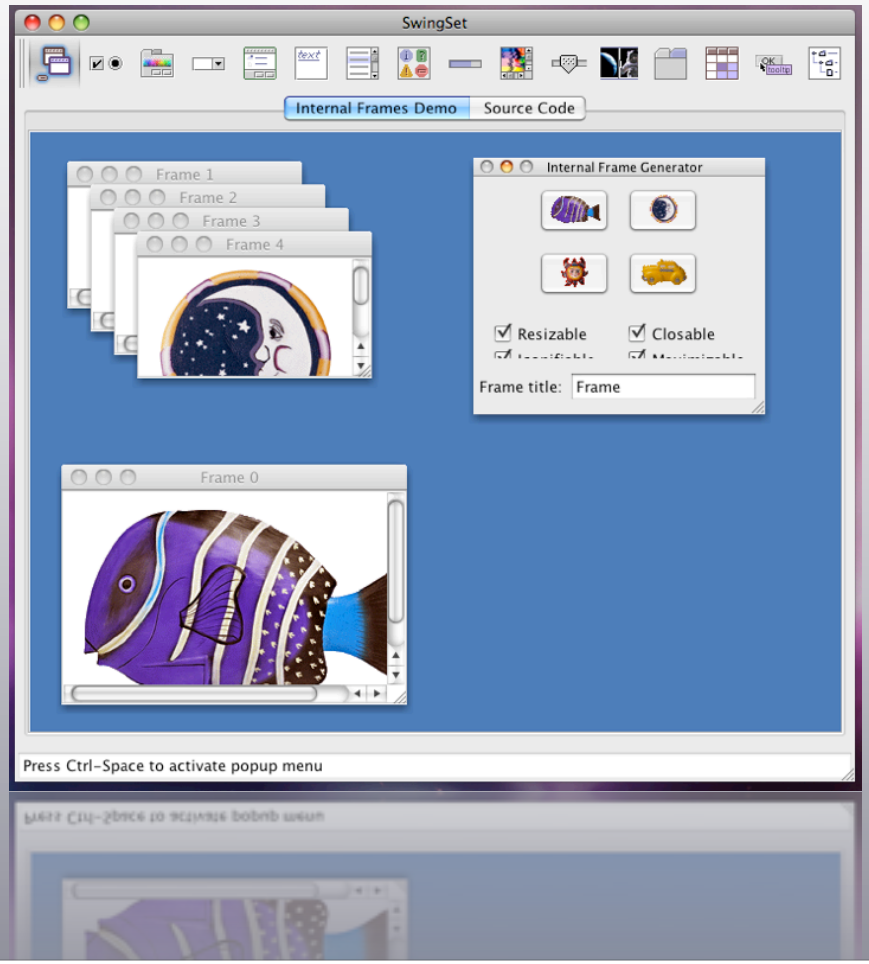
Java as a Second Language

Week 5: GUIs

CS3773
Simson Garfinkel



Making GUIs with Java & Swing

GUIs in Java

AWT - Abstract Window Toolkit

- import java.awt.*;
- Introduced with Java 1.0
- Used *native implementations* for windows, buttons, text objects, menues, etc.
- A disaster: buttons were different on every platform!
- Included with Internet Explorer

JFC/Swing

- Introduced with Java 1.2
- Microsoft refused to adopt (because it hated Sun by then)

SWT - Standard Widget Toolkit (part of Eclipse)

& others

javax.swing
JFRAME

java.lang.Object
  java.awt.Component
    java.awt.Container
      java.awt.Window
        java.awt.Frame
          javax.swing.JFrame

Everything that appears on the screen is drawn inside a JFrame

Frames have:

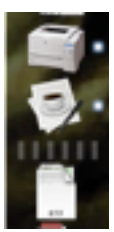
- contentPane
- Other stuff

http://java.sun.com/j2se/1.5.0/docs/api/javax/swing/JFrame.html

http://java.sun.com/docs/books/tutorial/uiswing/components/rootpane.html

Let's look at a simple Swing Program:

```
import javax.swing.*;

public class FirstGUI {
   public static void main(String[] args) {
       JFrame frame = new JFrame();
       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   }
}
```

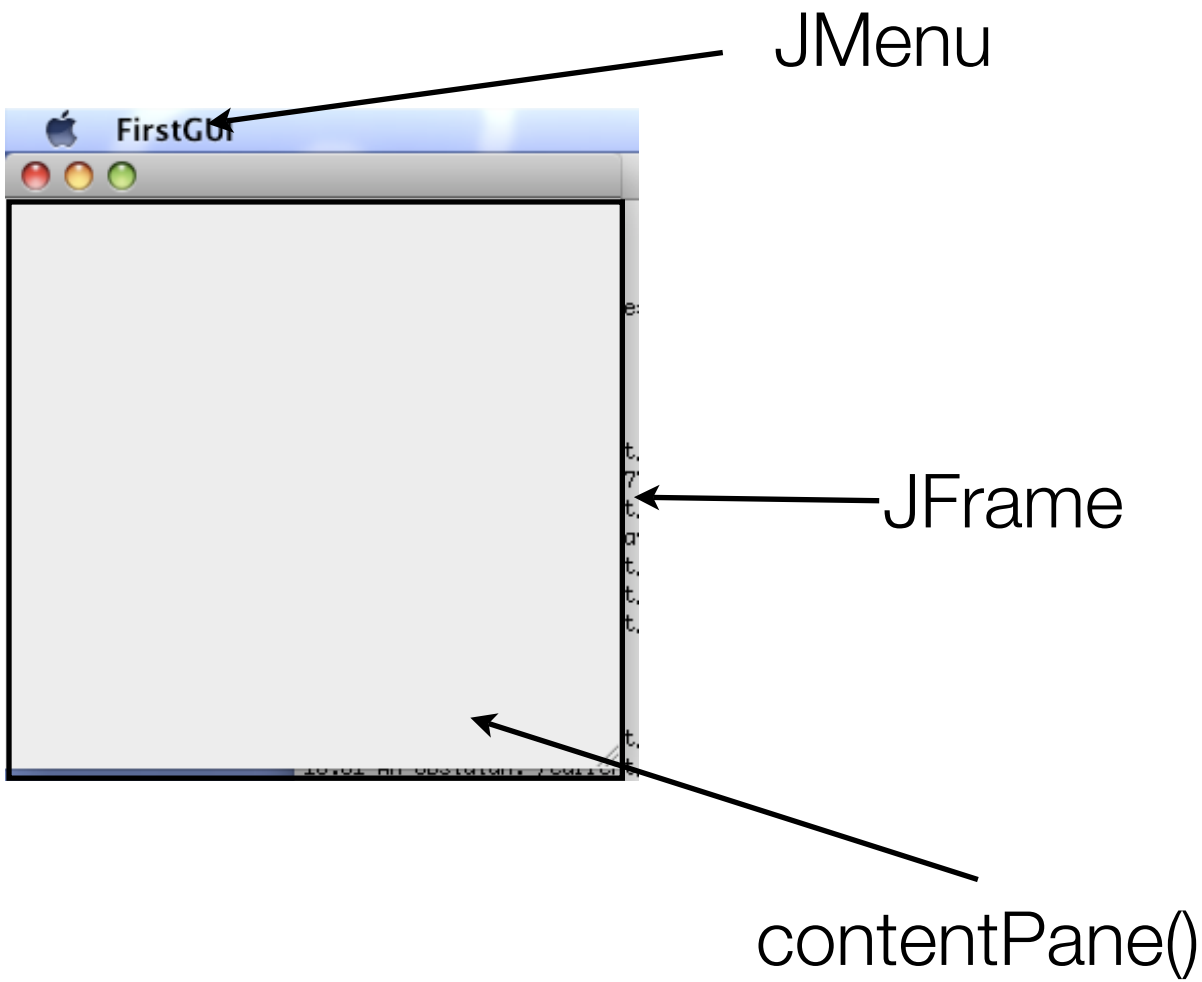


Let's look at a simple Swing Program:

```
import javax.swing.*;

public class FirstGUI {
   public static void main(String[] args) {
       JFrame frame = new JFrame();
       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       frame.setVisible(true);
   }
}
```

Let's look at a simple Swing Program:

```
import javax.swing.*;

public class FirstGUI {
   public static void main(String[] args) {
       JFrame frame = new JFrame();
       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       frame.setSize(300,300);
       frame.setVisible(true);
   }
}
```


JMenu
FirstGUI
JFrame
contentPane()

Swing gives us a lot "for free."

Window resizing

Window redrawing

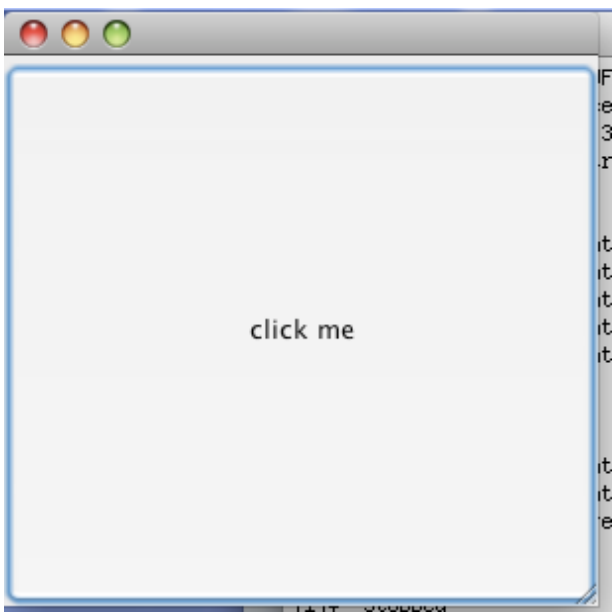
Menu with "quit"

But it's really doing much more.


```
import javax.swing.*;

public class FirstGUI {
   public static void main(String[] args) {
       JFrame frame = new JFrame();
       frame.setDefaultCloseOperation(
         JFrame.EXIT_ON_CLOSE);
       frame.setSize(300,300);
       frame.setVisible(true);
   }
}
```

Add a clickable button.

```
public class FirstGUI {
    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JButton button = new JButton("click me");
        frame.getContentPane().add(button);
        frame.setSize(300,300);
        frame.setVisible(true);
    }
}
```



Swing components allow HTML for titles...

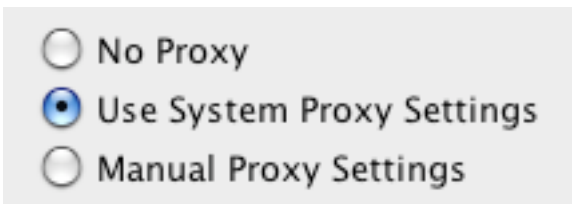
```
JButton button =
new JButton("<html><b>bold</b><br><i>itallic</i></html>");
```

javax.swing
JButton

java.lang.Object
  java.awt.Component
    java.awt.Container
      javax.swing.JComponent
        javax.swing.AbstractButton
          javax.swing.JButton

Terminology:

- *title* or *text* — what's displayed on the button
- *enabled/disabled* — Can the button be clicked?
- *pressed* — the button is being clicked on
- *selected* — is the the *selected* state
- *icon* — image displayed by the button
- *ButtonGroup* — a set of mutually exclusive buttons

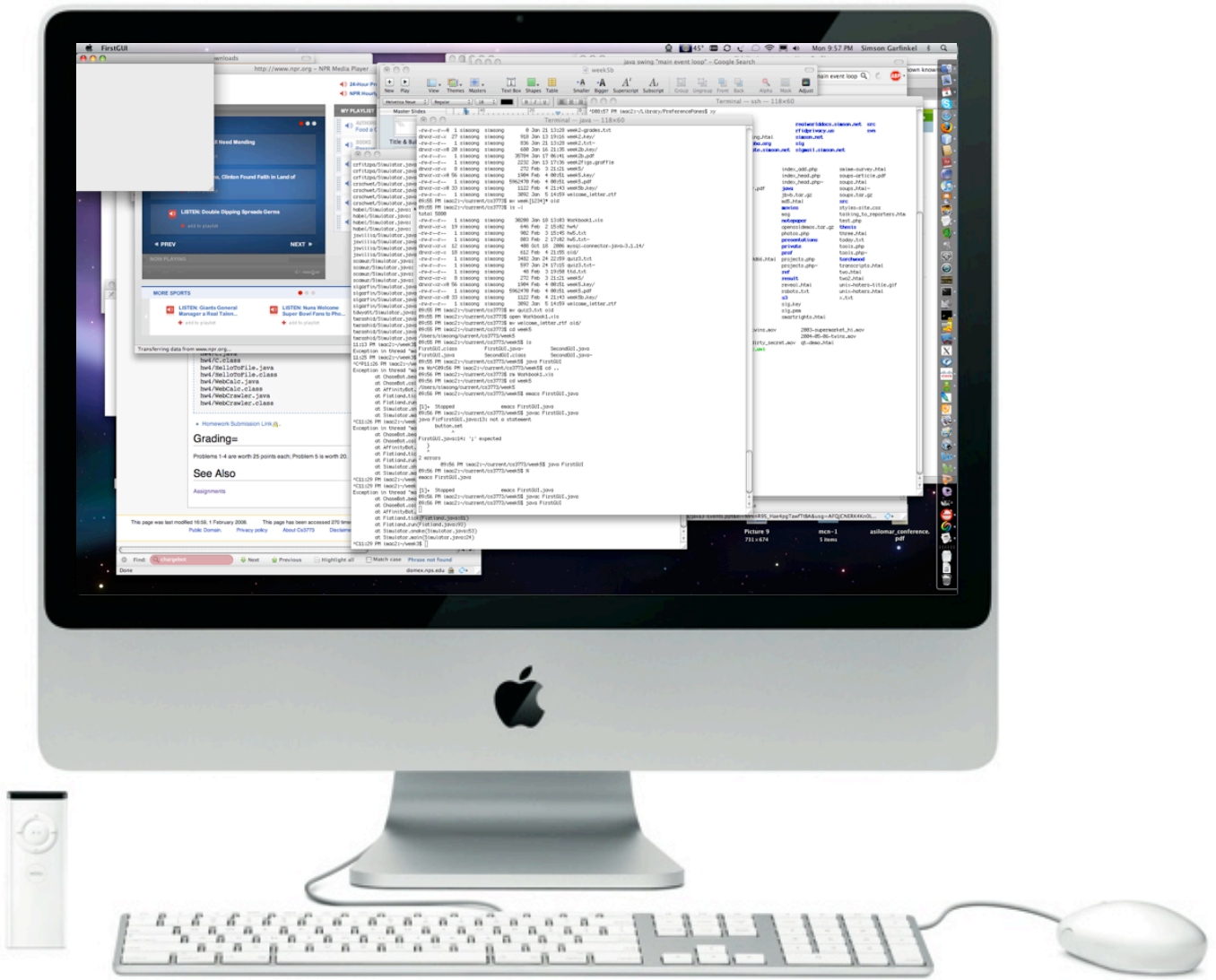


Reference:

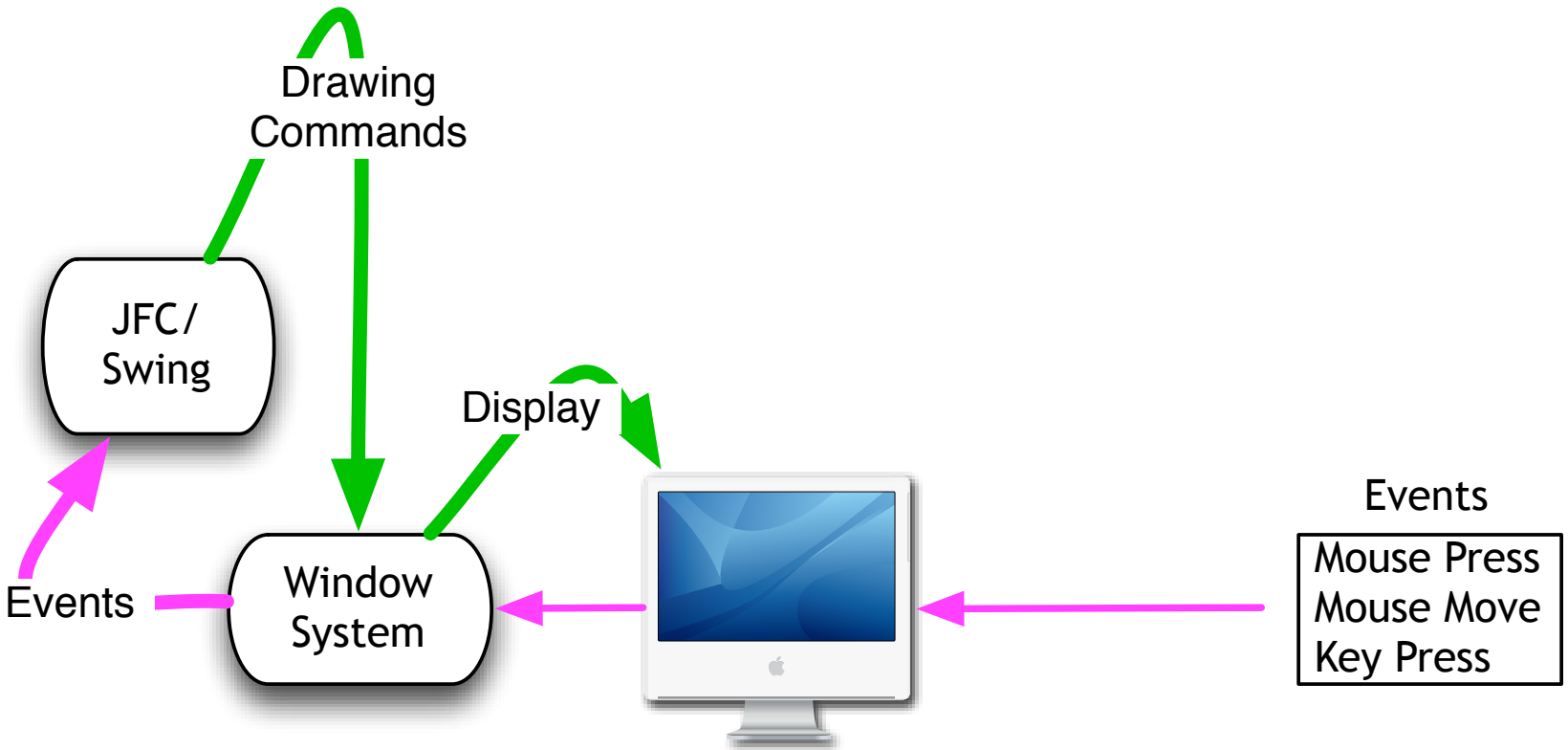
http://java.sun.com/docs/books/tutorial/uiswing/components/button.html

http://java.sun.com/j2se/1.5.0/docs/api/javax/swing/JButton.html

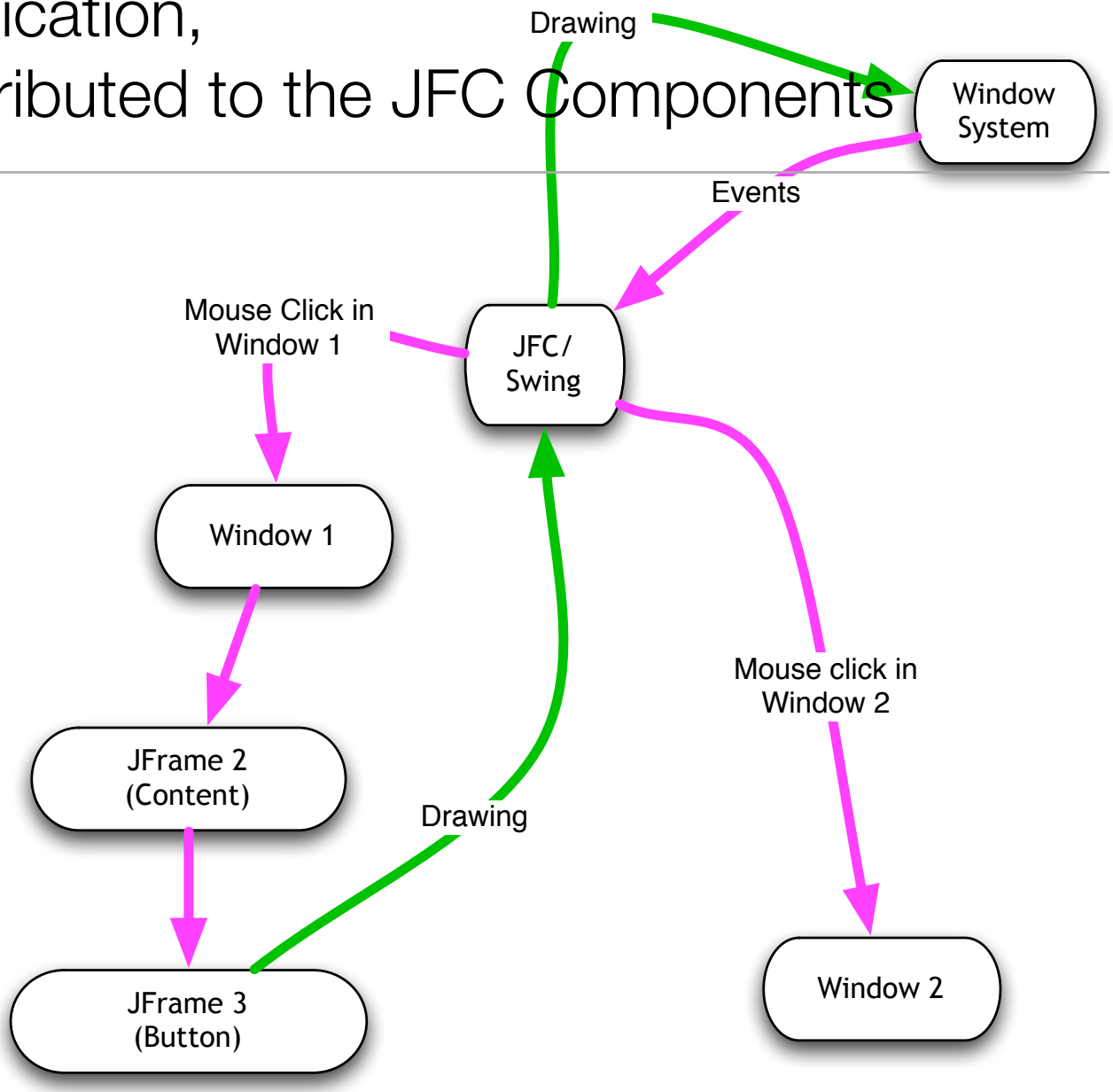
How does JFC/Swing make an application work?



Your JFC/Swing program communicates with the Window System to the outside world



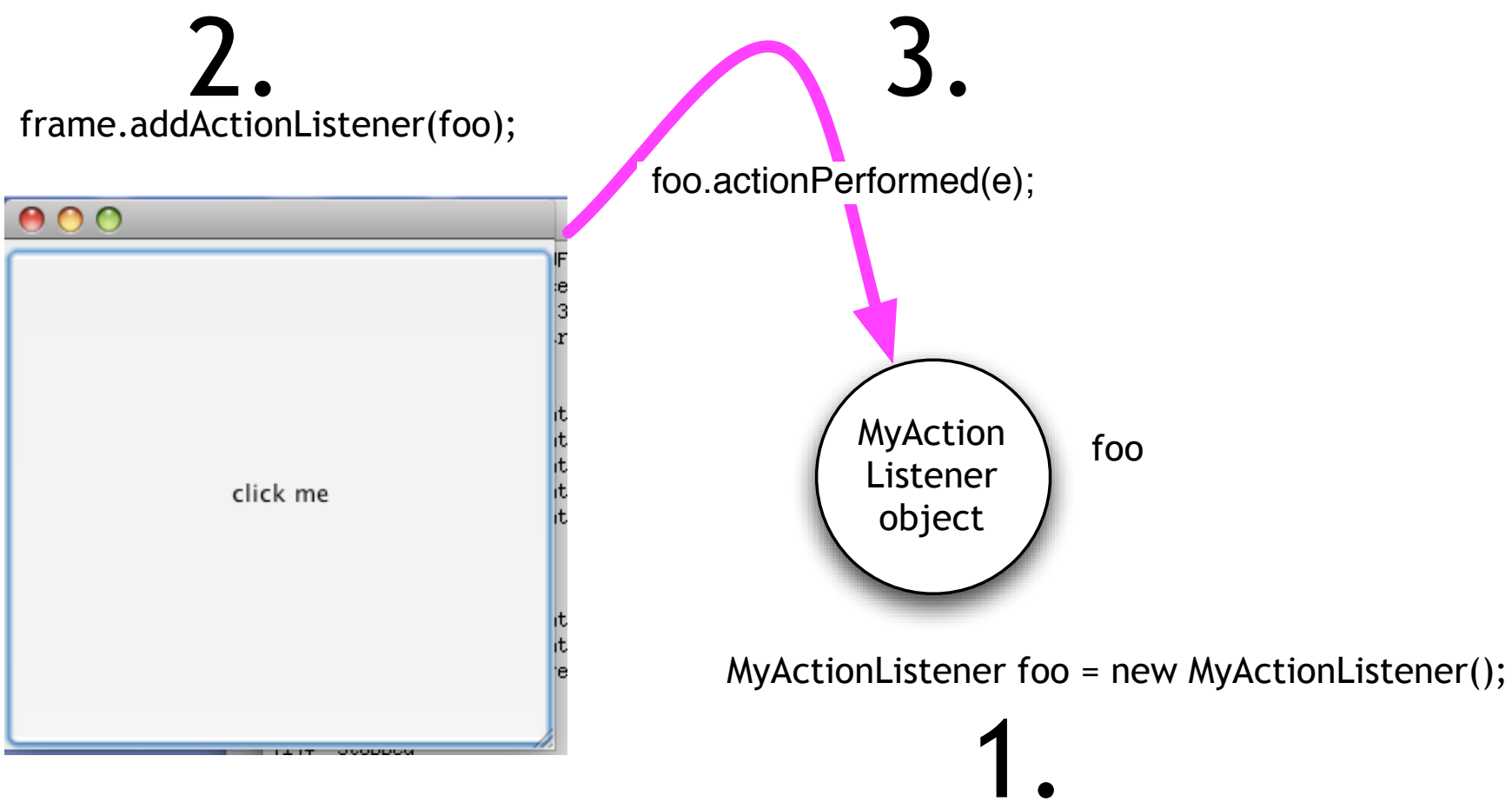
Inside the Application, events are distributed to the JFC Components



JFC/Swing provides a "main event loop"

```
main(){
  Initialize variables.
  Create the user interface.
  while(true){
      e = getNextEvent();
      switch(e){
        -- mouse events
        -- keyboard events
        -- timer events
        -- network events.
        -- program exit events.
      }
  }
```

Event Listeners: Objects whose method's are called when an event happens



javax.swing
JEvent

JEvent is a package that provides Interfaces and Classes

Listeners — Interfaces for catching events

- MenuListener — listen for menu events
- ActionListener — Listens for button presses
- MouseListener — Mouse clicks
- MouseMotionListener — Mouse Motion

Classes — Specific events

- MenuEvent

NOTE: Some events are still in **java.awt.event:**

- java.awt.event.MouseEvent

See:

http://java.sun.com/j2se/1.5.0/docs/api/javax/swing/event/package-summary.html
http://java.sun.com/docs/books/tutorial/uiswing/events/

Here is what the code looks like:

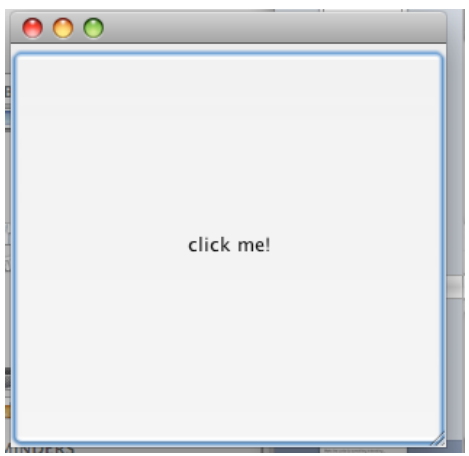
```
import javax.swing.*;
import java.awt.event.*;

class MyActionListener implements ActionListener {
    public void actionPerformed(ActionEvent e){
    System.out.println("clicked!");
    }
}

public class ThirdGUI  {
    static int times_clicked = 0;
    public static void main(String[] args) {
      JFrame frame = new JFrame();
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      JButton button = new JButton("click me!");
      button.addActionListener(new MyActionListener());
      frame.getContentPane().add(button);
      frame.setVisible(true);
      frame.setSize(300,300);
   }
}
```

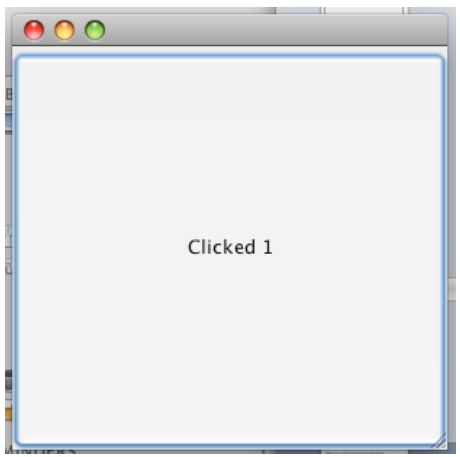
Make the code actually do something...

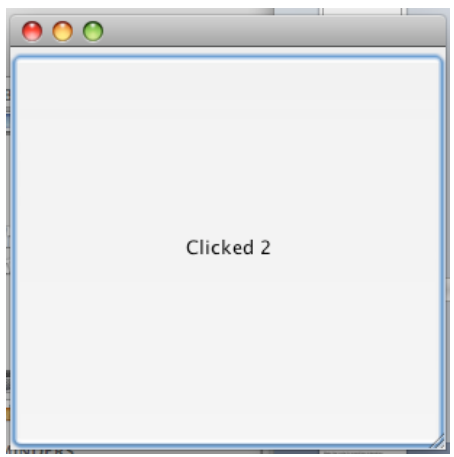
```
class MyActionListener implements ActionListener {
    public void actionPerformed(ActionEvent e){
        ThirdGUI.button.setText("nosmis");
    }
}

public class ThirdGUI  {
    static int times_clicked = 0;
    static JButton button = null;
    public static void main(String[] args) {
      JFrame frame = new JFrame();
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      button = new JButton("click me!");
      button.addActionListener(new MyActionListener());
      frame.getContentPane().add(button);

      frame.setVisible(true);
      frame.setSize(300,300);
   }
}
```

Make the code do something interesting...

```
class MyActionListener implements ActionListener {
    public void actionPerformed(ActionEvent e){
        FourthGUI.times_clicked++;
        FourthGUI.button.setText("Clicked "+FourthGUI.times_clicked);
    }
}
```







Listener API

Listeners must implement Java Listener interface

- ActionListener
- KeyListener
- MouseListener
- MouseMotionListener
- WindowListener

MouseListener:

- mouseClicked()
- mouseEntered()
- mouseExited()
- mousePressed()
- mouseReleased

Anonymous classes:
Allow you to create a class with a single instance

```
public class FifthGUI  {
    static int times_clicked = 0;
    static JButton button = null;
    public static void main(String[] args) {
      JFrame frame = new JFrame();
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      button = new JButton("click me!");
      button.addActionListener(new ActionListener() {
              public void actionPerformed(ActionEvent e){
                  times_clicked++;
                  button.setText("Clicked "+times_clicked);
              }
          });
      frame.getContentPane().add(button);
      frame.setVisible(true);
      frame.setSize(300,300);
   }
}
```

Advantages of anonymous classes:

```
button.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e){
            times_clicked++;
            button.setText("Clicked "+times_clicked);
        }
    });
```

Less code!

No need to come up with a name or put code in a file.

Can easily access instance variables of containing class.