JAVA NOTES

GRAPHICAL USER INTERFACES

Terry Marris July 2001

APPENDIX A SPLIT PANES

A split pane has two parts with an adjustable boundary between them. Figure A1 below show a horizontally split pane.

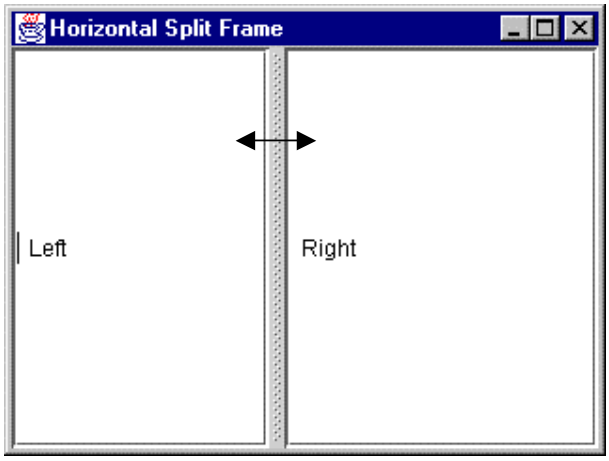

Figure A1 *A Split Pane*

We create two components, one for each part of the split pane.

```
JTextField left = new JTextField("  Left  ");
JTextField right = new JTextField("  Right  ");
```

We create a pane with a horizontal split and place a component in each part.

```
JSplitPane horizontalSplitPane = new JSplitPane(
                JSplitPane.HORIZONTAL_SPLIT, left, right);
```

We set continuous layout to true so that the contents are continuously repainted whenever the user adjusts the splitter.

```
horizontalSplitPane.setContinuousLayout(true);

getContentPane().add(horizontalSplitPane);
```

Here is the entire program.

```
/* HorizontalSplitFrame.java
   Terry Marris  13 July 2001
*/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

public class HorizontalSplitFrame extends JFrame {
  public HorizontalSplitFrame()
  {
    setTitle("Horizontal Split Frame");

    Toolkit tk = Toolkit.getDefaultToolkit();
    Dimension screen = tk.getScreenSize();
    setLocation(screen.width/4, screen.height/4);
    setSize(screen.width/2, screen.height/2);

    addWindowListener(new WindowAdapter()
      {
        public void windowClosing(WindowEvent e)
        {
          System.exit(0);
        }
      });

    JTextField left = new JTextField("  Left  ");
    JTextField right = new JTextField("  Right  ");

    JSplitPane horizontalSplitPane = new JSplitPane(
                    JSplitPane.HORIZONTAL_SPLIT, left, right);
    horizontalSplitPane.setContinuousLayout(true);

    getContentPane().add(horizontalSplitPane);
  }


  public static void main(String[] s)
  {
    new HorizontalSplitFrame().show();
  }
}
```

We can place a split pane within a split pane like this.

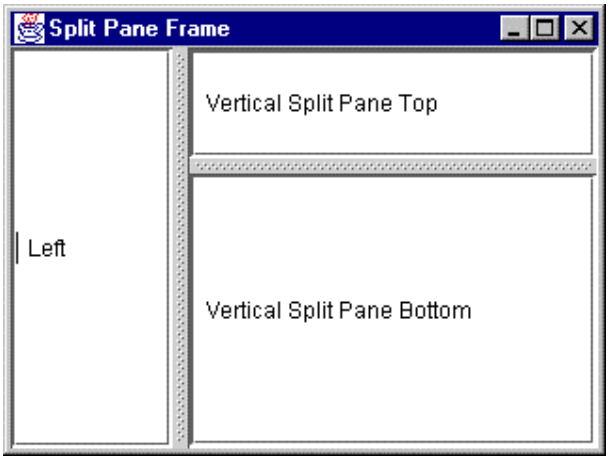

Figure A2 *A Split Pane within a Split Pane*

```
/* SplitPaneFrame.java
   Terry Marris  13 July 2001
*/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

public class SplitPaneFrame extends JFrame {
  public SplitPaneFrame()
  {
    setTitle("Split Pane Frame");

    Toolkit tk = Toolkit.getDefaultToolkit();
    Dimension screen = tk.getScreenSize();
    setLocation(screen.width/4, screen.height/4);
    setSize(screen.width/2, screen.height/2);

    addWindowListener(new WindowAdapter()
      {
        public void windowClosing(WindowEvent e)
        {
          System.exit(0);
        }
      });
```

```
    JTextField left = new JTextField("  Left  ");
    JTextField top = new JTextField(
                          "  Vertical Split Pane Top  ");
    JTextField bottom = new JTextField(
                          "  Vertical Split Pane Bottom  ");

    JSplitPane verticalSplitPane = new JSplitPane(
                       JSplitPane.VERTICAL_SPLIT, top, bottom);
    verticalSplitPane.setContinuousLayout(true);

    JSplitPane horizontalSplitPane = new JSplitPane(
        JSplitPane.HORIZONTAL_SPLIT, left, verticalSplitPane);
    horizontalSplitPane.setContinuousLayout(true);

    getContentPane().add(horizontalSplitPane);
  }


  public static void main(String[] s)
  {
    new SplitPaneFrame().show();
  }
}
```

javax.swing.JSplitPane

Fields		
static String	BOTTOM	used to add a component below the other component.
static int	HORIZONTAL_SPLIT	the components are split along the x-axis, the two components will be split one to the left of the other.
static String	LEFT	used to add a component to the left of the other component
static String	RIGHT	used to add a component to the right of the other component
static int	VERTICAL_SPLIT	the components are split along the y-axis, with one component below the other.

Constructors	
JSplitPane()	initialises a new split pane so that its components are side-by-side horizontally and with no continuous layout.
JSplitPane(int orientation)	Initialises a new split pane with the given orientation, HORIZONTAL_SPLIT or VERTICAL_SPLIT, and with no continuous layout.
JSplitPane(int orientation, boolean continuousLayout)	initialises a new split pane with the given orientation and with continuous layout, which, if true, redraws the components continuously when the user resizes them.
JSplitPane(int orientation, Component first, Component second)	initialises a new split pane with the given orientation, placing the first component is in the left or top part, the second component in the right or bottom part, and with no continuous layout.
JSplitPane(int orientation, boolean continuousLayout, Component first, Component second)	initialises a new split pane with the given orientation, components and continuous layout.

Methods		
Component	getBottomComponent()	returns the component below the divider.
Component	getLeftComponent()	returns the component to the left of the divider.
Component	getRightComponent()	returns the component to the right of the divider.
Component	getTopComponent(0	returns the component above the divider.
void	setContinousLayout(boolean b)	if true repaints the components continuously whenever the user moves the divider.
void	setDividerLocation(double d)	sets the divider location as a percentage of the JSplitPane's size, 0.0 <= d <= 1.0
void	setDividerLocation(int p)	sets the location of the divider, typically a pixel count.
void	setDividerSize(int s)	sets the size of the divider

Further reading: HORSTMANN & CORNELL *Core Java 2 Volume 2* pp 432