**GUI11 TextField**

**Directions**

The next GUI component we will look at is the textfield. A textfield is small window that allows a user to enter simple text. **JTextField** is the Java class that is used to create textfields. Look at the sample code below that allows a user to type the name of color into a textfield then changes the background color of the frame to that color.

Copy the code into your source file.

import java.util.\*;

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

public class GUI11 extends JFrame implements ActionListener

{

// instance variables

private JButton button;

private JTextField textField;

// Step 1: declare JTextfield variable

// constructor

public GUI11()

{

// create button

button = new JButton("Submit");

// set button attributes

button.setLocation(50, 50);

button.setSize(100, 25);

// add button to frame

getContentPane().add(button);

// register listener with button

button.addActionListener(this);

// Step 2 : instantiate JTextField object

textField = new JTextField();

// Step 3 : set textfield attributes

textField.setLocation(175, 50);

textField.setSize(150, 25);

// Step 4 : add textfield to frame's content pane

getContentPane().add(textField);

// set frame attributes

setLayout(null);

setSize(500, 500);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent event)

{

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

{

// Step 5 : retrieve text from textfield

String str = textField.getText();

textField.setText("");

// clear textfield

if(str.equals("red"))

{

getContentPane().setBackground(Color.red);

}

if(str.equals("green"))

{

getContentPane().setBackground(Color.green);

}

if(str.equals("blue"))

{

getContentPane().setBackground(Color.blue);

}

if(str.equals("white"))

{

getContentPane().setBackground(Color.white);

}

if(str.equals("black"))

{

getContentPane().setBackground(Color.black);

}

if(str.equals("orange"))

{

getContentPane().setBackground(Color.orange);

}

if(str.equals("yellow"))

{

getContentPane().setBackground(Color.yellow);

}

}

}

// main method

public static void main(String[] args)

{

GUI11 app = new GUI11(); // run program

}

}

The same process used to add a label or button to a frame is also used with a textfield.

**Step 1 :** create a JTextField variable

private JTextField textField;

**Step 2 :** create a JTextField object and store its reference in the JTextField variable

textField = new JTextField();

**Step 3 :** set the attributes for the textfield (location, size, color, etc.)

textField.setLocation(175, 50);

textField.setSize(150, 25);

**Step 4 :** add the JTextField to the content pane of the frame

getContentPane().add(textField);

**Step 5 :** retrieving text from JTextField

To retrieve the text typed into a textfield use the JTextField method getText. If you would like to clear the textfield after retrieving the input text use the JTextField method setText. Both of these methods are demonstrated in the actionPerformed method in the code above.

String str = textField.getText(); // retrieve text from textfield

textField.setText(""); // clear textfield

**Exercise**

Modify the program so that it displays three textfields that will allow a user to enter the red, green, and blue values of a color. When the submit button is pressed the background color is change to the color created by the values entered into the three textfields. The values of the RGB colors must be in the range of 0 - 255.

1. Compile and execute the program.
2. Remove the code highlighted in red.

import java.util.\*;

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

public class GUI11 extends JFrame implements ActionListener

{

// instance variables

private JButton button;

private JTextField textField;

// Step 1: declare JTextfield variable

// constructor

public GUI12()

{

// create button

button = new JButton("Submit");

// set button attributes

button.setLocation(50, 50);

button.setSize(100, 25);

// add button to frame

getContentPane().add(button);

// register listener with button

button.addActionListener(this);

// Step 2 : instantiate JTextField object

textField = new JTextField();

// Step 3 : set textfield attributes

textField.setLocation(175, 50);

textField.setSize(150, 25);

// Step 4 : add textfield to frame's content pane

getContentPane().add(textField);

// set frame attributes

setLayout(null);

setSize(500, 500);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent event)

{

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

{

// Step 5 : retrieve text from textfield

String str = textField.getText();

textField.setText("");

if(str.equals("red"))

{

getContentPane().setBackground(Color.red);

}

if(str.equals("green"))

{

getContentPane().setBackground(Color.green);

}

if(str.equals("blue"))

{

getContentPane().setBackground(Color.blue);

}

if(str.equals("white"))

{

getContentPane().setBackground(Color.white);

}

if(str.equals("black"))

{

getContentPane().setBackground(Color.black);

}

if(str.equals("orange"))

{

getContentPane().setBackground(Color.orange);

}

if(str.equals("yellow"))

{

getContentPane().setBackground(Color.yellow);

}

}

}

// main method

public static void main(String[] args)

{

GUI11 app = new GUI11(); // run program

}

}

1. Create a textfield named **redTextField** located at (100, 50) with a size of 50 x 25.
2. Add **redTextField** to contentPane.
3. Create a textfield named **greenTextField** located at (100, 100) with a size of 50 x 25.
4. Add **greenTextField** to contentPane.
5. Create a textfield named **blueTextField** located at (100, 150) with a size of 50 x 25.
6. Add **blueTextField** to contentPane.

1. Create a label named **redLabel** located at (50, 50) with a size of 50 x 25.
2. Add **redLabel** to contentPane.
3. Create a label named **greenLabel** located at (50, 100) with a size of 50 x 25.
4. Add **greenLabel** to contentPane.
5. Create a label named **blueLabel** located at (50, 150) with a size of 50 x 25.
6. Add **blueLabel** to contentPane.

1. Position the **submit** button at location (50, 200).

1. Modify the **actionPerformed** method so that when the submit button is pressed the frame's background color changes to the color specified by the three values in the textfields.   
     
   **Step 1 :** Retrieve text from textfields
2. String red = redTextField.getText(); // retrieve text from textfield
3. String green = greenTextField.getText(); // retrieve text from textfield
4. String blue = blueTextField.getText(); // retrieve text from textfield

**Step 2 :** Convert text to integers  
  
The **getText** method used to retrieve the text in a textfield always returns a string.  
Inorder to create a color using the numbers typed into the three textfields they must   
be first converted from their string form into an integer.

int r = Integer.parseInt(red); // convert text to an integer

int g = Integer.parseInt(green); // convert text to an integer

int b = Integer.parseInt(blue); // convert text to an integer

The parseInt method converts a number in string form to integer form - Example: "25" becomes 25.   
  
**Step 3 :**Change background color

getContentPane().setBackground(new Color(r, g, b));

**Source File**

GUI11.java

**Sample Run**

