Lab 11: Event Handling

Objective(s):

1. Overview of Event Handling

1: Overview of Event Handling

Any program that uses GUI (graphical user interface) such as Java application written for windows, is event driven. Event describes the change in state of any object.

For Example: Pressing a button, entering a character in Textbox, Clicking or dragging a mouse, etc.

Components of Event Handling

Event handling has three main components,

- **Events:** An event is a change in state of an object.
- **Events Source:** Event source is an object that generates an event.
- **Listeners:** A listener is an object that listens to the event. A listener gets notified when an event occurs.

Java Event classes and Listener interfaces

Event Classes	Listener Interfaces
ActionEvent	ActionListener
MouseEvent	MouseListener and MouseMotionListener
MouseWheelEvent	MouseWheelListener
KeyEvent	KeyListener
ItemEvent	ItemListener
TextEvent	TextListener
AdjustmentEvent	AdjustmentListener
WindowEvent	WindowListener

ComponentEvent	ComponentListener
ContainerEvent	ContainerListener
FocusEvent	FocusListener

Steps to perform Event Handling

Following steps are required to perform event handling:

1. Register the component with the Listener

Registration Methods

For registering the component with the Listener, many classes provide the registration methods. For example:

- **Button**
  - public void addActionListener(ActionListener a){}
- **MenuItem**
  - public void addActionListener(ActionListener a){}
- **TextField**
  - public void addActionListener(ActionListener a){}
  - public void addTextListener(TextListener a){}
- **TextArea**
  - public void addTextListener(TextListener a){}
- **Checkbox**
  - public void addItemListener(ItemListener a){}
- **Choice**
  - public void addItemListener(ItemListener a){}
- **List**
  - public void addActionListener(ActionListener a){}
  - public void addItemListener(ItemListener a){}

Java Event Handling Code

We can put the event handling code into one of the following places:

1. Within class

2. Other class
3. Anonymous class

Java event handling by implementing ActionListener

```
import java.awt.*;
import java.awt.event.*;
class AEvent extends Frame implements ActionListener{
        TextField tf;
        AEvent(){

                //create components
                tf=new TextField();
                tf.setBounds(60,50,170,20);
                Button b=new Button("click me");
                b.setBounds(100,120,80,30);

                //register listener
                b.addActionListener(this);//passing current instance

                //add components and set size, layout and visibility
                add(b);add(tf);
                setSize(300,300);
                setLayout(null);
                setVisible(true);
        }
        public void actionPerformed(ActionEvent e){
                tf.setText("Welcome");
        }
        public static void main(String args[]){
                new AEvent();
        }
}
```

public void setBounds(int xaxis, int yaxis, int width, int height); have been used in the above example that sets the position of the component it may be button, textfield etc.

Java event handling by outer class

```
import java.awt.*;
import java.awt.event.*;
class AEvent2 extends Frame{
        TextField tf;
        AEvent2(){
        //create components
        tf=new TextField();
        tf.setBounds(60,50,170,20);
        Button b=new Button("click me");
        b.setBounds(100,120,80,30);
        //register listener
        Outer o=new Outer(this);
        b.addActionListener(o);//passing outer class instance
        //add components and set size, layout and visibility
        add(b);add(tf);
        setSize(300,300);
        setLayout(null);
        setVisible(true);
}
public static void main(String args[]){
new AEvent2();
}
}
```

```
import java.awt.event.*;
class Outer implements ActionListener{
        AEvent2 obj;
        Outer(AEvent2 obj){
        this.obj=obj;
        }
        public void actionPerformed(ActionEvent e){
                obj.tf.setText("welcome");
        }
}
```

Java event handling by anonymous class

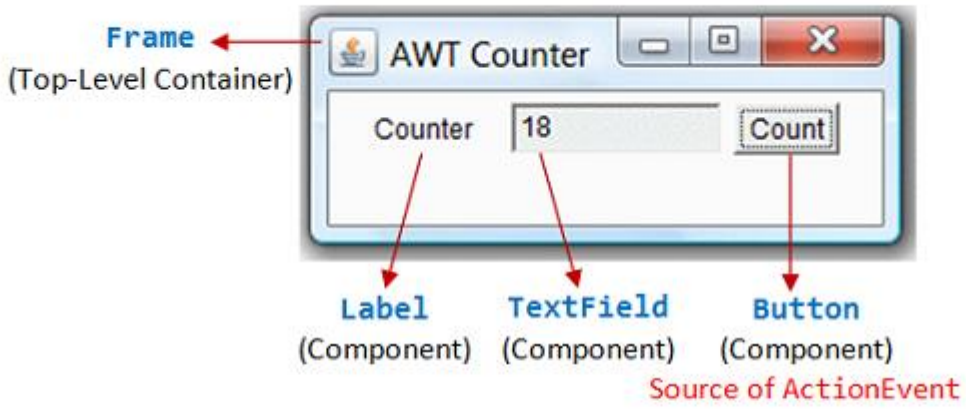
```
 import java.awt.*;
 import java.awt.event.*;
 class AEvent3 extends Frame{
        TextField tf;
        AEvent3(){
        tf=new TextField();
        tf.setBounds(60,50,170,20);
        Button b=new Button("click me");
        b.setBounds(50,120,80,30);

        b.addActionListener(new ActionListener(){
                public void actionPerformed(){
                tf.setText("hello");
                }
        });
        add(b);add(tf);
        setSize(300,300);
        setLayout(null);
        setVisible(true);
}
public static void main(String args[]){
        new AEvent3();
        }
}
```

Lab Tasks:

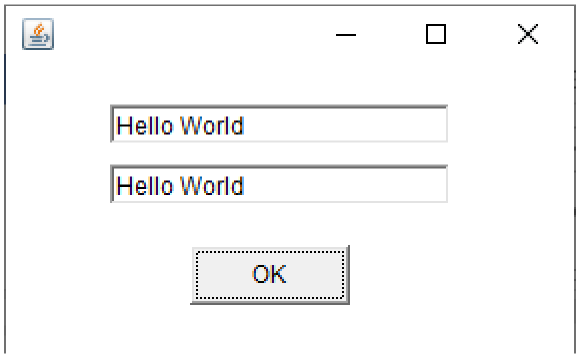
Exercise(s)

Complete following Tasks:

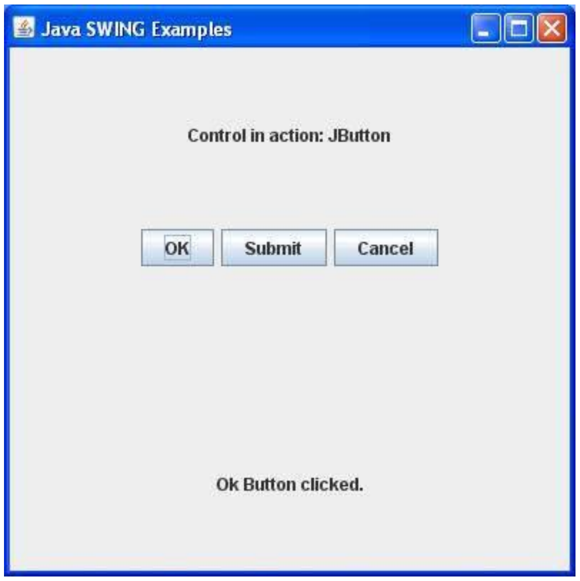
1. Write GUI application as shown in the Figure. Each time the "Count" button is clicked, the counter value shall increase by 1.



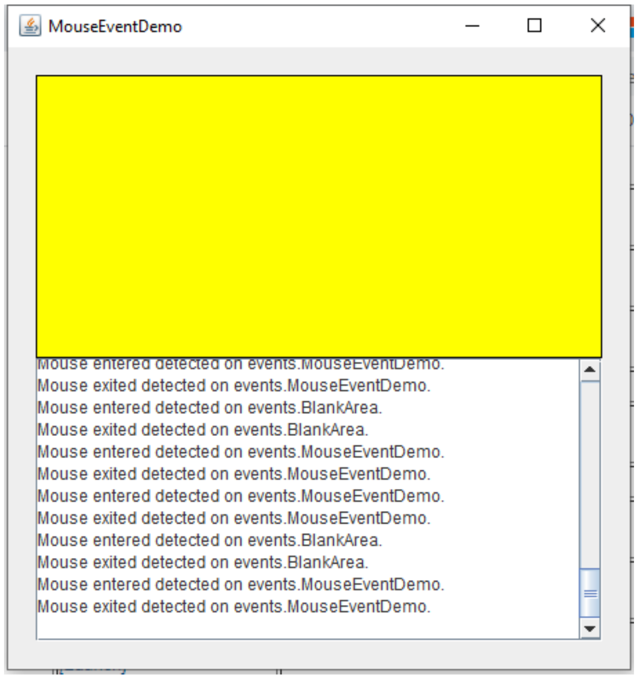
2. Examine the following picture and create a given GUI. Add 2 textboxes and one button. When a user write a text in textbox1 and click on OK button, text should be copied from textbox1 and added to textbox2.



3. Examine the following picture and wrote a program for given picture.

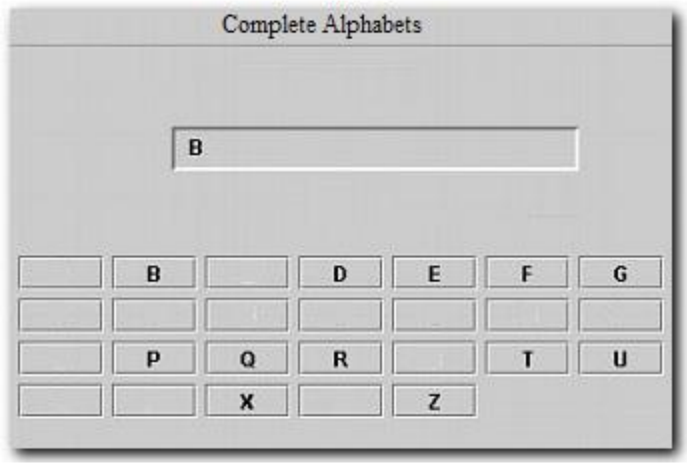


4. Examine following picture and write an application for given GUI to detect Mouse events. When a user click on yellow screen it should also detect button click.



```
Mouse entered detected on events.MouseEventDemo.
Mouse exited detected on events.MouseEventDemo.
Mouse entered detected on events.BlankArea.
Mouse exited detected on events.BlankArea.
Mouse entered detected on events.MouseEventDemo.
Mouse exited detected on events.MouseEventDemo.
Mouse entered detected on events.MouseEventDemo.
Mouse exited detected on events.MouseEventDemo.
Mouse entered detected on events.BlankArea.
Mouse exited detected on events.BlankArea.
Mouse entered detected on events.MouseEventDemo.
Mouse exited detected on events.MouseEventDemo.
```

5. Examine the following picture and write a code to create a given GUI. For this GUI you need to create and add 27 buttons and one textbox/textfield. Once a user run this application, all alphabets of buttons should not be visible until unless a user click on a particular button then it should display its label. Also if a user write a character in textbox/textfield and hit a enter key then button label should be visible. As given in picture when a user entered B and hit enter button then button B's text is visible now.



END