# COP 3330, Spring 2013

# GUIs – III (Recap- Detailed Discussion on AWT and Swing API)

Instructor : Arup Ghosh

04-01-13

School of Electrical Engineering and Computer Science
University of Central Florida

There are two sets of Java APIs for graphics programming: AWT (Abstract Windowing Toolkit) and Swing.

AWT API was introduced in JDK 1.0. Most of the AWT components have become obsolete and should be replaced by newer Swing components.

Swing API, a much more comprehensive set of graphics libraries that enhances the AWT, was introduced as part of Java Foundation Classes (JFC) after the release of JDK 1.1. JFC, which consists of Swing, Java2D, Accessibility API, Internationalization, and Pluggable Look-and-Feel Support, was an add-on to JDK 1.1 but has been integrated into core Java since JDK 1.2.

- Other than AWT/Swing Graphics APIs provided in JDK, others have also provided Graphics APIs that work with Java, such as Eclipse's Standard Widget Toolkit (SWT), Google Web Toolkit (GWT), 3D Graphics API such as Java bindings for OpenGL (JOGL) and Java3D.

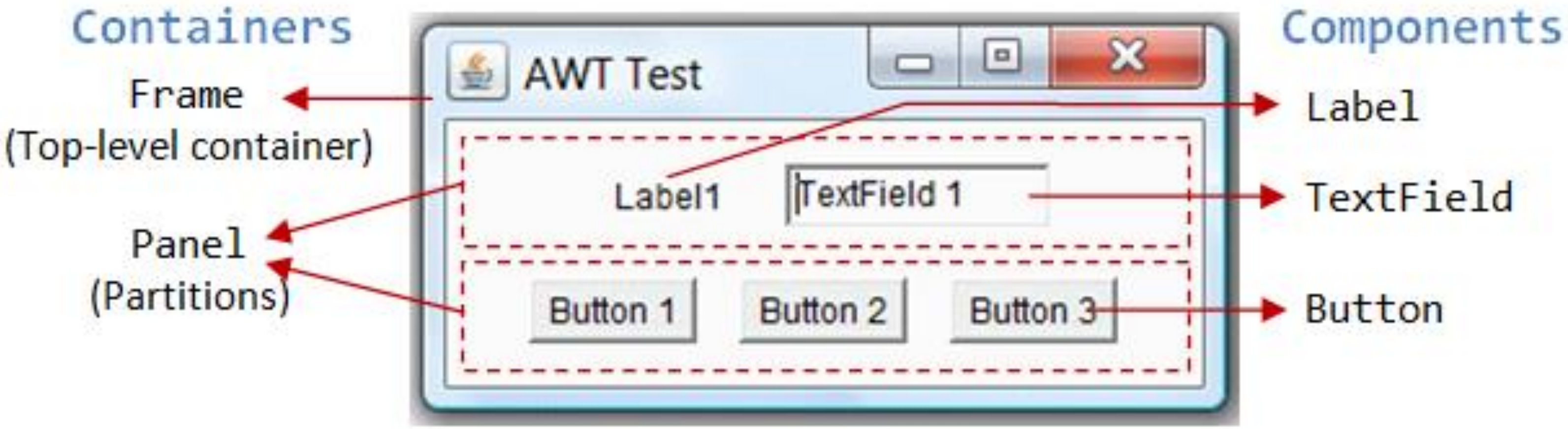
# Programming GUI with AWT

- AWT packages
  - AWT is huge! It consists of 12 packages (Swing is even bigger, with 18 packages as of JDK 1.7!). Fortunately, only 2 packages - java.awt and java.awt.event - are commonly-used.

- The java.awt package contains the *core* AWT graphics classes:
  - GUI Component classes (such as Button, TextField, and Label),
  - GUI Container classes (such as Frame, Panel, Dialog and ScrollPane),
  - Layout managers (such as FlowLayout, BorderLayout and GridLayout),
  - Custom graphics classes (such as Graphics, Color and Font).

- The java.awt.event package supports event handling:
  - Event classes (such as ActionEvent, MouseEvent, KeyEvent and WindowEvent),
  - Event Listener Interfaces (such as ActionListener, MouseListener, KeyListener and WindowListener),
  - Event Listener Adapter classes (such as MouseAdapter, KeyAdapter, and WindowAdapter).

# Components and Containers



- In a GUI program, a component must be kept in a container. You need to identify a container to hold the components. Every container has a method called add(Component c).

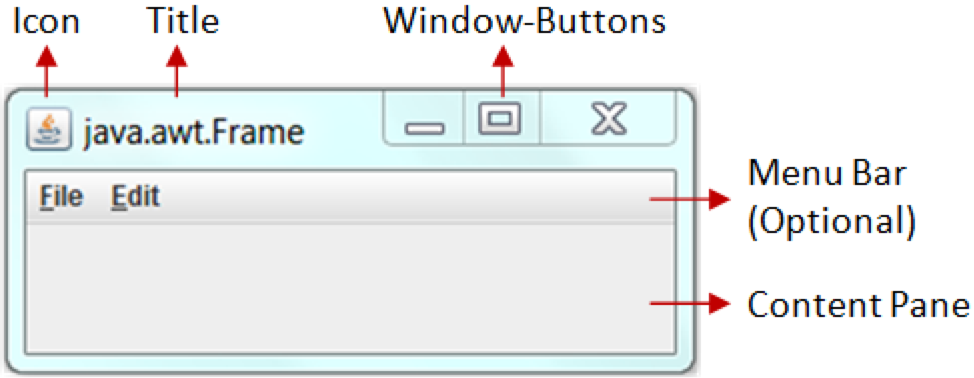
# AWT container classes

- Top-Level Containters:
  - Frame, Dialog and Applet

- Secondary Containers:
  - Panel and ScrollPane

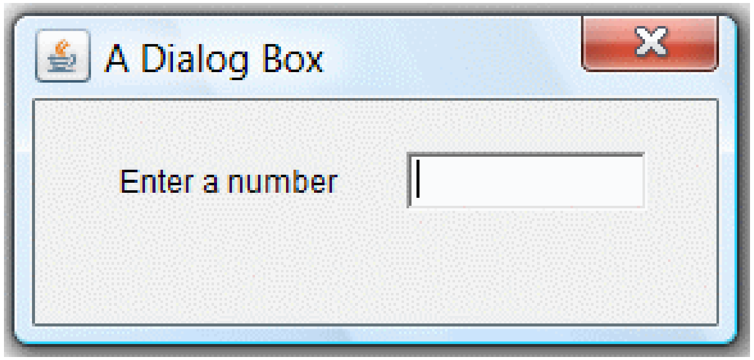
# Frame

- A Frame provides the "main window" for the GUI application, which has a title bar (containing an icon, a title, the minimize, maximize/restore-down and close buttons), an optional menu bar, and the content display area.



# Dialog

- An AWT Dialog is a "*pop-up window*" used for interacting with the users. A Dialog has a title-bar (containing an icon, a title and a close button) and a content display area, as illustrated.



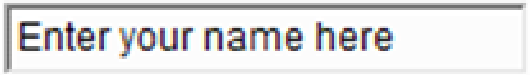
# Applet

- An AWT Applet (in package java.applet) is the top-level container for an applet, which is a Java program running inside a browser. Applet will be discussed in the later chapter.

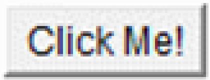
# Panel and ScrollPane

- A Panel is a rectangular box (partition) under a higher-level container, used to *layout* a set of related GUI components. See the above examples for illustration.
- Others, such as ScrollPane (which provides automatic horizontal and/or vertical scrolling for a single child component).

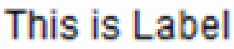
# AWT Component Classes


Enter your name here

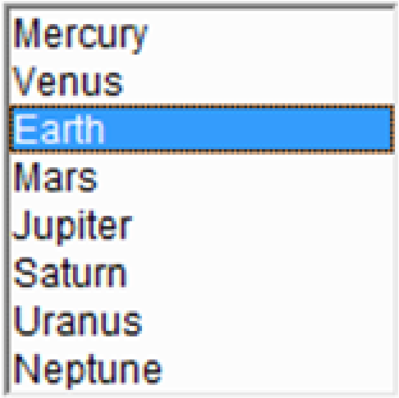
TextField



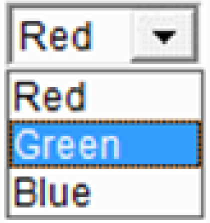
Button


This is Label

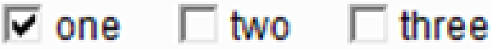
Label



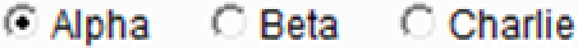
List



Choice


☑ one □ two □ three

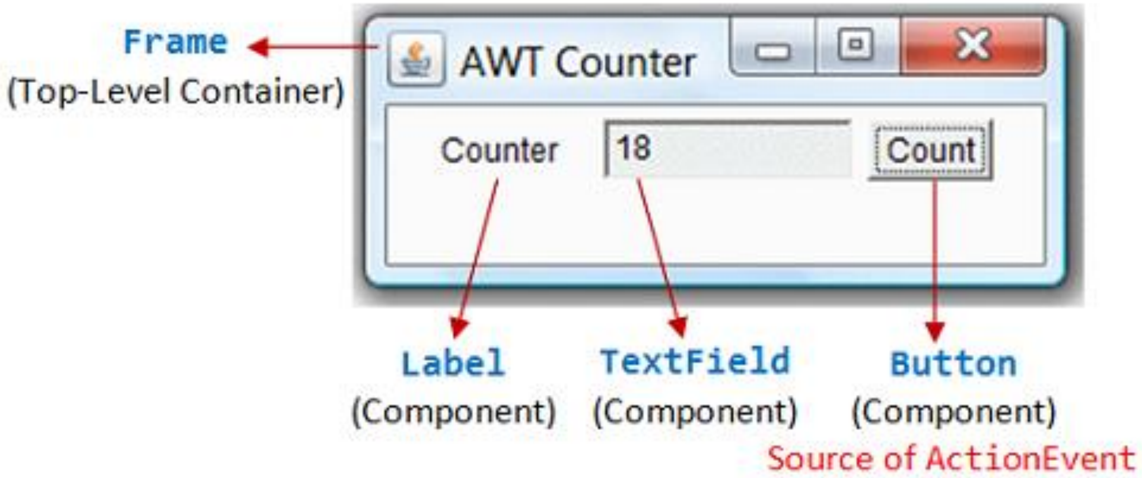
CheckBox


● Alpha ○ Beta ○ Charlie

CheckBoxGroup

# Example 1: AWTCounter

- Let's assemble some components together into a simple GUI counter program, as illustrated. It has a top-level containerFrame, which contains three components - a Label "Counter", a non-editable TextField to display the current count, and a "Count" Button. The TextField displays "0" initially.
- Each time you click the button, the counter's value increases by 1.



# Example 1: AWTCounter

- To exit this program, you have to close the CMD-shell (or press "control-c"); or push the red-square close button in Eclipse's Application Console. This is because we have yet to write the handler for the window's close button. We shall do that in the later example.