

Creating Internal Frames

NOTE: This supplement can be used after Chapter 6, "Containers and Layout Managers."

You can create multiple windows as shown in Example 11.12, "Creating Multiple Windows," in Chapter 11, "Creating User Interfaces," in *Introduction to Java Programming with JBuilder 4/5/6*. Java also allows you to use the JInternalFrame class to create windows within a window. This user interface is commonly known as *multiple document interface* or *MDI*. This user interface was quite popular and used in the earlier versions of many popular Windows software, but it is now rarely used. For this reason, this feature is not covered in the main text. Instead it is introduced in this supplement for your reference.

The JInternalFrame class is almost the same as the external JFrame class. The components are added to the internal frame in the same way as they are added to the external frame. The internal frame can have menus, the title, the Close icon, the Minimize icon, and the Maximize icon just like the external frame. The following are the major differences:

1. JInternalFrame extends JComponent, and JFrame extends the AWT Frame class. Therefore, JInternalFrame is a swing lightweight component, and JFrame is a swing heavyweight component.
2. Both JInternalFrame and JFrame are used to hold other components. JFrame is a top-level window component, and JInternalFrame must be contained inside a JDesktopPane of a JFrame or a JApplet.

Here are the steps to create an internal frame inside another window:

1. Use a JFrame or a JApplet to be the outer window.
2. Create a JDesktopPane and add it to the content pane of a JFrame or JApplet. Usually, the JDesktopPane is added to the center of the content pane.
3. Create a JInternalFrame and add it to the JDesktopPane using the add method.
4. Use the setVisible(true) method to display the internal frame.

Example Creating Internal Frames

This example creates internal frames to display flags in an applet. You can select flags from the Flags menu. Clicking a menu item causes a flag to be

displayed in an internal frame, as shown in Figure 1. The source code of the example is shown in Listing 1.

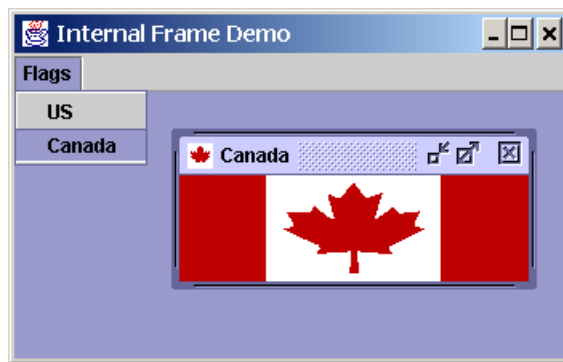


Figure 1

The flag image is displayed in an internal frame.

Listing 1: InternalFrameApplet.java

```
package internalframedemo;

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
import javax.swing.*;
import imageviewerdemo.ImageViewer;

public class InternalFrameApplet extends JApplet {
    boolean isStandalone = false;
    JMenuBar jMenuBar1 = new JMenuBar();
    JMenu jMenu1 = new JMenu();
    JMenuItem jmiUS = new JMenuItem();
    JMenuItem jmiCanada = new JMenuItem();
    ImageViewer imageView = new ImageViewer();

    // Create JDesktopPane to hold the internal frame
    JDesktopPane desktop = new JDesktopPane();
    JInternalFrame internalFrame = new JInternalFrame("US", true,
        true, true, true);

    // Create images
    Image imageUS = ImageViewer.createImage(
        "../image/us.gif", this);
    Image imageCanada = ImageViewer.createImage(
        "../image/ca.gif", this);

    // Create image icons
    ImageIcon imageUSIcon = ImageViewer.createImageIcon(
        "../image/usIcon.gif", this);
    ImageIcon imageCanadaIcon = ImageViewer.createImageIcon(
        "../image/caIcon.gif", this);
}
```

```

    /**Get a parameter value*/
    public String getParameter(String key, String def) {
        return isStandalone ? System.getProperty(key, def) :
            (getParameter(key) != null ? getParameter(key) : def);
    }

    /**Construct the applet*/
    public InternalFrameApplet() {
    }

    /**Initialize the applet*/
    public void init() {
        try {
            jbInit();
        }
        catch(Exception e) {
            e.printStackTrace();
        }
    }

    /**Component initialization*/
    private void jbInit() throws Exception {
        desktop.add(internalFrame);

        this.setSize(new Dimension(400,300));
        this.getContentPane().add(desktop, BorderLayout.CENTER);

        imageView.setImage(imageUS);
        internalFrame.setFrameIcon(imageUSIcon);

        internalFrame.getContentPane().add(imageViewer);
        internalFrame.setLocation(20, 20);
        internalFrame.setSize(100, 100);
        internalFrame.setVisible(true);

        jMenuBar1.add(jMenu1);
        jMenu1.add(jmiUS);
        jMenu1.add(jmiCanada);
        jMenu1.setText("Flags");
        jmiUS.setText("US");
        jmiCanada.setText("Canada");
        this.setJMenuBar(jMenuBar1);

        jmiUS.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(ActionEvent e) {
                jmiUS.actionPerformed(e);
            }
        });

        jmiCanada.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(ActionEvent e) {
                jmiCanada.actionPerformed(e);
            }
        });
    }

```

```

    }

    /**Get Applet information*/
    public String getAppletInfo() {
        return "Applet Information";
    }

    /**Get parameter info*/
    public String[][] getParameterInfo() {
        return null;
    }

    /**Main method*/
    public static void main(String[] args) {
        InternalFrameApplet applet = new InternalFrameApplet();
        applet.isStandalone = true;
        JFrame frame = new JFrame();
        //EXIT ON CLOSE == 3
        frame.setDefaultCloseOperation(3);
        frame.setTitle("Internal Frame Demo");
        frame.getContentPane().add(applet, BorderLayout.CENTER);
        applet.init();
        applet.start();
        frame.setSize(400,320);
        Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
        frame.setLocation((d.width - frame.getSize().width) / 2,
            (d.height - frame.getSize().height) / 2);
        frame.setVisible(true);
    }

    //static initializer for setting look & feel
    static {
        try {
            //UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
            //UIManager.setLookAndFeel(UIManager.getCrossPlatformLookAndFeelClassName());
        }
        catch(Exception e) {
        }
    }

    void jmiUS actionPerformed(ActionEvent e) {
        imageView.setImage(imageUS);
        internalFrame.setFrameIcon(imageUSIcon);
        internalFrame.setTitle("US");
    }

    void jmiCanada actionPerformed(ActionEvent e) {
        imageView.setImage(imageCanada);
        internalFrame.setFrameIcon(imageCanadaIcon);
        internalFrame.setTitle("Canada");
    }
}

```

Example Review

As shown in Figure 1, an internal frame looks similar to an external frame. Internal frames can be used just like an external frame, except that internal frames are always placed inside a JDesktopPane. JDesktopPane is a subclass of JLayeredPane. Since JDesktopPane is also a subclass of JComponent, it can be placed into the content pane of a JFrame or a JApplet.

The ImageViewer class was presented in Example 4.1, "Creating an Image Viewer." Its static methods createImage and createImageIcon create instances of Image and ImageIcon, respectively.

The properties of the JInternalFrame and JFrame are very similar. You can set a title, an internal frame icon, size, and visible for an internal frame. You may modify this example to add menus to the internal frame too.