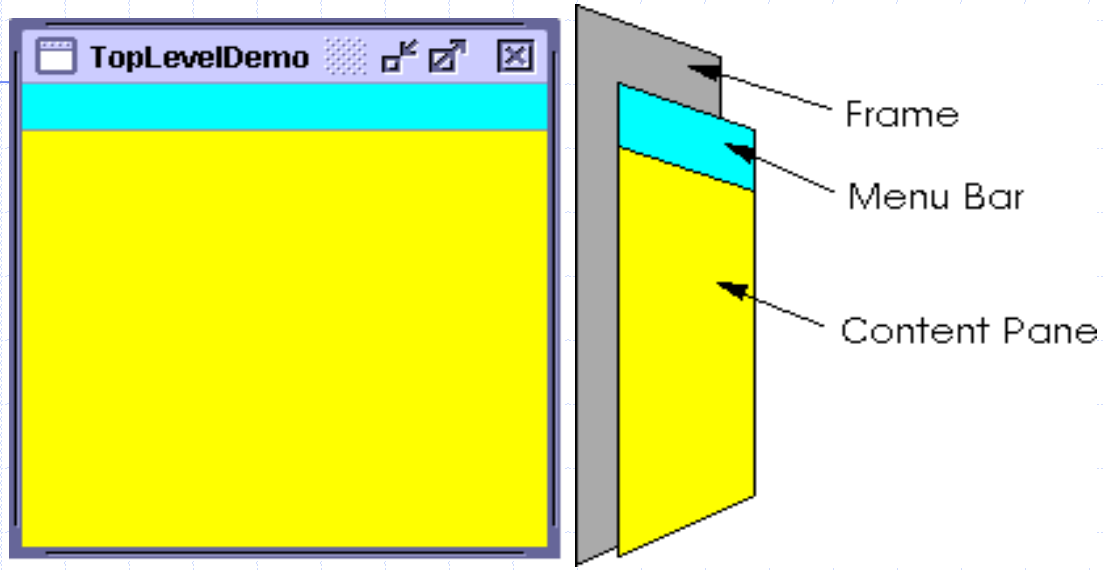
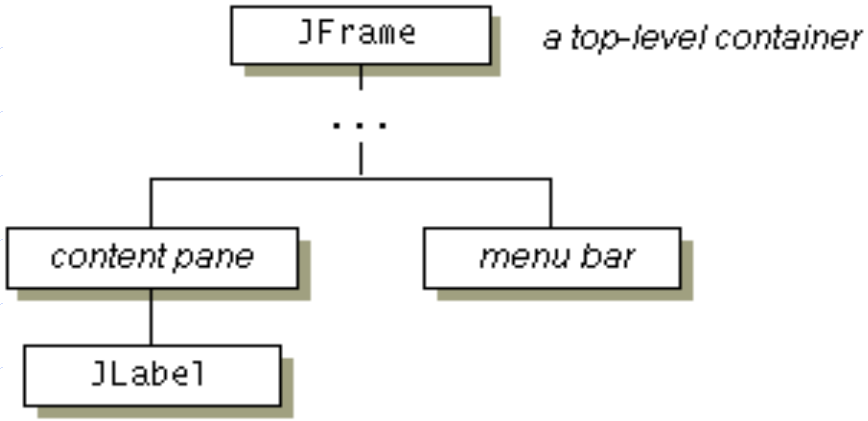
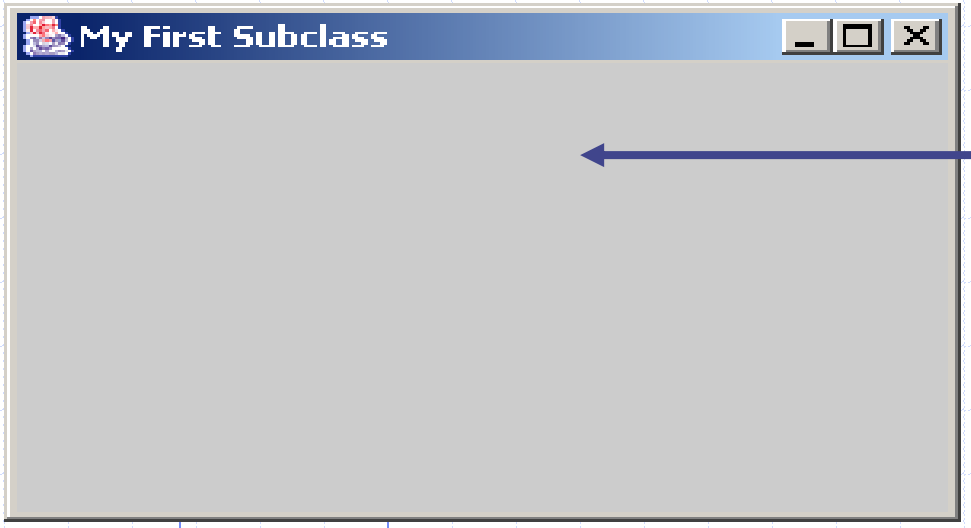
Use of Top Level Container





Reminder: The Content Pane of a Frame

- The content pane is where we put GUI objects such as buttons, labels, scroll bars, and others.
- We access the content pane by calling the frame's **getContentPane** method.



Adding Components to the Content Pane

```
myFrame.getContentPane().add(myLabel,BorderLayout.CENTER);
```

```
JPanel contentPane = new JPanel(new BorderLayout());
contentPane.setBorder(someBorder);
contentPane.add(someComponent, BorderLayout.CENTER);
contentPane.add(anotherComponent, BorderLayout.PAGE_END);
//Make it the content pane.
contentPane.setOpaque(true);
myFrame.setContentPane(contentPane);
```

Adding Components to the Content Pane

```
myFrame.getContentPane().add(myLabel,BorderLayout.CENTER);
```

```
JPanel contentPane = myFrame.getContentPane();
contentPane.setBorder(someBorder);
contentPane.add(someComponent, BorderLayout.CENTER);
contentPane.add(anotherComponent, BorderLayout.PAGE_END);
//Make it the content pane.
contentPane.setOpaque(true);
```

JLabel

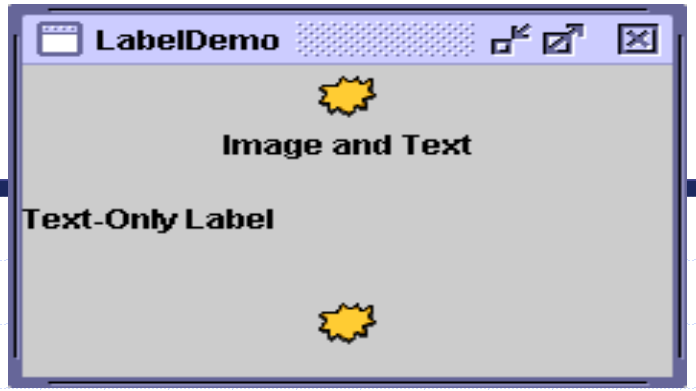
- We use a JLabel object to display a label.
- A label can be a text or an image.
- When creating an image label, we pass ImageIcon object instead of a string.

```
JLabel textLabel = new JLabel("Please enter your name");
contentPane.add(textLabel);

JLabel imgLabel = new JLabel(new ImageIcon("cat.gif"));
contentPane.add(imgLabel);
```

Use of Labels

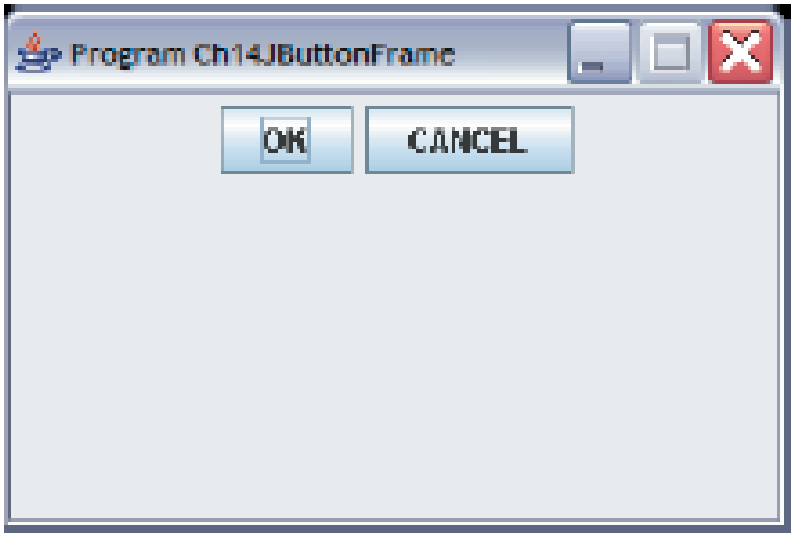
```
ImageIcon icon = createImageIcon("images/middle.gif");
label1 = new JLabel("Image and Text", icon,JLabel.CENTER);
//Set the position of the text, relative to the icon:
label1.setVerticalTextPosition(JLabel.BOTTOM);
label1.setHorizontalTextPosition(JLabel.CENTER);
label2 = new JLabel("Text-Only Label");
label3 = new JLabel(icon);
```



Placing a Button

- A **JButton** object is a GUI component that represents a pushbutton.
- Here's an example of how we place a button with **FlowLayout**.

```
contentPane.setLayout(
        new FlowLayout());
okButton
    = new JButton("OK");
cancelButton
    = new JButton("CANCEL");
contentPane.add(okButton);
contentPane.add(cancelButton);
```



Handling Text

- The Swing GUI classes **JLabel**, **JTextField**, and **JTextArea** deal with text.

- A **JLabel** object displays uneditable text (or image).
- A **JTextField** object allows the user to enter a single line of text.
- A **JTextArea** object allows the user to enter multiple lines of text. It can also be used for displaying multiple lines of uneditable text.

JTextField

- We use a JTextField object to accept a single line to text from a user. An action event is generated when the user presses the ENTER key.
- The getText method of JTextField is used to retrieve the text that the user entered.

```
JTextField input = new JTextField( );
input.addActionListener(eventListener);
contentPane.add(input);
```

TextFrame2

JLabel
(with an image)

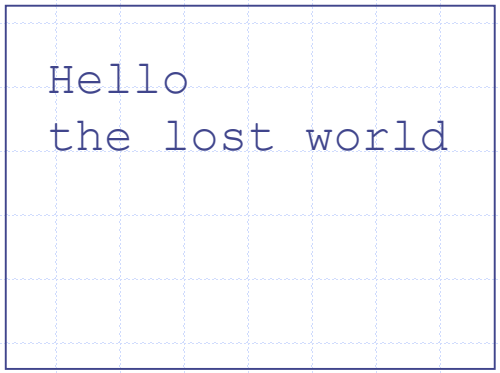
JLabel
(with a text)

JTextField

JTextArea

- We use a JTextArea object to display or allow the user to enter multiple lines of text.
- The setText method assigns the text to a JTextArea, replacing the current content.
- The append method appends the text to the current text.

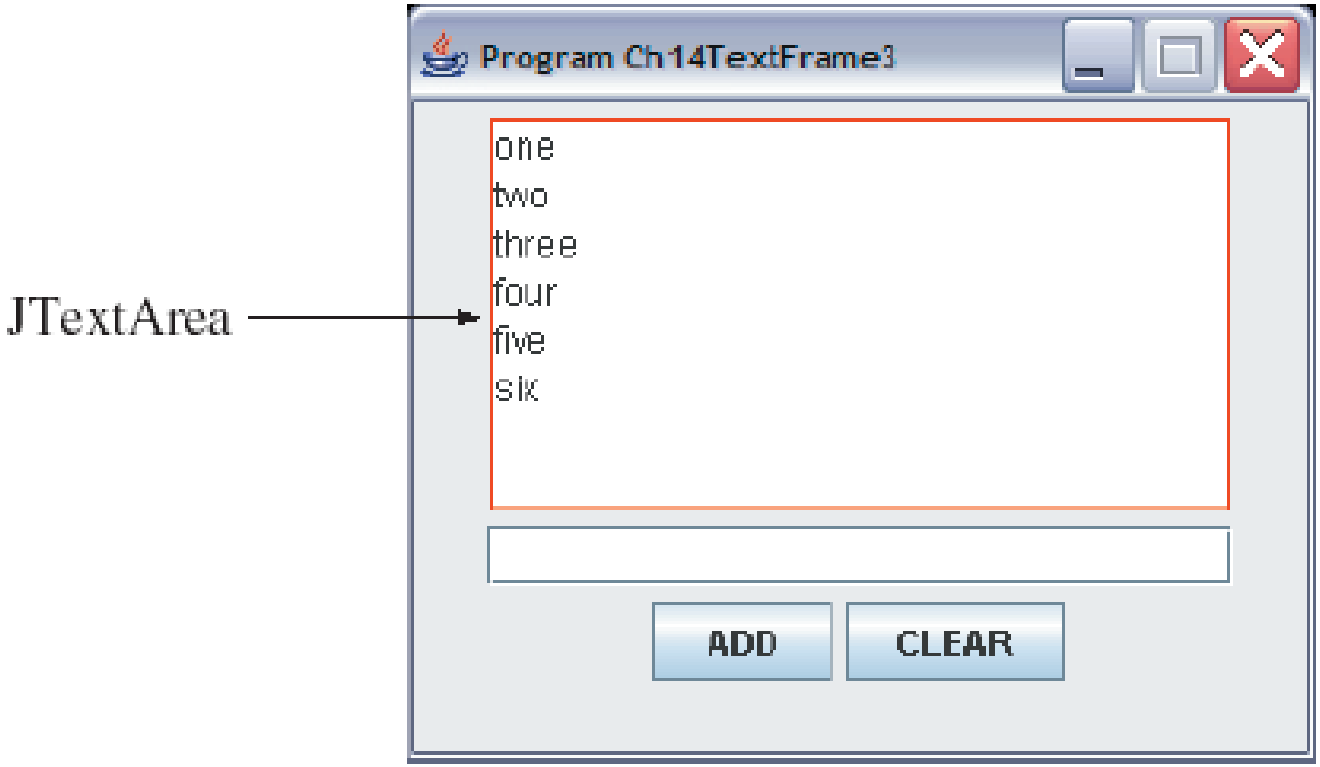
```
JTextArea textArea
        = new JTextArea( );
. . .
textArea.setText("Hello\n");
textArea.append("the lost ");
textArea.append("world");
```



JTextArea

TextFrame3

- The state of a **TextFrame3** window after six words are entered.



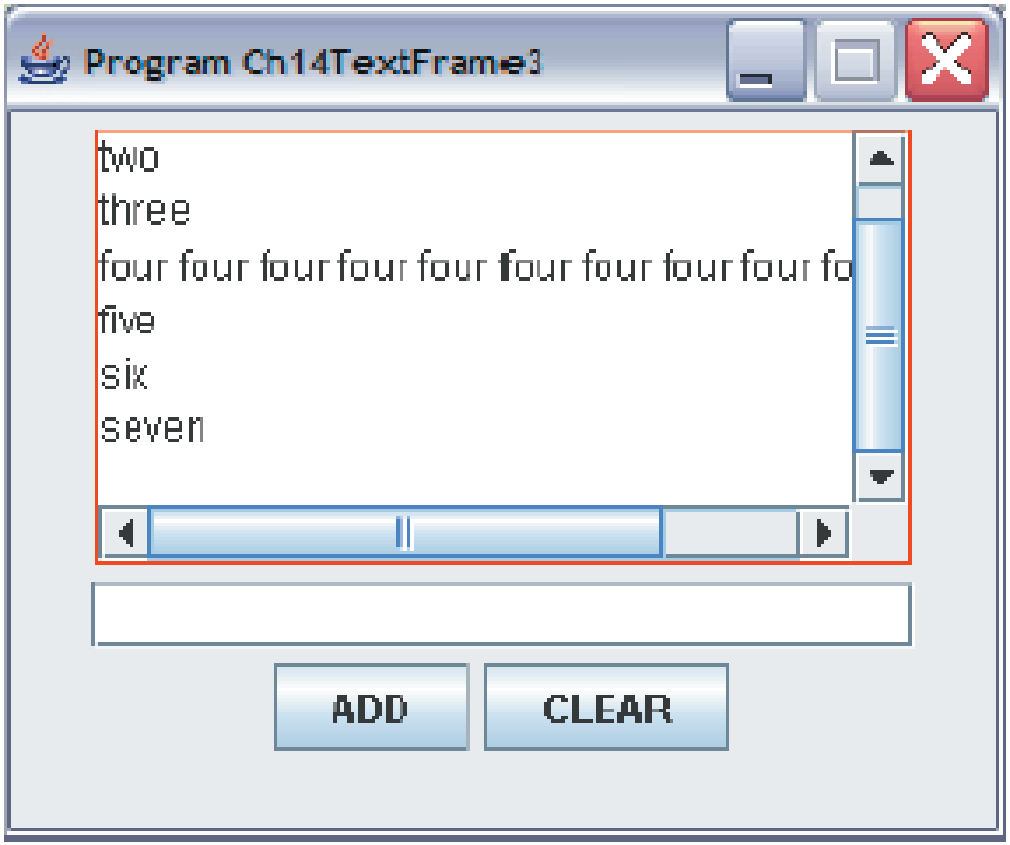
Adding Scroll Bars to JTextArea

- By default a JTextArea does not have any scroll bars. To add scroll bars, we place a JTextArea in a JScrollPane object.

```
JTextArea   textArea   = new JTextArea();
. . .
JScrollPane scrollText = new JScrollPane(textArea);
. . .
contentPane.add(scrollText);
```

TextFrame3 with Scroll Bars

- A sample TextFrame3 window when a JScrollPane is used.



Other Common GUI Components

- JCheckBox
  - see Ch14JCheckBoxSample1.java and Ch14JCheckBoxSample2.java
- JRadioButton
  - see Ch14JRadioButtonSample.java
- JComboBox
  - see Ch14JComboBoxSample.java
- JList
  - see Ch14JListSample.java
- JSlider
  - see Ch14JSliderSample.java

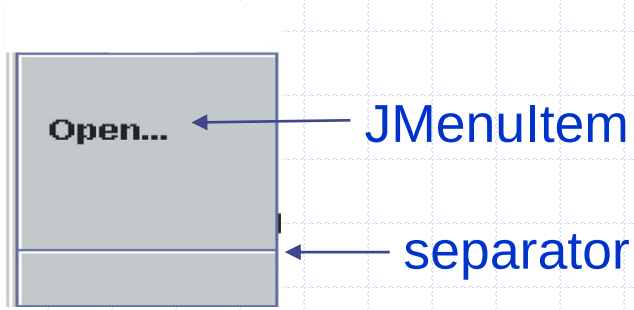
Menus

- The javax.swing package contains three menu-related classes: JMenuBar, JMenu, and JMenuItem.
- JMenuBar is a bar where the menus are placed. There is one menu bar per frame.
- JMenu (such as File or Edit) is a group of menu choices. JMenuBar may include many JMenu objects.
- JMenuItem (such as Copy, Cut, or Paste) is an individual menu choice in a JMenu object.
- Only the JMenuItem objects generate events.

Menu Components

JMenuBar

JMenu



Sequence for Creating Menus

1. Create a JMenuBar object and attach it to a frame.
2. Create a JMenu object.
3. Create JMenuItem objects and add them to the JMenu object.
4. Attach the JMenu object to the JMenuBar object.