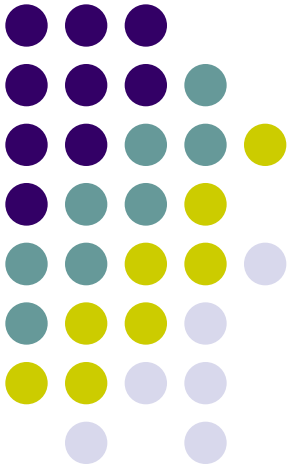
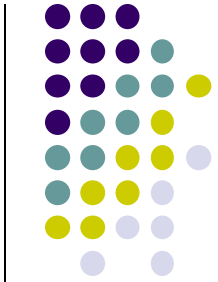
Vidyalankar Polytechnic

SWINGS

10 Marks

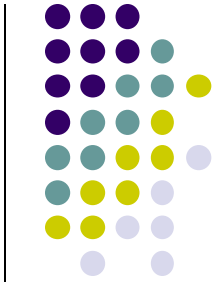


Unit Outcomes



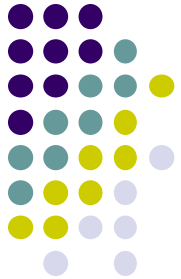
- Differentiate between AWT and Swing on the given aspects.
- Develop Graphical User Interface (GUI) programs using swing components for the given problem.
- Use the given type of button in Java based GUI
- Develop Graphical User Interface (GUI) programs using advanced swing components for the given problem.

Introduction



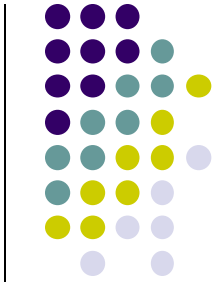
- Swing API is a set of extensible GUI Components to ease the developer's life to create JAVA based Front End/GUI Applications.
- It is build on top of AWT API and acts as a replacement of AWT API, since it has almost every control corresponding to AWT controls.

Introduction (cont..)



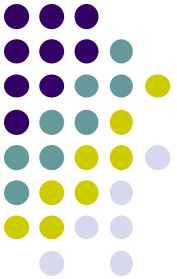
- Swing component follows a Model-View-Controller architecture to fulfill the following criteria's.
  - A single API is to be sufficient to support multiple look and feel.
  - API is to be model driven so that the highest level API is not required to have data.
  - API is to use the Java Bean model so that Builder Tools and IDE can provide better services to the developers for use.

MVC Architecture



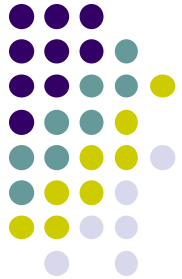
- Swing API architecture follows loosely based MVC architecture in the following manner.
  - Model represents component's data.
  - View represents visual representation of the component's data.
  - Controller takes the input from the user on the view and reflects the changes in Component's data.
  - Swing component has Model as a separate element, while the View and Controller part are clubbed in the User Interface elements. Because of which, Swing has a pluggable look-and-feel architecture.

Swing Features



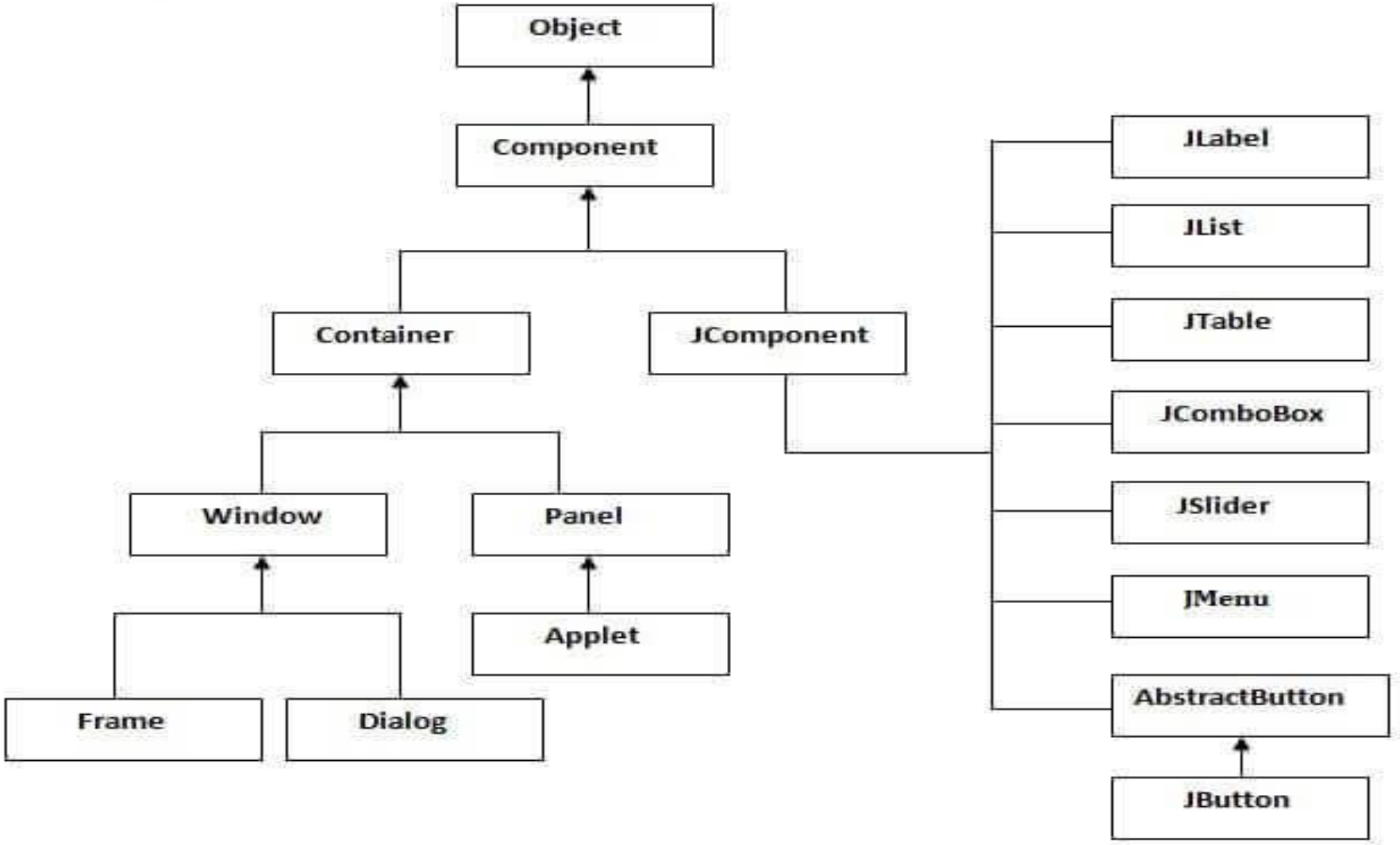
- **Light Weight** − Swing components are independent of native Operating System's API as Swing API controls are rendered mostly using pure JAVA code instead of underlying operating system calls.
- **Rich Controls** − Swing provides a rich set of advanced controls like Tree, TabbedPane, slider, colorpicker, and table controls.

Swing Features (cont..)

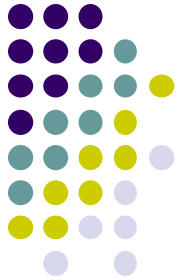


- **Highly Customizable** − Swing controls can be customized in a very easy way as visual apperance is independent of internal representation.
- **Pluggable look-and-feel** − SWING based GUI Application look and feel can be changed at run-time, based on available values.

Hierarchy of Java Swing classes

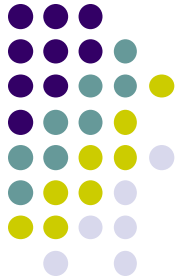


Difference between AWT & Swing



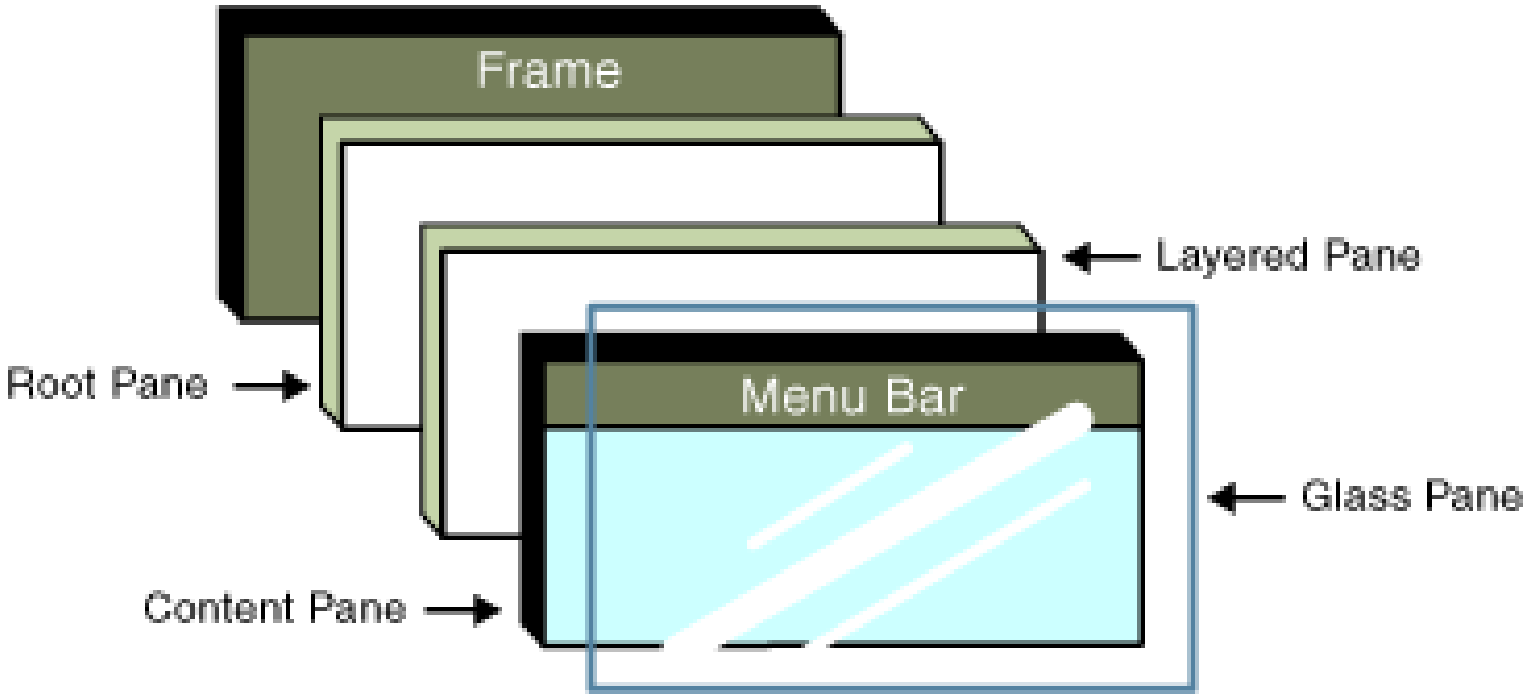
No.	Java AWT	Java Swing
1)	AWT components are **platform-dependent**.	Java swing components are **platform-independent**.
2)	AWT components are **heavyweight**.	Swing components are **lightweight**.
3)	AWT **doesn't support pluggable look and feel**.	Swing **supports pluggable look and feel**.
4)	AWT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

Commonly used Methods of Component class



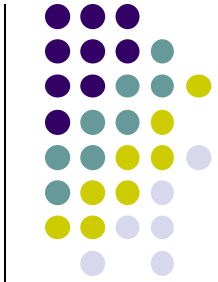
Method	Description
public void add(Component c)	add a component on another component.
public void setSize(int width,int height)	sets size of the component.
public void setLayout(LayoutManager m)	sets the layout manager for the component.
public void setVisible(boolean b)	sets the visibility of the component.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

JFrame



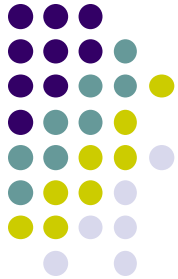
From Sun Microsystems(tm) Website

JFrame



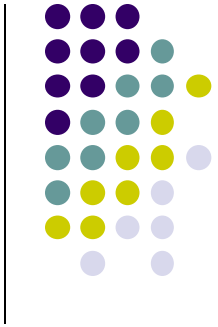
- JFrame works like the main window where components like labels, buttons, textfields are added to create a GUI.
- Unlike Frame, JFrame has the option to hide or close the window with the help of setDefaultCloseOperation(int) method.

JFrame (cont..)



Constructor	Description
JFrame()	It constructs a new frame that is initially invisible.
JFrame(GraphicsConfiguration gc)	It creates a Frame in the specified GraphicsConfiguration of a screen device and a blank title.
JFrame(String title)	It creates a new, initially invisible Frame with the specified title.
JFrame(String title, GraphicsConfiguration gc)	It creates a JFrame with the specified title and the specified GraphicsConfiguration of a screen device.

JFrame (cont..)



Modifier and Type	Method	Description
protected void	addImpl(Component comp, Object constraints, int index)	Adds the specified child Component.
protected JRootPane	createRootPane()	Called by the constructor methods to create the default rootPane.
protected void	frameInit()	Called by the constructors to init the JFrame properly.
void	setContentPane(Containe contentPane)	It sets the contentPane property
static void	setDefaultLookAndFeelDecorated(boolean defaultLookAndFeelDecorated)	Provides a hint as to whether or not newly created JFrames should have their Window decorations (such as borders, widgets to close the window, title...) provided by the current look and feel.

JFrame (cont..)

Modifier and Type	Method	Description
void	setIconImage(Image image)	It sets the image to be displayed as the icon for this window.
void	setJMenuBar(JMenuBar menubar)	It sets the menubar for this frame.
void	setLayeredPane(JLayeredPane layeredPane)	It sets the layeredPane property.
JRootPane	getRootPane()	It returns the rootPane object for this frame.
TransferHandler	getTransferHandler()	It gets the transferHandler property.

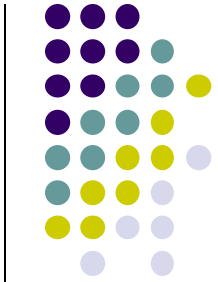
Program

JApplet

- We can use JApplet that can have all the controls of swing.

Program

JPanel

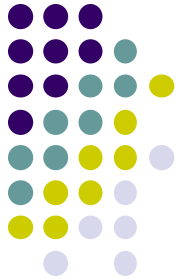


- The JPanel is a simplest container class.
- It provides space in which an application can attach any other component.
- It inherits the JComponents class.

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.

Program

ImageIcon



- In Swing, icons are encapsulated by the **ImageIcon** class, which paints an icon from an image

Constructors

ImageIcon()

ImageIcon(byte[] imageData)

ImageIcon(byte[] imageData, String description)

ImageIcon(Image image)

ImageIcon(Image image, String description)

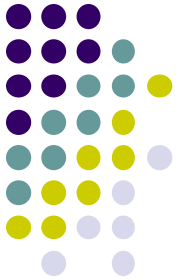
ImageIcon(String filename)

ImageIcon(String filename, String description)

ImageIcon(URL location)

ImageIcon(URL location, String description)

ImageIcon (cont..)



Methods

String getDescription()

int getIconHeight()

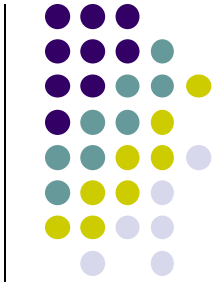
int getIconWidth()

Image getImage()

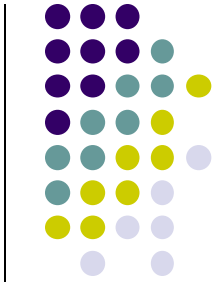
void setDescription(String description)

void setImage(Image image)

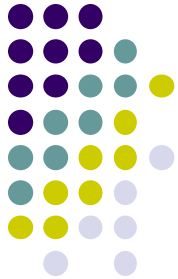
JLabel



- 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.



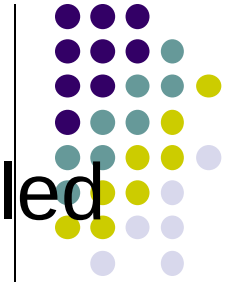
Constructor	Description
JLabel()	Creates a JLabel instance with no image and with an empty string for the title.
JLabel(String s)	Creates a JLabel instance with the specified text.
JLabel(Icon i)	Creates a JLabel instance with the specified image.
JLabel(String s, Icon i, int horizontalAlignment)	Creates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



Methods	Description
String getText()	t 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.

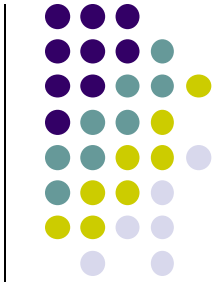
Program

JButton



- The JButton class is used to create a labeled button that has platform independent implementation.
- The application result in some action when the button is pushed.
- It inherits AbstractButton class.

Constructor	Description
JButton()	It creates a button with no text and icon.
JButton(String s)	It creates a button with the specified text.
JButton(Icon i)	It creates a button with the specified icon object.
JButton(String s, Icon i)	It creates a button with the specified text and icon.
JButton(Action a)	Creates a button where properties are taken from the Action supplied



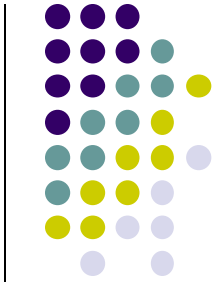
Methods	Description
void setText(String s)	It is used to set specified text on button
String getText()	It is used to return the text of the button.
void setEnabled(boolean b)	It is used to enable or disable the button.
void setIcon(Icon b)	It is used to set the specified Icon on the button.
Icon getIcon()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

Program

JTextField

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

Constructor	Description
JTextField()	Creates a new TextField
JTextField(String text)	Creates a new TextField initialized with the specified text.
JTextField(String text, int columns)	Creates a new TextField initialized with the specified text and columns.
JTextField(int columns)	Creates a new empty TextField with the specified number of columns.
JTextField(Document doc, String text, int columns)	Constructs a new JTextField that uses the given text storage model and the given number of columns.



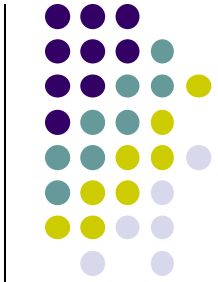
Methods	Description
void addActionListener(ActionListener l)	It is used to add the specified action listener to receive action events from this textfield.
Action getAction()	It returns the currently set Action for this ActionEvent source, or null if no Action is set.
void setFont(Font f)	It is used to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

Program

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

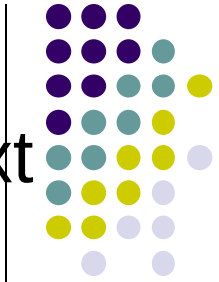
Constructor	Description
JTextArea()	Creates a text area that displays no text initially.
JTextArea(String s)	Creates a text area that displays specified text initially.
JTextArea(int row, int column)	Creates a text area with the specified number of rows and columns that displays no text initially.
JTextArea(String s, int row, int column)	Creates a text area with the specified number of rows and columns that displays specified text.
JTextArea(Document doc)	Constructs a new JTextArea with the given document model, and defaults for all of the other arguments (null, 0, 0).
JTextArea(Document doc, String text, int rows, int columns)	Constructs a new JTextArea with the specified number of rows and columns, and the given model.



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

Program

JPasswordField

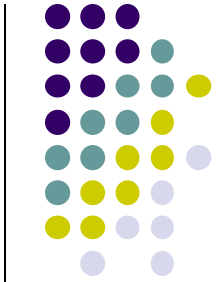


- The object of a JPasswordField class is a text component specialized for password entry.
- It allows the editing of a single line of text.
- It inherits JTextField class.

Program

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.
JPasswordField(Document doc, String txt, int columns)	Constructs a new JPasswordField that uses the given text storage model and the given number of columns.

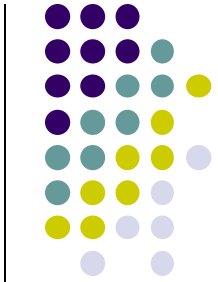
JCheckBox



- 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.

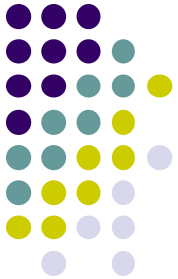
Constructor	Description
JJCheckBox()	Creates an initially unselected check box button with no text, no icon.
JCheckBox(String s)	Creates an initially unselected check box with text.
JCheckBox(String text, boolean selected)	Creates a check box with text and specifies whether or not it is initially selected.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

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.

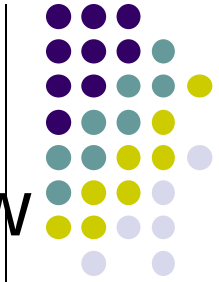
Constructor	Description
JRadioButton()	Creates an unselected radio button with no text.
JRadioButton(String s)	Creates an unselected radio button with specified text.
JRadioButton(String s, boolean selected)	Creates a radio button with the specified text and selected status.
JRadioButton(Action a)	Creates a radiobutton where properties are taken from the Action supplied.
JRadioButton(Icon icon)	Creates an initially unselected radio button with the specified image but no text
JRadioButton(Icon icon, boolean selected)	Creates a radio button with the specified image and selection state, but no text.
JRadioButton(String text, Icon icon)	Creates a radio button that has the specified text and image, and which is initially unselected.
JRadioButton(String text, Icon icon, boolean selected)	Creates a radio button that has the specified text, image, and selection state.



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

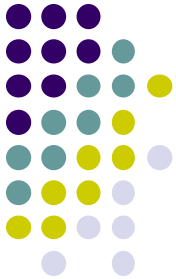
Program

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.

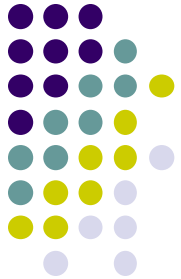
Constructor	Description
JComboBox()	Creates a JComboBox with a default data model.
JComboBox(Object[] items)	Creates a JComboBox that contains the elements in the specified array.
JComboBox(Vector<?> items)	Creates a JComboBox that contains the elements in the specified Vector.
JComboBox(ComboBoxModel aModel)	Creates a JComboBox that takes its items from an existing ComboBoxModel.



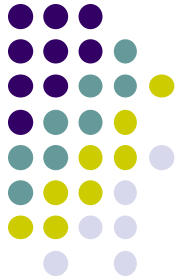
Methods	Description
void addItem(Object anObject)	It is used to add an item to the item list.
void removeItem(Object anObject)	It is used to delete an item to the item list.
void removeAllItems()	It is used to remove all the items from the list.
void setEditable(boolean b)	It is used to determine whether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

Program

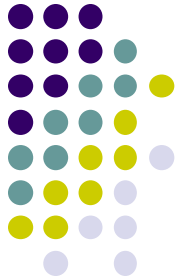
JList



- 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.



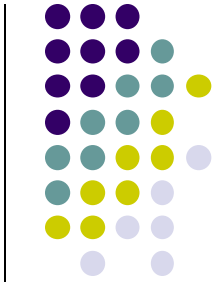
Constructor	Description
JList()	Creates a JList with an empty, read-only, model.
JList(ary[] listData)	Creates a JList that displays the elements in the specified array.
JList(ListModel<ary> dataModel)	Creates a JList that displays elements from the specified, non-null, model
JList(Vector<?> listData)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



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.

Program

JTabbedPane



- The JTabbedPane class is used to switch between a group of components by clicking on a tab with a given title or icon.
- It inherits JComponent class.

Program

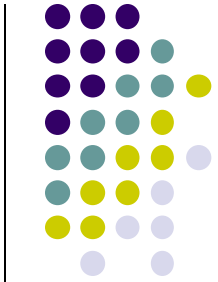
Constructor	Description
JTabbedPane()	Creates an empty TabbedPane with a default tab placement of JTabbedPane.Top.
JTabbedPane(int tabPlacement)	Creates an empty TabbedPane with a specified tab placement.
JTabbedPane(int tabPlacement, int tabLayoutPolicy)	Creates an empty TabbedPane with a specified tab placement and tab layout policy.

JScrollPane

- A JscrollPane is used to make scrollable view of a component. When screen size is limited, we use a scroll pane to display a large component or a component whose size can change dynamically.

Constructor	Purpose
JScrollPane()	It creates a scroll pane. The Component parameter, when present, sets the scroll pane's client. The two int parameters, when present, set the vertical and horizontal scroll bar policies (respectively).
JScrollPane(Component)	
JScrollPane(int, int)	
JScrollPane(Component, int, int)	

JTree

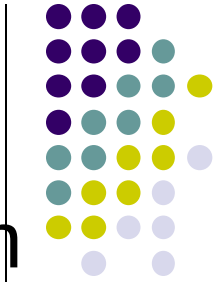


- The JTree class is used to display the tree structured data or hierarchical data. JTree is a complex component. It has a 'root node' at the top most which is a parent for all nodes in the tree. It inherits JComponent class.

Constructor	Description
JTree()	Creates a JTree with a sample model.
JTree(Object[] value)	Creates a JTree with every element of the specified array as the child of a new root node.
JTree(TreeNode root)	Creates a JTree with the specified TreeNode as its root, which displays the root node.

Program

JTable



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

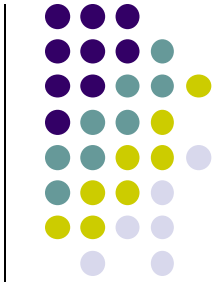
Constructor	Description
JTable()	Creates a table with empty cells.
JTable(Object[][] rows, Object[] columns)	Creates a table with the specified data.

Program

JProgressBar

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

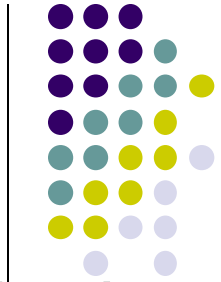
Constructor	Description
JProgressBar()	It is used to create a horizontal progress bar but no string text.
JProgressBar(int min, int max)	It is used to create a horizontal progress bar with the specified minimum and maximum value.
JProgressBar(int orient)	It is used to create a progress bar with the specified 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.



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

Program

ToolTip



- You can create a tool tip for any JComponent with **setToolTipText()** method. This method is used to set up a tool tip for the component.

For example, to add tool tip to PasswordField, you need to add only one line of code:

```
field.setToolTipText("Enter your Password");
```

Program