Introduction

• Up till now you have written programs that communicate with the end user through a text-based interface
  — Using System.out for output
  — Using Keyboard for input.
• Java provides two sets of facilities for developing GUIs:
  — The Abstract Window Toolkit (AWT): package java.awt
  — Swing: package javax.swing
Swing is a package that lets you create applications that use GUI instead of console interface.

Components and containers

• A component is any GUI element, such as a window, button or label.
• A container is a type of component that has the purpose of containing other components.
• Types of containers:
  — Top-level containers: Every Swing program contains at least one top-level container (e.g. JFrame, JDialog or JApplet). Top-level containers cannot be added to other containers.
  — Intermediate containers: used to group components so that they can be handled as a single component (e.g JPanel, JTabbedPane).
  — Atomic components (basic controls): cannot contain other components (e.g JButton, JTextField).

Containers - Layout

• Each container has a layout manager
  — Determines the size, location of contained widgets.
• To use layout managers, you have to import java.awt.*.
• Setting the current layout of a container: void setLayout(LayoutManager lm)
• LayoutManager implementing classes:
  — BorderLayout
  — BoxLayout
  — FlowLayout
  — GridLayout
Events, Listeners

- A program needs to react to the user's actions

- Examples:
  - When the user presses a button
  - When the user closes the program

- Swing defines all sorts of Listener interfaces
  - E.g.: ActionListener, MouseMotionListener, WindowListener, ...

public interface ActionListener extends EventListener {
  public void actionPerformed(ActionEvent e);
}

public interface MouseMotionListener extends EventListener {
  public void mouseDragged(MouseEvent e);
  public void mouseMoved(MouseEvent e);
}

Events, Listeners (cont.)

- A listener is an object that implements a listener interface

- If we need to react to an event on a component we register a listener object with that component.

  E.g.: addActionListener() registers an action listener with its receiver:
  
  JButton button = new JButton();
  ActionListener listener = ...;
  button.addActionListener(listener);

- When an event occurs, all registered listeners are notified
  - The appropriate listener method (e.g: actionPerformed()) is invoked
  - An object describing the event is passed as a parameter

Event Handling Demo: Code

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

public class Events implements ActionListener {
  public Events() {
    JFrame frame = new JFrame("Events");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.getContentPane().setLayout(new FlowLayout());
    JButton b = new JButton("Click me!");
    b.addActionListener(this);
    frame.getContentPane().add(b);
    frame.pack();
    frame.setVisible(true);
  }

  public void actionPerformed(ActionEvent e) {
    JOptionPane.showMessageDialog(null, "Thank you");
  }

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