An Introduction To Graphical User Interfaces

You will learn about the event-driven model and how to create simple graphical user interfaces (GUI's) in Java

James Tam

Note: GUI Code Cannot Be Run Through A Text-Only Connection: SSH

```
[csb exampleTwo 45 ]> ls
Driver.class*    Driver.java      MyListener.class*  MyListener.java

[csb exampleTwo 46 ]> java Driver
Exception in thread "main" java.lang.InternalError: Can't connect to X11 window server using
  ':0.0' as the value of the DISPLAY variable.
      at sun.awt.X11GraphicsEnvironment.initDisplay(Native Method)
      at sun.awt.X11GraphicsEnvironment.<clinit>(X11GraphicsEnvironment.java:125)
      at java.lang.Class.forName0(Native Method)
      at java.lang.Class.forName(Class.java:140)
      at
  java.awt.GraphicsEnvironment.getLocalGraphicsEnvironment(GraphicsEnvironment.java:62)
      at java.awt.Window.init(Window.java:223)
      at java.awt.Window.<init>(Window.java:267)
      at java.awt.Frame.<init>(Frame.java:398)
      at java.awt.Frame.<init>(Frame.java:363)
      at Driver.main(Driver.java:7)
```

James Tam

Components

- They are many types of graphical controls and displays available:
  - JButton, JFrame, JLabel, JTextArea, JWindow, JList
- A graphical component is also known as “widgets”
- For Sun’s online documentation refer to the url:
  - *http://download.oracle.com/javase/7/docs/api/* (especially java.awt.event, javax.swing.event, and javax.swing).

James Tam

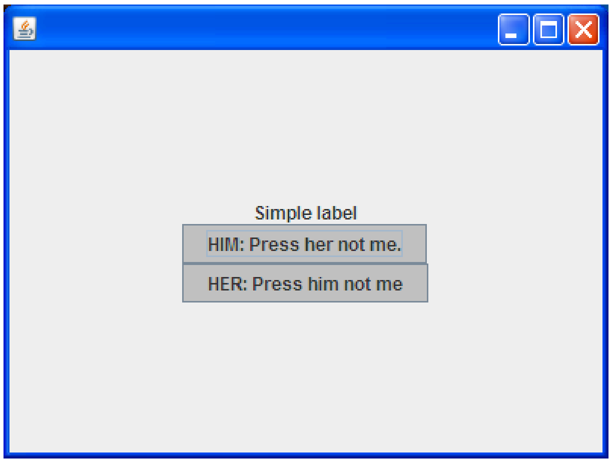
Containers

- A special type of component that is used to hold/contain the components (subclass of the basic component class).
- Can be used to group components on the screen (i.e., one container holds another container which in turn groups a number of controls).
- You must have at least one container object for your GUI:
  - JPanel, JWindow, JDialog, JFrame
- Components which have been added to a container will appear/disappear and be garbage collected along with the container.

James Tam

Containers

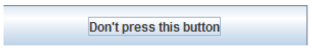
- You must have at least one container object for your GUI:
  - JPanel, JWindow, JDialog, J**Frame**



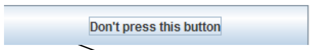
James Tam

Some Relevant Java GUI libraries

1. Java classes for the components and containers
   - e.g., JButton class
   - javax.swing (import javax.swing.* or import javax.swing.<*class name*>)



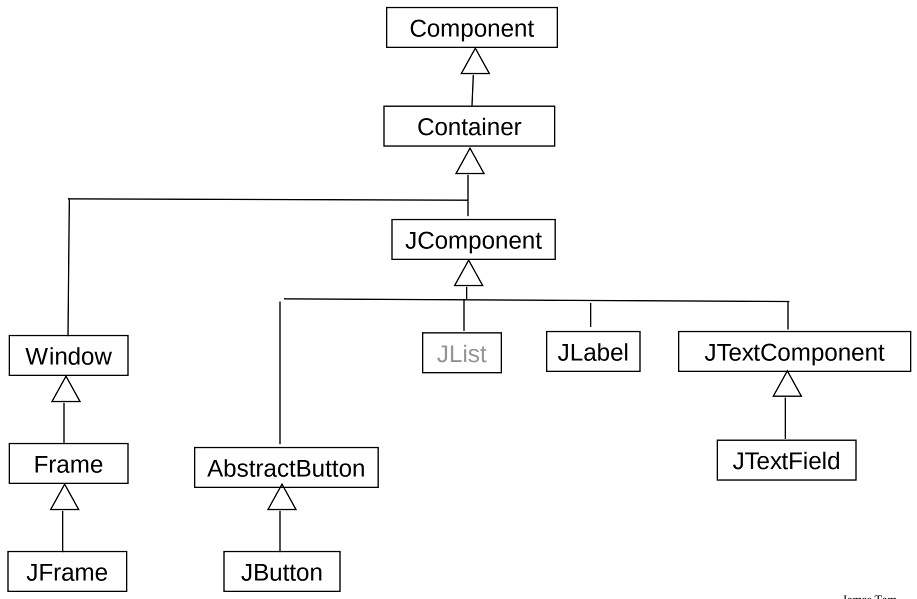
2. Java classes with the code to react to user-initiated events
   - e.g., code that executes when a button is pressed
   - java.awt.event (import java.awt.event.*, import javax.swing.event.*)



```
class ButtonListener implements ActionListener
{
    public void actionPerformed(ActionEvent e)
    {
            :          :          :
    }
}
```

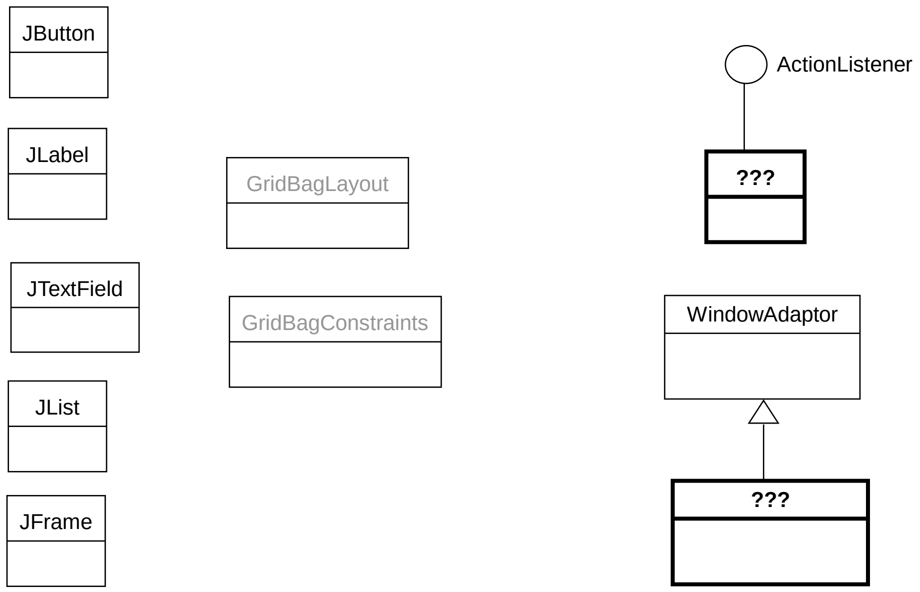
James Tam

Hierarchy: Important Widget Classes



James Tam

Some Relevant Java GUI Classes For This Section



James Tam

Traditional Software

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

Example

```
    :
    if (num >= 0)
    {
        // Statements for the body of the if
    }
    else
    {
        // Statements for the body of the else
    }
```

When num is non-negative

Num is negative

James Tam

Traditional Software

•The user can only interact with the program at places that are specified by the program (e.g., when an input statement is encountered).

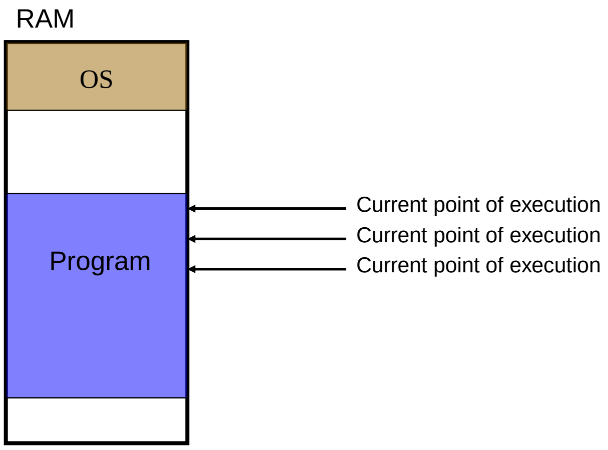
Example

```
Scanner aScanner = new Scanner (System.in);
System.out.print(“Enter student ID number: “);
id = aScanner.nextInt ();
```

James Tam

Event-Driven Software

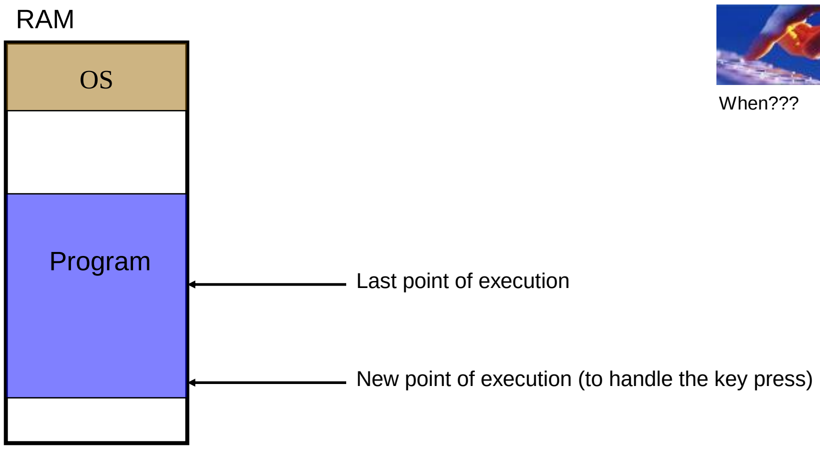
•Program control can also be sequential


RAM
OS
Program
Current point of execution
Current point of execution
Current point of execution

James Tam

Event-Driven Software

- In addition program control *can also* be determined by events



When???

James Tam

Characteristics Of Event Driven Software

- Program control can be determined by events as well as standard program control statements.
- A typical source of these events is the user.
- These events can occur at any time.

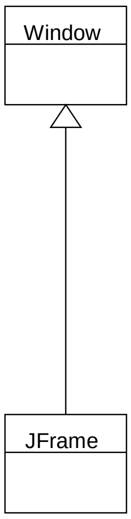
James Tam

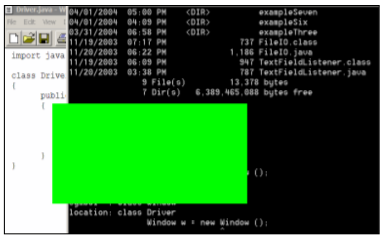
Most Components Can Trigger Events

- Graphical objects can be manipulated by the user to trigger events.
- Each graphical object can have 0, 1 or many events that can be triggered.

James Tam

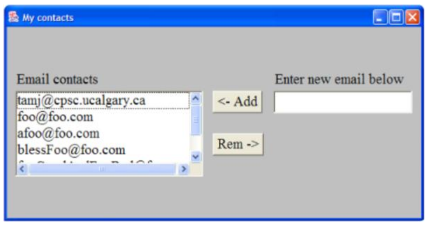
Window Classes





```
import java

class Drive
{
        publi
        {

        }
}
```

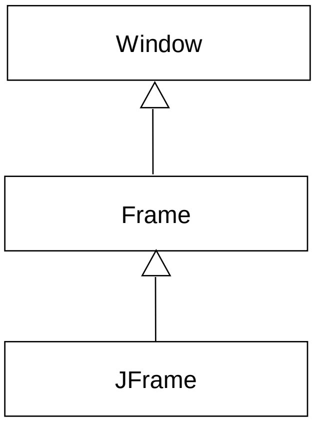
```
04/01/2004  05:00 PM    <DIR>          exampleSeven
04/01/2004  04:09 PM    <DIR>          exampleSix
03/31/2004  06:58 PM    <DIR>          exampleThree
11/19/2003  07:17 PM               737 FileIO.class
11/20/2003  06:22 PM             1,186 FileIO.java
11/19/2003  06:09 PM               947 TextFieldListener.class
11/20/2003  03:38 PM               787 TextFieldListener.java
               9 File(s)         13,378 bytes
               7 Dir(s)   6,389,465,088 bytes free

                                     ();

location: class Driver
                Window w = new Window ();
                               ^
```



tamj@cpsc.ucalgary.ca
foo@foo.com
afoo@foo.com
blessFoo@foo.com

James Tam

The Window Class Hierarchy



James Tam

Class JFrame

- For full details look at the online API:
  - http://download.oracle.com/javase/7/docs/api/javax/swing/JFrame.html
- Some of the more pertinent methods:
  - JFrame ("<*Text on the title bar*>")
  - setSize (<*pixel width*>, <*pixel height*>)
  - setVisible (<*true/false*>)
  - setDefaultCloseOperation (<*class constants*>[1])

1 DISPOSE_ON_CLOSE, HIDE_ON_CLOSE, DO_NOTHING_ON_CLOSE

James Tam

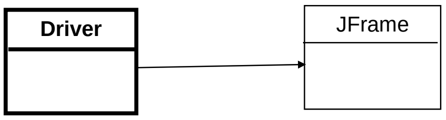
Example: Creating A Frame That Can Close (And Cleanup Memory After Itself)

- Location of the example:

/home/233/examples/gui/first_frame

OR

www.cpsc.ucalgary.ca/~tamj/233/examples/gui



James Tam

Example: Creating A Frame That Can Close (And Cleanup Memory After Itself)

```
import javax.swing.*;
public class Driver
{
    public static void main (String [] args)
    {
        JFrame mf = new JFrame ("Insert title here");
        mf.setSize (300,200);
        mf.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        mf.setVisible(true);
    }
}
```

James Tam

Pitfall 1: Showing Too Early

- When a container holds a number of components the components must be added to the container (later examples).
- To be on the safe side the call to the "setVisible()" method should be done after the contents of the container have already been created and added.

James Tam

Window Events

•The basic JFrame class provides basic capabilities for common windowing operations: minimize, maximize, resize, close.

•However if a program needs to perform other actions (i.e., your own custom code) when these events occur the built in approach won't be sufficient.

- E.g., the program is to automatically save your work to a file when you close the window.

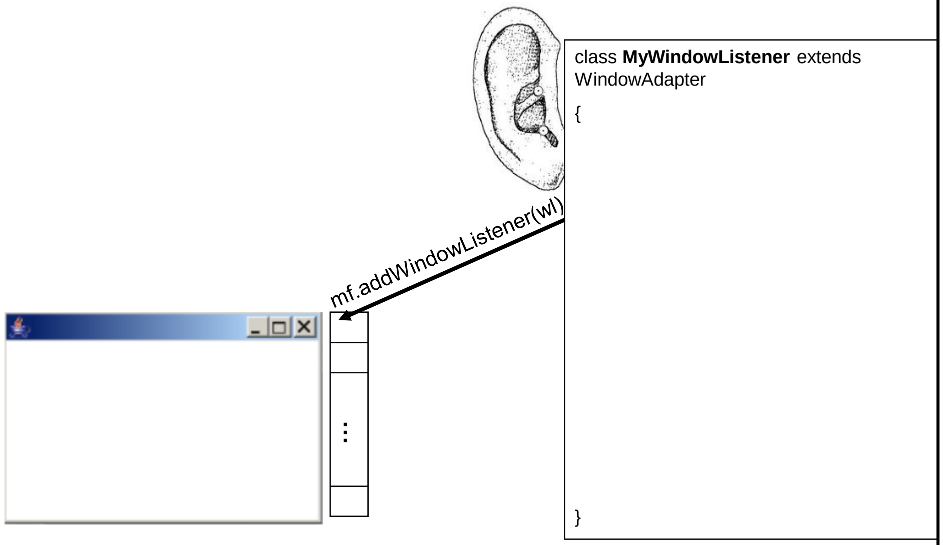
James Tam

Steps In The Event Model For Handling A Frame Event: Window Closing

1) The frame must register all interested event listeners.
2) The user triggers the event by closing the window
3) The window sends a message to all listeners of that event.
4) The window event listener runs the code to handle the event (e.g., save information to a file).

James Tam

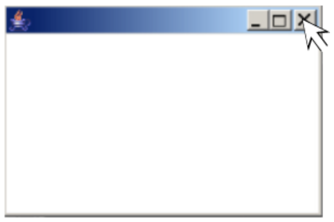
1. The Frame Must Register All Interested Event Listeners.



```
class MyWindowListener extends
WindowAdapter
{
}
```

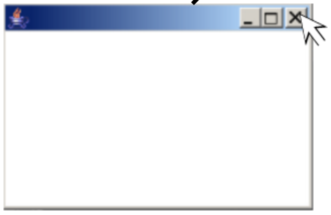
James Tam

2. The User Triggers The Event By Closing The Window



James Tam

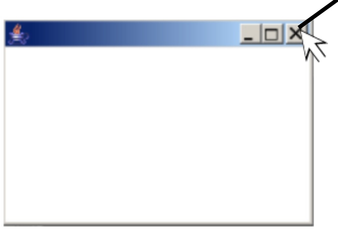
3. The Window Sends A Message To All Listeners Of That Event.



```
public class **MyWindowListener** extends
WindowAdapter
{
        public void windowClosing  (WindowEvent e)
        {



         }
}
```

James Tam

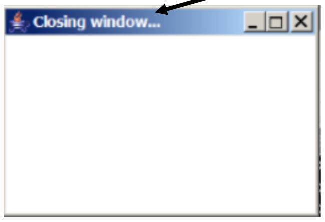
4. The Event Listener Runs The Code To Handle The Event.



```
public class MyWindowListener extends
WindowAdapter
{
        public void windowClosing (WindowEvent e)
        {
           /* Code to react to event * /
           JFrame aFrame = (JFrame) e.getWindow();
           aFrame.setTitle("Closing window...");
          // Pause program so user can see the message.
           aFrame.setVisible(false);
           aFrame.dispose();
        }
}
```

James Tam

4. The Event Listener Runs The Code To Handle The Event.



```
public class MyWindowListener extends
WindowAdapter
{
        public void windowClosing (WindowEvent e)
        {
            /* Code to react to event * /
            JFrame aFrame = (JFrame) e.getWindow();
            aFrame.setTitle("Closing window...");
           // Pause program so user can see the message.
            aFrame.setVisible(false);
            aFrame.dispose();
        }
}
```

James Tam

An Example Of Handling A Frame Event

•Location of the example:

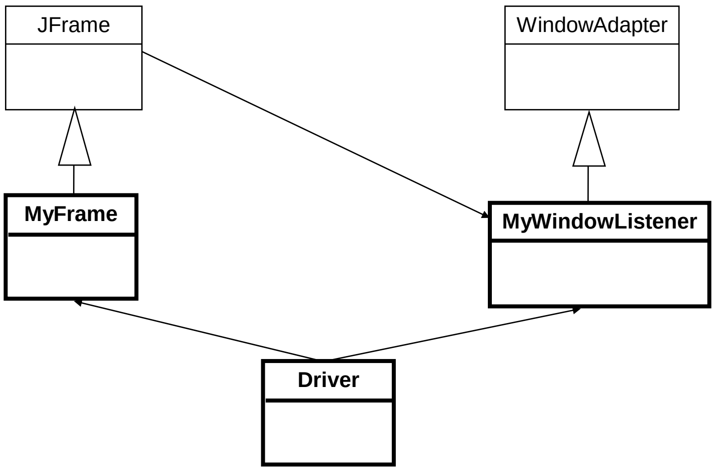
/home/233/examples/gui/second_window_events

OR

www.cpsc.ucalgary.ca/~tamj/233/examples/gui/second_window_events

James Tam

An Example Of Handling A Frame Event (2)



James Tam

The Driver Class

```
import javax.swing.JFrame;

public class Driver
{
    public static final int WIDTH = 300;
    public static final int HEIGHT = 200;
    public static void main (String [] args)
    {
        MyFrame aFrame = new MyFrame ();
        MyWindowListener aListener = new MyWindowListener() ;
        aFrame.addWindowListener(aListener);
        aFrame.setSize (WIDTH,HEIGHT);
        aFrame.setVisible(true);
    }
}
```

James Tam

Class MyFrame

```
import javax.swing.JFrame;

public class MyFrame extends JFrame
{
    // More code will be added in later examples.
}
```

James Tam

Class MyWindowListener

```
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import javax.swing.JFrame;

public class MyWindowListener extends WindowAdapter {
        public void windowClosing (WindowEvent e) {
        JFrame aFrame = (JFrame) e.getWindow();
        aFrame.setTitle("Closing window...");
        try
           Thread.sleep(3000);
        catch (InterruptedException ex)
           System.out.println("Pausing of program was interrupted");
        aFrame.setVisible(false);
        aFrame.dispose();
      }
}
```

James Tam

Callback

- The code that handles the event (the code that is called when a GUI event such as a window closing occurs) is commonly referred to as a "callback".
  - An old IBM IDE (VisualAge) used to refer to these as 'event to code'.
    - Evaluation copy: http://download.cnet.com/IBM-VisualAge-for-Java/3000-2247_4-18868.html
    - IBM page: http://www-142.ibm.com/software/products/us/en/atoz

James Tam

Callback (2)

- Example callbacks:

```
// Window event callback (you have already seen this example)
 public void windowClosing (WindowEvent e) {
     << Called when window event occurs >>
 }

// Button event callback (you haven't yet seen this example)
public void actionPerformed (ActionEvent e)  {
     << Called when button event occurs >>


}
```

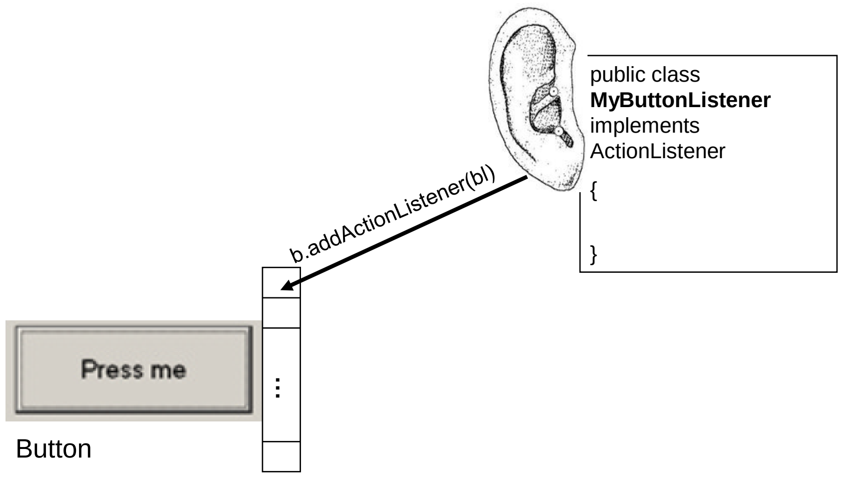
James Tam

Steps In The Event Model For Handling A Button Event

1) The button must register all interested event listeners.
2) The user triggers an event by pressing a button.
3) The button sends a message to all listeners of the button press event.
4) The button listener runs the code to handle the button press event.

James Tam

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

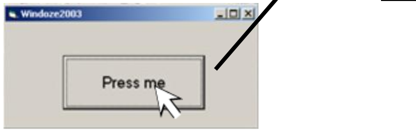


James Tam

2. The User Triggers An Event By Pressing The Button

James Tam

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

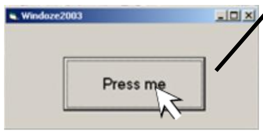


```
public class **MyButtonListener** implements ActionListener
{
     public void actionPerformed (ActionEvent e)
    {


    }
}
```

James Tam

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

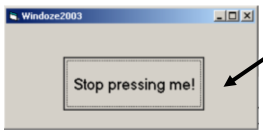
```
public class MyButtonListener implements ActionListener
{
    public void actionPerformed (ActionEvent e)
    {
        JButton b = (JButton) e.getSource();
        b.setLabel("Stop pressing me!");
    }
}
```



James Tam

4. The Event Listener Runs The Code To Handle The Event

```
public class MyButtonListener implements ActionListener
{
    public void actionPerformed (ActionEvent e)
    {
        JButton b = (JButton) e.getSource();
        b.setLabel("Stop pressing me!");
    }
}
```



James Tam

An Example Of Handling A Button Event

•Location of the example:

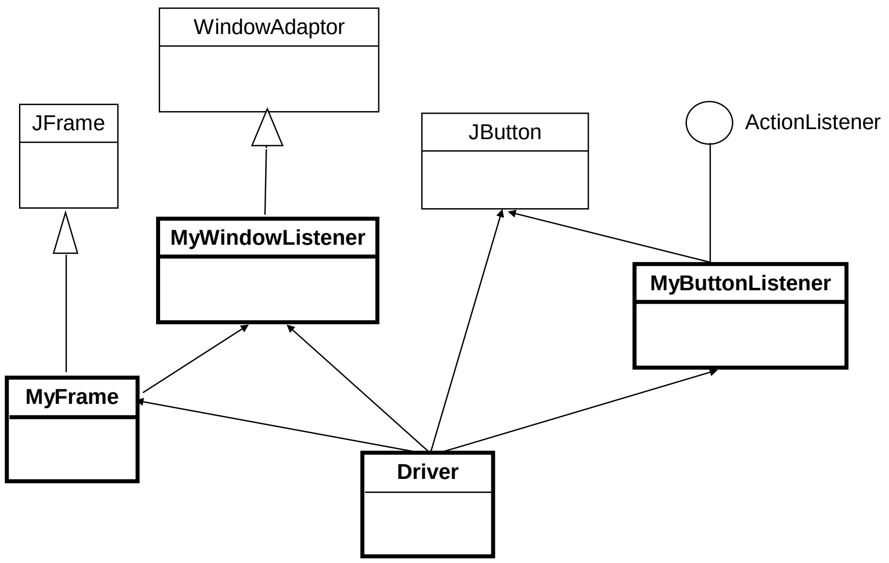
/home/233/examples/gui/three_button_events

OR

www.cpsc.ucalgary.ca/~tamj/233/examples/gui/three_button_events

James Tam

An Example Of Handling A Button Event (2)



James Tam

An Example Of Handling A Button Event: The Driver Class

```
import javax.swing.JButton;

public class Driver
{
    public static final int WIDTH = 300;
    public static final int HEIGHT = 200;
    public static void main (String [] args)
    {
        MyFrame aFrame = new MyFrame ();
        MyWindowListener aWindowListener = new MyWindowListener();
        aFrame.addWindowListener(aWindowListener);
        aFrame.setSize (WIDTH,HEIGHT);
```

James Tam

An Example Of Handling A Button Event: The Driver Class (2)

```
        JButton aButton = new JButton("Press me.");
        MyButtonListener aButtonListener = new
        MyButtonListener();
        aButton.addActionListener(aButtonListener);
        aFrame.add (aButton);
        aFrame.setVisible(true);
    }
}
```

James Tam

An Example Of Handling A Button Event: The ButtonListener Class

```
import javax.swing.JButton;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class MyButtonListener implements ActionListener
{
    public void actionPerformed (ActionEvent e)
    {
        JButton aButton = (JButton) e.getSource();
        aButton.setText("Stop pressing me!");
    }
}
```

James Tam

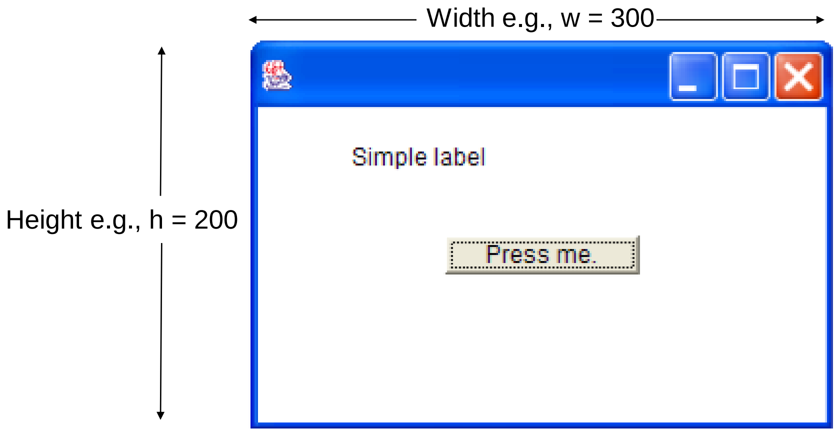
How To Handle The Layout Of Components

1. Manually set the coordinates yourself
2. Use one of Java’s built-in layout manager classes

James Tam

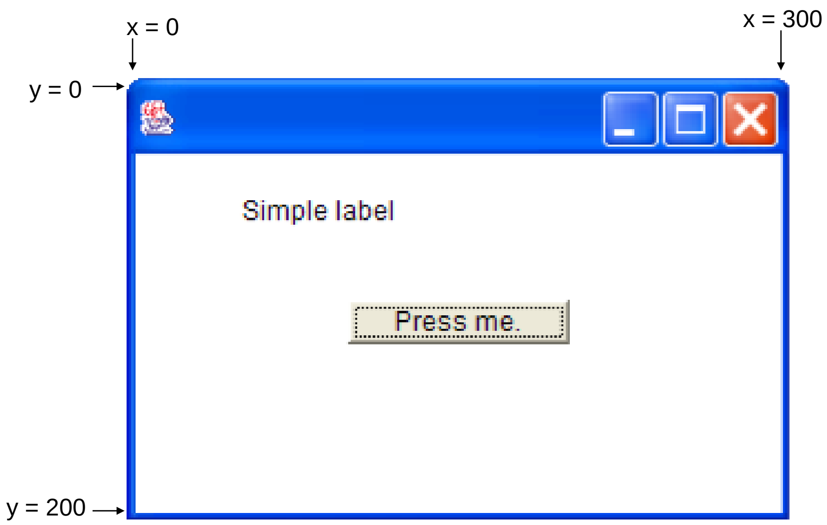
Layout Is Based On Spatial Coordinates

```
e.g. MyFrame my =new MyFrame ();
     my.setSize(300,200);
```



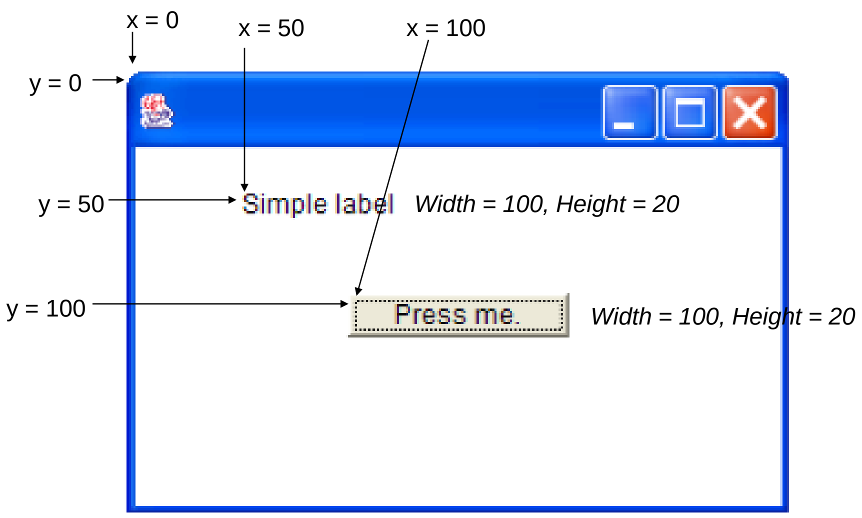
James Tam

Layout Is Based On Spatial Coordinates



James Tam

Coordinates Of Components: Relative To The Container



James Tam

Pitfall 2: Invisible Component

- Don’t forget that coordinates (0,0) are covered by the title bar of the frame.
- Components added at this location may be partially or totally hidden by the title bar.

James Tam

A Example Showing Manual Layout

•Location of the example:

/home/233/examples/gui/fourth_manual_layout

OR

www.cpsc.ucalgary.ca/~tamj/233/examples/gui/fourth_manual_layout

James Tam

An Example Showing Manual Layout: The Driver Class

```
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JFrame;

public class Driver
{
    public static final int WIDTH_FRAME = 300;
    public static final int HEIGHT_FRAME = 300;
    public static final int X_COORD_BUTTON = 100;
    public static final int Y_COORD_BUTTON = 100;
    public static final int WIDTH_BUTTON = 100;
    public static final int HEIGHT_BUTTON = 20;
    public static final int X_COORD_LABEL = 50;
    public static final int Y_COORD_LABEL = 50;
    public static final int WIDTH_LABEL = 100;
    public static final int HEIGHT_LABEL = 20;
```

James Tam

An Example Showing Manual Layout: The Driver Class (2)

```
    public static void main (String [] args) {
        JFrame aFrame = new JFrame ();
        aFrame.setLayout(null);
        aFrame.setSize (WIDTH_FRAME,HEIGHT_FRAME);
        JButton aButton = new JButton("Press me.");
        aButton.setBounds(X_COORD_BUTTON,
                              Y_COORD_BUTTON,
                              WIDTH_BUTTON,
                              HEIGHT_BUTTON);
        JLabel aLabel = new JLabel ("Simple label");
        aLabel.setBounds(X_COORD_LABEL,
                            Y_COORD_LABEL,
                            WIDTH_LABEL,
                            HEIGHT_LABEL);
        aFrame.add(aButton);
        aFrame.add(aLabel);
        aFrame.setVisible(true);
    }
}
```

James Tam

Components Effecting The State Of Other Components

•Location of the example:

/home/233/examples/gui/sixth_controls_affect_controls

OR

www.cpsc.ucalgary.ca/~tamj/233/examples/gui/sixth_controls_affect_controls

James Tam

Components Effecting The State Of Other Components: The Driver Class

```
public class Driver
{
    public static final int WIDTH = 800;
    public static final int HEIGHT = 600;
    public static void main (String [] args)
    {
        MyFrame aFrame = new MyFrame ();
        aFrame.setSize(WIDTH,HEIGHT);
        aFrame.setVisible(true);
    }
}
```

James Tam

Components Effecting The State Of Other Components: Class MyFrame

```
public class MyFrame extends JFrame
{
   private JLabel aLabel1;
   private JButton aButton;
   private MyButtonListener aButtonListener;
```

James Tam

Components Effecting The State Of Other Components: Class MyFrame (2)

```
public MyFrame ()
{
    MyWindowListener aWindowListener = new MyWindowListener ();
    JLabel aLabel2;
    addWindowListener(aWindowListener);
    aLabel1 = new JLabel("Label 1");
    aLabel2 = new JLabel("Label 2");
    aLabel1.setBounds(100,100,100,30);
    aLabel2.setBounds(300,100,100,30);
```

James Tam

Components Effecting The State Of Other Components: Class MyFrame (3)

```
        aLabel = new JLabel("Simple label");
        aLayout = new GridBagLayout();
        setLayout(aLayout);     // Calling method of super class.
        addWidget(aLabel, 0, 0, 1, 1);
        addWidget(himButton, 0, 1, 1, 1);
        addWidget(herButton, 0, 2, 1, 1);
    }
```

James Tam

Components Effecting The State Of Other Components: Class MyFrame (4)

```
     aButtonListener = new MyButtonListener();
     aButton = new JButton("Press for multiple effects");
     aButton.addActionListener(aButtonListener);
     aButton.setBounds(150,300,200,50);
     add(aLabel1);
     add(aLabel2);
     add(aButton);
     setLayout(null);
   }

   public JButton getAButton () { return aButton; }
   public JLabel getLabel1 () { return aLabel1; }
   // JT: Note that label2 has no accessor – not the effect in Button listener
}
```

James Tam

Components Effecting The State Of Other Components: Class MyFrame (5)

```
   public class MyWindowListener extends WindowAdapter
   {
       public void windowClosing (WindowEvent e)
       {
          JFrame f = (JFrame) e.getWindow();
          f.setTitle("Closing window…");
           try {
                Thread.sleep(3000);
           }
           catch (InterruptedException ex) {
                System.out.println("Pausing of program was interrupted");
           }
          f.setVisible(false);
          f.dispose();
       }
}
```

James Tam

Components Effecting The State Of Other Components: Class ButtonListener

```
public class MyButtonListener implements ActionListener
{
   public void actionPerformed (ActionEvent e)
   {
              JButton aButton = (JButton) e.getSource();
              MyFrame aFrame = (MyFrame)
                   aButton.getRootPane().getParent();
              JLabel aLabel1 = aFrame.getLabel1();  // Has accessor
              aLabel1.setText("Effect1");

              JLabel aLabel2 = null;                  // No accessor
              Container aContainer = aFrame.getContentPane();
              Component aComponent = aContainer.getComponent(1)
              if (aComponent instanceof JLabel)
                     aLabel2 = (JLabel) aComponent;
```

James Tam

Components Effecting The State Of Other Components: Class ButtonListener

```
              JLabel aLabel2 = null;                  // No accessor
              Container aContainer = aFrame.getContentPane();
              Component aComponent = aContainer.getComponent(1)
              if (aComponent instanceof JLabel)
                     aLabel2 = (JLabel) aComponent;

              if (aLabel2 != null)
                     aLabel2.setText("Effect2");
   }
}
```

James Tam

Important Concepts And Terms

- GUI
- Event-based software
- Registering listeners
- Call back (event-to-code)

James Tam

References

- Books:
  - "*Java Swing*" by Robert Eckstein, Marc Loy and Dave Wood (O'Reilly)
  - "*Absolute Java*" (4th Edition) by Walter Savitch (Pearson)
  - "*Java: How to Program*" (6th Edition) by H.M. Deitel and P.J. Deitel (Pearson)
- Websites:
  - Java API specifications: http://download.oracle.com/javase/7/docs/api/
  - Java tutorials: http://download.oracle.com/javase/tutorial/uiswing/
  - Java tutorial (layout): http://docs.oracle.com/javase/tutorial/uiswing/layout/using.html

James Tam

You Should Now Know

- The difference between traditional and event driven software
- How event-driven software works (registering and notifying event listeners)
- How some basic Swing controls work
  - Example: Capturing common events for the controls such as a button press, Window events
- How to layout components manually using a coordinate system

James Tam