in the tick-box column:



btnJob1 [JToggleButton] - Properties

Properties

action	<none>
background	[240,240,240]
buttonGroup	<none>
font	Tahoma 11 Plain
foreground	[0,0,0]
icon	
mnemonic	
selected	✓
text	off

Run the program again. The window should now be the correct size, and displayed with the correct colour scheme. The toggle button will show the caption '***off***'.

Check that the picture image can be hidden or displayed by repeatedly clicking the toggle button, and that the button caption changes correctly with each mouse click.

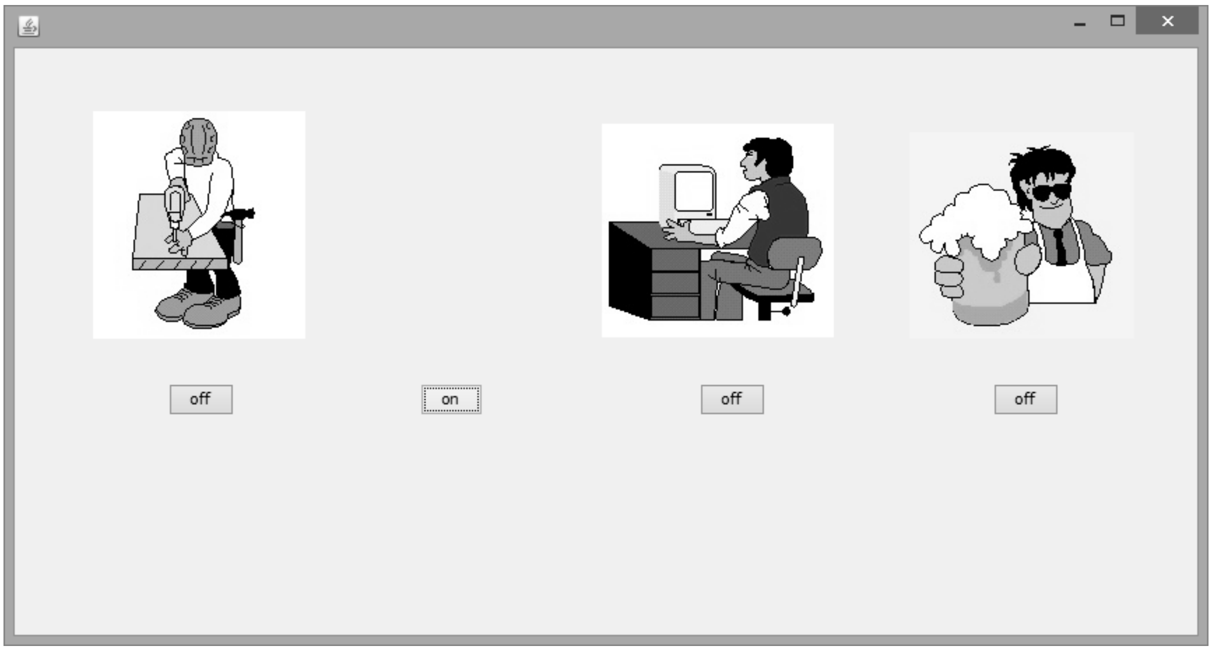
As a challenge, complete the programming for the remaining three images.

- Add toggle buttons
- Create button click methods
- Add program code to make the image visible or hidden.

Remember to change the variable names for each of the images in turn. For example, the button click method for the second picture image should be:

```
private void btnJob2ActionPerformed(java.awt.event.ActionEvent evt) {

        JToggleButton btn = (JToggleButton) evt.getSource();
        if (btn.isSelected())
        {
            lblJob2.setVisible(true);
            btnJob2.setText("off");
        }
        else
        {
            lblJob2.setVisible(false);
            btnJob2.setText("on");
        }

    }
```

Test your completed program to check that each of the picture images can be displayed or hidden correctly:


off
on
off
off