Java Swing Tutorial

Java Swing tutorial 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.

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.	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.

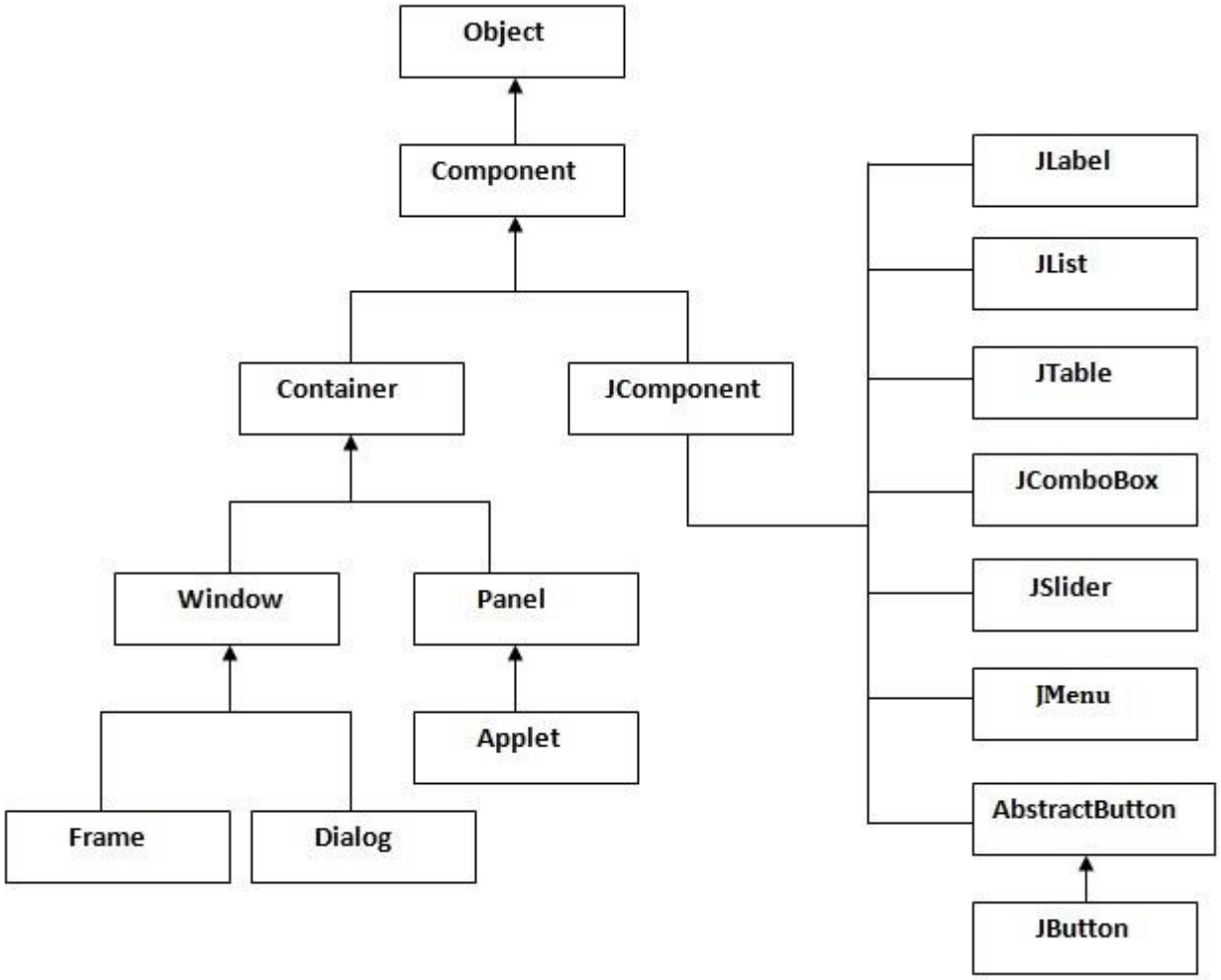
5)	AWT **doesn't follows MVC**(Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing **follows MVC**.

What is JFC

The Java Foundation Classes (JFC) are a set of GUI components which simplify the development of desktop applications.

Hierarchy of Java Swing classes

The hierarchy of java swing API is given below.



Swing is a part of Java Foundation classes (JFC), the other parts of JFC are java2D and Abstract window toolkit (AWT). AWT, Swing & Java 2D are used for building graphical user interfaces (GUIs) in java. In this tutorial we will mainly discuss about Swing API which is used for building GUIs on the top of AWT and are much more light-weight compared to AWT.

A Simple swing example

In the below example we would be using several swing components that you have not learnt so far in this tutorial. We will be discussing each and everything in detail in the coming swing tutorials.

The below swing program would create a login screen.

```
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
public class SwingFirstExample {

    public static void main(String[] args) {
        // Creating instance of JFrame
        JFrame frame = new JFrame("My First Swing Example");
        // Setting the width and height of frame
        frame.setSize(350, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        /* Creating panel. This is same as a div tag in HTML
         * We can create several panels and add them to specific
         * positions in a JFrame. Inside panels we can add text
         * fields, buttons and other components.
         */
        JPanel panel = new JPanel();
        // adding panel to frame
        frame.add(panel);
        /* calling user defined method for adding components
         * to the panel.
         */
        placeComponents(panel);

        // Setting the frame visibility to true
        frame.setVisible(true);
    }

    private static void placeComponents(JPanel panel) {

        /* We will discuss about layouts in the later sections
         * of this tutorial. For now we are setting the layout
         * to null
         */
        panel.setLayout(null);

        // Creating JLabel
        JLabel userLabel = new JLabel("User");
        /* This method specifies the location and size
         * of component. setBounds(x, y, width, height)
         * here (x,y) are cordinates from the top left
         * corner and remaining two arguments are the width
         * and height of the component.
         */
        userLabel.setBounds(10,20,80,25);
        panel.add(userLabel);
```

```
        /* Creating text field where user is supposed to
         * enter user name.
         */
        JTextField userText = new JTextField(20);
        userText.setBounds(100,20,165,25);
        panel.add(userText);

        // Same process for password label and text field.
        JLabel passwordLabel = new JLabel("Password");
        passwordLabel.setBounds(10,50,80,25);
        panel.add(passwordLabel);

        /*This is similar to text field but it hides the user
         * entered data and displays dots instead to protect
         * the password like we normally see on login screens.
         */
        JPasswordField passwordText = new JPasswordField(20);
        passwordText.setBounds(100,50,165,25);
        panel.add(passwordText);

        // Creating login button
        JButton loginButton = new JButton("login");
        loginButton.setBounds(10, 80, 80, 25);
        panel.add(loginButton);
    }

}
```

Output: