THE BASIC GUI APPLICATION

THERE ARE TWO BASIC TYPES of GUI program in Java: stand-alone applications and applets. An applet is a program that runs in a rectangular area on a Web page. Applets are generally small programs, meant to do fairly simple things, although there is nothing to stop them from being very complex. Applets were responsible for a lot of the initial excitement about Java when it was introduced, since they could do things that could not otherwise be done on Web pages. However, there are now easier ways to do many of the more basic things that can be done with applets, and they are no longer the main focus of interest in Java. Nevertheless, there are still some things that can be done best with applets, and they are still somewhat common on the Web. We will look at applets in the next section.

A stand-alone application is a program that runs on its own, without depending on a Web browser. You've been writing stand-alone applications all along. Any class that has a `main()` routine defines a stand-alone application; running the program just means executing this `main()` routine. However, the programs that you've seen up till now have been "command-line" programs, where the user and computer interact by typing things back and forth to each other. A GUI program offers a much richer type of user interface, where the user uses a mouse and keyboard to interact with GUI components such as windows, menus, buttons, check boxes, text input boxes, scroll bars, and so on. The main routine of a GUI program creates one or more such components and displays them on the computer screen. Very often, that's all it does. Once a GUI component has been created, it follows its **own** programming---programming that tells it how to draw itself on the screen and how to respond to events such as being clicked on by the user.

A GUI program doesn't have to be immensely complex. We can, for example, write a very simple GUI "Hello World" program that says "Hello" to the user, but does it by opening a window where the greeting is displayed:

```
import javax.swing.JOptionPane;

public class HelloWorldGUI1 {

   public static void main(String[] args) {
      JOptionPane.showMessageDialog( null, "Hello World!" );
   }

}
```

When this program is run, a window appears on the screen that contains the message "Hello World!". The window also contains an "OK" button for the user to click after reading the message. When the user clicks this button, the window closes and the program ends. By the way, this program can be placed in a file named *HelloWorldGUI1.java*, compiled, and run just like any other Java program.

Now, this program is already doing some pretty fancy stuff. It creates a window, it draws the contents of that window, and it handles the event that is generated when the user clicks the button. The reason the program was so easy to write is that all the work is done by `showMessageDialog()`, a `static` method in the built-in class *JOptionPane*. (Note that the source code "imports" the class `javax.swing.JOptionPane` to make it possible to refer to the *JOptionPane* class using its simple name. See Subsection 4.5.3 for information about importing classes from Java's standard packages.)

If you want to display a message to the user in a GUI program, this is a good way to do it: Just use a standard class that already knows how to do the work! And in fact, *JOptionPane* is regularly used for just this purpose (but as part of a larger program, usually). Of course, if you want to do anything serious in a GUI program, there is a lot more to learn. To give you an idea of the types of things that are

involved, we'll look at a short GUI program that does the same things as the previous program -- open a window containing a message and an OK button, and respond to a click on the button by ending the program -- but does it all by hand instead of by using the built-in *JOptionPane* class. Mind you, this is **not** a good way to write the program, but it will illustrate some important aspects of GUI programming in Java.

Here is the source code for the program. You are not expected to understand it yet. I will explain how it works below, but it will take the rest of the chapter before you will really understand completely. In this section, you will just get a brief overview of GUI programming.

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

public class HelloWorldGUI2 {

   private static class HelloWorldDisplay extends
JPanel {
      public void paintComponent(Graphics g) {
         super.paintComponent(g);
         g.drawString( "Hello World!", 20, 30 );
      }
   }

   private static class ButtonHandler implements
ActionListener {
      public void actionPerformed(ActionEvent e) {
         System.exit(0);
      }
   }

   public static void main(String[] args) {

      HelloWorldDisplay displayPanel = new
HelloWorldDisplay();
      JButton okButton = new JButton("OK");
      ButtonHandler listener = new ButtonHandler();
      okButton.addActionListener(listener);

      JPanel content = new JPanel();
```

```
            content.setLayout(new BorderLayout());
            content.add(displayPanel, BorderLayout.CENTER);
            content.add(okButton, BorderLayout.SOUTH);

            JFrame window = new JFrame("GUI Test");
            window.setContentPane(content);
            window.setSize(250,100);
            window.setLocation(100,100);
            window.setVisible(true);

        }

    }
```

Source : http://math.hws.edu/javanotes/c6/s1.html