Swing: 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:

Java AWT	Java Swing
AWT components are platform-dependent.	Java swing components are platform-independent.
AWT components are heavyweight.	Swing components are lightweight.
AWT doesn't support pluggable look and feel.	Swing supports pluggable look and feel.
AWT provides less components than Swing.	Swing provides more powerful components such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.
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.

There are two ways to create a frame:

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

For Example:

```
import javax.swing.*;
class fs {
public static void main(String[] args) {
JFrame f=new JFrame();//creating instance of JFrame

JButton b=new JButton("click");//creating instance of JButton
b.setBounds(130,100,100, 40);//x axis, y axis, width, height
```

```
f.add(b);//adding button in JFrame

f.setSize(400,500);//400 width and 500 height
f.setLayout(null);//using no layout managers
f.setVisible(true);//making the frame visible
}
}
```

Program: In this program use in the class with constructor.

```
import javax.swing.*;
public class Simple {
JFrame f;
Simple(){
f=new JFrame();//creating instance of JFrame
JButton b=new JButton("click");//creating instance of JButton
b.setBounds(130,100,100, 40);

f.add(b);//adding button in JFrame

f.setSize(400,500);//400 width and 500 height
f.setLayout(null);//using no layout managers
f.setVisible(true);//making the frame visible
}
public static void main(String[] args) {
new Simple();
}
}
```

Program: In this program use in the inheritance in swing.

```
import javax.swing.*;
public class Simple2 extends JFrame
{                                       //inheriting JFrame
JFrame f;
Simple2(){
JButton b=new JButton("click");//create button
b.setBounds(130,100,100, 40);
```

```
add(b);//adding button on frame
setSize(400,500);
setLayout(null);
setVisible(true);
}
public static void main(String[] args) {
new Simple2();
}
}
```

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.

Syntax:

public class JButton extends AbstractButton implements Accessible

For Example:

```
import javax.swing.*;
public class ButtonExample {
public static void main(String[] args) {
    JFrame f=new JFrame("Button Example");
    JButton b=new JButton("Click Here");
    b.setBounds(50,100,95,30);
    f.add(b);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
}
```

Program: In this program use the button in action perform.

```
import java.awt.event.*;
import javax.swing.*;
public class ButtonExample {
public static void main(String[] args) {
    JFrame f=new JFrame("Button Example");
    final JTextField tf=new JTextField();
```

```
    tf.setBounds(50,50, 150,20);
    JButton b=new JButton("Click Here");
    b.setBounds(50,100,95,30);
    b.addActionListener(new ActionListener(){
public void actionPerformed(ActionEvent e){
            tf.setText("Welcome to Javatpoint.");
        }
    });
    f.add(b);f.add(tf);
    f.setSize(400,400);
    f.setLayout(null);

    f.setVisible(true);
}
}
```

Program: In this program use in the button and perform action is print the picture in button

```
import javax.swing.*;
public class ButtonExample{
ButtonExample(){
JFrame f=new JFrame("Button Example");
JButton b=new JButton(new ImageIcon("D:\\icon.png"));
b.setBounds(100,100,100, 40);
f.add(b);
f.setSize(300,400);
f.setLayout(null);
f.setVisible(true);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
public static void main(String[] args) {
    new ButtonExample();
}
}
```

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.

For Example:

```
import javax.swing.*;
class LabelExample
{
public static void main(String args[])
    {
    JFrame f= new JFrame("Label Example");
    JLabel l1,l2;
    l1=new JLabel("First Label.");
    l1.setBounds(50,50, 100,30);
    l2=new JLabel("Second Label.");
    l2.setBounds(50,100, 100,30);
    f.add(l1); f.add(l2);
    f.setSize(300,300);
    f.setLayout(null);
    f.setVisible(true);
    }
  }
```

Program: In this program use the JLabel and perform the action in label.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class LabelExample extends Frame implements ActionListener{
    JTextField tf; JLabel l; JButton b;
    LabelExample(){
        tf=new JTextField();
        tf.setBounds(50,50, 150,20);
        l=new JLabel();
        l.setBounds(50,100, 250,20);
        b=new JButton("Find IP");
        b.setBounds(50,150,95,30);
        b.addActionListener(this);
```

```
        add(b);add(tf);add(l);
        setSize(400,400);
        setLayout(null);
        setVisible(true);
    }
    public void actionPerformed(ActionEvent e) {
        try{
        String host=tf.getText();
        String
ip=java.net.InetAddress.getByName(host).getHostAddress();
        l.setText("IP of "+host+" is: "+ip);
        }catch(Exception ex){System.out.println(ex);}
    }
    public static void main(String[] args) {
        new LabelExample();
    } }
```

JTextField: The object of a JTextField class is a text component that allows the editing of a single line text. It inherits JTextComponent class.Write a program using the function add the two values and print, but without argument and without return value.

For Example:

```
import javax.swing.*;
class TextFieldExample
{
public static void main(String args[])
    {
    JFrame f= new JFrame("TextField Example");
    JTextField t1,t2;
    t1=new JTextField("Welcome to Javatpoint.");
    t1.setBounds(50,100, 200,30);
    t2=new JTextField("AWT Tutorial");
    t2.setBounds(50,150, 200,30);
    f.add(t1); f.add(t2);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
```

```
    }
    }
```

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.

For Example:

```
import javax.swing.*;
public class TextAreaExample
{
     TextAreaExample(){
        JFrame f= new JFrame();
        JTextArea area=new JTextArea("Welcome to javatpoint");
        area.setBounds(10,30, 200,200);
        f.add(area);
        f.setSize(300,300);
        f.setLayout(null);
        f.setVisible(true);
     }
public static void main(String args[])
    {
   new TextAreaExample();
    }}
```

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.

For Example:

```
import javax.swing.*;
public class PasswordFieldExample {
    public static void main(String[] args) {
    JFrame f=new JFrame("Password Field Example");
     JPasswordField value = new JPasswordField();
     JLabel l1=new JLabel("Password:");
        l1.setBounds(20,100, 80,30);
         value.setBounds(100,100,100,30);
            f.add(value);  f.add(l1);
            f.setSize(300,300);
```

```
            f.setLayout(null);
            f.setVisible(true);
}
}
```

Program: In this program use the Jpassword and actionlistener.

```
import javax.swing.*;
import java.awt.event.*;
public class pass1
{
    public static void main(String[] args)
    {
    JFrame f=new JFrame("Password Field Example");
     JLabel l = new JLabel();
     l.setBounds(20,150, 200,50);
     JPasswordField p = new JPasswordField();
     p.setBounds(100,75,100,30);
     JLabel l1=new JLabel("Username:");
        l1.setBounds(20,20, 80,30);
        JLabel l2=new JLabel("Password:");
        l2.setBounds(20,75, 80,30);
        JButton b = new JButton("Login");
        b.setBounds(100,120, 80,30);
        final JTextField t = new JTextField();
        t.setBounds(100,20, 100,30);
                f.add(p); f.add(l1); f.add(l); f.add(l2); f.add(b);
f.add(t);
                f.setSize(300,300);
                f.setLayout(null);
                f.setVisible(true);
                b.addActionListener(new ActionListener()
                 {
                public void actionPerformed(ActionEvent e)
                {
                   String data = "Username " + t.getText();
                   data += ", Password: "
                   + new String(p.getPassword());
```

```
                    l.setText(data);
                }
            });
}
}
```

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.

For Example:

```
import javax.swing.*;
public class radio {
JFrame f;
radio(){
f=new JFrame();
JRadioButton r1=new JRadioButton("A) Male");
JRadioButton r2=new JRadioButton("B) Female");
r1.setBounds(75,50,100,30);
r2.setBounds(75,100,100,30);
ButtonGroup bg=new ButtonGroup();
bg.add(r1);bg.add(r2);
f.add(r1);f.add(r2);
f.setSize(300,300);
f.setLayout(null);
f.setVisible(true);
}
public static void main(String[] args) {
    new radio();
}
}
```

Program: Using ActionListener in RadioButton.

```
import javax.swing.*;
import java.awt.event.*;
class radio1 extends JFrame implements ActionListener
{
```

```
JRadioButton rb1,rb2;
JButton b;
radio1()
{
rb1=new JRadioButton("Male");
rb1.setBounds(100,50,100,30);
rb2=new JRadioButton("Female");
rb2.setBounds(100,100,100,30);
ButtonGroup bg=new ButtonGroup();
bg.add(rb1);bg.add(rb2);
b=new JButton("click");
b.setBounds(100,150,80,30);
b.addActionListener(this);
add(rb1);add(rb2);add(b);
setSize(300,300);
setLayout(null);
setVisible(true);
}
public void actionPerformed(ActionEvent e)
{
if(rb1.isSelected())
{
JOptionPane.showMessageDialog(this,"You are Male.");
}
if(rb2.isSelected())
{
JOptionPane.showMessageDialog(this,"You are Female.");
}
}
public static void main(String args[])
{
new radio1();
}
}
```

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

For Example:

```
import javax.swing.*;
public class tabel {
    JFrame f;
    tabel(){
    f=new JFrame();
    String data[][]={ {"101","Amit","670000"},
                      {"102","Jai","780000"},
                      {"101","Sachin","700000"}};
    String column[]={"ID","NAME","SALARY"};
    JTable jt=new JTable(data,column);
    jt.setBounds(50,60,200,300);
    JScrollPane sp=new JScrollPane(jt);
    f.add(sp);
    f.setSize(500,500);
    f.setVisible(true);
}
public static void main(String[] args)
 {
    new tabel();
}
}
```

Layout: Java provides various layout managers to position the controls. Properties like size, shape, and arrangement vary from one layout manager to the other. When the size of the applet or the application window changes, the size, shape, and arrangement of the components also changes in response, i.e. the layout managers adapt to the dimensions of the appletviewer or the application window. The layout manager is associated with every Container object. Each layout manager is an object of the class that implements the LayoutManager interface.

Different types of layouts:

FlowLayout: The Java FlowLayout class is used to arrange the components in a line, one after another (in a flow). It is the default layout of the applet or panel.

For Example:

```
import java.awt.*;
```

```
import javax.swing.*;

public class flow{
JFrame f;
flow(){
    f=new JFrame();

    JButton b1=new JButton("1");
    JButton b2=new JButton("2");
    JButton b3=new JButton("3");
    JButton b4=new JButton("4");
    JButton b5=new JButton("5");

     // adding buttons to the frame
    f.add(b1); f.add(b2); f.add(b3); f.add(b4); f.add(b5);

     // setting flow layout of right alignment
    f.setLayout(new FlowLayout(FlowLayout.LEFT,20,25));

    f.setSize(300,300);
    f.setVisible(true);
}
public static void main(String[] args) {
    new flow();
}
}
```

GridLayout: The Java GridLayout class is used to arrange the components in a rectangular grid. One component is displayed in each rectangle.

For Example:

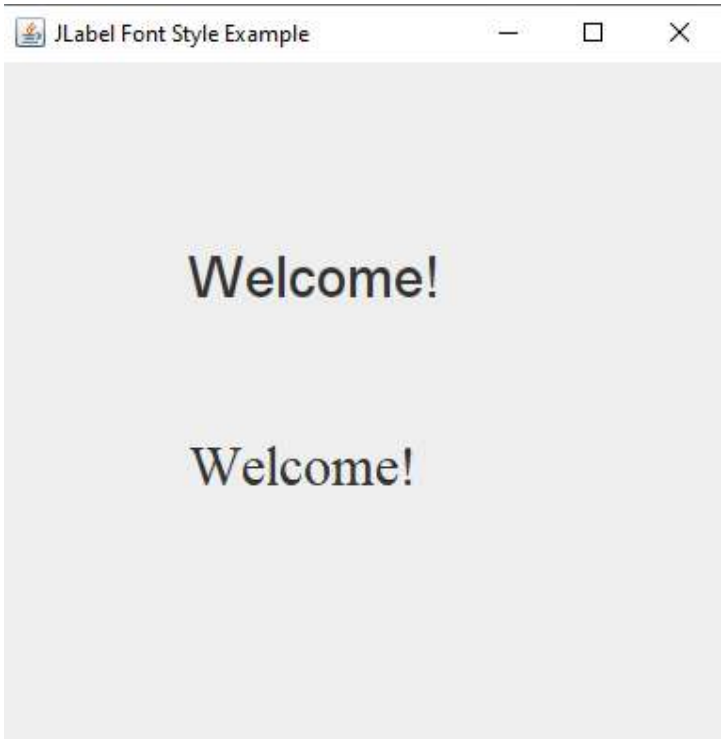
```
import java.awt.*;
import javax.swing.*;

public class flow{
JFrame f;
flow(){
```

```
    f=new JFrame();

    JButton b1=new JButton("1");
    JButton b2=new JButton("2");
    JButton b3=new JButton("3");
    JButton b4=new JButton("4");
    JButton b5=new JButton("5");

     // adding buttons to the frame
    f.add(b1); f.add(b2); f.add(b3); f.add(b4); f.add(b5);

     // setting flow layout of right alignment
    f.setLayout(new GridLayout(3,3,50,50));

    f.setSize(300,300);
    f.setVisible(true);
}
public static void main(String[] args) {
    new flow();
}
}
```

For Example: In this example use the styling in Lable.

```
//Necessary Imports
import java.awt.*;
import javax.swing.*;

public class FontSyle extends JFrame {
    public static void main(String[] args) {

        JFrame frame = new JFrame("JLabel Font Style Example"); //
Creating a new JFrame with a title
```

```
        frame.setLayout(null); // To terminate the default flow
layout
        frame.setVisible(true); // To display the frame

frame.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE); // To
Terminate JFrame and the program on close
        frame.setBounds(100, 200, 400, 400); // To set the bounds of
the frame.

        Container c = frame.getContentPane(); //To get content pane
layer and add objects in the frame.

        //Creating First JLabel Object
        JLabel obj1 = new JLabel("Welcome!"); //Creating a JLabel
with title as Welcome
        obj1.setBounds(100, 100, 300, 30); //Setting the bounds of
the label.
        obj1.setFont(new Font("Arial", Font.PLAIN, 30)); //Creating
an Arial Font Style with size 30

        //Creating Second Jlabel Object with different font
        JLabel obj2 = new JLabel("Welcome!"); //Creating same JLabel
title as before.
        obj2.setBounds(100, 200, 300, 30); //Setting the same bounds
as before.
        obj2.setFont(new Font("Times New Roman", Font.PLAIN, 30));
//Creating an Times New Roman Font Style with size 30

        //Adding the objects in the container frame
        c.add(obj1);
        c.add(obj2);
    }
}
```


JLabel Font Style Example
Welcome!
Welcome!