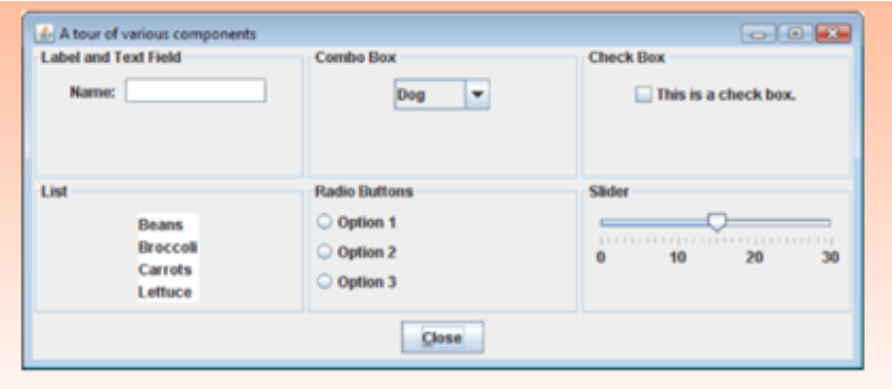
Ch 7 GUI Review

GUI Classes

Chapter covers: JLabel, JTextField, JComboBox, JCheckBox, JList, JRadioButton, JSlider, and JButton. All these are shown below… “Close” is a JButton.



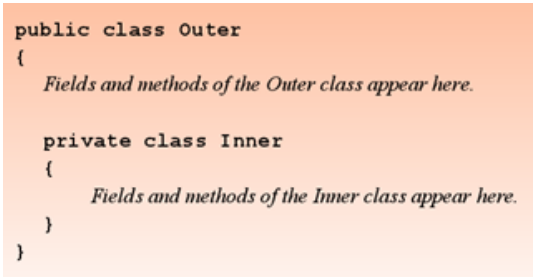
AWT = Abstract Windowing Toolkit, classes for drawing graphics and creating GUI’s.
Swing = an alternate library alongside AWT to build GUI’s and Applets

Code of interest in our textbook... I’ll **bold** the featured class:

- p 367 ShowWindow.java - create an empty window using **JFrame**
- p 369 SimpleWindow.java - using inheritance to extend **JFrame** (“more common technique”)
- p 373 KiloConverterWindow.java - adding **JPanel** of components to the “content pane” of a JFrame. Common methodology: add JLabel, JTextField, JButton to JPanel... then, add Jpanel to JFrame.
- p 385 ColorWindow.java - using the **Color** class to set fg and bg colors of GUI objects
- p 389 EventObjectWindow.java - a complete example handling GUI events using **ActionListener** and **ActionEvent**.
- p 414 MetricConverterWindow.java - a **JRadioButton** example
- p 420 ColorCheckBoxWindow.java - a **JCheckBox** example

ActionListeners

Capture GUI events. Typically implemented as private inner classes of GUI classes.



Typical sequence... 1) We use the ActionListener interface from AWT.

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

2) We implement it.

```
public class my Listener implements ActionListener {
   public void actionPerformed( ActionEvent e) {
      // your code to handle event here
   }
}
```

3) And then we register it with some GUI object, like a JButton.

```
…
JButton but = new JButton( "Example");
but.addActionListener( new myListener());
…
```

You can create a separate4 ActionListener for each GUI object, or share a listener by using the getSource() method for the ActionEvent.

Layout Managers

Use a layout manager to a control the position of components (JButton, JTextField, etc) within a container (like JFrame or JPanel). Three classes:

- **FlowLayout** - arranges components in rows; default for JPanel
- **BorderLayout** - arranges components in regions: north, south, east, west, and center; default for JFrame
- **GridLayout** - arranges components into a grid with rows and columns

PS - These guys are often very finicky and difficult to get just right.

Debugging!

Sometimes, your debugger won't help in a GUI application. Use good, ole print statements!