Lab 5: Event Handling in Java

- Integral for GUI based programs
- Events are supported by packages like java.awt
- Response is generated when the user interacts with a GUI-based elements.

Event:

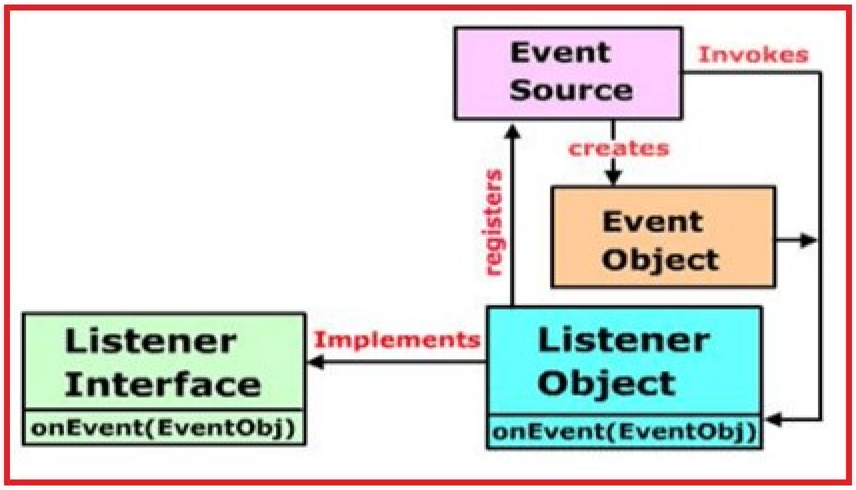
- change in the state of object or source

Event Handling:

- Controls the event and decides what should happen if an event occurs
- Delegation Event Model

Delegation Event Model

- Defines standard way for getting and processing events
- Mechanism:
    - Event Generation:
        - a source generates an event and sends it to listener(s)
        - E.g. Button Click event, etc
    - Event Listen & Handle:
        - Listeners waits until some event occurs, once an event is received, the listener processes the event and then returns.
        - Implement the interface in the listener so that it will receive the type of event desired .E.g. ActionListener is implemented for handling Button Click event.



Advantages of Delegation Event Model

- **A**ppliance logic(i.e. processing of events) is separated from the interface logic which generates those events
- An interface element is in a position to “delegate” the processing of an occasion to a separate piece of code.
- In the delegation event model, listeners must register with a source so as to receive an occasional notification:
    - Hence, notifications are sent only to listeners that want to receive them.
    - This is a more efficient way to handle events.

	Events	**Source Object**	**Listener Interface**	**Methods**
1	ActionEvent	Button, List, MenuItem, TextField	ActionListener	ActionPerformed( )
2	AdjustmentEvent	Component	ComponentListener	AdjustmentValueChanged( )

3	FocusEvent	Component	FocusListener	focusGained( )
				focusLost( )

4	TextEvent	Text Component	TextListener	TextChanged( )
5	ItemEvent	Checkbox,choice	ItemListener	ItemStateChanged( )

6	MouseEvent	Mouse Movement	MouseListener	MousePressed( )
				mouseClicked( )
				mouseEntered( )
				mouseExited( )
				mouseReleased( )

7	WindowEvent	Window	WindowListener	windowActivated( )
				windowDeactivated( )
				windowOpened( )
				windowClosed( )
				windowClosing( )

8	KeyEvent	TextComponent	KeyListener	keyTyped( )

				keyReleased()
				
				keyPressed( )

1.1 Example showing Steps to Handle Event- ActionListener

Let us discuss about click event handling in a button

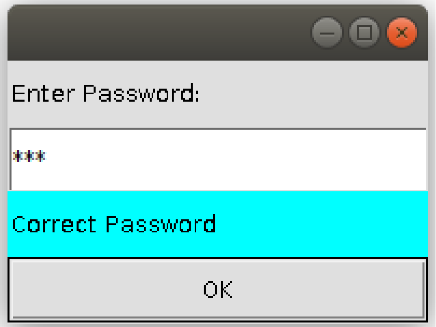
1. **Event Generation:** Whenever the user clicks the button an event is generated.

2. **Object Creation:** Object of the concerned event class will be automatically created and information about the source and the event gets populated within the same object.

3. **Listener Invocation:** Then the event object is forwarded to the method of the registered listener class.

4. **Process Event:** Now the method will get executed and returned.

```java

import java.awt.*;
import java.awt.event.*;
public class CheckPassword extends Frame implements ActionListener{
    Button b;   Label l1,l2;
    TextField t;
    GridLayout glay;
    CheckPassword(){

        b = new Button("OK");
        l1=new Label("Enter Password:");
        l2= new Label();
        t= new TextField(10);
        t.setEchoChar('*');
        glay=new GridLayout(4,1);
        setSize(200,150);
        setLayout(glay);
        add(l1);        add(t);
        add(l2);        add(b);
        b.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e){
        String s= e.getActionCommand();
        String psw;
        if(s.equals("OK")){
            psw=t.getText();
            if(psw.equals("BIT")){
                l2.setText("Correct Password");
                l2.setBackground (Color.cyan);
            }
            else{
                l2.setText("Inorrect Password");
                l2.setBackground (Color.red);
            }
        }
    }
    public static void main(String args[]){
        CheckPassword cp= new CheckPassword();
        cp.setVisible(true);
    }
}
```

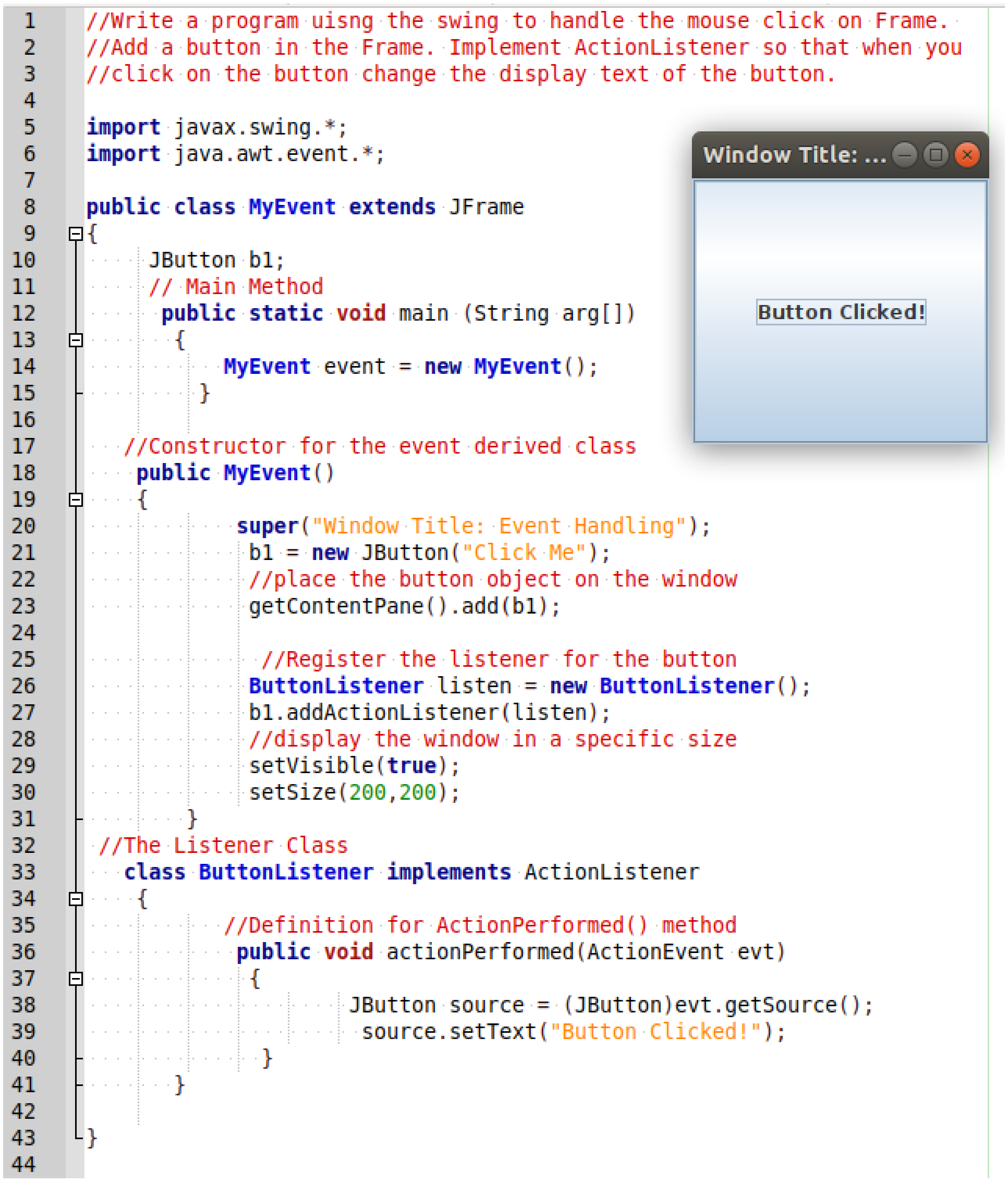


Enter Password:

***

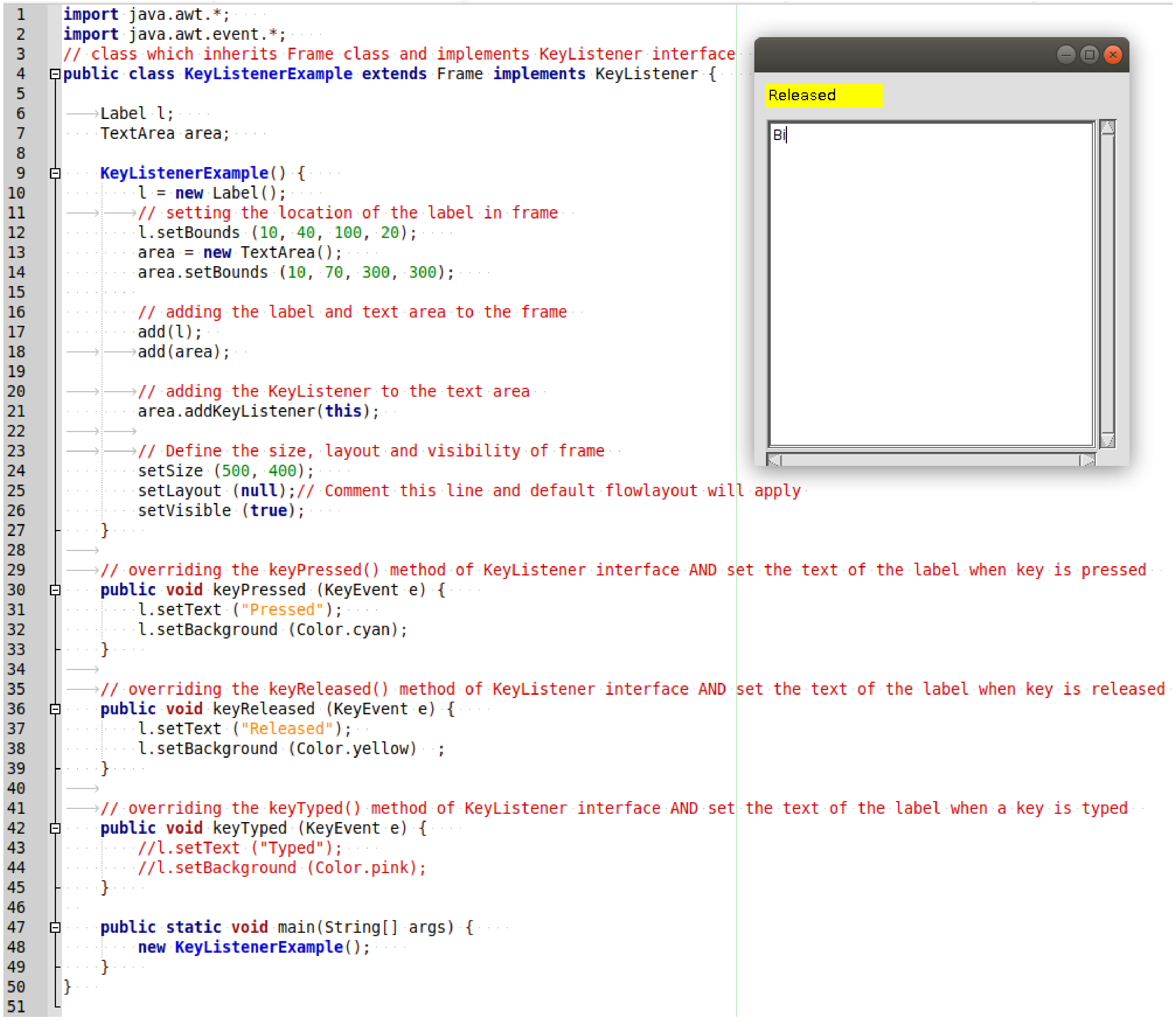
Correct Password

1.2 MyEvent.java : same example BUT a new outer class implements the ActionListener interface

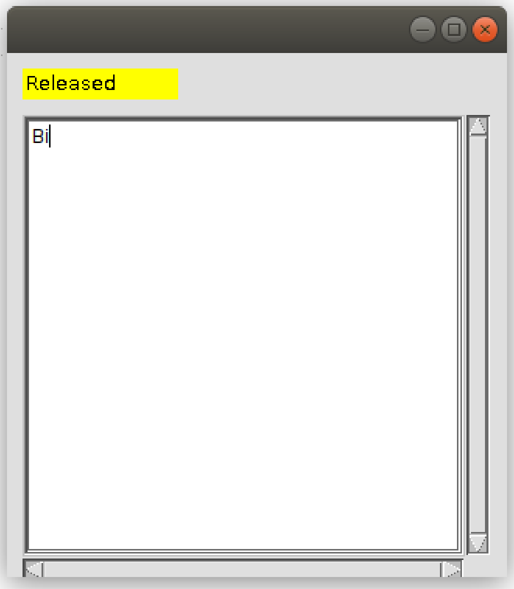


```
//Write a program uisng the swing to handle the mouse click on Frame.
//Add a button in the Frame. Implement ActionListener so that when you
//click on the button change the display text of the button.

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

public class MyEvent extends JFrame
{
    JButton b1;
    // Main Method
     public static void main (String arg[])
      {
           MyEvent event = new MyEvent();
        }

  //Constructor for the event derived class
    public MyEvent()
    {
            super("Window Title: Event Handling");
             b1 = new JButton("Click Me");
             //place the button object on the window
             getContentPane().add(b1);

              //Register the listener for the button
             ButtonListener listen = new ButtonListener();
             b1.addActionListener(listen);
             //display the window in a specific size
             setVisible(true);
             setSize(200,200);
        }
 //The Listener Class
   class ButtonListener implements ActionListener
    {
           //Definition for ActionPerformed() method
            public void actionPerformed(ActionEvent evt)
             {
                     JButton source = (JButton)evt.getSource();
                      source.setText("Button Clicked!");
              }
       }

}
```

2.1 Example showing KeyListener Example

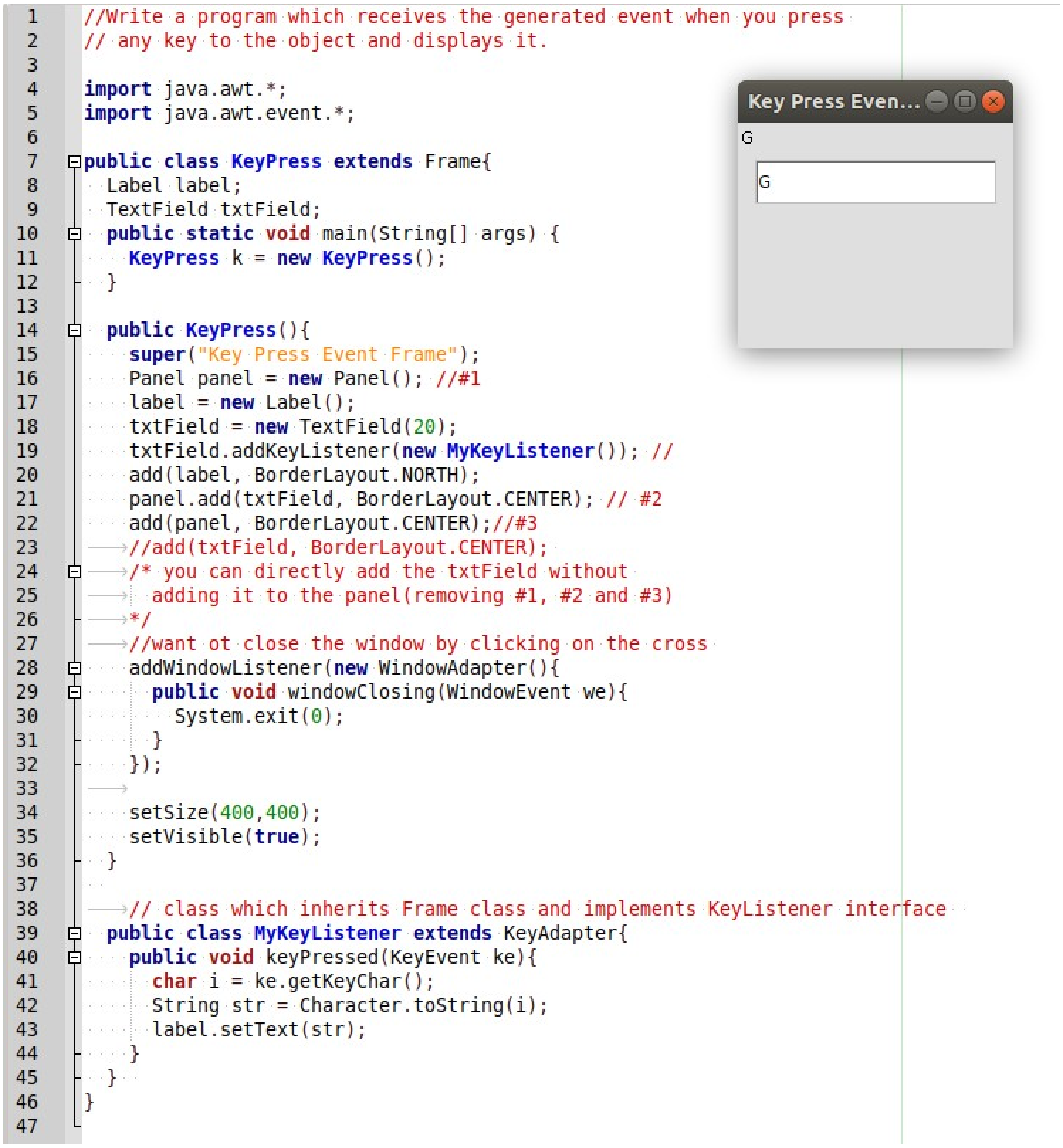


```java
import java.awt.*;
import java.awt.event.*;
// class which inherits Frame class and implements KeyListener interface
public class KeyListenerExample extends Frame implements KeyListener {

    Label l;
    TextArea area;

    KeyListenerExample() {
        l = new Label();
        // setting the location of the label in frame
        l.setBounds (10, 40, 100, 20);
        area = new TextArea();
        area.setBounds (10, 70, 300, 300);

        // adding the label and text area to the frame
        add(l);
        add(area);

        // adding the KeyListener to the text area
        area.addKeyListener(this);

        // Define the size, layout and visibility of frame
        setSize (500, 400);
        setLayout (null);// Comment this line and default flowlayout will apply
        setVisible (true);
    }

    // overriding the keyPressed() method of KeyListener interface AND set the text of the label when key is pressed
    public void keyPressed (KeyEvent e) {
        l.setText ("Pressed");
        l.setBackground (Color.cyan);
    }

    // overriding the keyReleased() method of KeyListener interface AND set the text of the label when key is released
    public void keyReleased (KeyEvent e) {
        l.setText ("Released");
        l.setBackground (Color.yellow)  ;
    }

    // overriding the keyTyped() method of KeyListener interface AND set the text of the label when a key is typed
    public void keyTyped (KeyEvent e) {
        //l.setText ("Typed");
        //l.setBackground (Color.pink);
    }

    public static void main(String[] args) {
        new KeyListenerExample();
    }
}
```



```
Released
Bi
```

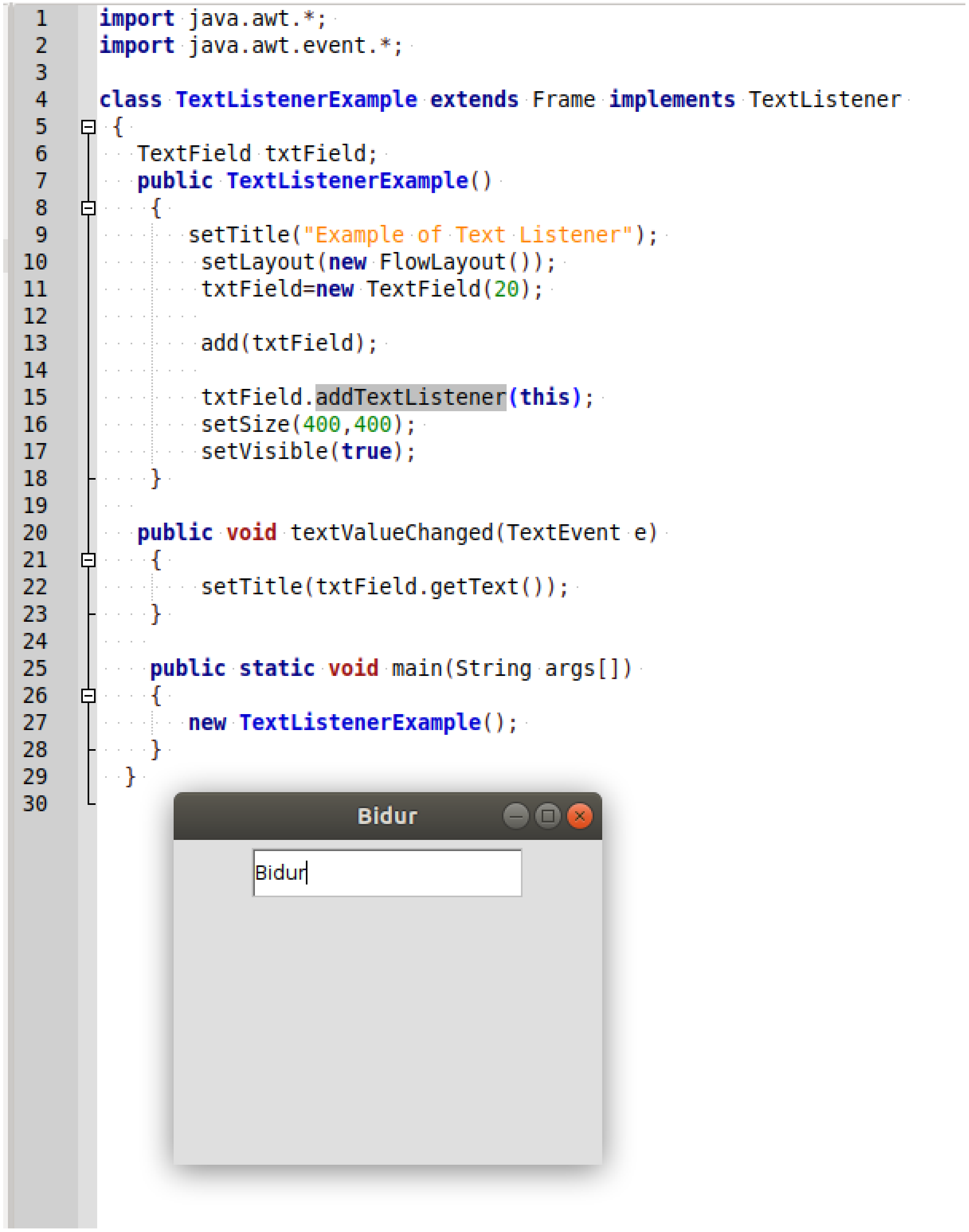
2.2 KeyPress.java : same example BUT a new class implements the KeyListener interface



```
1  //Write a program which receives the generated event when you press
2  // any key to the object and displays it.
3
4  import java.awt.*;
5  import java.awt.event.*;
6
7  public class KeyPress extends Frame{
8    Label label;
9    TextField txtField;
10   public static void main(String[] args) {
11     KeyPress k = new KeyPress();
12   }
13
14   public KeyPress(){
15     super("Key Press Event Frame");
16     Panel panel = new Panel(); //#1
17     label = new Label();
18     txtField = new TextField(20);
19     txtField.addKeyListener(new MyKeyListener()); //
20     add(label, BorderLayout.NORTH);
21     panel.add(txtField, BorderLayout.CENTER); // #2
22     add(panel, BorderLayout.CENTER);//#3
23     //add(txtField, BorderLayout.CENTER);
24     /* you can directly add the txtField without
25       adding it to the panel(removing #1, #2 and #3)
26     */
27     //want ot close the window by clicking on the cross
28     addWindowListener(new WindowAdapter(){
29       public void windowClosing(WindowEvent we){
30         System.exit(0);
31       }
32     });
33
34     setSize(400,400);
35     setVisible(true);
36   }
37
38     // class which inherits Frame class and implements KeyListener interface
39   public class MyKeyListener extends KeyAdapter{
40     public void keyPressed(KeyEvent ke){
41       char i = ke.getKeyChar();
42       String str = Character.toString(i);
43       label.setText(str);
44     }
45   }
46 }
47
```

Line 28: WindowEvent (implementing WindowListener) is also shown in this example 2.2

3. Example showing **TextListener Example**



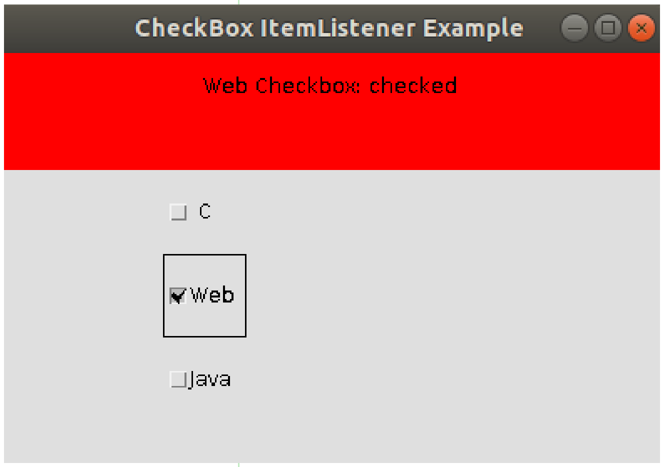
```
import java.awt.*;
import java.awt.event.*;

class TextListenerExample extends Frame implements TextListener
{
  TextField txtField;
  public TextListenerExample()
   {
      setTitle("Example of Text Listener");
       setLayout(new FlowLayout());
       txtField=new TextField(20);

       add(txtField);

       txtField.addTextListener(this);
       setSize(400,400);
       setVisible(true);
   }

  public void textValueChanged(TextEvent e)
   {
       setTitle(txtField.getText());
   }

   public static void main(String args[])
   {
      new TextListenerExample();
   }
 }
```

Bidur

Bidur

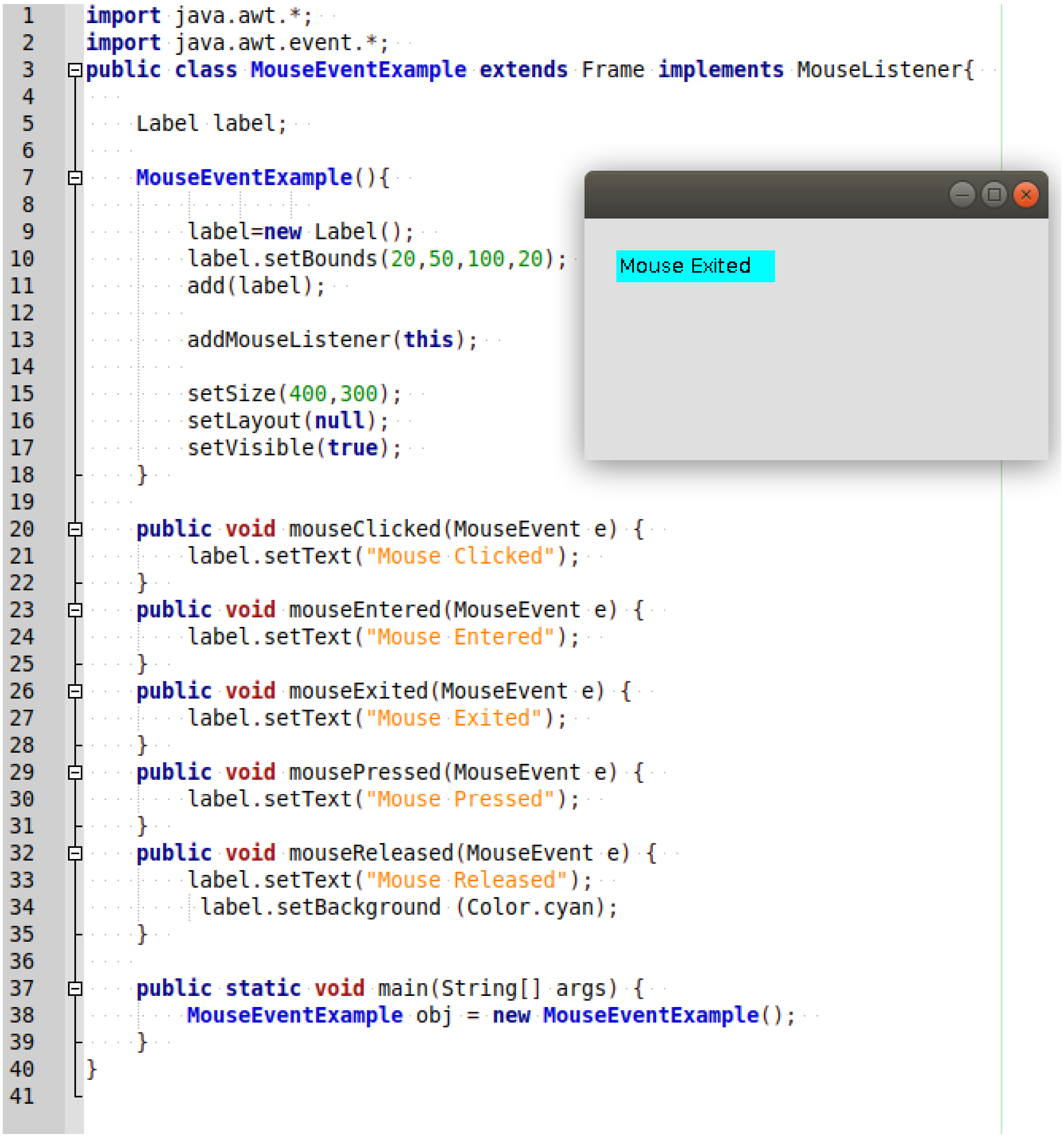
4. **Example showing CheckboxItemListener Example**

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

public class CheckboxItemListenerExample implements ItemListener{

    Checkbox checkBox1, checkBox2, checkBox3;
    Label label;
    CheckboxItemListenerExample(){
        Frame f= new Frame("CheckBox ItemListener Example");
        label = new Label();
        label.setAlignment(Label.CENTER);
        label.setSize(400,100);

        checkBox1 = new Checkbox("C");
        checkBox1.setBounds(100,100, 50,50);
        checkBox2 = new Checkbox("Web");
        checkBox2.setBounds(100,150, 50,50);
        checkBox3 = new Checkbox("Java");
        checkBox3.setBounds(100,200, 50,50);
        f.add(checkBox1);        f.add(checkBox2);
        f.add(checkBox3);        f.add(label);

        checkBox1.addItemListener(this);
        checkBox2.addItemListener(this);
        checkBox3.addItemListener(this);

        f.setSize(400,300);
        f.setLayout(null);
        f.setVisible(true);
    }

    public void itemStateChanged(ItemEvent e) {
        if(e.getSource()==checkBox1){
            label.setText("C Checkbox: " + (e.getStateChange()==1?"checked":"unchecked"));
            label.setBackground (Color.cyan);
        }

        if(e.getSource()==checkBox2)  {
            label.setText("Web Checkbox: " + (e.getStateChange()==1?"checked":"unchecked"));
            label.setBackground (Color.red);
        }
        if(e.getSource()==checkBox3)  {
            label.setText("Java Checkbox: " + (e.getStateChange()==1?"checked":"unchecked"));
            label.setBackground (Color.yellow);
        }
    }

    public static void main(String args[]) {
        CheckboxItemListenerExample obj = new CheckboxItemListenerExample();
    }
}
```



```
Web Checkbox: checked
```

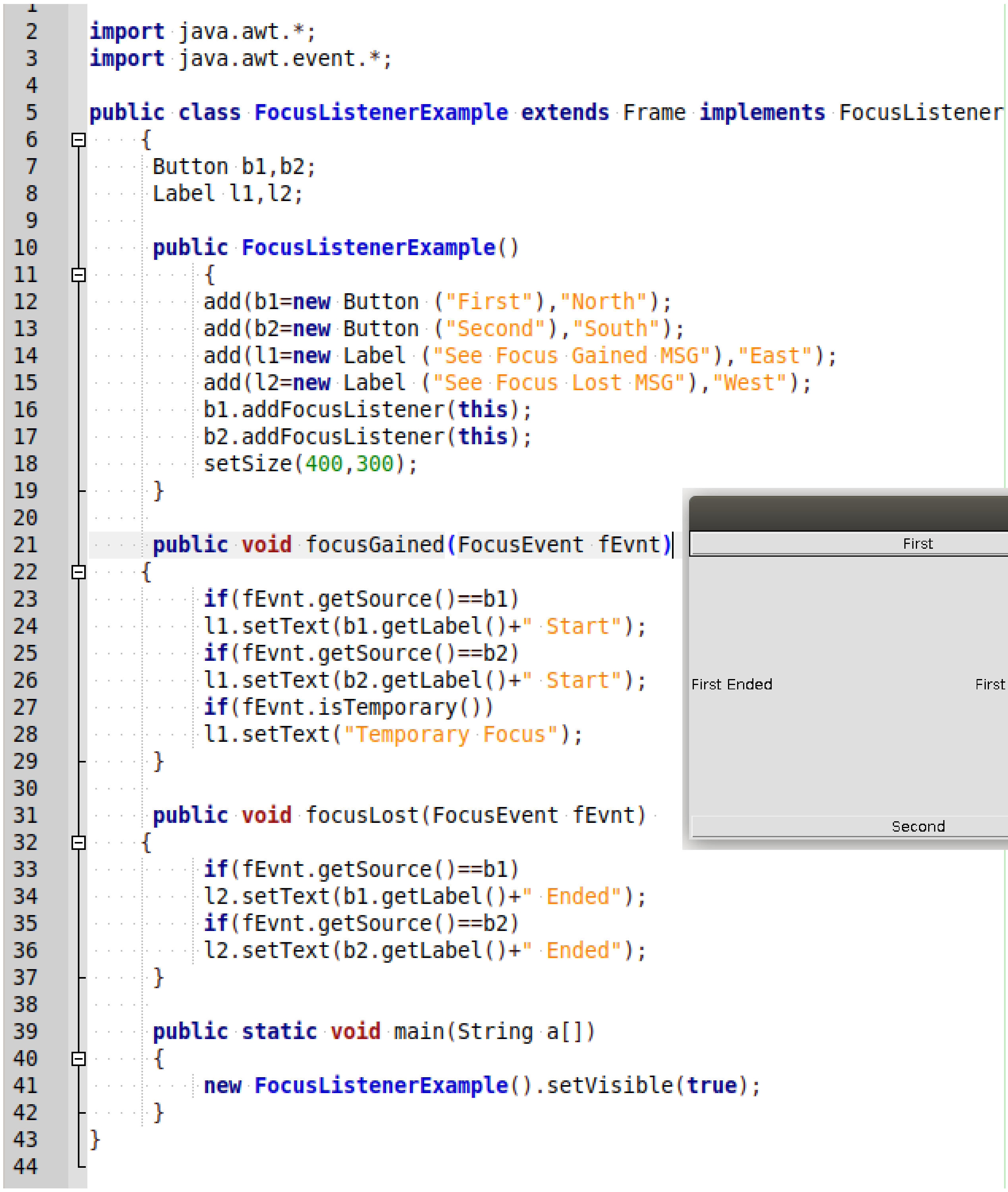
5. **Example showing MouseEvent Example**



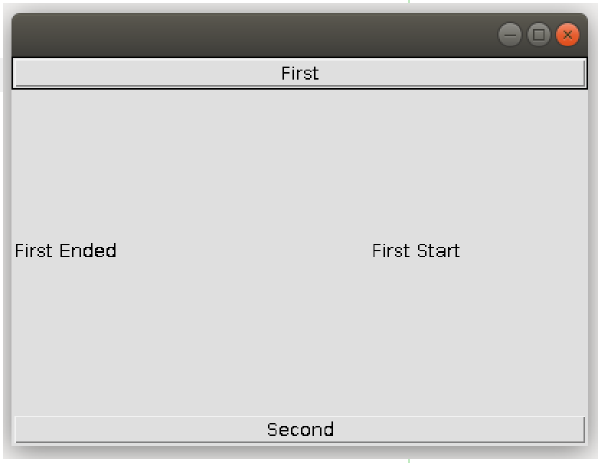
```java
import java.awt.*;
import java.awt.event.*;
public class MouseEventExample extends Frame implements MouseListener{

    Label label;

    MouseEventExample(){

        label=new Label();
        label.setBounds(20,50,100,20);
        add(label);

        addMouseListener(this);

        setSize(400,300);
        setLayout(null);
        setVisible(true);
    }

    public void mouseClicked(MouseEvent e) {
        label.setText("Mouse Clicked");
    }
    public void mouseEntered(MouseEvent e) {
        label.setText("Mouse Entered");
    }
    public void mouseExited(MouseEvent e) {
        label.setText("Mouse Exited");
    }
    public void mousePressed(MouseEvent e) {
        label.setText("Mouse Pressed");
    }
    public void mouseReleased(MouseEvent e) {
        label.setText("Mouse Released");
         label.setBackground (Color.cyan);
    }

    public static void main(String[] args) {
        MouseEventExample obj = new MouseEventExample();
    }
}
```

```
Mouse Exited
```

6. Example showing FocusListener Example

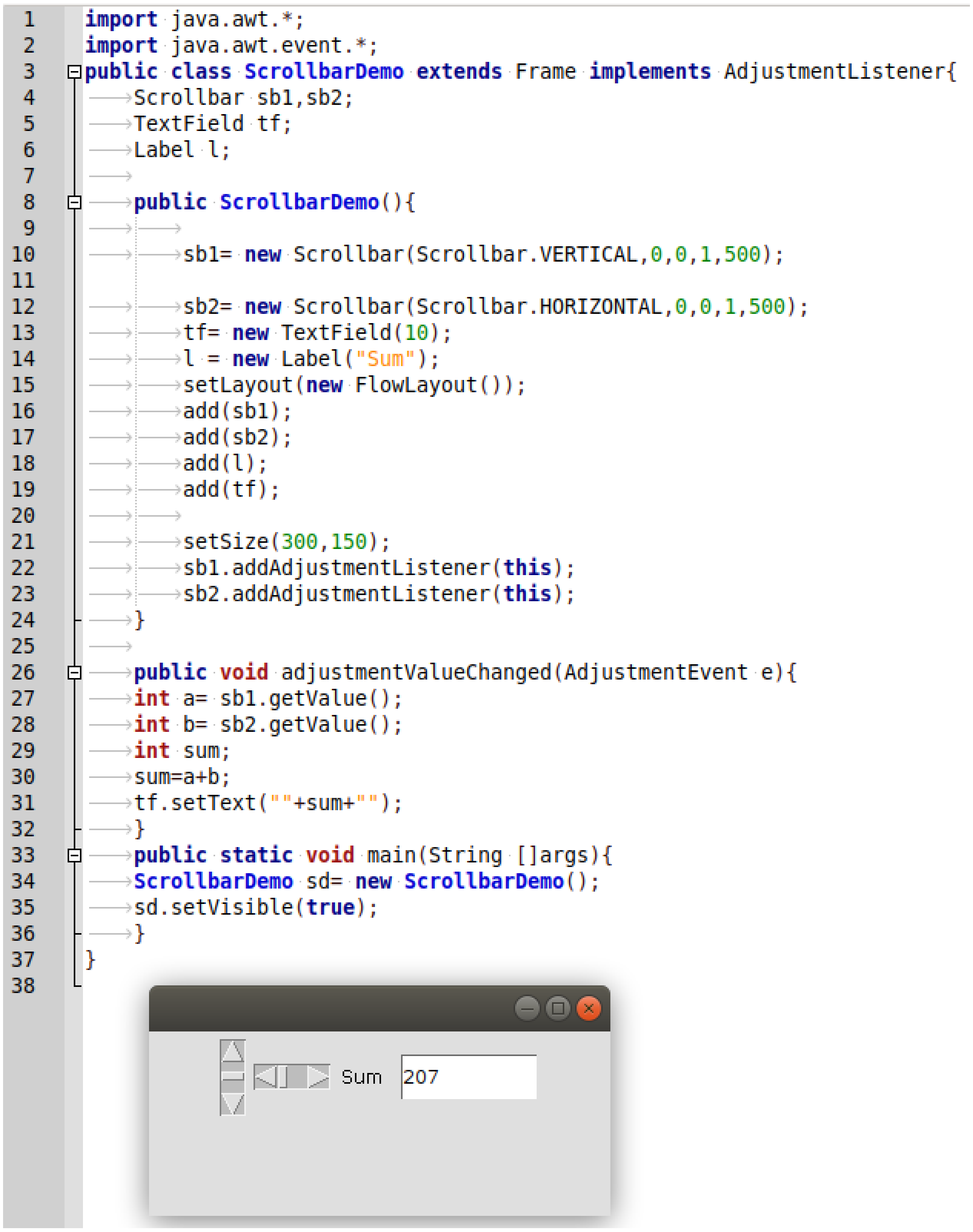


```
import java.awt.*;
import java.awt.event.*;

public class FocusListenerExample extends Frame implements FocusListener
    {
     Button b1,b2;
     Label l1,l2;

     public FocusListenerExample()
        {
        add(b1=new Button ("First"),"North");
        add(b2=new Button ("Second"),"South");
        add(l1=new Label ("See Focus Gained MSG"),"East");
        add(l2=new Label ("See Focus Lost MSG"),"West");
        b1.addFocusListener(this);
        b2.addFocusListener(this);
        setSize(400,300);
     }

     public void focusGained(FocusEvent fEvnt)
    {
        if(fEvnt.getSource()==b1)
        l1.setText(b1.getLabel()+" Start");
        if(fEvnt.getSource()==b2)
        l1.setText(b2.getLabel()+" Start");
        if(fEvnt.isTemporary())
        l1.setText("Temporary Focus");
     }

     public void focusLost(FocusEvent fEvnt)
    {
        if(fEvnt.getSource()==b1)
        l2.setText(b1.getLabel()+" Ended");
        if(fEvnt.getSource()==b2)
        l2.setText(b2.getLabel()+" Ended");
     }

     public static void main(String a[])
     {
        new FocusListenerExample().setVisible(true);
     }
}
```



7. Example showing AdjustmentListener Example

- Create two scrollbars and display the sum of their values in a TextField



```java
import java.awt.*;
import java.awt.event.*;
public class ScrollbarDemo extends Frame implements AdjustmentListener{
    Scrollbar sb1,sb2;
    TextField tf;
    Label l;
    
    public ScrollbarDemo(){
        
        sb1= new Scrollbar(Scrollbar.VERTICAL,0,0,1,500);

        sb2= new Scrollbar(Scrollbar.HORIZONTAL,0,0,1,500);
        tf= new TextField(10);
        l = new Label("Sum");
        setLayout(new FlowLayout());
        add(sb1);
        add(sb2);
        add(l);
        add(tf);
        
        setSize(300,150);
        sb1.addAdjustmentListener(this);
        sb2.addAdjustmentListener(this);
    }
    
    public void adjustmentValueChanged(AdjustmentEvent e){
    int a= sb1.getValue();
    int b= sb2.getValue();
    int sum;
    sum=a+b;
    tf.setText(""+sum+"");
    }
    public static void main(String []args){
    ScrollbarDemo sd= new ScrollbarDemo();
    sd.setVisible(true);
    }
}
```