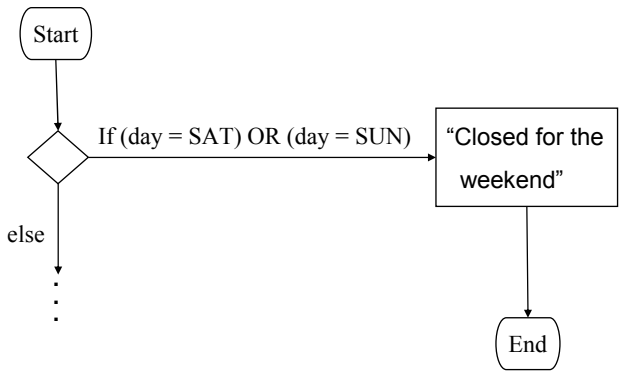
An Introduction To Graphical User Interfaces

The event-driven model

Building a simple graphical interface in Java

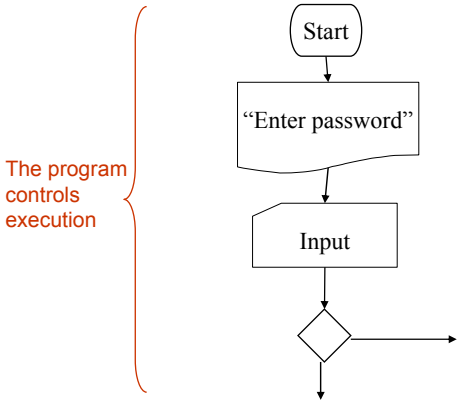
Traditional Software

Program control is largely determined by the program through a series of statements.



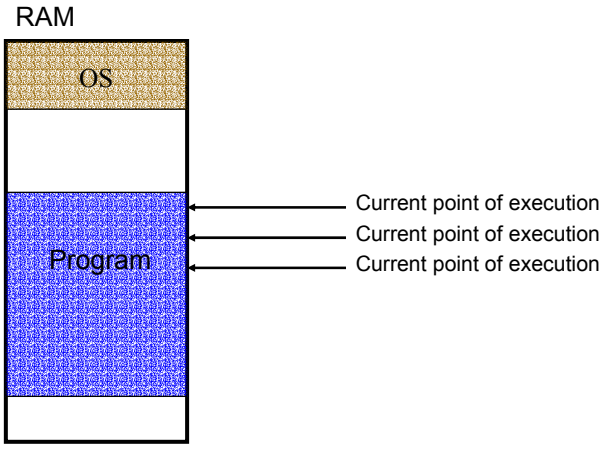
Traditional Software

Program control is largely determined by the program through a series of statements.



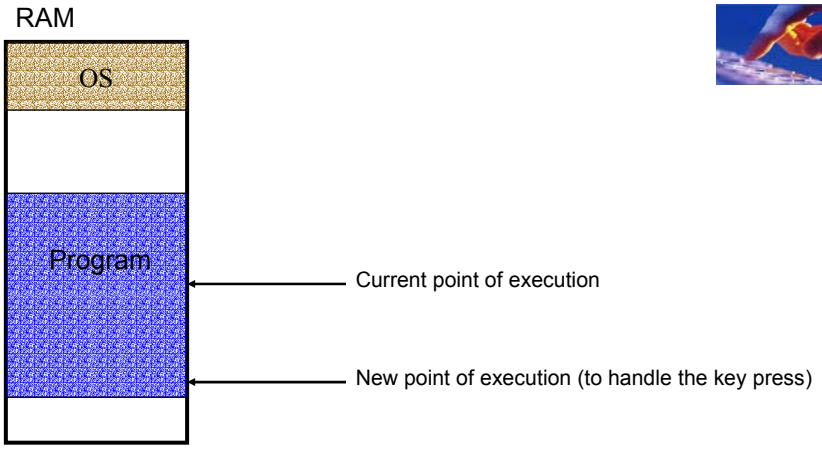
Event-Driven Software

Program control *can also* be determined by events



Event-Driven Software

Program control *can also* be determined by events



Characteristics Of Event Driven Software

Program control can be determined by events as well standard program control statements

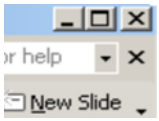
The typical source of these events is the user

These events can occur at any time

Onscreen Objects Can Trigger Events

Graphical objects can be manipulated by the user to trigger events.

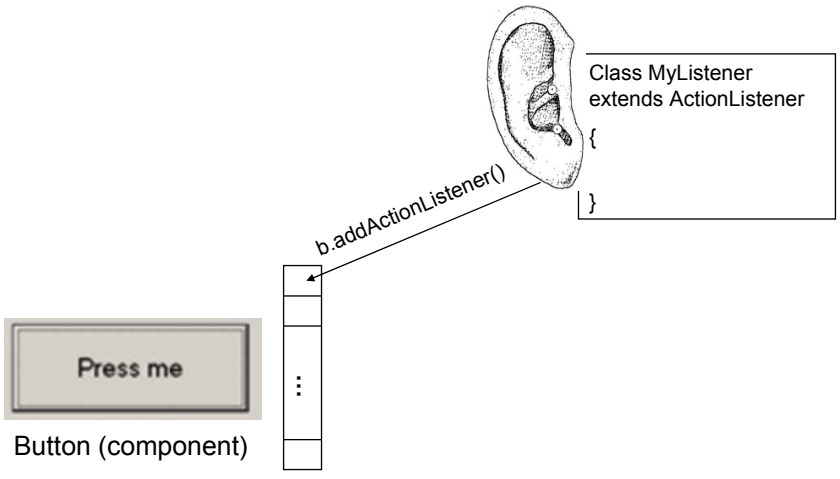
Each object can have 0, 1 or many events that can be triggered.



Steps In The Event Model

1) The graphical component must register all interested event listeners.
2) The user triggers an event by manipulating that component
3) The component sends a message to all listeners of that event
4) The event listener provides code to handle the event

1. The Graphical Component Must Register All Interested Event Listeners.



2. The User Triggers An Event By Manipulating That Component

3. The Component Sends A Message To All Registered Listeners For That Event

```
class MyListener extends ActionListener
{
    public void actionPerformed (ActionEvent e)
    {

    }
```

3. The Component Sends A Message To All Registered Listeners For That Event

```
class MyListener extends ActionListener
{
    public void actionPerformed (ActionEvent e)
    {
        button.setText("Stop pressing me");
    }
```

4. The Event Listener Provides Code To Handle The Event

```
class MyListener extends ActionListener
{
    public void actionPerformed (ActionEvent e)
   {
        button.setText("Stop pressing me");
    }
```

Building A Simple GUI

This example can be found in the directory:
/home/profs/tamj/233/examples/GUIs

The Driver Class

```
import java.awt.*;
import java.awt.event.*;
public class Driver
{
  public static void main(String[] args)
  {
     Frame f = new Frame ();
     Panel p = new Panel ();
     f.add(p);

     Button makePopup = new Button ("Popup");
     Button goesInvisible = new Button("Press me");
     Button quitProgram = new Button ("Quit");
     MyListener simpleListener = new MyListener ();
```


The Driver Class (2)

```
     // Para1 = x coordinate, Para2 = y coordinate, Para3 = width, Para4 = height
     f.setBounds(100,200,300,200);
     p.add(goesInvisible);
     p.add(makePopup);
     p.add(quitProgram);

     // Register simpleListener as a listener to when the buttons get pressed
     goesInvisible.addActionListener(simpleListener);
     makePopup.addActionListener(simpleListener);
     quitProgram.addActionListener(simpleListener);

     // Make the frame visible
     f.show();
     }
  }
```

Class MyListener

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

public class MyListener implements java.awt.event.ActionListener
{
  public void actionPerformed(ActionEvent e)
  {
     String s = e.getActionCommand();
     if (s.equals("Popup"))
     {
          Frame f = new Frame ("Popup window");
          f.setBounds(0,0,200,100);
          f.show();
          for (int i = 0; i < 100000000; i++);
          f.setTitle("Going away soon");
          for (int i = 0; i < 100000000; i++);
          f.setVisible(false);
     }
```


Class MyListener

```
     else if (s.equals("Press me"))
     {
          Button b = (Button) e.getSource();
          b.setVisible(false);
     }
     else if (s.equals("Quit"))
     {
          System.exit(0);
     }
  }
}
```

Summary

You should now know:

- The difference between traditional and event driven software
- How event-driven software works
- A simple graphical interface example