# Introduction to Java Swing

Last Updated : 30 Jul, 2024

**Swing** is a Java Foundation Classes [JFC] library and an extension of the Abstract Window Toolkit [AWT]. Java Swing offers much-improved functionality over AWT, new components, expanded components features, and excellent event handling with drag-and-drop support.

## Introduction of Java Swing

Swing has about four times the number of User Interface [UI] components as AWT and is part of the standard Java distribution. By today's application GUI requirements, AWT is a limited implementation, not quite capable of providing the components required for developing complex GUIs required in modern commercial applications. The AWT component set has quite a few bugs and does take up a lot of system resources when compared to equivalent Swing resources. Netscape introduced its Internet Foundation Classes [IFC] library for use with Java. Its Classes became very popular with programmers creating GUI's for commercial applications.

- Swing is a Set of API (API- Set of Classes and Interfaces)
- Swing is Provided to Design Graphical User Interfaces
- Swing is an Extension library to the AWT (Abstract Window Toolkit)
- Includes New and improved Components that have been enhancing the looks and Functionality of GUIs'
- Swing can be used to build (Develop) The Standalone swing GUI Apps as Servlets and Applets
- It Employs model/view design architecture.
- Swing is more portable and more flexible than AWT, the Swing is built on top of the AWT.
- Swing is Entirely written in Java.

- Java Swing Components are Platform-independent, and The Swing Components are lightweight.
- Swing Supports a Pluggable look and feel and Swing provides more powerful components.
- such as tables, lists, Scrollpanes, Colourchooser, tabbed pane, etc.
- Further Swing Follows MVC.

## Difference between Java Swing and Java AWT

There are certain points from which Java Swing is different than Java AWT as mentioned below:

| Java AWT | Java Swing |
|---|---|
| Java AWT is an API to develop GUI applications in Java. | Swing is a part of Java Foundation Classes and is used to create various applications. |
| Components of AWT are heavy weighted. | The components of Java Swing are lightweight. |
| Components are platform dependent. | Components are platform independent. |
| Execution Time is more than Swing. | Execution Time is less than AWT. |
| AWT components require java.awt package. | Swing components requires javax.swing package. |

To know more about the topic, refer to Java Swing vs Java AWT.

## What is JFC?

JFC stands for Java Foundation Classes. JFC is the set of GUI components that simplify desktop Applications. Many programmers think that JFC and Swing are one and the same thing, but that is not so. JFC contains Swing [A UI component package] and quite a number of other items:

- Cut and paste: Clipboard support.
- Accessibility features: Aimed at developing GUIs for users with disabilities.
- The Desktop Colors Features were first introduced in Java 1.1
- Java 2D: it has Improved colors, images, and text support.

## Features Of Swing Class

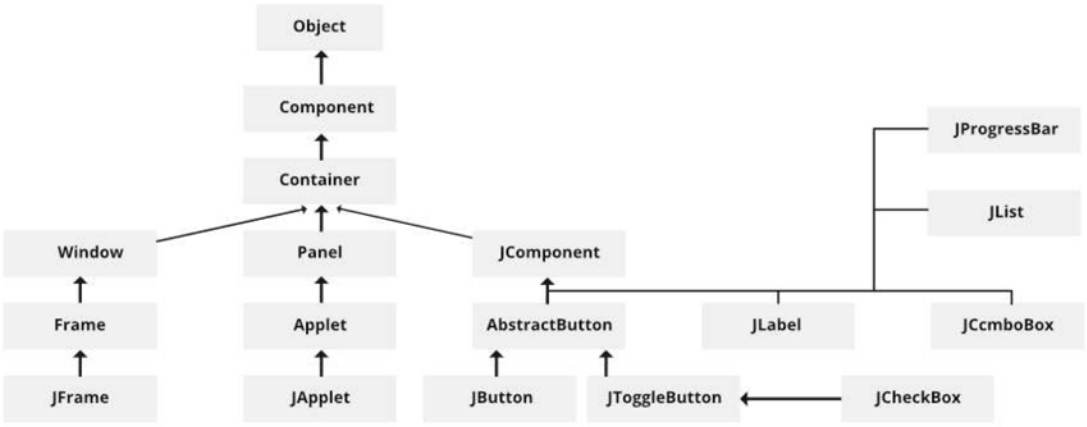
- Pluggable look and feel.
- Uses MVC architecture.
- Lightweight Components
- Platform Independent
- Advanced features such as JTable, JTabbedPane, JScollPane, etc.
- Java is a platform-independent language and runs on any client machine, the GUI look and feel, owned and delivered by a platform-specific O/S, simply does not affect an application's GUI constructed using Swing components.



lightweight component development. For a component to qualify as lightweight, it must not depend on any non-Java [O/s based) system classes. Swing components have their own view supported by Java's look and feel classes.

- **Pluggable Look and Feel:** This feature enable the user to switch the look and feel of Swing components without restarting an application. The Swing library supports components' look and feels that remain the same across all platforms wherever the program runs. The Swing library provides an API that gives real flexibility in determining the look and feel of the GUI of an application
- **Highly customizable** – Swing controls can be customized in a very easy way as visual appearance is independent of internal representation.
- **Rich controls**– Swing provides a rich set of advanced controls like Tree TabbedPane, slider, colorpicker, and table controls.

## Swing Classes Hierarchy




## The MVC Connection

- In general, a visual component is a composite of **three distinct aspects:**
  1. The way that the component looks when rendered on the screen.
  2. The way such that the component reacts to the user.
  3. The state information associated with the component.

- Over the years, one component architecture has proven itself to be exceptionally effective: – **Model-View-Controller** or **MVC** for short.
- In MVC terminology, the **model** corresponds to the state information associated with the Component.
- The **view** determines how the component is displayed on the screen, including any aspects of the view that are affected by the current state of the model.
- The **controller** determines how the component reacts to the user.

**The simplest Swing components have capabilities far beyond AWT components as follows:**

- Swing buttons and labels can be displaying images instead of or in addition to text.
- The borders around most Swing components can be changed easily. For example, it is easy to put a 1-pixel border around the outside of a Swing label.
- Swing components do not have to be rectangular. Buttons, for example, can be round.

- Now The Latest Assertive technologies such as screen readers can easily get information from Swing components. Example: A screen reader tool can easily capture the text that is displayed on a Swing button or label.

## Example of Java Swing Programs

**Example 1: Develop a program using label (swing) to display the message "GFG WEB Site Click":**

Java

```java
// Java program using label (swing)
// to display the message "GFG WEB Site Click"
import java.io.*;
import javax.swing.*;

// Main class
class GFG {

    // Main driver method
    public static void main(String[] args)
    {
        // Creating instance of JFrame
        JFrame frame = new JFrame();

        // Creating instance of JButton
        JButton button = new JButton(" GFG WebSite Click");

        // x axis, y axis, width, height
        button.setBounds(150, 200, 220, 50);

        // adding button in JFrame
        frame.add(button);

        // 400 width and 500 height
        frame.setSize(500, 600);

        // using no layout managers
        frame.setLayout(null);

        // making the frame visible
        frame.setVisible(true);
    }
```

```
33 }
```

Output:

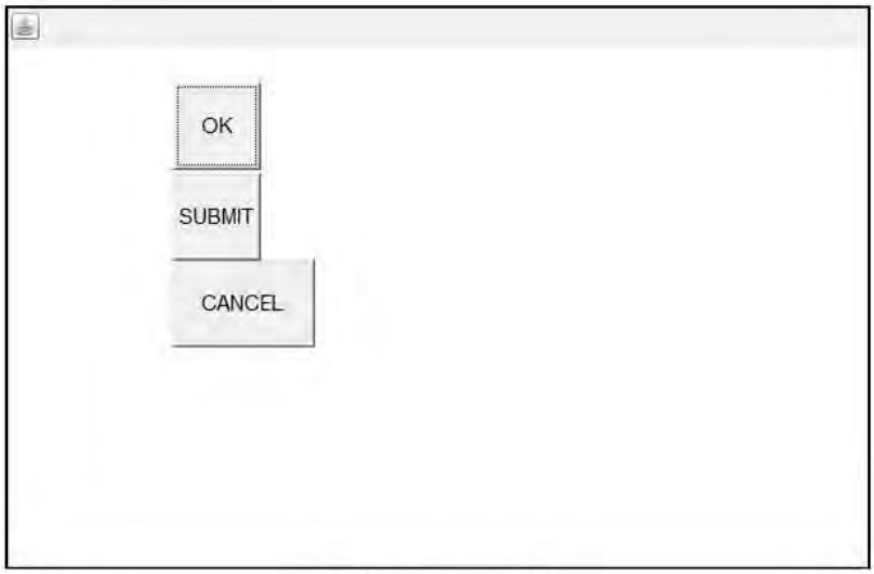

Example 2: Write a program to create three buttons with caption OK, SUBMIT, CANCEL.

Java

```java
// Java program to create three buttons
// with caption OK, SUBMIT, CANCEL
import java.awt.*;

class button {
    button()
    {
        Frame f = new Frame();

        // Button 1 created
        // OK button
        Button b1 = new Button("OK");
        b1.setBounds(100, 50, 50, 50);
        f.add(b1);

        // Button 2 created
        // Submit button
        Button b2 = new Button("SUBMIT");
        b2.setBounds(100, 101, 50, 50);
        f.add(b2);

        // Button 3 created
        // Cancel button
```

```
        Button b3 = new Button("CANCEL");
        b3.setBounds(100, 150, 80, 50);
        f.add(b3);

        f.setSize(500, 500);
        f.setLayout(null);
        f.setVisible(true);
    }

    public static void main(String a[]) { new button(); }
}
```

Output:




*Output of Example 2*

### Example 3: Program to Add Checkbox in the Frame

Java

```
// Java Swing Program to Add Checkbox
// in the Frame
import java.awt.*;

// Driver Class
class Lan {
    // Main Function
    Lan()
    {
        // Frame Created
        Frame f = new Frame();
```

```
12
13          Label l1 = new Label("Select known Languages");
14
15          l1.setBounds(100, 50, 120, 80);
16          f.add(l1);
17
18          // CheckBox created
19          Checkbox c2 = new Checkbox("Hindi");
20          c2.setBounds(100, 150, 50, 50);
21          f.add(c2);
22
23          // CheckBox created
24          Checkbox c3 = new Checkbox("English");
25          c3.setBounds(100, 200, 80, 50);
26          f.add(c3);
27
28          // CheckBox created
29          Checkbox c4 = new Checkbox("marathi");
30          c4.setBounds(100, 250, 80, 50);
31          f.add(c4);
32
33          f.setSize(500, 500);
34          f.setLayout(null);
35          f.setVisible(true);
36      }
37
38      public static void main(String ar[]) { new Lan(); }
39  }
```

Output:

*Output of Example 3*

## Components of Swing Class the task's percentage

| Class | Description |
|---|---|
| Component | A Component is the Abstract base class for about the non-menu user-interface controls of Java SWING. Components are representing an object with a graphical representation. |
| Container | A Container is a component that can container Java SWING Components |
| JComponent | A JComponent is a base class for all swing UI Components In order to use a swing component that inherits from JComponent, the component must be in a containment hierarchy whose root is a top-level Java Swing container. |
| JLabel | A JLabel is an object component for placing text in a container. |
| JButton | This class creates a labeled button. |
| JColorChooser | A JColorChooser provides a pane of controls designed to allow the user to manipulate and select a color. |
| JCheckBox | A JCheckBox is a graphical (GUI) component that can be in either an on-(true) or off-(false) state. |
| JRadioButton | The JRadioButton class is a graphical (GUI) component that can be in either an on-(true) or off-(false) state. in the group |
| JList | A JList component represents the user with the scrolling list of text items. |
| JComboBox | A JComboBox component is Presents the User with a show up Menu of choices. |

| Class | Description |
| --- | --- |
| JTextField | A JTextField object is a text component that will allow for the editing of a single line of text. |
| JPasswordField | A JPasswordField object it is a text component specialized for password entry. |
| JTextArea | A JTextArea object is a text component that allows for the editing of multiple lines of text. |
| ImageIcon | A ImageIcon control is an implementation of the Icon interface that paints Icons from Images |
| JScrollbar | A JScrollbar control represents a scroll bar component in order to enable users to Select from range values. |
| JOptionPane | JOptionPane provides set of standard dialog boxes that prompt users for a value or Something. |
| JFileChooser | A JFileChooser it Controls represents a dialog window from which the user can select a file. |
| JProgressBar | As the task progresses towards completion, the progress bar displays the tasks percentage on its completion. |
| JSlider | A JSlider this class is letting the user graphically (GUI) select by using a value by sliding a knob within a bounded interval. |
| JSpinner | A JSpinner this class is a single line input where the field that lets the user select by using a number or an object value from an ordered sequence. |