CSC 172 – Introduction to Algorithms and Data Structures

Auxiliary Lecture 1 – An Introduction to the Java Swing Toolkit

Source: zetcode.com

About Swing

- Swing library is an official Java GUI toolkit released by Sun Microsystems. It is used to create Graphical user interfaces with Java.
- Swing is based Java's GUI widget toolkit.
- It is part of Oracle's Java Foundation Classes (JFC), which is the APIs that are used to provide a graphical user interface (GUI) for Java programs.

About Swing

- The main characteristics of the Swing toolkit
  - **Platform Independent** – It doesn't just work on PCs
  - **Customizable** – you have control of the details for the components that you add
  - **Extensible** – Swing is modular so you can add your own components if you wish
  - **Configurable** – it can respond to changes that you make to it during runtime.
  - **Lightweight** – it uses its own APIs and not the operating system's

Swing and Its Packages

- The Swing API has 18 public packages:

javax.accessibility	javax.swing
javax.swing.border	javax.swing.colorchooser
javax.swing.event	javax.swing.filechooser
javax.swing.plaf	javax.swing.plaf.basic
javax.swing.plaf.metal	javax.swing.plaf.multi
javax.swing.plaf.synth	javax.swing.table
javax.swing.text	javax.swing.text.html
javax.swing.text.html.parser	javax.swing.text.rtf
javax.swing.tree	javax.swing.undo

SwingExample1.java

```
import javax.swing.JFrame;
import javax.swing.SwingUtilities;

public class Example extends JFrame {

    public Example() {
       setTitle("Simple example");
       setSize(300, 200);
       setLocationRelativeTo(null);
       setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
```

```
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                Example ex = new Example();
                ex.setVisible(true);
            }
        });
    }
}
```

Dissecting **SwingExample1.java**

```
//   Allows the use of the Jframe
//   package, which allows us to use
//   the Jframe class
import javax.swing.JFrame;

//   Allows the use of SwingUtilities,
//   which contains seeral methods that
//   we may need
import javax.swing.SwingUtilities;

//   Our class will be a derived from the
//   Jframe class
public class SwingExample1 extends JFrame {
```

Dissecting **SwingExample1.java (continued)**

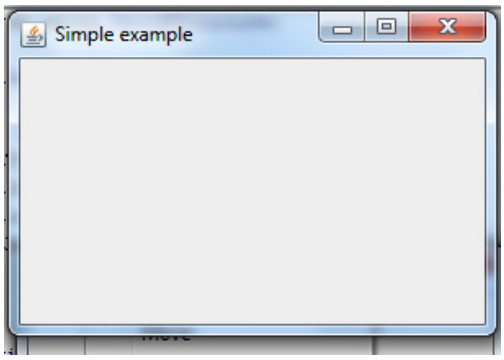
```
// The constructor for the class
 public SwingExample1() {

   // We set the window's title
   setTitle("Simple example");

   //  Our windows is 300 pixels wide by
   //  200 pixels high
   setSize(300, 200);
```

Dissecting **SwingExample1.java (continued)**

```
   //  It centers the windows because the
   // parameter is a null reference
   setLocationRelativeTo(null);
   // If we click the close button, the
   // window closes - otherwise, nothing
   // happens
   setDefaultCloseOperation(EXIT_ON_CLOSE);
}
```

Dissecting **SwingExample1.java (continued)**

```
//  We are creating an instance of a
//  method that will create our window and
//  execute whatevr it's going to do.
  SwingUtilities.invokeLater
    (new Runnable() {
      public void run() {
        Example ex = new Example();
        ex.setVisible(true);
      }
     }
   );
```



SwingExample2.java

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.SwingUtilities;

public class SwingExample2 extends JFrame {

    public SwingExample2() {
        initUI();
    }
```

```
public final void initUI() {

   JPanel panel = new JPanel();
   getContentPane().add(panel);
   panel.setLayout(null);

   JButton quitButton = new JButton("Quit");
   quitButton.setBounds(50, 60, 80, 30);
   quitButton.addActionListener
     (new ActionListener() {
      public void actionPerformed
                (ActionEvent event) {
          System.exit(0);
      }
   });
```

```
   panel.add(quitButton);

   setTitle("Quit button");
   setSize(300, 200);
   setLocationRelativeTo(null);
   setDefaultCloseOperation(EXIT_ON_CLOSE);
}
```

```
    public static void main(String[] args) {
      SwingUtilities.invokeLater
        (new Runnable() {
            public void run() {
                SwingExample2 ex
                         = new SwingExample2();
                ex.setVisible(true);
            }
          }
         );
    }
}
```

Dissecting **SwingExample2.java**

```
//  To make it listen to the button, we will
//  need an ActionEvent and an
//  ActionListener
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

Dissecting **SwingExample2.java** (continued)

```
//  It is a good programming practice to
//  put the code that creates the GUI inside
//  a specific method. We are using the
class's constructor for that purpose.
public SwingExample2() {
        initUI();
}
```

Dissecting **SwingExample2.java** (continued)

```
//  This is the method that initializes the
//  window
public final void initUI() {
  // We set up a panel as a container for
  // the button (a standard practice)
  JPanel panel = new JPanel();

  // We add the panel to the frame.
  getContentPane().add(panel);
```

Dissecting **SwingExample2.java** (continued)

```
//  JPanel has a FlowLayout manager, which
//  is used to place widgets (buttons,
//  etc.)onto containers.  By calling
//  setLayout(null), we can position the
//  components absolutely (not relative to
//  anything).
panel.setLayout(null);
```

Dissecting **SwingExample2.java** (continued)

```
//  Create the button
JButton quitButton = new JButton("Quit");

//  We position it 50 pixels from the left
//  edge and 60 pixels from the top edge.
//  It's 80 pixels wide and 30 pixels high
quitButton.setBounds(50, 60, 80, 30);
```

Dissecting **SwingExample2.java** (continued)

```
//  We add an action listener. The
//  action listener will be called, when
//  we perform an action on the button. In
//  our case, if we click on the button. The
//  click will terminate the application.

quitButton.addActionListener
  (new ActionListener() {
    public void actionPerformed
                (ActionEvent event) {
        System.exit(0);
    }
   }
);
```

Dissecting **SwingExample2.java** (continued)

```
//  We need to add the button in order to
//  make it visible.
panel.add(quitButton);
```

Tooltips

- Tooltips be used as a part of the application's help system. Swing will show a small rectangular window, if we hover a mouse pointer over an object.

SwingExample3.java

```
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.SwingUtilities;

public class SwingExample3 extends JFrame {

    public Example() {
        initUI();
    }
```

```
    public final void initUI() {

        JPanel panel = new JPanel();
        getContentPane().add(panel);

        panel.setLayout(null);
        panel.setToolTipText("A Panel container");

        JButton button = new JButton("Button");
        button.setBounds(100, 60, 100, 30);

        //  Add the tooltip to the panel and the
        //  button
        button.setToolTipText("A button component");
```

```
        panel.add(button);

        setTitle("Tooltip");
        setSize(300, 200);
        setLocationRelativeTo(null);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                Example ex = new Example();
                ex.setVisible(true);
            }
        });
    }
}
```