# CHAPTER-7

# GUI Programming in JAVA Using Swing Control -II

**Prepared By:**

**(Cs Basics team)**

# Objective

In this presentation you will learn about the following swing controls i.e. Purpose, their design time Properties and Run time Methods.

- Label (jLabel)
- Simple Dialog (jDialog)
- Text Field (jTextField)
- Password Field (jPasswordField)
- Text Area (jTextArea)
- Scroll Bar (jScrollBar)
- Slider (jSlider)
- Check Box (jCheckBox)
- Radio Buttons (jRadioButton)
- Understanding Focus

# Working with jLabel control

A Label control belongs to jLabel class and used to display non-editable text. The text to be displayed is controlled by text property (design time) and setText() method at run time. jLabel offers the following features- It can

- display Text or Image or both.
- It may have bordered appearance.
- Supports HTML for formatted text.

# Commonly used Properties of jButton

| Properties | Value | Description |
|---|---|---|
| Background | Color | Sets the background color. It works only when opaque is set to True. |
| Border | Border setting as in Jpanel | Specifies the type of Border applies on the boundary of the panel. |
| Foreground | Color | Sets the foreground color. |
| ToolTipText | Text | Sets the text for tooltip. |
| Minimum Size | X, Y values | Defines the minimum width and height |
| Maximum Size | X, Y values | Defines the maximum(x,y) size. |
| Preferred Size | X, Y values | Defines the preferred (x,y) size. |
| Horizontal & VericalAlignment | Right, Trailing, Left, Leading, Center (Horizontal)<br>Top, Bottom , Center (vertical) | Specifies the horizontal and vertical alignment of text. |
| Text | Text | Sets the text to be displayed. |
| Font | Font name and size | Defines the font and size of text. |
| enabled | True/False | Determines whether Active or not |
| Icon | Image file to be displayed | Specifies the image file to be displayed. |
| Horizontal/ Vertical TextPosition | Right, Trailing, Left, Leading, Center<br>Top, Bottom , Center (vertical) | Specifies the horizontal and vertical position of text along with image. |

# Commonly used Methods of jButton

| Methods | Description |
|---|---|
| Void setText(String) | Sets the string of text to be displayed.<br>Ex. jLabel1.setText("I am OK"); |
| String getText() | Returns the text displayed by the label.<br>Ex. String st=jLabel1.getText(); |
| Void setIcon(Image) | Sets the image file to be displayed.<br>Ex. jLabel1.setIcon(new ImageIcon("c:\\abc.gif"); |
| Void setHorizontalAlignment(int)<br>Void setVerticalAlignment(int) | Sets the horizontal/Vertical alignment of text.<br>Ex. jLabel1.setHorizontalAlignment(2);<br>SwingConstants.LEFT / CENTER/ RIGHT etc. can be used. |
| int getHorizontalAlignment( )<br>Int getVerticalAlignment() | Return the horizontal/ vertical alignment status of text.<br>Ex. int a =jLabel1.getHorizontalAlignment(); |
| Void setHorizontalTextposition(int)<br>Void setVerticalTextposition(int)<br><br>int getHorizontalTextposition( )<br>Int getVerticalTextposition() | Sets the horizontal/Vertical position of text to be displayed along with Image.<br>Ex. jLabel1.setHorizontalTextposition(2);<br>SwingConstants.LEFT / CENTER/ RIGHT etc. can be used in place of number.<br>Return the horizontal/ vertical text position status of text.<br>Ex. int a =jLabel1.getHorizontalTextposition(); |

# Displaying Image with jLabel

- Setting up Image at Design Time :-
  - Add jLabel control and click on ellipse ( ) of Icon property in property window.
  - In the dialogue box, select Image chooser option.
  - Specify the path and file name in External Image option.
  - Open source editor and go to top of the code and write-

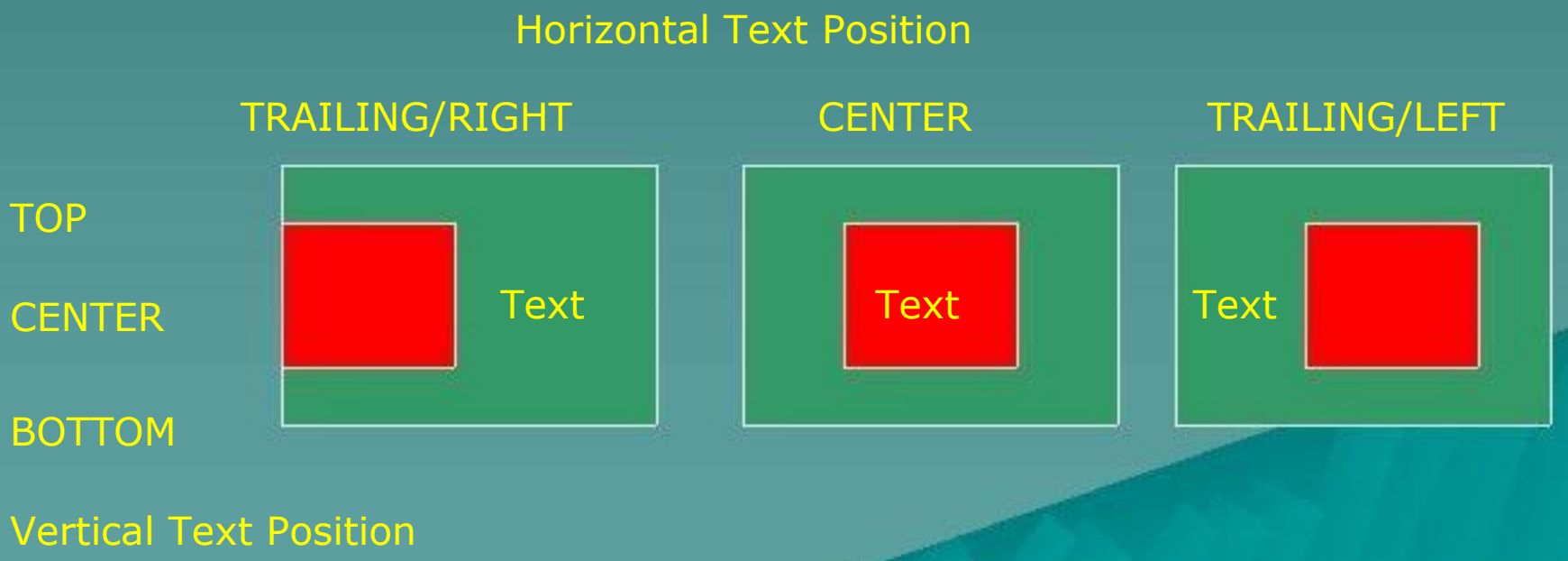
    import javax.swing.ImageIcon;
- Setting up Image at Run time:-
  - Import the javax library by placing following command at top of the code.

    import.javax.swing.ImageIcon;
  - Use the following command in a Event method where image to be displayed or changed.

    jLabel.setIcon(new ImageIcom("c:\\abc.gif"))

# Displaying Text with Image with jLabel

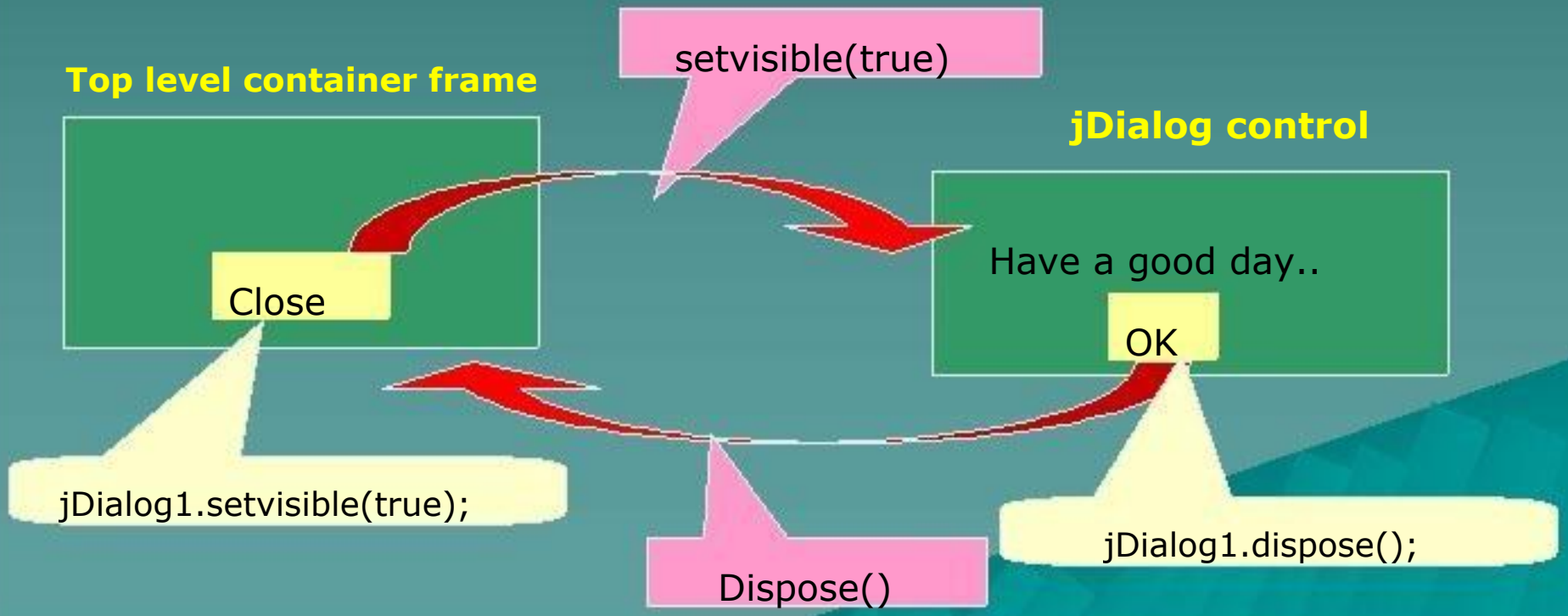
- Specify Text /message  in text propery.
- Set Image file path and file name with icon property.
- Use Horizontal and Vertical text position.
- Use Horizontal and vertical alignment if required.



# Working with Dialog Control

**A dialog control is a control that can be used to display messages in the application. It may contains message , image and buttons etc.**

- **Add jDialog control from swing controls and customize it as per you requirement with text, image and buttons.**
- **It can be invoked in ActionPerformed Event of a button in top level container jframe by jDialog1.setvisible(true) command.**
- **You can set text at run time by jDialog1.setText() method before invoking it.**



# Steps to add Dialog Control (jDailog)

**A dialog control belongs to Swing Window class available in Swing control's palette. The following step may be followed - Design and Application using jFrame and other controls, as you did earlier. Drag jDialog control from Swing Window tab of Swing Tool box.**

- **This will add jDialog1 node under Other component tab in Inspector window.**
- **Now double click jDialog1 node.. In Inspector window, this will open a blank Dialog frame in Design Area.**
- **Attach jLabel and jButton controls as per your choice and customize them Text messages.**
- **Double Click on the jButton control and write the given code in //todo section, to close the dialog box.**

```
jDialog1.dispose()
```

- **Now double on jFrame node in Inspector window to open jFrame control in design area.**
- **Double click on the button on which you want to attach Dialog window and write the command in /TODo section , to invoke the Dialog control.**

**jDialog1.setVisible(true)**

**Now RUN the application.**

# Working with jTextField control

A jTextField is a versatile control, used to get input from user or to display text. It is an object of jTextField class and allow the user to enter a single line of text. jTextField offers the following features-

- You can insert and select text.
- You can scroll the text, if not fit in visible area.
- You can use selected text in other application using clipboard.

# Commonly used Properties of jTextField

| Properties | Value | Description |
|---|---|---|
| Background | Color | Sets the background color. It works only when opaque is set to True. |
| Border | Border setting as in JPanel | Specifies the type of Border applies on the boundary of the panel. |
| Foreground | Color | Sets the foreground color. |
| ToolTipText | Text | Sets the text for tooltip. |
| MinimumSize | X, Y values | Defines the minimum width and height |
| MaximumSize | X, Y values | Defines the maximum(x,y) size. |
| PreferredSize | X, Y values | Defines the preferred (x,y) size. |
| Horizontal Alignment | Right, Trailing, Left, Leading, Center (Horizontal) | Specifies the horizontal and vertical alignment of text. |
| Text | Text | Sets the text to be displayed. |
| Font | Font name and size | Defines the font and size of text. |
| enabled | True/False | Determines whether Active or not |
| editable | True/False | Allow user to edit text, if set to true. |
| focusable | True/False | Allow the text field can receive the focus, if set to true. |
| nextFocusable Component | Component control | Specifies the component that should receive focus, if Tab key pressed |

# Commonly used Methods of jTextField

| Methods | Description |
|---|---|
| Void setText(String) | Sets the string of text to be displayed.<br>Ex. jTextField1.setText("I am OK"); |
| String getText() | Returns the text displayed by the label.<br>Ex. String st=jTextField1.getText(); |
| Void setEditable(Boolean) | Sets the TextField editable.<br>Ex. jTextField1.setEditable(True); |
| boolean isEditable( ) | Returns the setting whether it is editable or not.<br>Ex. Boolean b=jTextField1.isEditable( ); |
| int getHorizontalAlignment( ) | Return the horizontal alignment status of Text field.<br>Ex. int a =jTextField1.getHorizontalAlignment(); |
| Void setHorizontalAlignment(int)<br><br>Void selectAll( ) | Sets the horizontal of text to be displayed.<br>Ex. jTextField1.setHorizontalAlignment(2);<br>jTextFieldLEFT / CENTER/ RIGHT etc. can be used in place of number.<br>Selects all characters in the Text Field.<br>Ex. jTextField1.selectAll(); |

**Most common event of Text Field is Action Event, fired when Enter key is pressed.**

# Working with Password Field

A jPasswordField is a type of Text field that shows encrypted text i.e. actual text is not shown, rather than '*' is displayed.

The character displayed in place of typed character is called echo Character, which is controlled by echoChar property.

# Commonly used Properties of jPasswordField

| Properties | Value | Description |
|---|---|---|
| Background | Color | Sets the background color. It works only when opaque is set to True. |
| Border | Border setting as in Jpanel | Specifies the type of Border applies on the boundary of the panel. |
| Foreground | Color | Sets the foreground color. |
| ToolTipText | Text | Sets the text for tooltip. |
| MinimumSize | X, Y values | Defines the minimum width and height |
| MaximumSize | X, Y values | Defines the maximum(x,y) size. |
| PreferredSize | X, Y values | Defines the preferred (x,y) size. |
| Horizontal Alignment | Right, Trailing, Left, Leading, Center (Horizontal) | Specifies the horizontal and vertical alignment of text. |
| Text | Text | Sets the text to be displayed. |
| Font | Font name and size | Defines the font and size of text. |
| enabled | True/False | Determines whether Active or not |
| editable | True/False | Allow user to edit text, if set to true. |
| nextFocusable Component | Component control | Specifies the component that should receive focus, if Tab key pressed |
| echoChar | Character | Specifies the character to be displayed in place of typed character. |

# Commonly used Methods of jPasswordField

| Methods | Description |
|---|---|
| Void setEchoChar(char) | Sets the echo character.<br>Ex. jPasswordField1.setEchoChar( '#' ); |
| Char getEchoChar( ) | Returns the echo character.<br>Ex. Char st=jPasswordField1.getEchoChar(); |
| Char [] getPassword( )* | Returns the text displayed by the password field.<br>Ex. String pwd= new String (jPasswordField1.getPassword()); |
| Void selectAll() | Selects all characters in the password field<br>Ex. jPasswordField1.selectAll( ); |

* getPassword() method returns a character array, not a string. To store it in a string variable you need to use constructor of string.

## Comparing Strings in JAVA

In Java two strings can not be compared directly, by using = operator.

Two methods (1) equals( ) (2) compareTo( ) are used for this purpose.

Equals() returns TRUE/FALSE but compareTo() returns 0 if both are equal otherwise non-zero value is returned.

**Ex. if (strname.equals("Amit")) {□ □ } else {□ ..}**

# Working with jTextArea control

A jTextArea control is a multi-line text component, used to get input from user or to display text. It is an object of jTextArea class.

By default, it does not wrap (move next line) lines of text like word processor, if line goes beyond the boundary. Some features are-

- You can insert and select multiple line of text.
- You can wrap text, if not fit in visible area.
- You can use selected text in other application using clipboard.

# Commonly used Properties of jTextArea

| Properties | Value | Description |
|---|---|---|
| Background | Color | Sets the background color. |
| Border | Border setting as in Jpanel | Specifies the type of Border applies on the boundary of the panel. |
| Foreground | Color | Sets the foreground color. |
| ToolTipText | Text | Sets the text for tooltip. |
| lineWrap | True/ false | Defines Wrapping feature enable/disable |
| rows | number | Set the number of rows of in text area |
| columns | number | Sets the number of columns |
| tabSize | number | Sets the number of space characters, when tab key pressed. |
| Text | Text | Sets the text to be displayed. |
| Font | Font name and size | Defines the font and size of text. |
| enabled | True/False | Determines whether Active or not |
| editable | True/False | Allow user to edit text, if set to true. |
| focusable | True/False | Allow the text area can receive the focus, if set to true. |
| nextFocusable Component | Component control | Specifies the component that should receive focus. |

Other commonly used properties are-
maximumSize, minimumSize, preferredSize etc.

# Commonly used Methods of jTextArea

| Methods | Description |
|---|---|
| Void setText(String) | Sets the string of text to be displayed.<br>Ex. jTextArea1.setText("I am OK"); |
| String getText() | Returns the text displayed by the label. |
| Void setEditable(Boolean) | Sets the TextArea editable. |
| boolean isEditable( ) | Returns the setting whether it is editable or not.<br>Ex. Boolean b=jTextArea1.isEditable( ); |
| Void setRows(int)<br>Void setColumns( ) | Sets number of rows and columns for the text area.<br>Ex. jTextArea1.setRows(5); |
| int getRows( )<br>Int getColumns( ) | Returns number of rows and columns for the text area.<br>Ex. Int x=jTextArea1.getRows(5); |
| Void selectAll( ) | Selects all characters in the Text Field.<br>Ex. jTextField1.selectAll(); |
| Void setTabSize() | Set the number of spaces for the tab key.<br>Ex: jTextArea1.setTabSize(5); |
| Void append(string) | Adds specified text in the text area.<br>Ex: jTaxtArea1.append("How are you"); |
| Void insert( string, int)<br><br>Void setLineWrap(boolean) | Inserts specified text at given position. Use 0 to insert at top.<br>Ex. jTextArea1.insert("Amit",1);<br>Enables or disables line wrap feature. |

# Working with jScrollBar control

A jScrollBar control belongs to jScrollBar class of Java. Generally it is used to set the input values to the application like Sound slider, Color contrast or brightness etc. Sometimes it is used to input numerical values to an application, in place of Keyboard. There are two types of scroll bars i.e. Horizontal and Vertical, depending on its Orientation or appearance on the Frame.

Scroll bars offers a range of numbers starting from it MINIMUM to MAXIMUM values. The current value of indicator can be used in the application, which can be changes by scrolling the indicator. When user changes the position of indicator, by clicking on Arrows of scroll bar, an Unit Increment is accessed, where as on clicking on scroll area, a Block Increment is occurred.

AdjustmentValueChanged method of Adjustment event is triggered when scrollbar is changed.

Some features of jScrollBar's are-

- It can be used to input some numerical values to the application.
- It may works as a slider to the various Multimedia controls to control the sound and properties of the picture or video.
- It may be Horizontal or Vertical, by setting its Orientation property.

# Commonly used Properties of jScrollBar

| Properties | Value | Description |
|---|---|---|
| enabled | True/False | Determines whether Active or not |
| minimum | Value | Determines the minimum value for the scrollbar |
| maximum | Value | Determines the maximum value |
| Orientation | value | Defines the orientation i.e. Horizontal or Vertical. Default is Vertical. |
| unitIncrement | value | Defines the amount of change when user clicks on Arrows. Default is 1. |
| value | value | Returns current value of indicator. Default is 0. |
| VisibleAmount | value | Defines the width of thumb/Indicator. Default is 10. |
| valueAdjusting | True/False | Thumb/Indicator is being dragged, when set to True. Default is False. |

# Commonly used Methods of jScrollBar

| Methods | Description |
|---|---|
| Void setMinimum(int) | Sets the minimum value for the scroll bar.<br>Ex. jScrollBar1.setMinimum(0); |
| Int getMinimum() | Returns the minimum value of the scroll bar.<br>Ex. int x= jScrollBar1.getMinimum(); |
| Void setMaximum(int) | Sets the maximum value for the scroll bar. |
| Int getMaximum() | Returns the maximum value of the scroll bar. |
| Void setOrientation(int) | Sets the orientation of the scroll bar.(0-vertical, 1- horizontal)<br>Ex. jScrollBar1.setOrientation(1); |
| Int getOrientation() | Returns the current orientation value of the scroll bar. |
| Void setValue(int) | Sets the current value for the scroll bar. |
| Int getValue() | Returns the current value of the scroll bar.<br>Ex. int x= jScrollBar1.getValue(); |
| Void setEnabled(boolean) | Returns true id scrollbar is enabled. |
| Void setUnitIncrement(int) | Sets the unit increment value for the scroll bar.<br>Ex. jScrollBar1.setUnitIncrement(1); |
| Int getUnitIncrement() | Returns the value of unit increment of the scroll bar. |
| Void setBlockIncrement(int)<br><br>Int getBlockIncrement() | Sets the Block increment value for the scroll bar.<br>Ex. jScrollBar1.setBlockIncrement(5);<br>Returns the value of block increment of the scroll bar. |

# Working with jSlider control

A jSlider control belongs to jSlider class of Swing controls. It is much similar with jScrollBar in functioning. The only difference is that, it displays values on it like a scale.

Technically Slider control is a Scroll bar with labeled scale.

Like Scroll bar it may be Horizontal and Vertical depending on its Orientation.

Slider control offers a much cleaner user interface and labeled values of scale from Minimum to Maximum ranges defined by the application.

Some features of jSlider's are

- It can be used to input some numerical values to the application.
- It may works as a scale and may used with various Multimedia controls to control the sound and properties of the picture or video.
- It may be Horizontal or Vertical, by setting its Orientation property.

# Commonly used Properties of jSlider

| Properties | Value | Description |
|---|---|---|
| enabled | True/False | Determines whether Active or not |
| minimum | Value | Determines the minimum value for the slider. (default is 0) |
| maximum | Value | Determines the maximum value (default is 100) |
| Orientation | value | Defines the orientation i.e. Horizontal or Vertical. Default is Horizontal. |
| value | value | Returns current value of slider knob indicator. Default is 50. |
| inverted | True/False | Slider values are listed in descending order (maximum to minimum), if set to true. |
| minerTickSpacing | value | Sets the number of values between the miner tick marks. Default is 0. |
| majorTickSpacing | value | Sets the major tick values between the major tick marks. Default is 0. |
| valueAdjusting | True/False | Thumb/Indicator is being dragged, when set to True. Default is False. |
| paintLabels | True/False | Labels are displayed when set to true. |
| paintTicks | True/False | Ticks are displayed when set to true. |
| paintTrack | True/False | Track of slider appears painted when set to true. |

# Commonly used Methods of jSlider

| Methods | Description |
|---|---|
| Void setMinimum(int) | Sets the minimum value for the slider.<br>Ex. jSlider1.setMinimum(0); |
| Int getMinimum() | Returns the minimum value of the slider.<br>Ex. int x= jSlider1.getMinimum(); |
| Void setMaximum(int) | Sets the maximum value for the slider. |
| Int getMaximum() | Returns the maximum value of the slider. |
| Void setOrientation(int) | Sets the orientation of the scroll bar.(0-vertical, 1- horizontal)<br>Ex. jSlider1.setOrientation(1); |
| Int getOrientation() | Returns the current orientation value of the slider. |
| Void setValue(int) | Sets the current value for the slider. |
| Int getValue() | Returns the current value of the slider.<br>Ex. int x= jSlider1.getValue(); |
| Void setEnabled(boolean) | Sets slider enabled (active), if set to true. |
| int getMajorTickSpacing()<br><br>int getMinerTickSpacing() | Returns the value of Major Tick spacing of the slider bar.<br>Ex. Int x= jSlider1.getMajorTickSpacing();<br>Returns the value of Miner Tick spacing of the slider bar.<br>Ex. Int x= jSlider1.getMinerTickSpacing(); |

# Working with jCheckBox control

A jCheckBox control belongs to jCheckBox class of Swing controls. It indicates whether a particular condition is on or off. You can use Check boxes to give users true/false or yes/no options.

Check Boxes works independently to each other, so that any number of check boxes can be selected at the same time.

- Some features of jCheckBox control's are- It can be used to input True/False or Yes/No typed input to the application.
- Multiple check boxes can be selected at the same time.

# Commonly used Properties of jCheckBox

| Properties | Value | Description |
|---|---|---|
| Background | Color | Sets the background color. |
| Border | Border setting as in Jpanel | Specifies the type of Border applies on the boundary of the panel. |
| Foreground | Color | Sets the foreground color. |
| ToolTipText | Text | Sets the text for tool tip. |
| Minimum Size | X, Y values | Defines the minimum width and height |
| Maximum Size | X, Y values | Defines the maximum(x,y) size. |
| Preferred Size | X, Y values | Defines the preferred (x,y) size. |
| Horizontal TextPosition | Right, Trailing, Left, Leading, Center | Specifies the horizontal position of text along with icon. (default is right) |
| Text | Text | Sets the text to be displayed. |
| Font | Font name and size | Defines the font and size of text. |
| enabled | True/False | Determines whether Active or not |
| mnemonic | Character | Specifies the shortcut (access) key |
| selected | True/false | Check box will be selected, if set to true. (default is false) |
| Focusable | True/False | It can receive Focus, when set to true. |
| nextFocusableComponent | Control | Specifies the next focusable control, when Tab key is pressed. |

# Commonly used Methods of jCheckBox

| Methods | Description |
|---|---|
| Void setText(String) | Sets the string of text to be displayed.<br>Ex. jCheckBox1.setText("Computer"); |
| String getText() | Returns the text displayed by on the check box.<br>Ex. String st=jCheckBox1.getText(); |
| Void setEnabled(boolean) | Sets the check box enables, if true is given.<br>Ex. jCheckBox1.setEnabled(true) |
| Void setVisible(boolean) | Sets the check box visible, if true is given.<br>Ex. jCheckBox1.setVisible(true) |
| Boolean isEnabled( ) | Returns the state whether check box is enabled.<br>Ex. Boolean st=jCheckBox1.isEnabled(true) |
| Boolean isVisible( ) | Returns the state whether check box is visible or not.<br>Ex. Boolean st=jCheckBox1.isVisible(true) |
| Boolean isSelected( ) | Returns the state whether check box is selected or not.<br>Ex. Boolean st=jCheckBox1.isSelected(true) |

# Working with jRadioButton control

A jRadioButton control belongs to jRadioButton class of Swing controls. It is used to get choices from the user. It is grouped control, so that only one can be selected at a time among them.

Check Boxes works in group, so that they must be kept in a container control like jPannel, and only one can be selected at the same time.

Some features of jRadioButton control's are-

- It can be used to input choices typed input to the application.
- Only one Radio button can be selected ata time.

They must be kept in a container control to form a group.

# Commonly used Properties of jRadioButton

| Properties | Value | Description |
|---|---|---|
| Background | Color | Sets the background color. |
| Foreground | Color | Sets the foreground color. |
| buttonGroup | Name of control | Specifies the name of group to which Radio Button belongs. |
| ToolTipText | Text | Sets the text for tool tip. |
| Minimum Size | X, Y values | Defines the minimum width and height |
| Maximum Size | X, Y values | Defines the maximum(x,y) size. |
| Preferred Size | X, Y values | Defines the preferred (x,y) size. |
| Horizontal TextPosition | Right, Trailing, Left, Leading, Center | Specifies the horizontal position of text along with icon. (default is right) |
| Text | Text | Sets the text to be displayed. |
| Font | Font name and size | Defines the font and size of text. |
| enabled | True/False | Determines whether Active or not |
| mnemonic | Character | Specifies the shortcut (access) key |
| selected | True/false | Check box will be selected, if set to true. (default is false) |
| Focusable | True/False | It can receive Focus, when set to true. |
| nextFocusableComponent | Control | Specifies the next focusable control, when Tab key is pressed. |

# Commonly used Methods of jRadioButton

| Methods | Description |
|---|---|
| Void setText(String) | Sets the string of text to be displayed.<br>Ex. jRadioButton1.setText("Science"); |
| String getText() | Returns the text displayed by on the check box.<br>Ex. String st=jRadioNutton1.getText(); |
| Void setEnabled(boolean) | Sets the Radio Button enables, if true is given.<br>Ex. jRadioButton1.setEnabled(true) |
| Void setVisible(boolean) | Sets the Radio Button visible, if true is given.<br>Ex. jRadioButton1.setVisible(true) |
| Boolean isEnabled( ) | Returns the state whether radio button is enabled.<br>Ex. Boolean st=jRadioButton1.isEnabled(true) |
| Boolean isVisible( ) | Returns the state whether Radio Button is visible or not.<br>Ex. Boolean st=jRadioButton1.isVisible(true) |
| Boolean isSelected( ) | Returns the state whether Radio Button is selected or not.<br>Ex. Boolean st=jRadioButton1.isSelected(true) |

# Understanding Focus

- A Focus is the ability to receive user input/ response through Mouse or Keyboard. When object or control has focus, it can receive input from user.
- An object or control can receive focus only if its enabled and visible property are set to true.
- Most of the controls provides FOCUS_GAINED() and FOCUS_LOST() method in FocusEvent by the FocusListener.
- FOCUS_LOST() is generally used for validation of data.

You can give focus to an object at run time by invoking the requestFocus() method in the code.

Ex. jTextBox2.requestFocus();