**GUI08 Listener**

**Directions**

In this assignment you learn how to give your GUI programs the ability to respond to a button being clicked.

The following program responds to a button event by changing the color of the frame's background.

Create a java file named GUI08 and copy the code below into the file.

import java.util.\*;

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

// Step 1: make frame a listener

public class GUI08 extends JFrame implements

ActionListener

{

// instance variables

private JButton button;

// constructor

public GUI08()

{

// create button

button = new JButton("Change Color");

// set button attributes

button.setLocation(175,50);

button.setSize(150, 50);

// add button to frame

getContentPane().add(button);

// Step 2: register listener with button

button.addActionListener(this);

// set frame attributes

setLayout(null);

setSize(500, 500);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

// Step 3: add an actionPerformed method

public void actionPerformed(ActionEvent event)

{

if(event.getSource() == button) // identify source

{

int r = (int)(Math.random() \* 256); // red intensity: 0..255

int g = (int)(Math.random() \* 256); // green intensity: 0..255

int b = (int)(Math.random() \* 256); // blue intensity:0..255

Color color = new Color(r, g, b); // make custom color

getContentPane().setBackground(color); // change background of frame

}

}

// main method

public static void main(String[] args)

{

GUI08 app = new GUI08(); // run program

}

}

The process we will use to respond to a button being clicked (button event) consists of three steps.

Step 1 : make the frame a listener

A **listener** in Java is a class that will be notified when an event, such as a button click, occurs. To make a Java class a listener you add the following code to its class definition.

public class GUI08 extends JFrame

implements ActionListener

Step 2 : register the frame (listener) with a button

Inorder for a listener to be notified when a particular button is pressed the listener must be registered with that button. The following line of code registers the frame as a listener for button. The keyword "this" is used in Java so that a class can refer to itself. The code is basically saying "register this class as a listener for button".

button.addActionListener(this);

Step 3 : implement the actionPerformed method

The last step is to add an actionPerformed method to your listener class. The way that the operating system is going to notify the listener, in our case the frame, that the button has been pressed is by calling the actionPerformed method of the listener class. **Any code placed inside this method will be executed each time the button is pressed.**

public void actionPerformed(ActionEvent event)

{

if(event.getSource() == button) // identify source

{

int r = (int)(Math.random() \* 256); // red intensity: 0..255

int g = (int)(Math.random() \* 256); // green intensity: 0..255

int b = (int)(Math.random() \* 256); // blue intensity:0..255

Color color = new Color(r, g, b); // make custom color

getContentPane().setBackground(color); // change background of frame

}

}

A listener can be registered by more than one graphical object. So the first line of code in an actionPerformed method is usually an if statement which determines which event has occurred. The if statement below will return true if button is the object that caused the event notification to occur.

if(event.getSource() == button) // identify the source of the event

**Modifications**

Add a second button to the frame named **"Draw Circle"**. Each time this button is pressed the actionPerformed method should draw a circle at a random location within the window. The circle should be drawn in a random color with a size of 100 x 100.

To make this modification do the following three steps. The code should be added to the **actionPerformed** method so that it will be executed when the "Draw Circle" button is clicked.

**Step 1** : Request Graphics Object

When you want to draw on the frame's content pane you must use a method of the **Graphics** class. The paint method provides a Graphics object in its parameter list. However, when you want to draw on the content pane and you are not in the paint method you must request a Graphics object from the content pane. The following line of code needs to be added to the actionPerformed method so that you can draw on the frame's content pane from within this method.

Graphics

page

= getGraphics();

The page variable can now be used to draw on the content pane (i.e. page.drawOval(50, 50, 100, 100)).

**Step 2**: Change the Graphics Object's Color to a Random Color

The following code can be used to generate a random color.

int r = (int)(Math.random() \* 256); // random red

int g = (int)(Math.random() \* 256); // random green

int b = (int)(Math.random() \* 256); // random blue

Color myColor = new Color(r, g, b); // create color

page

.setColor(myColor); // set graphics object to new color

**Step 3** : Draw Circle at Random Location

The following code can be used to generate a random circle with a size of 100 x 100.

int x = (int)(Math.random() \* 500);

int y = (int)(Math.random() \* 500);

page

.fillOval(x, y, 100, 100);

**Source File**

GUI08.java

**Sample Run**

