**Swing**

**Java Swing tutorial** is *used to create window-based applications*. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.

Unlike AWT, 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.

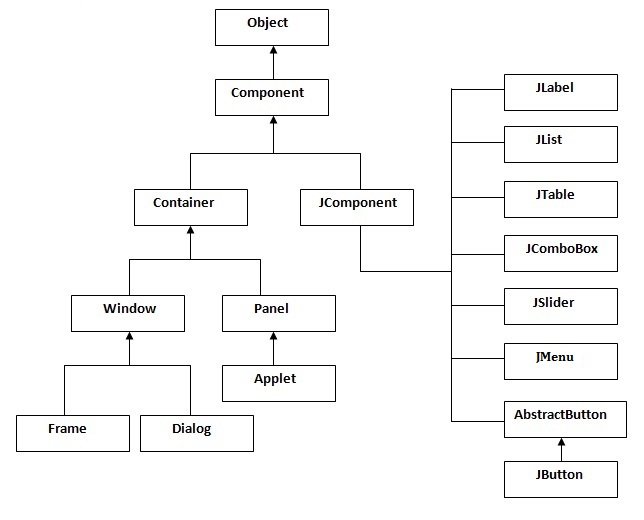
Difference between AWT and Swing

There are many differences between java awt and swing that are given below.

|  |  |  |
| --- | --- | --- |
| **No.** | **AWT** | **Swing** |
| 1) | AWT components are **platform-dependent**. | 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**. |

Hierarchy of Swing classes

The hierarchy of swing API is given below.



Commonly used Methods of Component class

The methods of Component class are widely used in java swing that are given below.

|  |  |
| --- | --- |
| **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. |

**Examples**

There are two ways to create a frame:

* By creating the object of Frame class (association)
* By extending Frame class (inheritance)

The setBounds(int xaxis, int yaxis, int width, int height)is used in the above example that sets the position of the button.

# JButton:

|  |
| --- |
| The JButton class is used to create a button that have plateform-independent implementation. |

### Constructors:

* **JButton():** creates a button with no text and icon.
* **JButton(String s):** creates a button with the specified text.
* **JButton(Icon i):** creates a button with the specified icon object.

### Methods of AbstractButton class:

|  |
| --- |
| **1) public void setText(String s):** is used to set specified text on button. |
| **2) public String getText():** is used to return the text of the button. |
| **3) public void setEnabled(boolean b):** is used to enable or disable the button. |
| **4) public void setIcon(Icon b):** is used to set the specified Icon on the button. |
| **5) public Icon getIcon():** is used to get the Icon of the button. |
| **6) public void setMnemonic(int a):** is used to set the mnemonic on the button. |
| **7) public void addActionListener(ActionListener a):** is used to add the action listener to this object. |

#### Note: The JButton class extends AbstractButton class.

# 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.

### Constructors:

* **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.

### Methods of AbstractButton class:

|  |
| --- |
| **1) public void setText(String s):** is used to set specified text on button. |
| **2) public String getText():** is used to return the text of the button. |
| **3) public void setEnabled(boolean b):** is used to enable or disable the button. |
| **4) public void setIcon(Icon b):** is used to set the specified Icon on the button. |
| **5) public Icon getIcon():** is used to get the Icon of the button. |
| **6) public void setMnemonic(int a):** is used to set the mnemonic on the button. |
| **7) public void addActionListener(ActionListener a):** is used to add the action listener to this object. |

#### Note: The JRadioButton class extends the JToggleButton class that extends AbstractButton class.

|  |
| --- |
| The ButtonGroup class can be used to group multiple buttons so that at a time only one button can be selected. |

JTextArea class:

The JTextArea class is used to create a text area. It is a multiline area that displays the plain text only.

Constructors:

* **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.

methods of JTextArea class:

|  |
| --- |
| **1) public void setRows(int rows):** is used to set specified number of rows. |
| **2) public void setColumns(int cols)::** is used to set specified number of columns. |
| **3) public void setFont(Font f):** is used to set the specified font. |
| **4) public void insert(String s, int position):** is used to insert the specified text on the specified position. |
| **5) public void append(String s):** is used to append the given text to the end of the document. |

JComboBox class:

The JComboBox class is used to create the combobox (drop-down list). At a time only one item can be selected from the item list.

Constructors:

|  |
| --- |
| JComboBox() |
| JComboBox(Object[] items) |
| JComboBox(Vector<?> items) |

methods of JComboBox class:

|  |
| --- |
| **1) public void addItem(Object anObject):** is used to add an item to the item list. |
| **2) public void removeItem(Object anObject):** is used to delete an item to the item list. |
| **3) public void removeAllItems():** is used to remove all the items from the list. |
| **4) public void setEditable(boolean b):** is used to determine whether the JComboBox is editable. |
| **5) public void addActionListener(ActionListener a):** is used to add the ActionListener. |
| **6) public void addItemListener(ItemListener i):** is used to add the ItemListener. |

JTable class

|  |
| --- |
| The JTable class is used to display the data on two dimensional tables of cells. |

Constructors:

* **JTable():** creates a table with empty cells.
* **JTable(Object[][] rows, Object[] columns):** creates a table with the specified data.

JColorChooser:

|  |
| --- |
| The JColorChooser class is used to create a color chooser dialog box so that user can select any color. |

Constructors:

|  |
| --- |
| * **JColorChooser():** is used to create a color chooser pane with white color initially. * **JColorChooser(Color initialColor):** is used to create a color chooser pane with the specified color initially. |

methods of JColorChooser class:

|  |
| --- |
| **public static Color showDialog(Component c, String title, Color initialColor):** is used to show the color-chooser dialog box. |

JProgressBar class:

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

Constructors:

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

methods of JProgressBar class:

|  |
| --- |
| **1) public void setStringPainted(boolean b):** is used to determine whether string should be displayed. |
| **2) public void setString(String s):** is used to set value to the progress string. |
| **3) public void setOrientation(int orientation):** is used to set the orientation, it may be either vertical or horizontal by using SwingConstants.VERTICAL and SwingConstants.HORIZONTAL constants.. |
| **4) public void setValue(int value):** is used to set the current value on the progress bar. |

JSlider class:

|  |
| --- |
| The JSlider is used to create the slider. By using JSlider a user can select a value from a specific range. |

Constructors of JSlider class:

* **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.

Methods of JSlider class:

|  |
| --- |
| **1) public void setMinorTickSpacing(int n):** is used to set the minor tick spacing to the slider. |
| **2) public void setMajorTickSpacing(int n):** is used to set the major tick spacing to the slider. |
| **3) public void setPaintTicks(boolean b):** is used to determine whether tick marks are painted. |
| **4) public void setPaintLabels(boolean b):** is used to determine whether labels are painted. |
| **5) public void setPaintTracks(boolean b):** is used to determine whether track is painted. |