

Java Swing

61FIT3NPR -Network Programming

Faculty of Information Technology
Hanoi University
Fall 2020



OVERVIEW:

- ❖ *What is Swing?*
- ❖ *Java swing components*
- ❖ *How to change TitleBar icon
Java Swing*
- ❖ *Some Java Swing Apps*



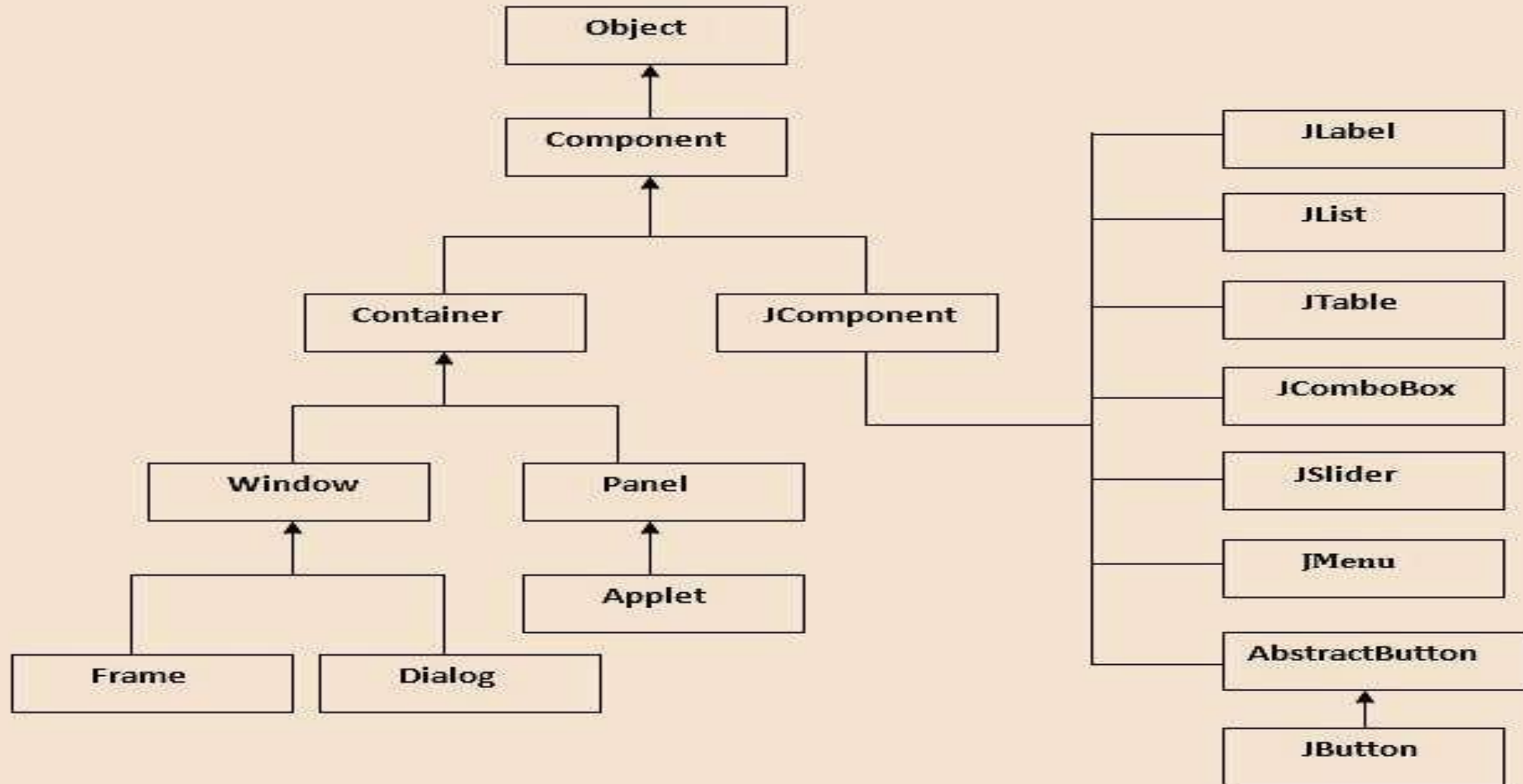
What is Java Swing

- Java Swing is a part of Java Foundation Classes (JFC) that is used to create window-based applications. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.
 - Java Swing provides platform-independent and lightweight components.
 - The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.
-

Difference between AWT and Swing

No.	JavaAWT	JavaSwing
1)	AWT components are platform-dependent.	Java swing components are platform-independent.
2)	AWT components are heavyweight.	Swing components are lightweight.
3)	AWT doesn't support pluggable look and feel.	Swing supports <i>pluggable look and feel</i> .
4)	AWT provides less components than Swing.	Swing provides more powerful components such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.
5)	AWT doesn't follow MVC(Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing follows MVC.

Hierarchy of Java Swing classes



Commonly used Methods of Component class

<i>Method</i>	<i>Description</i>
<i>public void add(Component c)</i>	add a component on another component.
<i>public void setSize(int width,int height)</i>	sets size of the component.
<i>public void setLayout(LayoutManager m)</i>	sets the layout manager for the component.
<i>public void setVisible(boolean b)</i>	sets the visibility of the component. It is by default false.

JavaJButton

Declaration of JButtonclass.

public class JButton extends AbstractButton implements Accessible

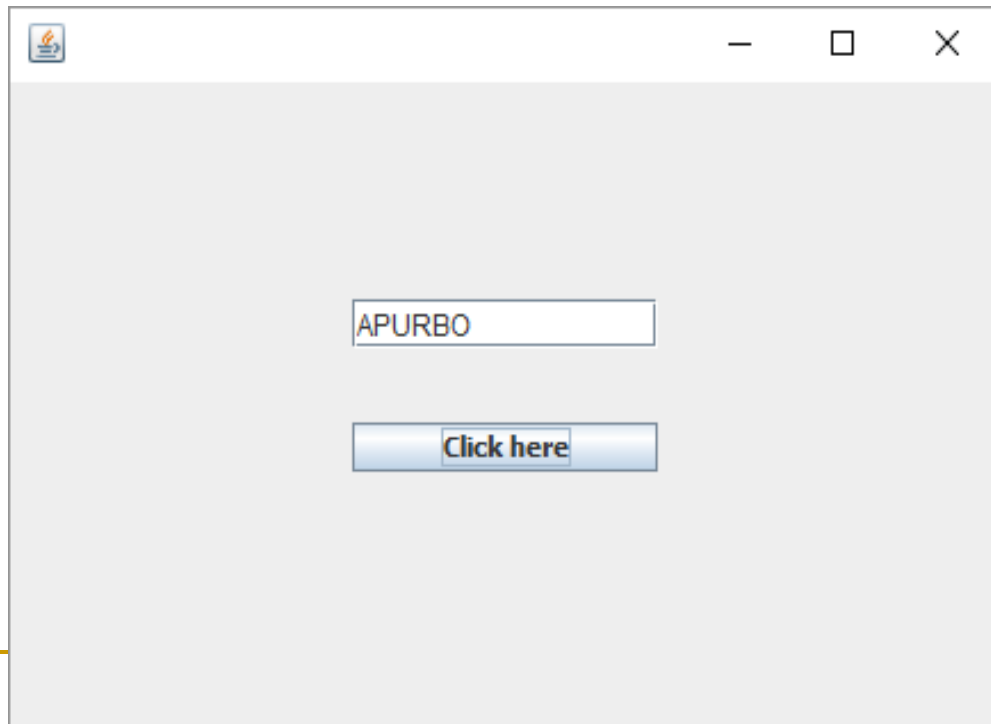
Commonly used Constructors of JButton:

<i>Constructor</i>	<i>Description</i>
<i>JButton()</i>	It creates a button with no text and icon.
<i>JButton(String s)</i>	It creates a button with the specified text.
<i>JButton(Icon i)</i>	It creates a button with the specified icon object.

Commonly used Methods of AbstractButton class:

<i>Methods</i>	<i>Description</i>
<i>void setText(String s)</i>	It is used to set specified text on button
<i>String getText()</i>	It is used to return the text of the button.
<i>void setEnabled(boolean b)</i>	It is used to enable or disable the button.
<i>void setIcon(Icon b)</i>	It is used to set the specified Icon on the button.
<i>Icon getIcon()</i>	It is used to get the Icon of the button.
<i>void setMnemonic(int a)</i>	It is used to set the mnemonic on the button.
<i>void addActionListener(ActionListener a)</i>	It is used to add the action listener to this object.

Example of Java JButton



JavaJPasswordField

The object of a JPasswordField class is a text component specialized for password entry. It allows the editing of a single line of text. It inherits JTextField class.

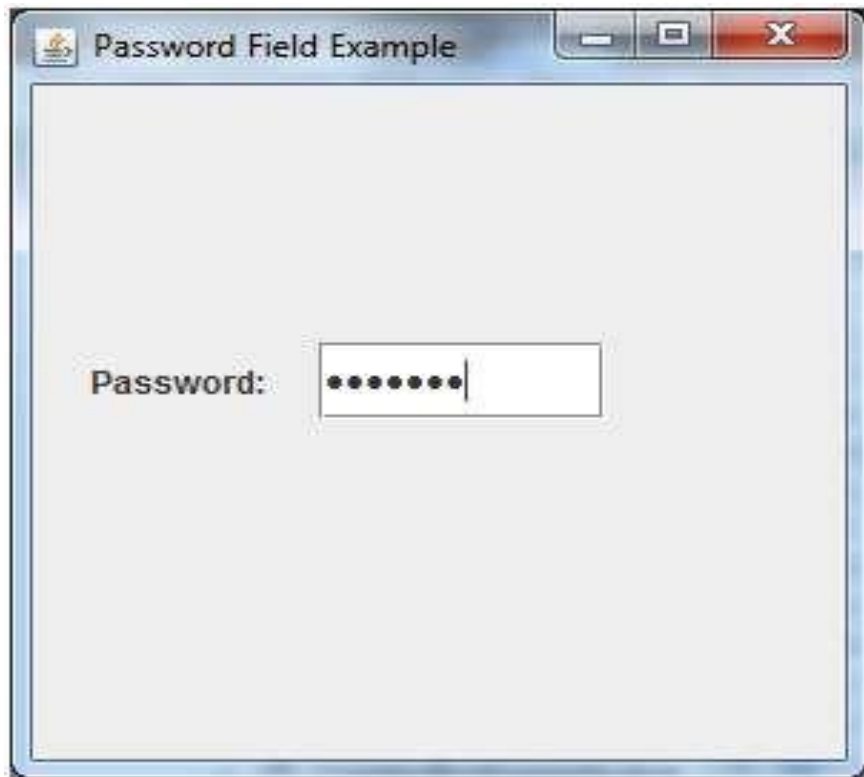
JPasswordField class declaration

```
public class JPasswordField extends JTextField
```

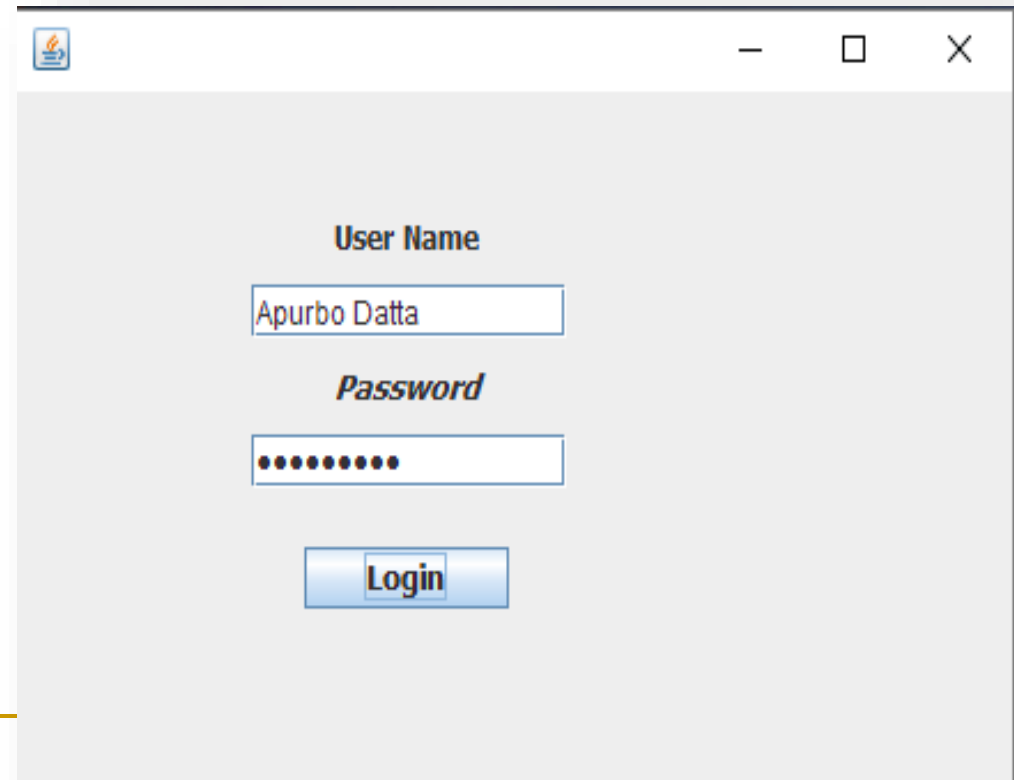
Commonly used Constructors JPasswordField:

Constructor	Description
JPasswordField()	Constructs a new JPasswordField, with a default document, null starting text string, and 0 column width.
JPasswordField(int columns)	Constructs a new empty JPasswordField with the specified number of columns.
JPasswordField(String text)	Constructs a new JPasswordField initialized with the specified text.
JPasswordField(String text, int columns)	Construct a new JPasswordField initialized with the specified text and columns.

Java JPasswordField Example



Java JPasswordField Example with ActionListener



Java JLabel

The object of JLabel class is a component for placing text in a container. It is used to display a single line of read only text. The text can be changed by an application but a user cannot edit it directly. It inherits JComponent class.

JLabel class declaration

```
public class JButton extends AbstractButton implements Accessible
```

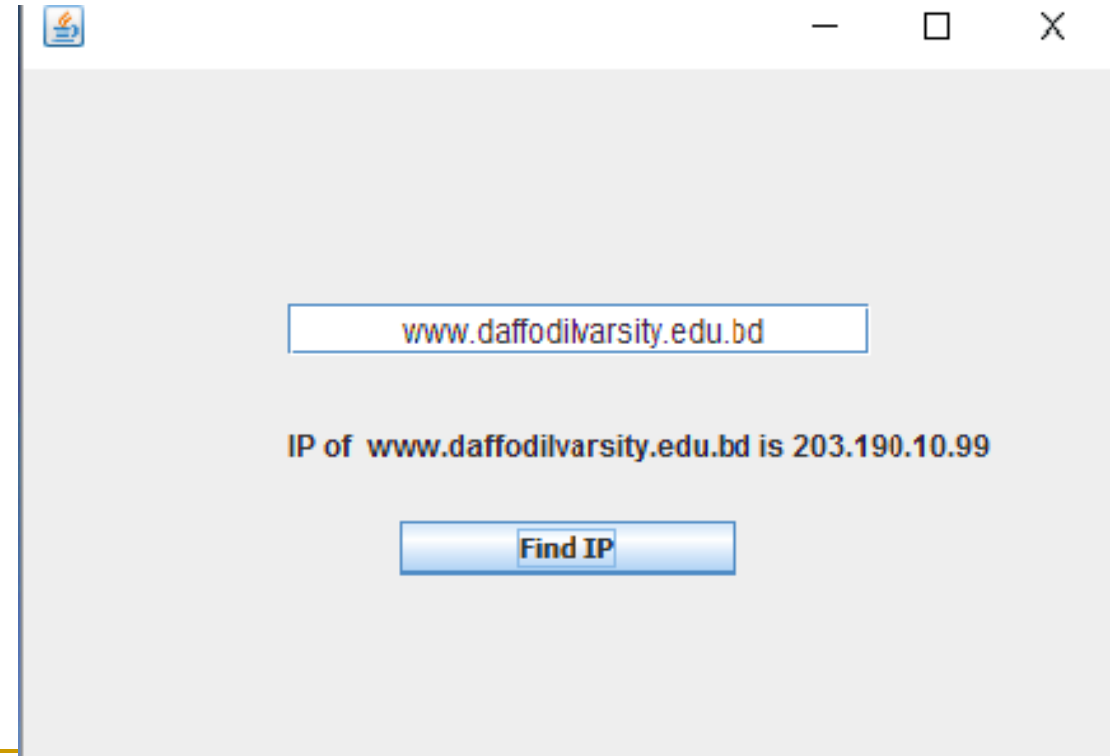
Commonly used Constructors of JLabel.

<i>Constructor</i>	<i>Description</i>
<i>JLabel()</i>	Creates a JLabel instance with no image and with an empty string for the title.
<i>JLabel(String s)</i>	Creates a JLabel instance with the specified text.
<i>JLabel(Icon i)</i>	Creates a JLabel instance with the specified image.
<i>JLabel(String s, Icon i, int horizontalAlignment)</i>	Creates a JLabel instance with the specified text, image, and horizontal alignment.

Commonly used Methods of JLabel:

Methods	Description
String getText()	It returns the text string that a label displays.
void setText(String text)	It defines the single line of text this component will display.
void setHorizontalAlignment(int alignment)	It sets the alignment of the label's contents along the X axis.
Icon getIcon()	It returns the graphic image that the label displays.
int getHorizontalAlignment()	It returns the alignment of the label's contents along the X axis.

Java JLabelExample



Java JTextField

The object of a JTextField class is a text component that allows the editing of a single line text. It inherits JTextComponent class.

JTextField class declaration

```
public class JTextField extends JTextComponent implements SwingConstants
```

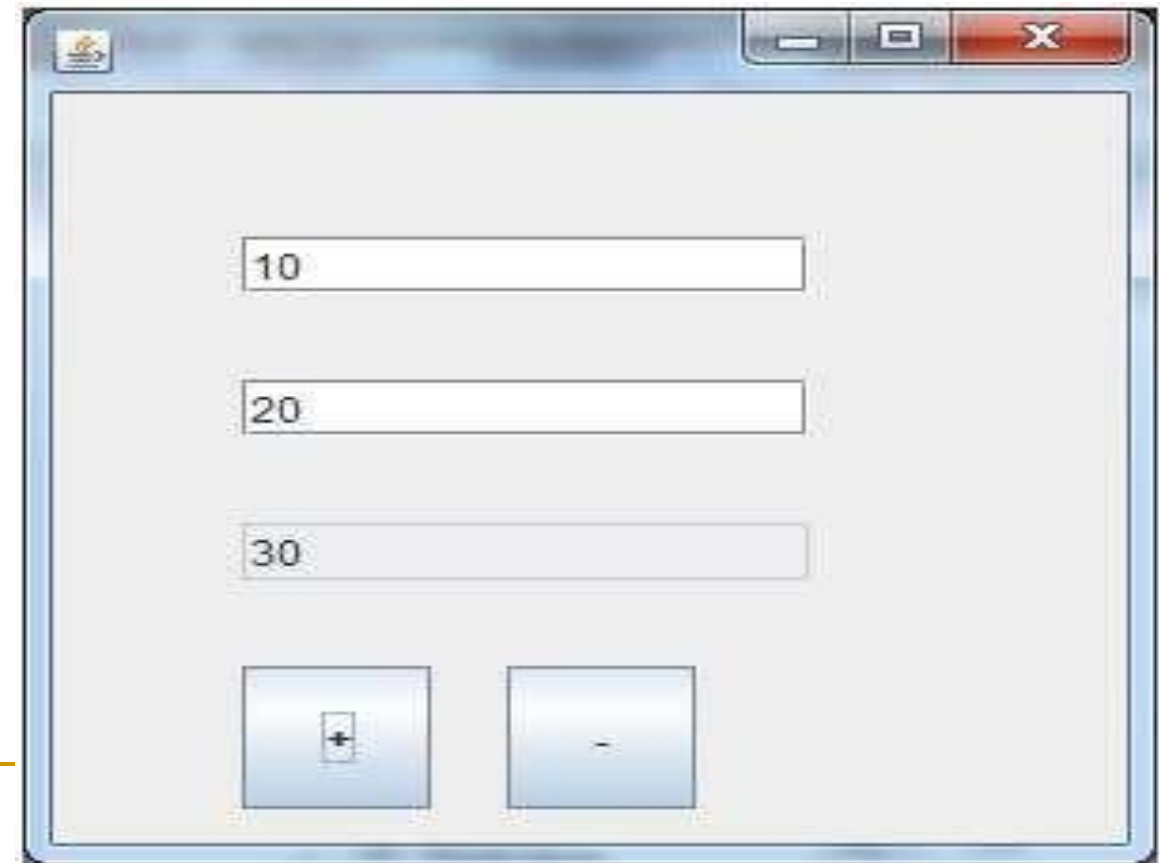
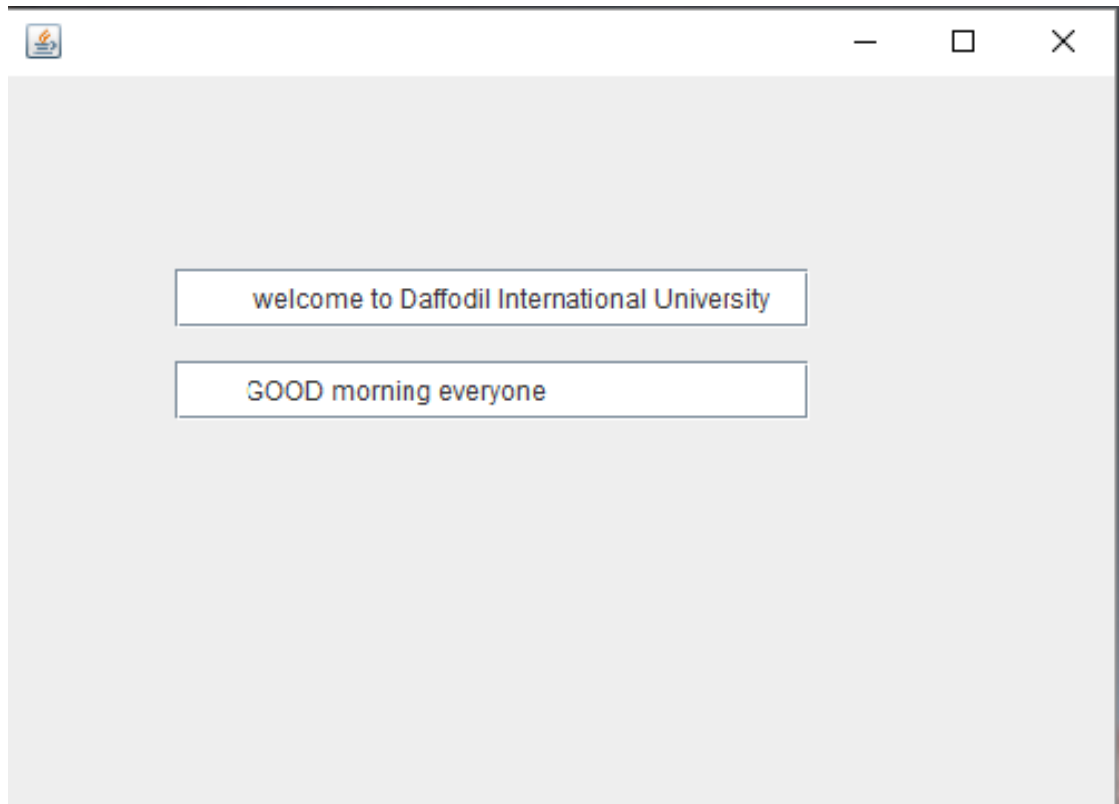

Commonly used Constructors of JTextField :

<i>Constructor</i>	<i>Description</i>
<code>JTextField()</code>	Creates a new TextField
<code>JTextField(String text)</code>	Creates a new TextField initialized with the specified text.
<code>JTextField(String text, int columns)</code>	Creates a new TextField initialized with the specified text and columns.
<code>JTextField(int columns)</code>	Creates a new empty TextField with the specified number of columns

Commonly used Methods:

Methods	Description
<code>void addActionListener(ActionListener l)</code>	It is used to add the specified action listener to receive action events from this textfield.
<code>Action getAction()</code>	It returns the currently set Action for this ActionEvent source, or null if no Action is set.
<code>void setFont(Font f)</code>	It is used to set the current font.
<code>void removeActionListener(ActionListener l)</code>	It is used to remove the specified action listener so that it no longer receives action events from this textfield.

Java JTextField Example



Java JTextArea

The object of a JTextArea class is a multi line region that displays text. It allows the editing of multiple line text. It inherits JTextComponent class

JTextArea class declaration

```
public class JTextArea extends JTextComponent
```

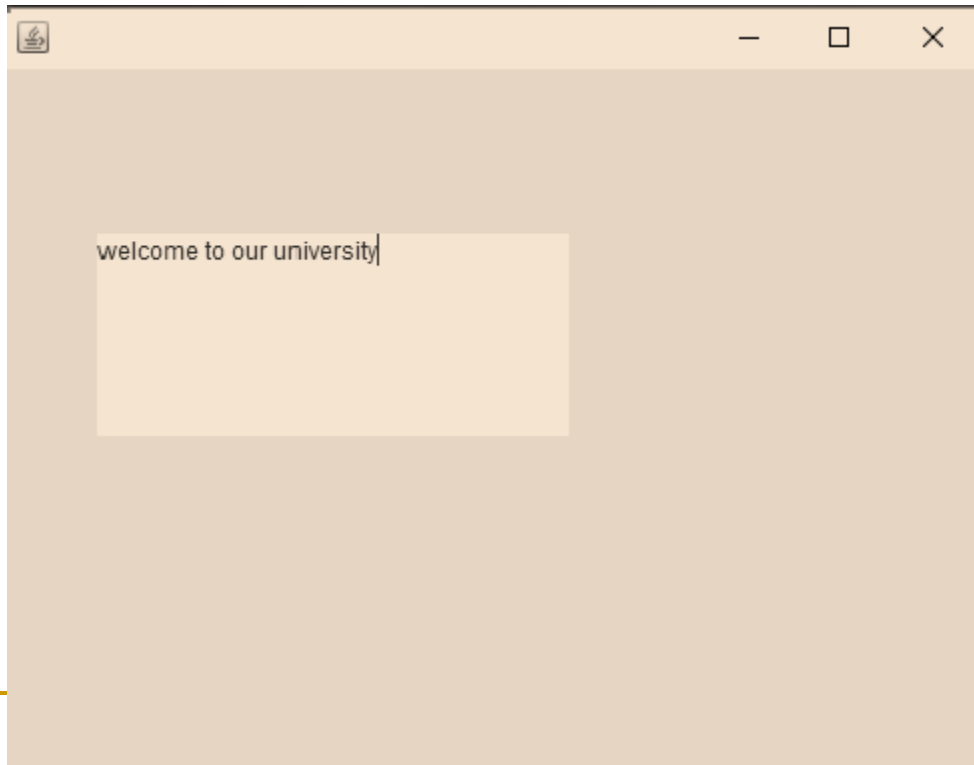
Commonly used Constructors of JTextArea:

Constructor	Description
JTextArea()	Creates a text area that displays no text initially.
JTextArea(String s)	Creates a text area that displays specified text initially.
JTextArea(int row, int column)	Creates a text area with the specified number of rows and columns that displays no text initially.
JTextArea(String s, int row, int column)	Creates a text area with the specified number of rows and columns that displays specified text.

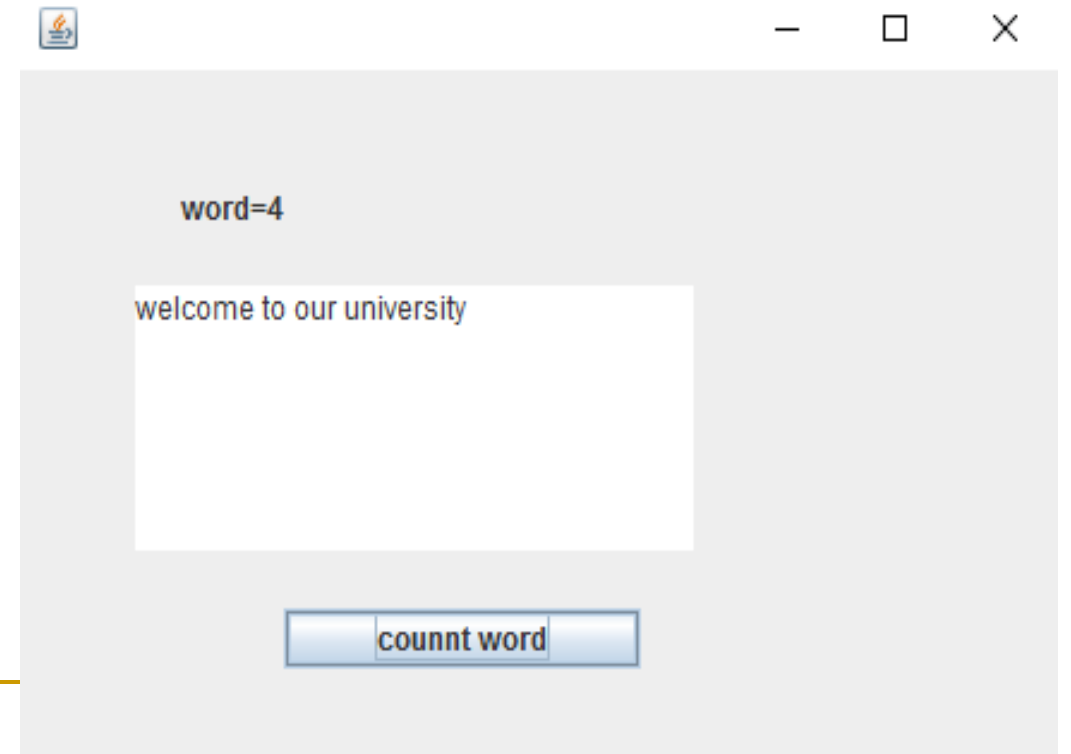
Commonly used Methods of JTextArea:

Methods	Description
<code>void setRows(int rows)</code>	It is used to set specified number of rows.
<code>void setColumns(int cols)</code>	It is used to set specified number of columns.
<code>void setFont(Font f)</code>	It is used to set the specified font.
<code>void insert(String s, int position)</code>	It is used to insert the specified text on the specified position.
<code>void append(String s)</code>	It is used to append the given text to the end of the document.

Java JTextArea Example



Java JTextArea Example with ActionListener



Java JCheckBox

The JCheckBox class is used to create a checkbox. It is used to turn an option on (true) or off (false). Clicking on a CheckBox changes its state from "on" to "off" or from "off" to "on ".It inherits JToggleButton class.

JCheckBox class declaration

```
public class JCheckBox extends JToggleButton implements Accessible
```

Commonly used Constructors:

Constructor	Description
JJCheckBox()	Creates an initially unselected check box button with no text, no icon.
JChechBox(String s)	Creates an initially unselected check box with text.
JCheckBox(String text, boolean selected)	Creates a check box with text and specifies whether or not it is initially selected.
JCheckBox(Action a)	Creates a check box where properties are taken from the Action supplied.

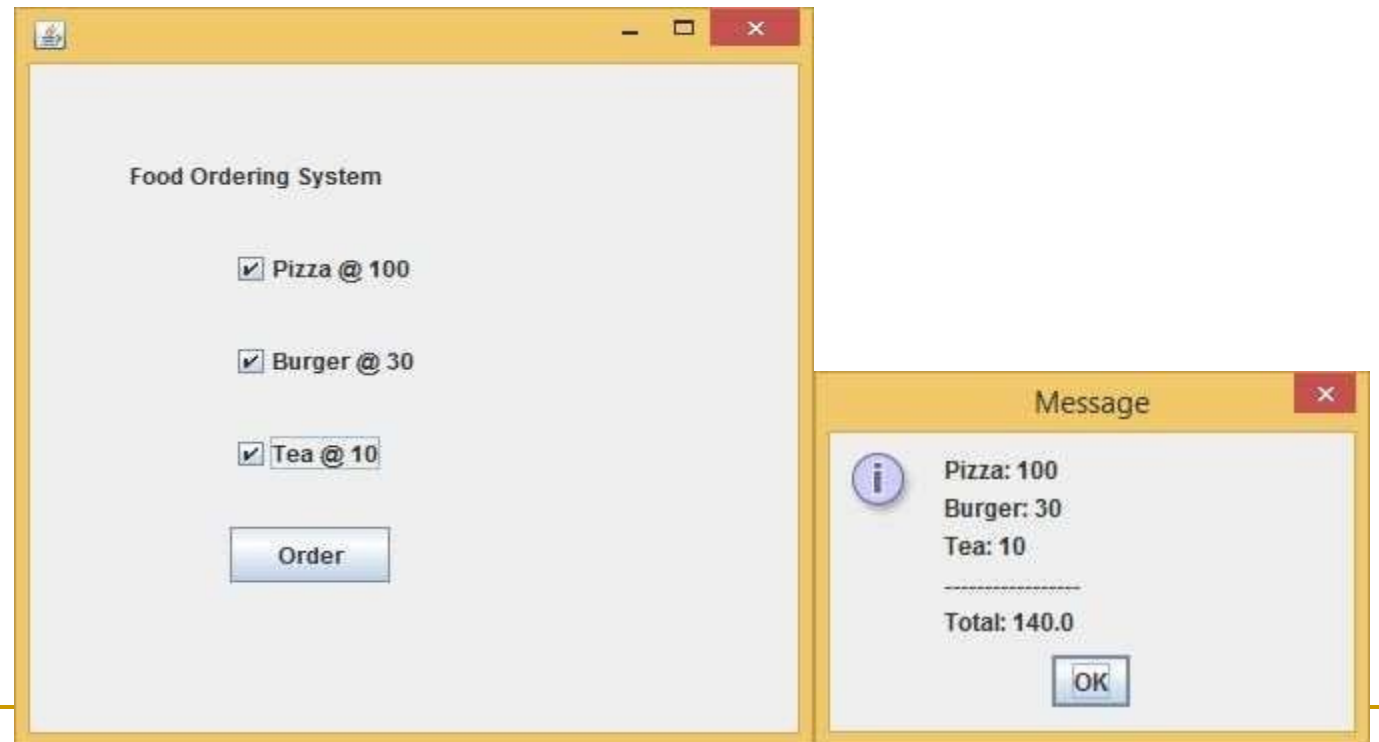
Commonly used Methods:

Methods	Description
AccessibleContext getAccessibleContext()	It is used to get the AccessibleContext associated with this JCheckBox.
protected String paramString()	It returns a string representation of this JCheckBox.

Java JCheckBox Example



Java JCheckBox Example with ItemListene



Java JRadioButton

The JRadioButton class is used to create a radio button. It is used to choose one option from multiple options. It is widely used in exam systems or quiz.

❖ It should be added in ButtonGroup to select one radio button only.

JRadioButton class declaration:

Public class JRadioButton extends JToggleButton implements Accessible

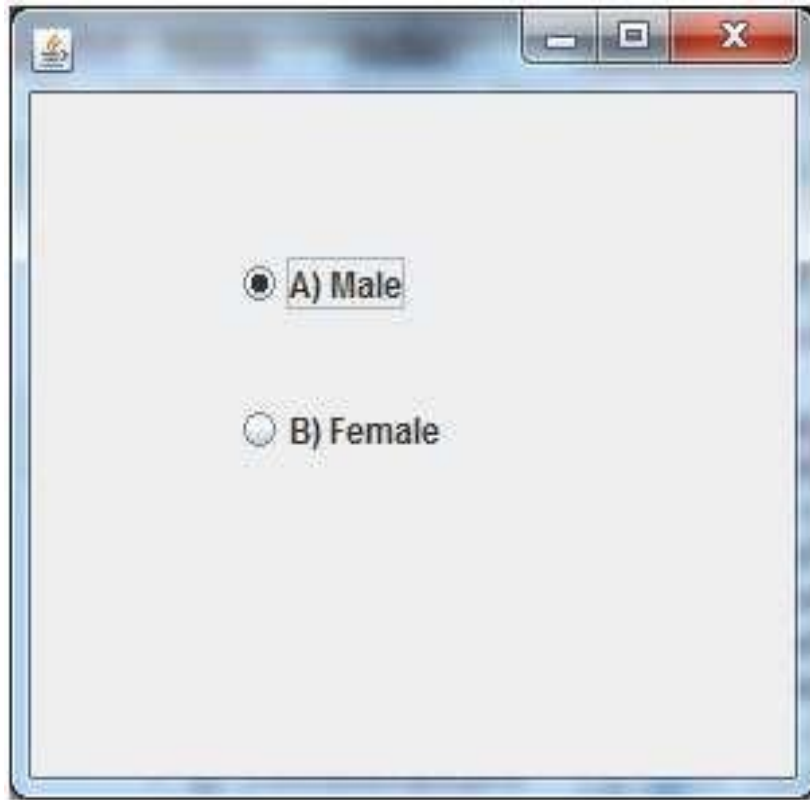
Commonly used Constructors of JRadioButton:

Constructor	Description
JRadioButton()	Creates an unselected radio button with no text.
JRadioButton(String s)	Creates an unselected radio button with specified text.
JRadioButton(String s, boolean selected)	Creates a radio button with the specified text and selected status.

Commonly used Methods:

Methods	Description
<code>void setText(Strings)</code>	It is used to set specified text on button.
<code>String getText()</code>	It is used to return the text of the button.
<code>void setEnabled(boolean b)</code>	It is used to enable or disable the button.
<code>void setIcon(Icon b)</code>	It is used to set the specified Icon on the button.
<code>Icon getIcon()</code>	It is used to get the Icon of the button.
<code>void setMnemonic(int a)</code>	It is used to set the mnemonic on the button.
<code>void addActionListener(ActionListener a)</code>	It is used to add the action listener to this object.

Java JRadioButton Example



Java JRadioButton Example with ActionListener



Java JComboBox

The object of Choice class is used to show popup menu of choices. Choice selected by user is shown on the top of a menu. It inherits JComponent class.

JComboBox class declaration

```
public class JComboBox extends JComponent implements ItemSelectable, ListDataListener,
ActionListener, Accessible
```

Commonly used Constructors Of JComboBox:

<i>Constructor</i>	<i>Description</i>
<i>JComboBox()</i>	Creates a JComboBox with a default data model.
<i>JComboBox(Object[] items)</i>	Creates a JComboBox that contains the elements in the specified array.
<i>JComboBox(Vector<?> items)</i>	Creates a JComboBox that contains the elements in the specified Vector.

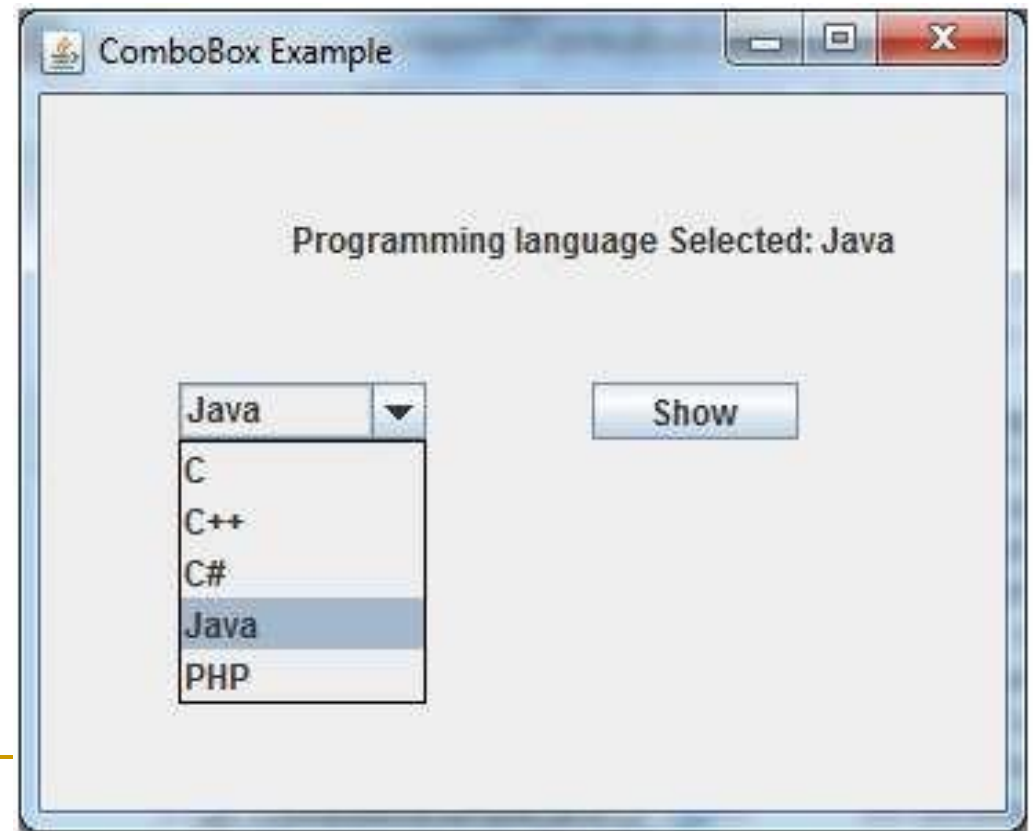
Commonly used Methods Of JComboBox:

Methods	Description
<code>void addItem(Object anObject)</code>	It is used to add an item to the item list.
<code>void removeItem(Object anObject)</code>	It is used to delete an item to the item list.
<code>void removeAllItems()</code>	It is used to remove all the items from the list.
<code>void setEditable(boolean b)</code>	It is used to determine whether the JComboBox is editable.
<code>void addActionListener(ActionListener a)</code>	It is used to add the ActionListener.
<code>void addItemListener(ItemListener i)</code>	It is used to add the ItemListener.

Java JComboBox Example



Java JComboBox Example with ActionListener



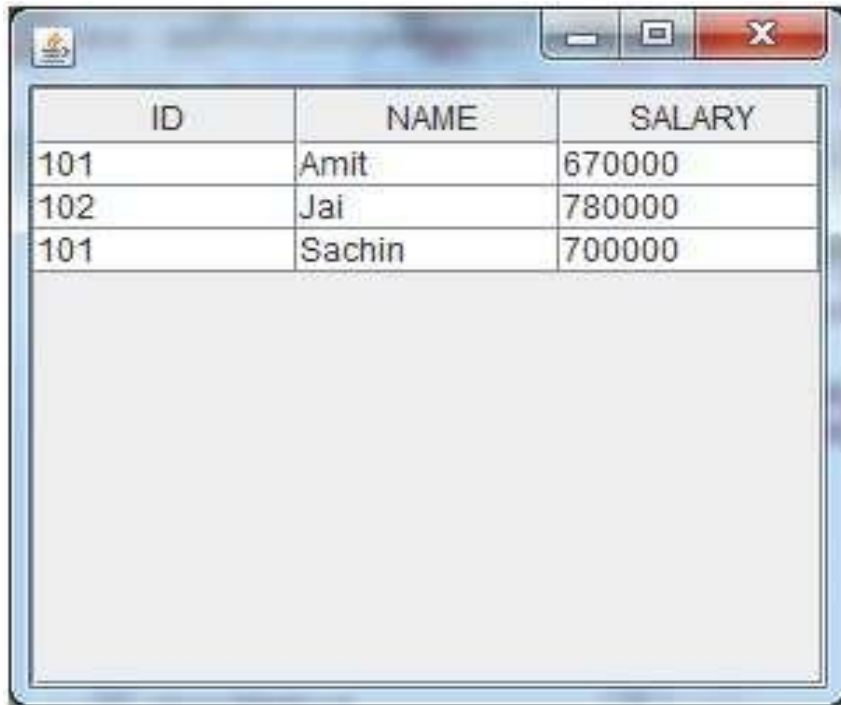
Java JTable

The JTable class is used to display data in tabular form. It is composed of rows and columns.

Commonly used Constructors of JTable:

Constructor	Description
JTable()	Creates a table with empty cells.
JTable(Object[][] rows, Object[] columns)	Creates a table with the specified data.

Java JTable Example



ID	NAME	SALARY
101	Amit	670000
102	Jai	780000
101	Sachin	700000

Java JTable Example with ListSelectionListen



ID	NAME	SALARY
101	Amit	670000
102	Jai	780000
101	Sachin	700000

If you select an element in column NAME, name of the element will be displayed on the console:

Table element selected is: Sachin

Java JList

The object of JList class represents a list of text items. The list of text items can be set up so that the user can choose either one item or multiple items. It inherits JComponent class.

JList class declaration

```
public class JList extends JComponent implements Scrollable, Accessibl  
e
```

Commonly used Constructors of JList :

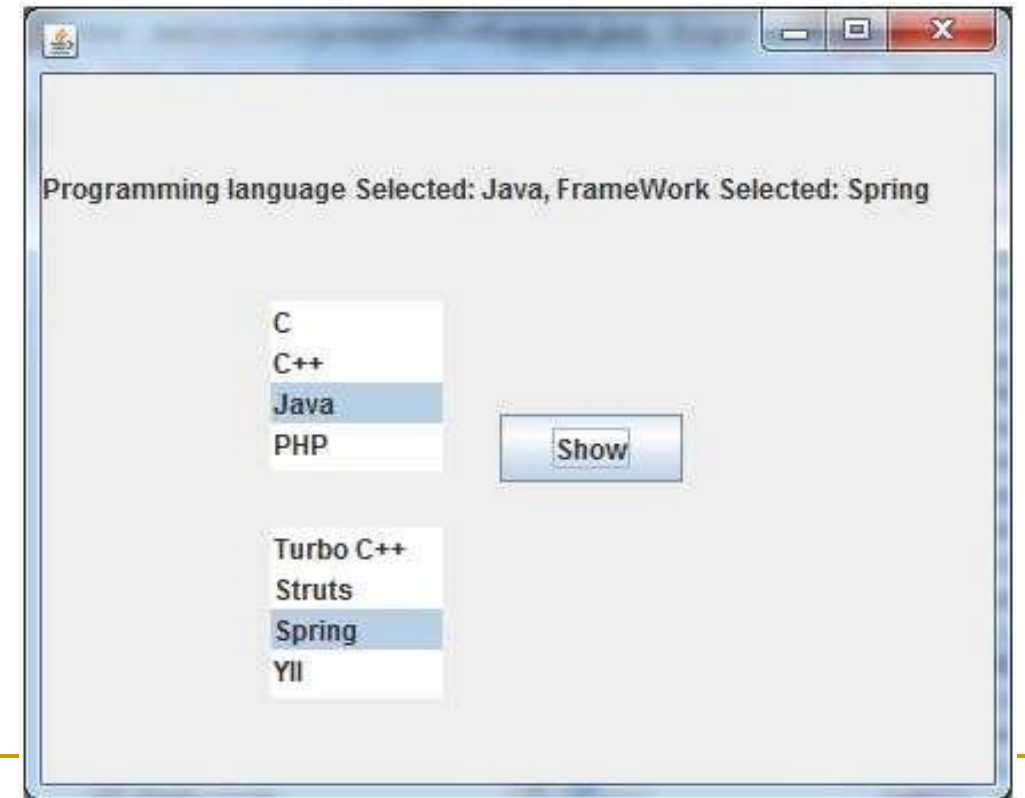
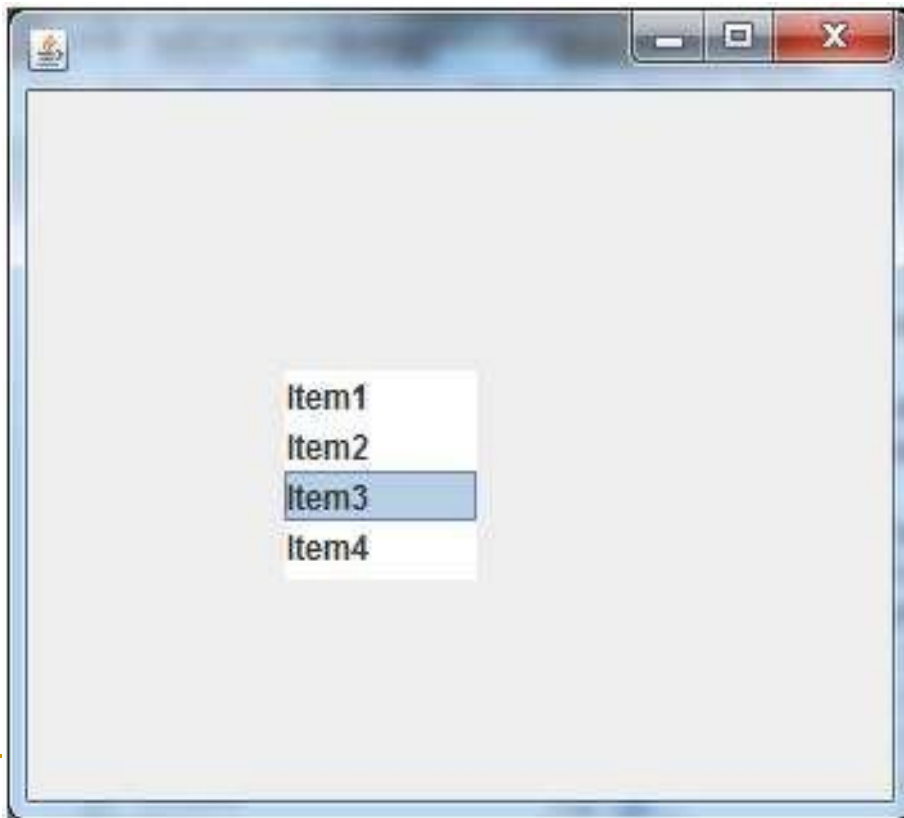
Constructor	Description
JList()	Creates a JList with an empty, read-only, model.
JList(ary[] listData)	Creates a JList that displays the elements in the specified array.
JList(ListModel<ary> dataModel)	Creates a JList that displays elements from the specified, non-null, model.

Commonly used Methods of JList

• Methods	Description
Void addListSelectionListener(ListSelectionListener listener)	It is used to add a listener to the list, to be notified each time a change to the selection occurs.
int getSelectedIndex()	It is used to return the smallest selected cell index.
ListModel getModel()	It is used to return the data model that holds a list of items displayed by the JList component.
void setListData(Object[] listData)	It is used to create a read-only ListModel from an array of objects.

JavaJListExample

Java JList Example with ActionListener



Java JPanel

The JPanel is a simplest container class. It provides space in which an application can attach any other component. It inherits the JComponent class.

It doesn't have title bar.

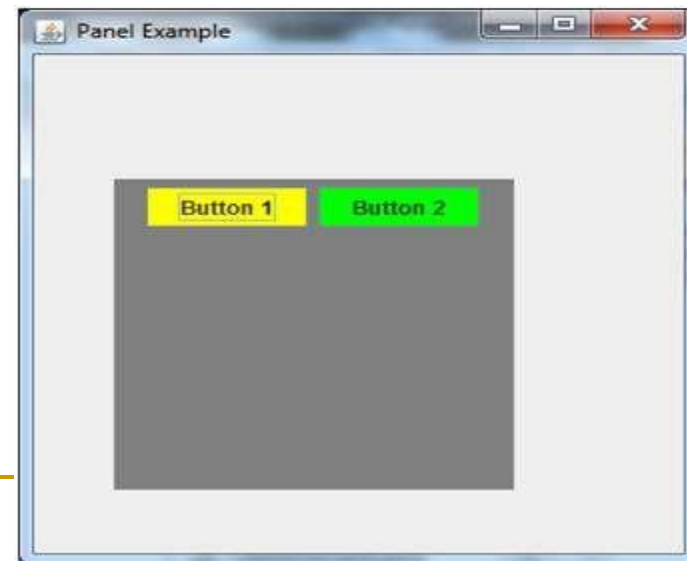
JPanel class declaration

```
public class JPanel extends JComponent implements Accessible
```

Commonly used Constructors of JPanel :

Constructor	Description
JPanel()	It is used to create a new JPanel with a double buffer and a flow layout.
JPanel(boolean isDoubleBuffered)	It is used to create a new JPanel with FlowLayout and the specified buffering strategy.
JPanel(LayoutManager layout)	It is used to create a new JPanel with the specified layout manager.

JavaJPanelExample



Java JProgressBar

The JProgressBar class is used to display the progress of the task. It inherits JComponent class.

JProgressBar class declaration

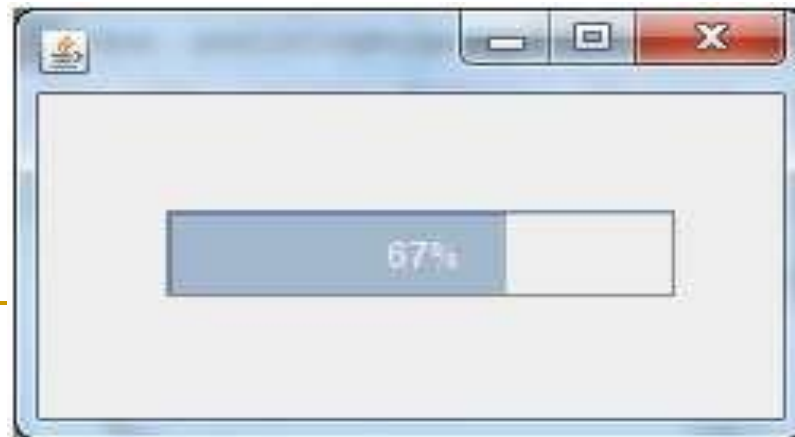
```
public class JProgressBar extends JComponent implements SwingConstants,  
Accessible
```

Commonly used Constructors:

<i>Constructor</i>	Description
JProgressBar()	It is used to create a horizontal progress bar but no string text.
JProgressBar(int min, int max)	It is used to create a horizontal progress bar with the specified minimum and maximum value.
JProgressBar(int orient)	It is used to create a progress bar with the specified orientation, it can be either Vertical or Horizontal by using SwingConstants.VERTICAL and SwingConstants.HORIZONTAL constants.
JProgressBar(int orient, int min, int max)	It is used to create a progress bar with the specified orientation, minimum and maximum value.

Commonly used Methods:

Method	Description
<code>void setStringPainted(boolean b)</code>	It is used to determine whether string should be displayed.
<code>void setString(String s)</code>	It is used to set value to the progress string.
<code>void setOrientation(int orientation)</code>	It is used to set the orientation, it may be either vertical or horizontal by using <code>SwingConstants.VERTICAL</code> and <code>SwingConstants.HORIZONTAL</code> constants.
<code>void setValue(int value)</code> <i>Java JProgressBar Example</i>	It is used to set the current value on the progress bar.



Java JSlider

The Java JSlider class is used to create the slider. By using JSlider, a user can select a value from a specific range.

Commonly used Constructors of JSlider class

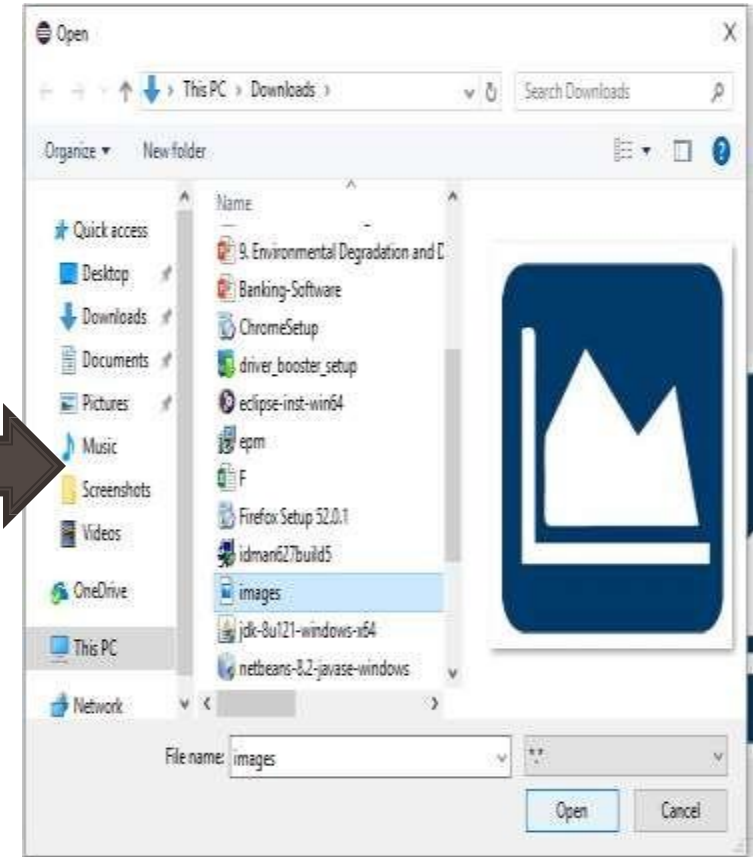
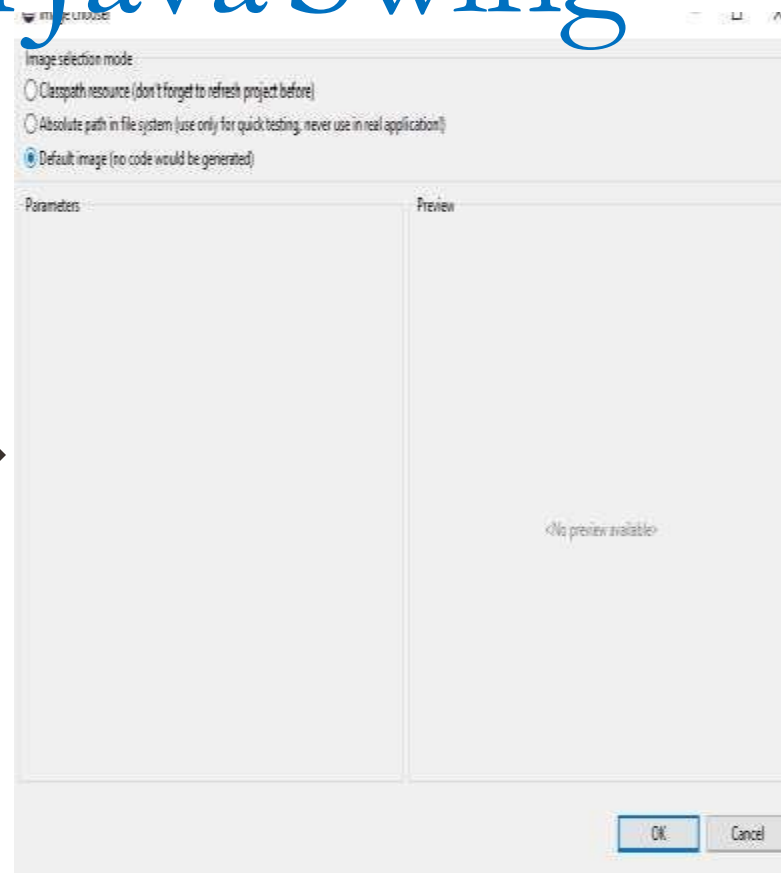
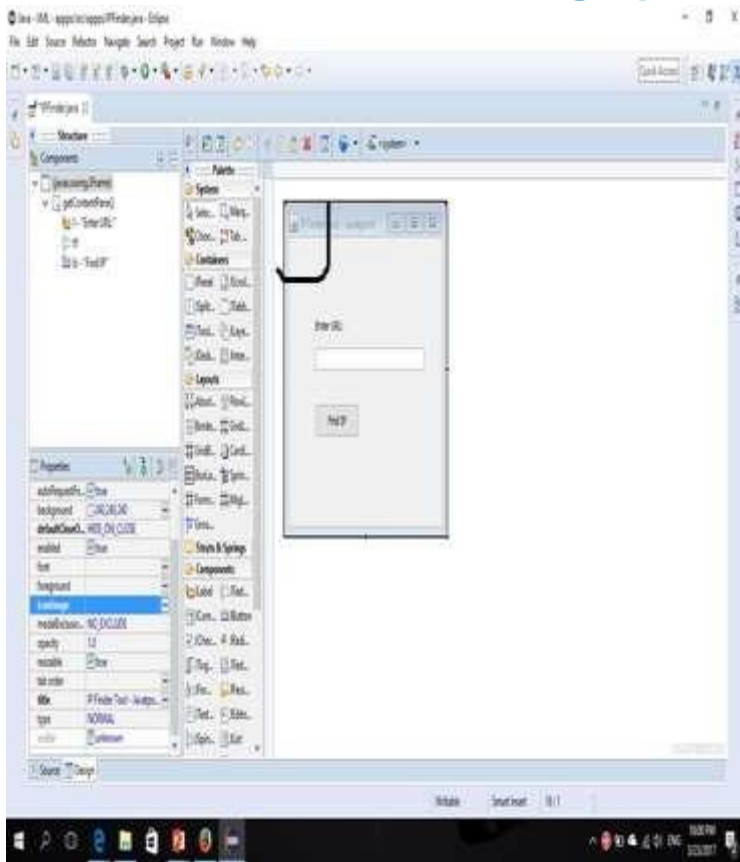
Constructor	Description
JSlider()	creates a slider with the initial value of 50 and range of 0 to 100.
JSlider(int orientation)	creates a slider with the specified orientation set by either JSlider.HORIZONTAL or JSlider.VERTICAL with the range 0 to 100 and initial value 50.
JSlider(int min, int max)	creates a horizontal slider using the given min and max.
JSlider(int min, int max, int value)	creates a horizontal slider using the given min, max and value.
JSlider(int orientation, int min, int max, int value)	creates a slider using the given orientation, min, max and value.

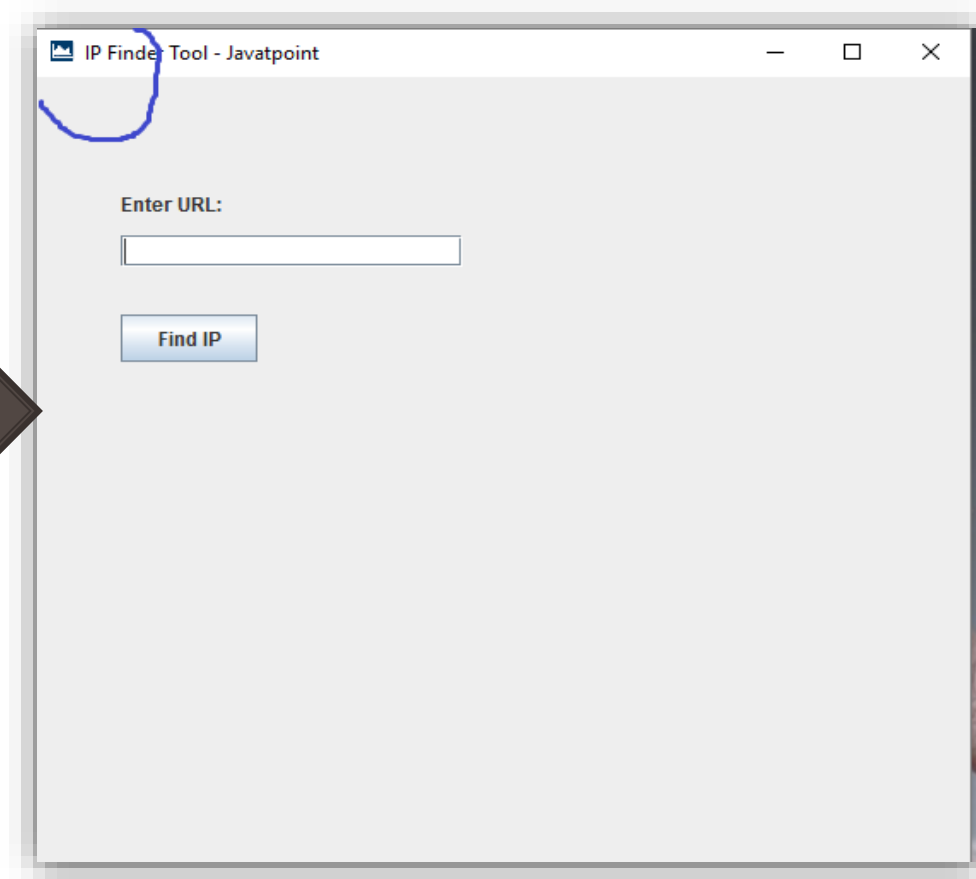
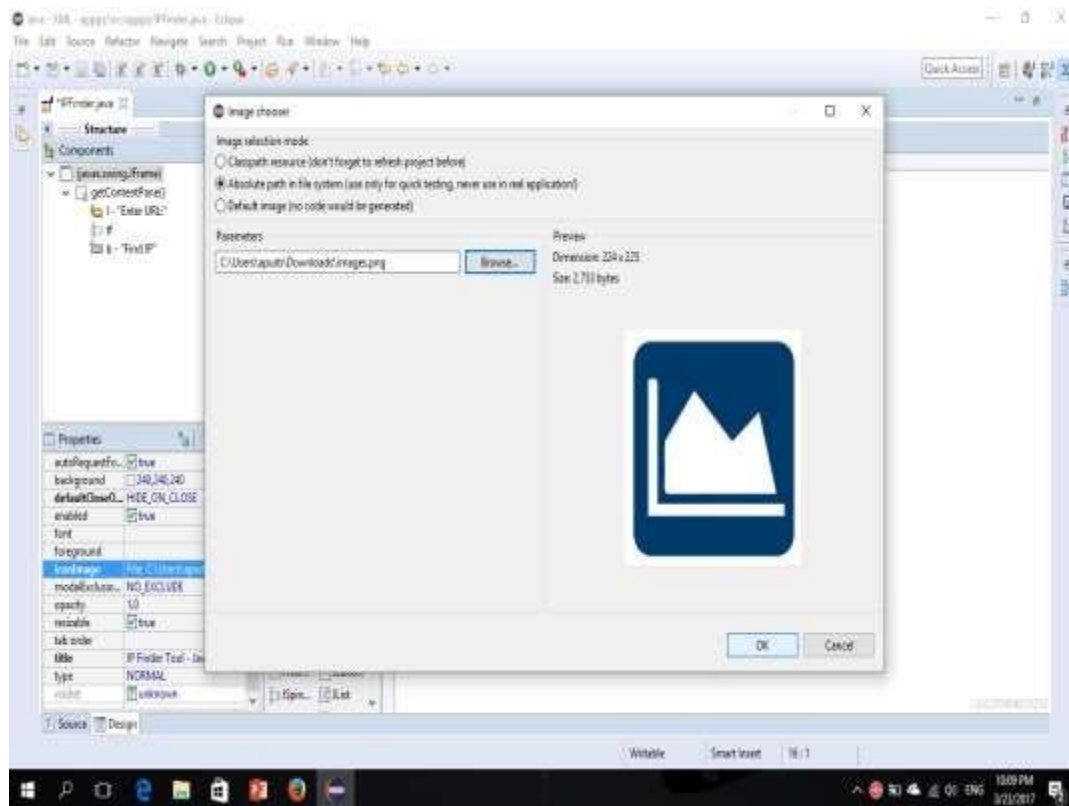
Commonly used Methods of JSlider class

Method	Description
<code>public void setMinorTickSpacing(int n)</code>	is used to set the minor tick spacing to the slider.
<code>public void setMajorTickSpacing(int n)</code>	is used to set the major tick spacing to the slider.
<code>public void setPaintTicks(boolean b)</code>	is used to determine whether tick marks are painted.
<code>public void setPaintLabels(boolean b)</code>	is used to determine whether labels are painted.
<code>Public void setPaintTracks(boolean b)</code>	Is used to determine whether track is painted



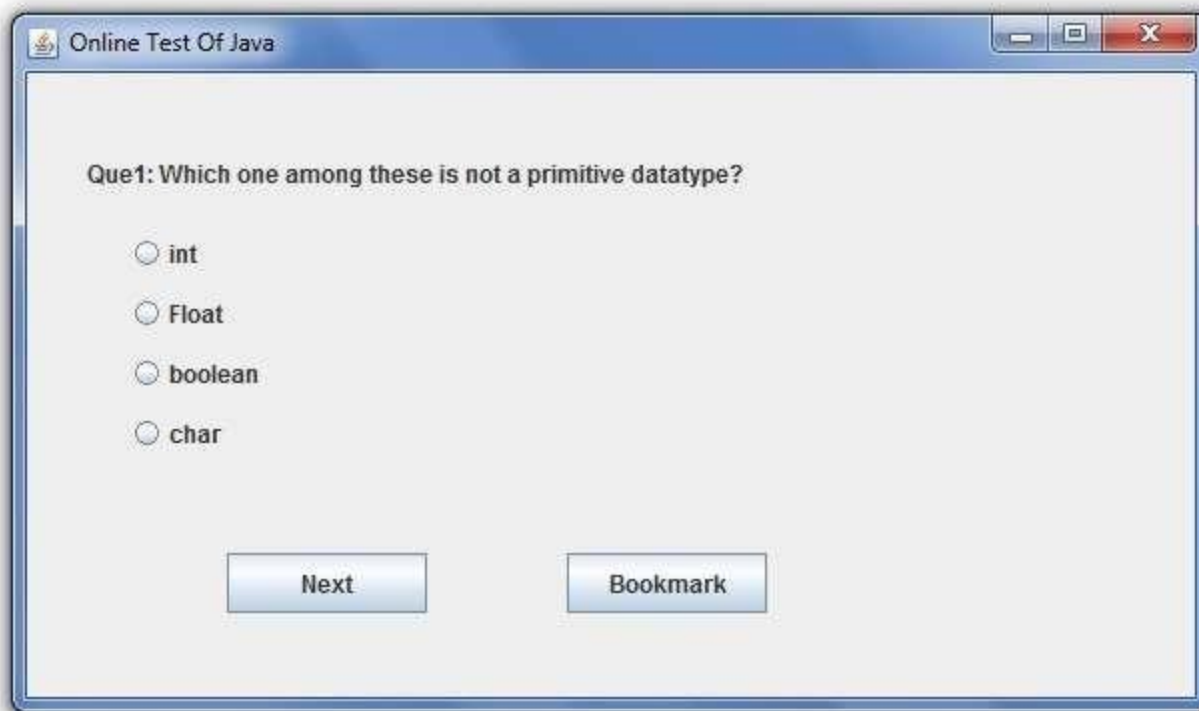
How to change TitleBar icon in Java Swing



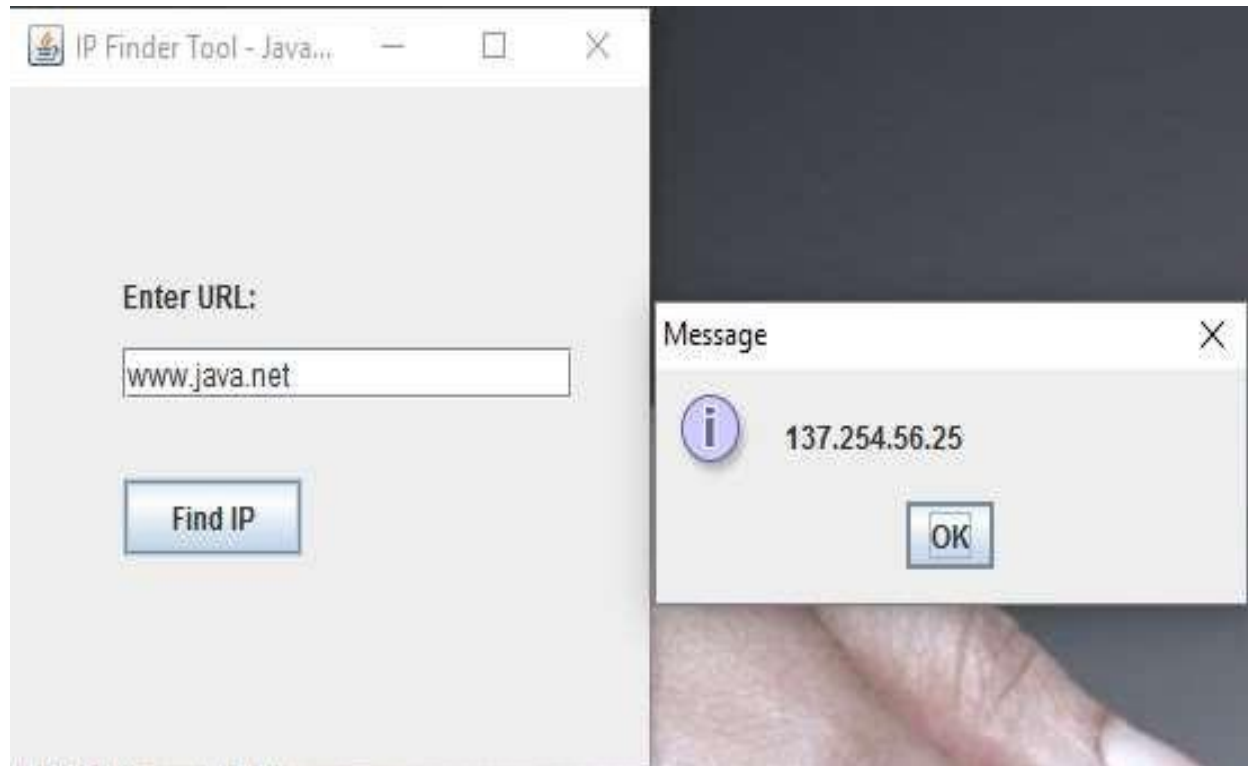


Some Java Swing Apps

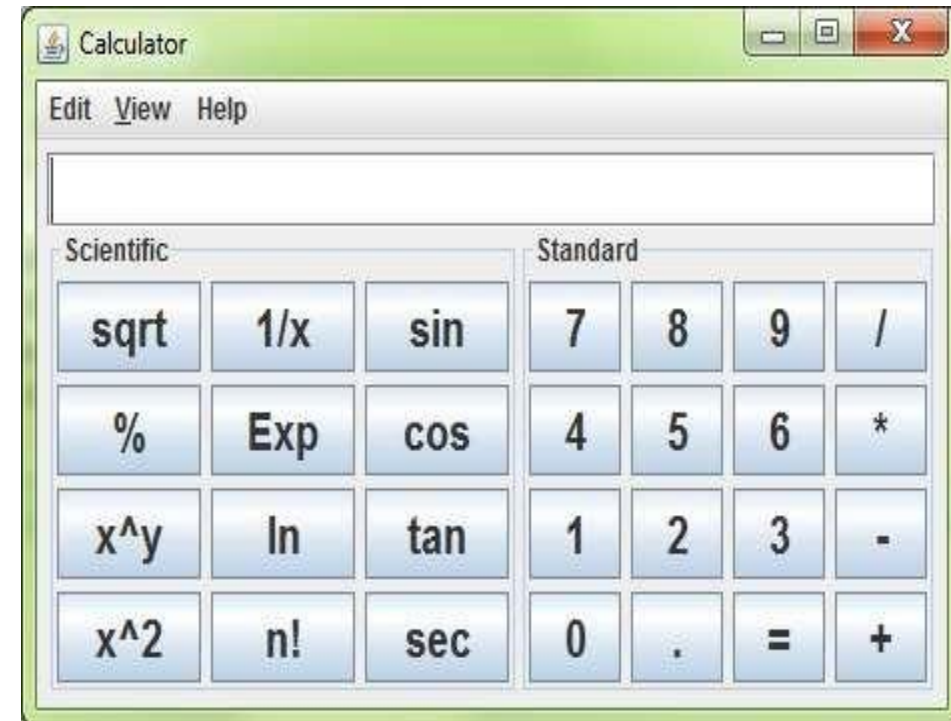
Online Exam



IPFinder



Calculator



AWT vs. SWT

You have to decide

- AWT / Swing (Abstract Windowing Toolkit)
 - Write once, run anywhere
 - Formerly ugly, with bad performance
 - Now look and work well
 - Use garbage collection
 - Come with the JDK and JRE
- SWT / JFace (Standard Window Toolkit)
 - The important fact is that Eclipse uses SWT, not AWT
 - Supposed to look better, run faster
 - A thin wrapper around native widgets
 - SWT components must be disposed (vs. garbage collected)
 - *Owing to need to free native resources*
 - Need JNI libraries for each platform
 - Distribution is through the Eclipse Foundation, not Sun

AWT vs. SWT

More Considerations

- It is not easy to convert between them
- The SWT look is not obviously better
- The performance difference may not be there either, today
- Eclipse uses SWT
 - They are supposed to mix and match, but ???
- Sun is unlikely to include SWT support in the JDK and JRE soon

