Experiment No.2

Aim of the Experiment: Write a program to create a frame using AWT. Implement mouseClicked, mouseEntered() and mouseExited() events. Frame should become visible when the mouse enters it.

Objective:

To Develop programs to handle events in Java Programming.

Resources:

Eclips IDE, JDK 1.8.0 is required

Course Outcome Addressed: CO2

Theory:

Event handling is fundamental to Java programming because it is used to create event driven programs eg • Applets • GUI based windows application • Web Application.

Event handling mechanism have been changed significantly between the original version of Java (1.0) and all subsequent versions of Java, beginning with version 1.1.

The modern approach to handling events is based on the delegation event model,

What is an Event?

Change in the state of an object is known as event i.e. event describes the change in state of source. Events are generated as result of user interaction with the graphical user interface components. For example, clicking on a button, moving the mouse, entering a character through keyboard, selecting an item from list, scrolling the page are the activities that causes an event to happen.

Types of Event:

The events can be broadly classified into two categories:

Foreground Events - Those events which require the direct interaction of user. They are generated as consequences of a person interacting with the graphical components in Graphical User Interface. For example, clicking on a button, moving the mouse, entering a character through keyboard, selecting an item from list, scrolling the page etc.

Background Events - Those events that require the interaction of end user are known as background events. Operating system interrupts, hardware or software failure, timer expires, an operation completion are the example of background events.

What is Event Handling?

Event Handling is the mechanism that controls the event and decides what should happen if an event occurs. This mechanism have the code which is known as event handler that is executed when an event occurs. Java Uses the Delegation Event Model to handle the events. This model defines the standard mechanism to generate and handle the events. Let's have a brief introduction to this model. The Delegation Event Model has the following key participants namely: Source - The source is an object on which event occurs. Source is responsible for providing information of the occurred event to it's handler. Java provides classes for source object. Listener - It is also known as event handler. Listener is responsible for generating response to an event. From java implementation point of view the listener is also an object. Listener waits until it receives an event. Once the event is received , the listener process the event an then returns. Advantages of

Advantages of event Handling:

The benefit of this approach is that the user interface logic is completely separated from the logic that generates the event. The user interface element is able to delegate the processing of an event to the separate piece of code. In this model ,Listener needs to be registered with the source object so that the listener can receive the event notification. This is an efficient way of handling the event because the event notifications are sent only to those listener that want to receive them.

Java MouseListener Interface:

The Java MouseListener is notified whenever you change the state of mouse. It is notified against MouseEvent. The MouseListener interface is found in java.awt.event package. It has five methods.

Methods of MouseListener interface

The signature of 5 methods found in MouseListener interface are given below:

1. public abstract void mouseClicked(MouseEvent e);
2. public abstract void mouseEntered(MouseEvent e);
3. public abstract void mouseExited(MouseEvent e);
4. public abstract void mousePressed(MouseEvent e);
5. public abstract void mouseReleased(MouseEvent e);

Program Logic:

1. Start the program

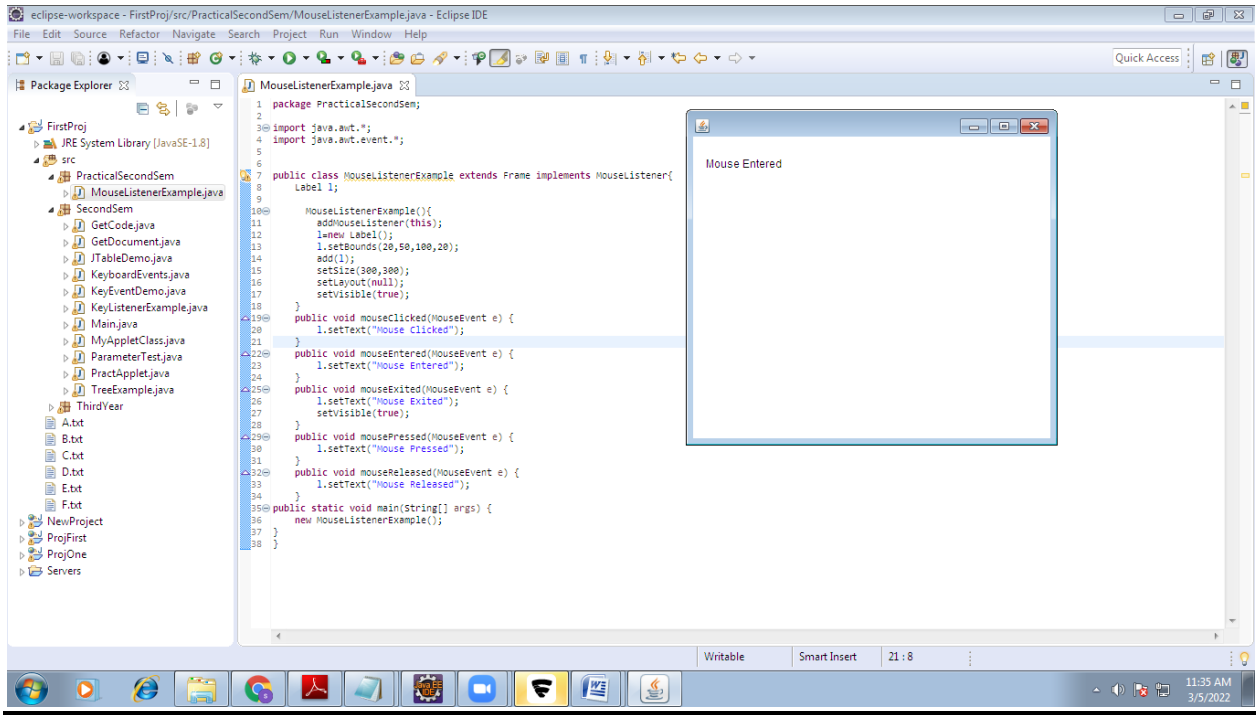
2. Create a class with the name: MouseListenerExample extends Frame implements MouseListener

3.

SOURCE CODE:

```
package PracticalSecondSem;
import java.awt.*;
import java.awt.event.*;
public class MouseListenerExample extends Frame implements MouseListener{
   Label l;
     MouseListenerExample(){
      addMouseListener(this);
      l=new Label();
      l.setBounds(20,50,100,20);
      add(l);
      setSize(300,300);
      setLayout(null);
      setVisible(true);
   }
   public void mouseClicked(MouseEvent e) {
      l.setText("Mouse Clicked");
   }
   public void mouseEntered(MouseEvent e) {
      l.setText("Mouse Entered");
   }
   public void mouseExited(MouseEvent e) {
      l.setText("Mouse Exited");
      setVisible(false);
   }
   public void mousePressed(MouseEvent e) {
      l.setText("Mouse Pressed");
   }
   public void mouseReleased(MouseEvent e) {
      l.setText("Mouse Released");
   }
public static void main(String[] args) {
   new MouseListenerExample();
}
}
```

OUTPUT:



Conclusion:-

References:

Herbert Schildt , "Java : The Complete Reference" Tata McGraw-Hill (7th Edition).

Practical Related Questions

1. Write algorithm of a program to create a frame using AWT. Implement mouseClicked, mouseEntered() and mouseExited() events. Frame should become visible when the mouse enters it.
2. List various methods of MouseListener and MouseMotionListener
3. Do all components generate the MouseEvent
4. Write the steps to obtain the coordinates of MouseClick
5. Write the steps to register for MosueEvents.