JAVA SWING CLASSES FOR GRAPHICAL USER INTERFACE IN JAVA

Java Swing 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.

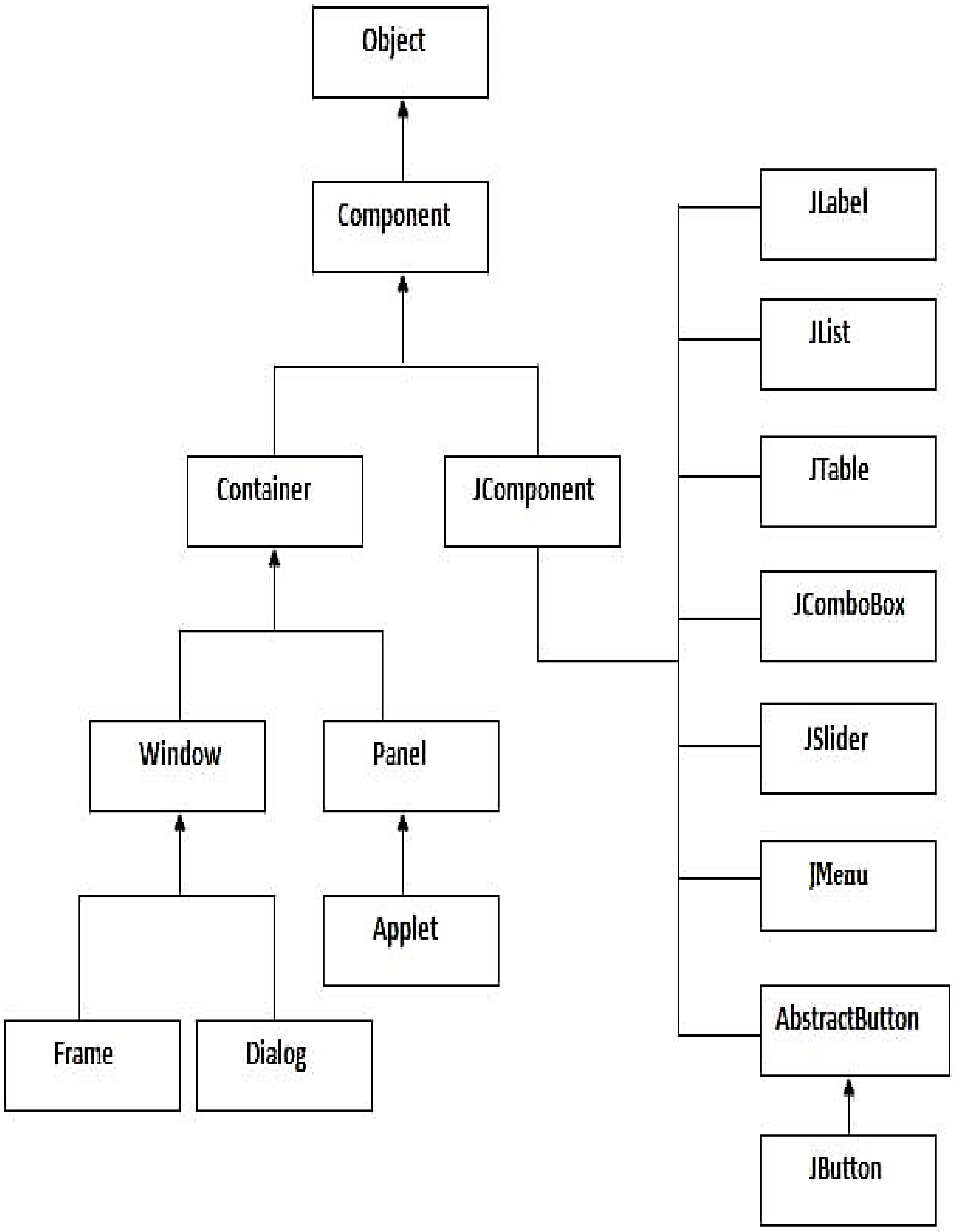
Java AWT	Java Swing
AWT components are **platform-dependent**.	Java swing components are **platform-independent**.
AWT components are **heavyweight**.	Swing components are **lightweight**.
AWT **doesn't support pluggable look and feel**.	Swing **supports pluggable look and feel**.
AWT provides **less 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**.

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.



Java Swing Examples

There are two ways to create a frame:

1. By extending Frame class (inheritance)
2. By creating the object of Frame class (association)

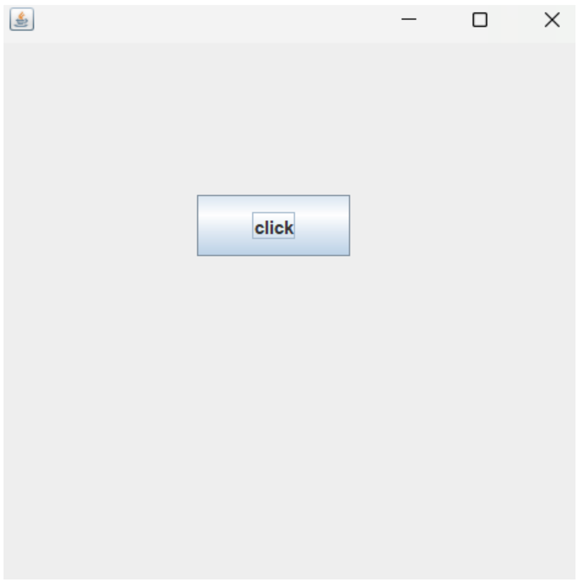
We can write the code of swing inside the main(), constructor or any other method.

Simple example of Swing by inheritance

We can also inherit the JFrame class, so there is no need to create the instance of JFrame class explicitly.

File: Simple2.java

```
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();
}
}
```

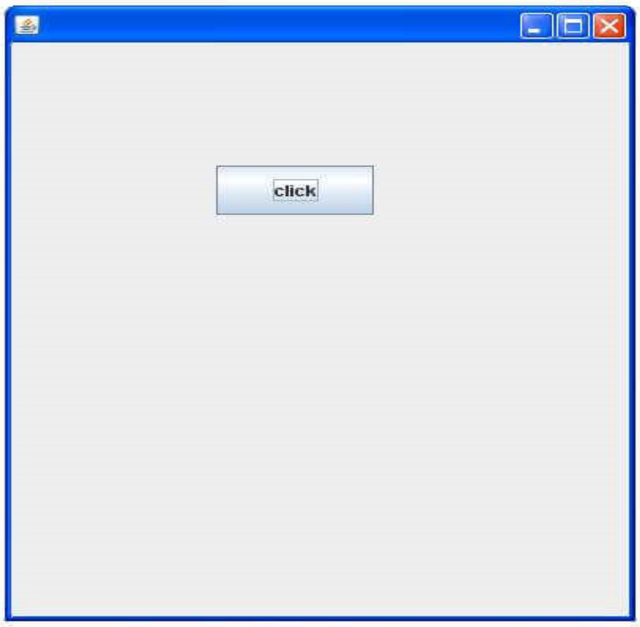


Simple Java Swing Example

Let's see a simple swing example where we are creating one button and adding it on the JFrame object inside the main() method.

File: FirstSwingExample.java

```
import javax.swing.*;
public class FirstSwingExample {
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
}
}
```

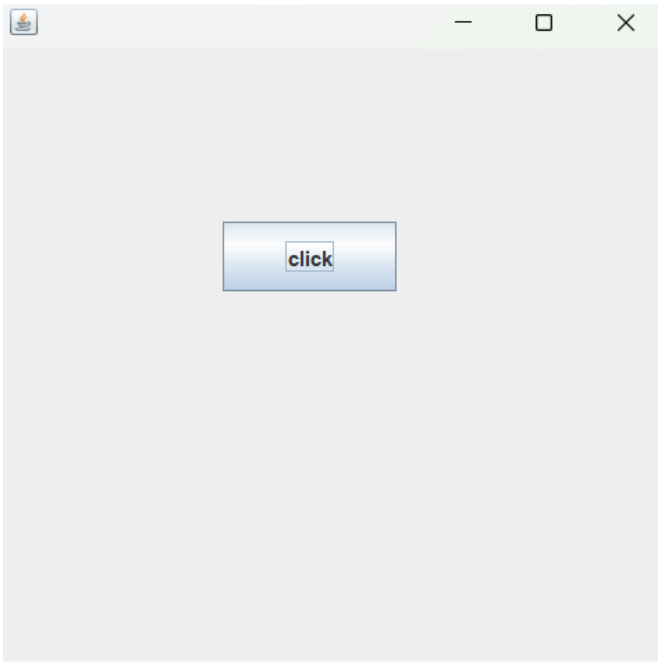


Example of Swing by Association inside constructor

We can also write all the codes of creating JFrame, JButton and method call inside the java constructor.

File: Simple.java

```
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();

}

}
```



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

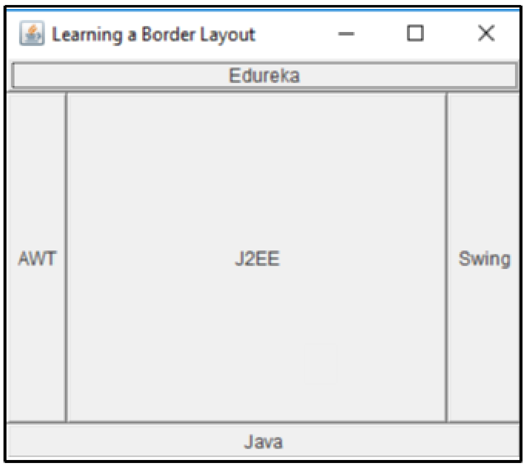
Layout Manager

To arrange the components inside a container we use the layout manager. Following are several layout managers:

1. Border layout
2. Flow layout
3. GridBag layout

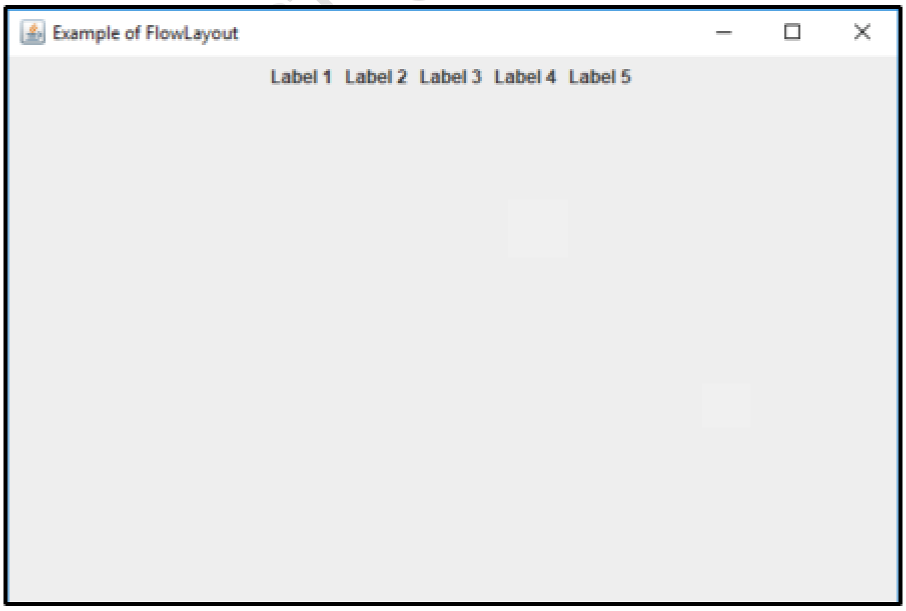
Border Layout

The default layout manager for every JFrame is BorderLayout. It places components in upto five places which is top, bottom, left, right and center.



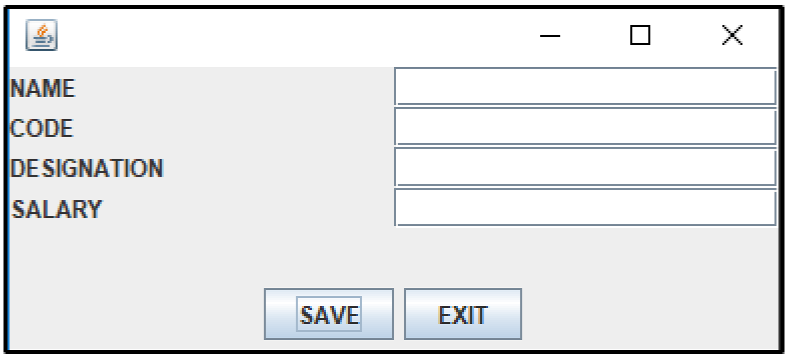
Flow Layout

FlowLayout simply lays the components in a row one after the other, it is the default layout manager for every JPanel.



GridBag Layout

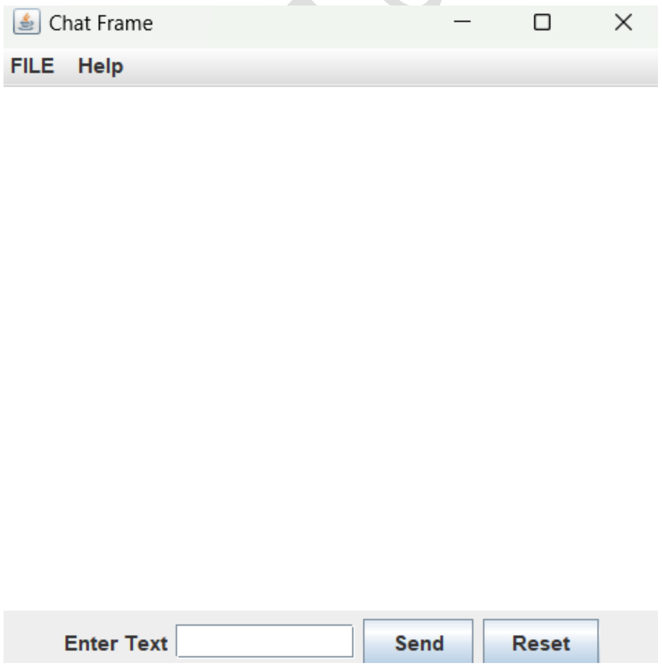
GridBagLayout places the components in a grid which allows the components to span more than one cell.



A simple example for creating a GUI using swing in Java.

Example: Chat Frame

```
import javax.swing.*;

import java.awt.*;

class Example {

    public static void main(String args[]) {


        JFrame frame = new JFrame("Chat Frame");

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        frame.setSize(400, 400);


        JMenuBar ob = new JMenuBar();

        JMenu ob1 = new JMenu("FILE");

        JMenu ob2 = new JMenu("Help");

        ob.add(ob1);

        ob.add(ob2);

        JMenuItem m11 = new JMenuItem("Open");

        JMenuItem m22 = new JMenuItem("Save as");

        ob1.add(m11);
        ob1.add(m22);


        JPanel panel = new JPanel(); // the panel is not visible in output
        JLabel label = new JLabel("Enter Text");
        JTextField tf = new JTextField(10); // accepts upto 10 characters
        JButton send = new JButton("Send");
        JButton reset = new JButton("Reset");
        panel.add(label); // Components Added using Flow Layout
        panel.add(label); // Components Added using Flow Layout
        panel.add(tf);
        panel.add(send);
        panel.add(reset);
        JTextArea ta = new JTextArea();


        frame.getContentPane().add(BorderLayout.SOUTH, panel);
        frame.getContentPane().add(BorderLayout.NORTH, ob);
        frame.getContentPane().add(BorderLayout.CENTER, ta);
        frame.setVisible(true);
    } }
```



Class Practice

JButton Class

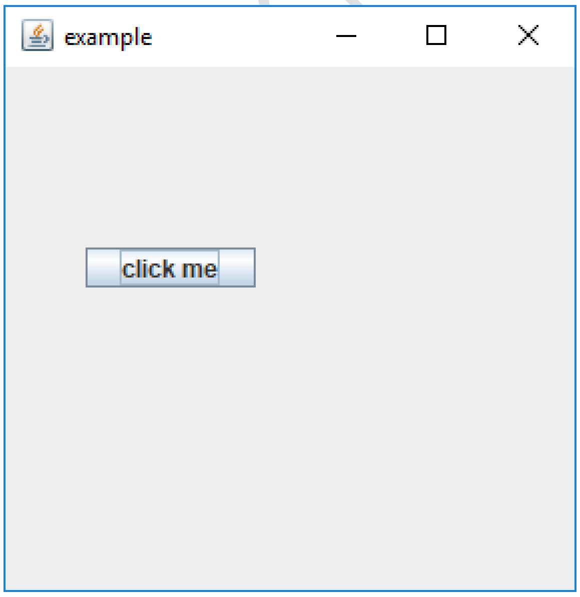
It is used to create a labelled button. Using the ActionListener it will result in some action when the button is pushed. It inherits the AbstractButton class and is platform independent.

Example:

```
import javax.swing.*;

public class example{

public static void main(String args[]) {

JFrame a = new JFrame("example");

JButton b = new JButton("click me");

b.setBounds(40,90,85,20);

a.add(b);

a.setSize(300,300);

a.setLayout(null);

a.setVisible(true);

}

}
```

Output:

JButton - Java Swing - Edureka



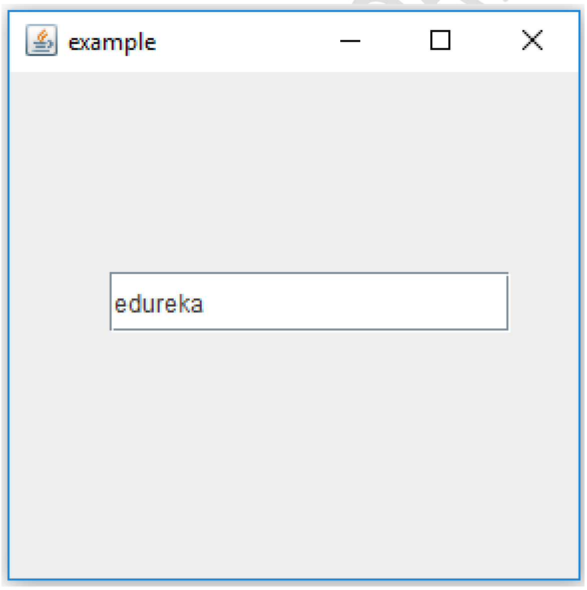
JTextField Class

It inherits the JTextComponent class and it is used to allow editing of single line text.

Example:

```
import javax.swing.*;
public class example{
public static void main(String args[]) {
JFrame a = new JFrame("example");
JTextField b = new JTextField("edureka");
b.setBounds(50,100,200,30);
a.add(b);
a.setSize(300,300);
a.setLayout(null);
a.setVisible(true);
}
}
```

Output:



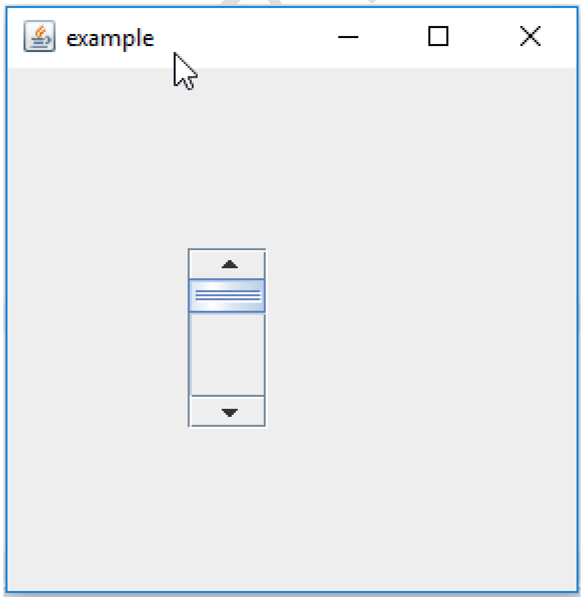
JScrollBar Class

It is used to add scroll bar, both horizontal and vertical.

Example:

```
import javax.swing.*;
class example{
example(){
JFrame a = new JFrame("example");
JScrollBar b = new JScrollBar();
b.setBounds(90,90,40,90);
a.add(b);
a.setSize(300,300);
a.setLayout(null);
a.setVisible(true);
}
public static void main(String args[]){
new example();
}
}
```

Output:

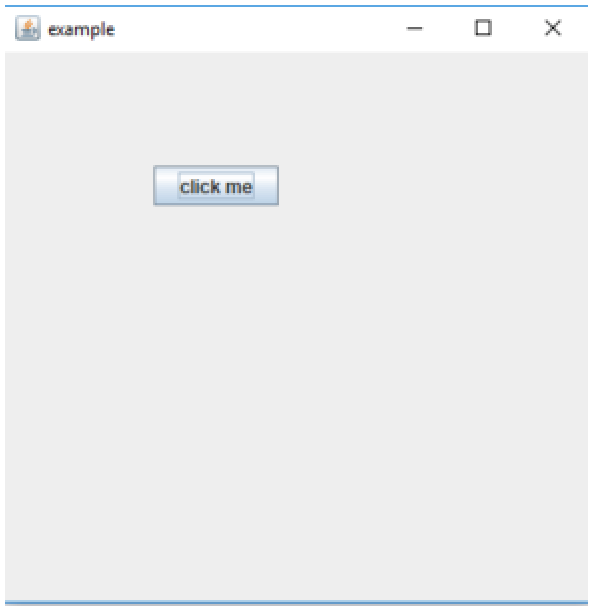


JPanel Class

It inherits the JComponent class and provides space for an application which can attach any other component.

```
import java.awt.*;
import javax.swing.*;
public class Example{
Example(){
JFrame a = new JFrame("example");
JPanel p = new JPanel();
p.setBounds(40,70,200,200);
JButton b = new JButton("click me");
b.setBounds(60,50,80,40);
p.add(b);
a.add(p);
a.setSize(400,400);
a.setLayout(null);
a.setVisible(true);
}
public static void main(String args[])
{
new Example();
}
}
```

Output:

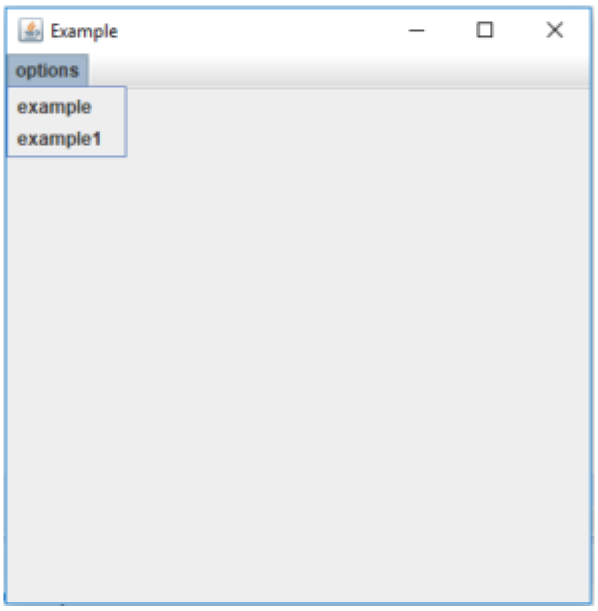


JMenu Class

It inherits the JMenuItem class, and is a pull down menu component which is displayed from the menu bar.

```
import javax.swing.*;
class Example{
JMenu menu;
JMenuItem a1,a2;
Example()
{
JFrame a = new JFrame("Example");
menu = new JMenu("options");
JMenuBar m1 = new JMenuBar();
a1 = new JMenuItem("example");
a2 = new JMenuItem("example1");
menu.add(a1);
menu.add(a2);
m1.add(menu);
a.setJMenuBar(m1);
a.setSize(400,400);
a.setLayout(null);
a.setVisible(true);
}
public static void main(String args[])
{
new Example();
}
}
```

Output:

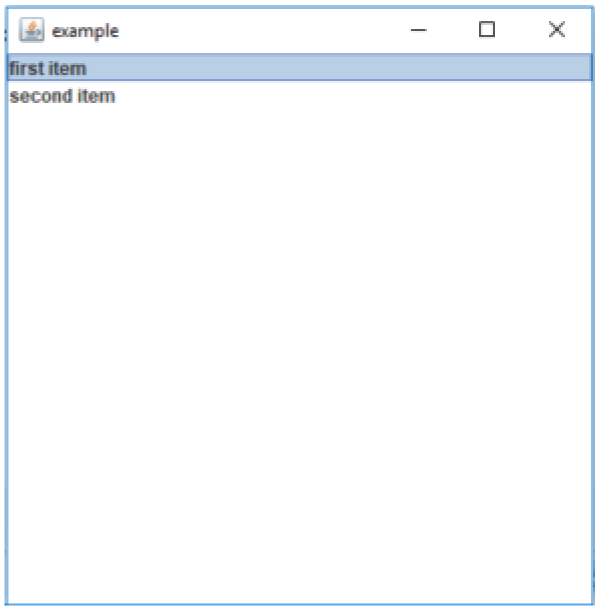


JList Class

It inherits JComponent class, the object of JList class represents a list of text items.

```
import javax.swing.*;
public class Example
{
Example(){
JFrame a  = new JFrame("example");
DefaultListModel<String> l = new DefaultListModel< >();
l.addElement("first item");
l.addElement("second item");
JList<String> b = new JList< >(l);
b.setBounds(100,100,75,75);
a.add(b);
a.setSize(400,400);
a.setVisible(true);
a.setLayout(null);
}
public static void main(String args[])
{
new Example();
}
}
```

Output:



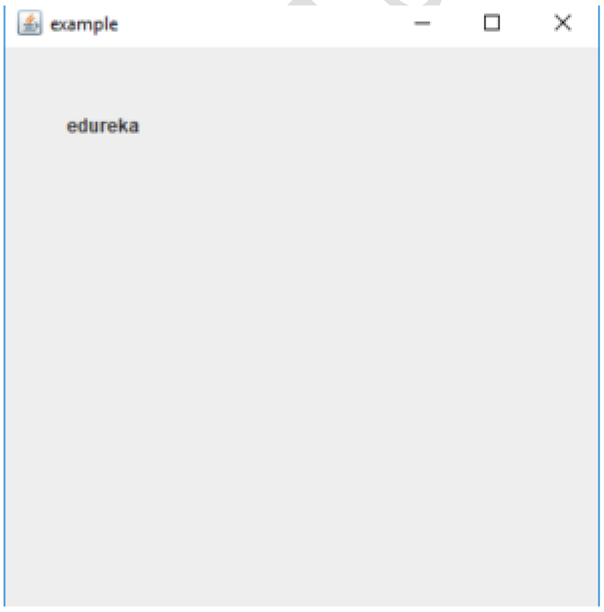
```
first item
second item
```

JLabel Class

It is used for placing text in a container. It also inherits JComponent class.

```
import javax.swing.*;
public class Example{
public static void main(String args[])
{
JFrame a = new JFrame("example");
JLabel b1;
b1 = new JLabel("edureka");
b1.setBounds(40,40,90,20);
a.add(b1);
a.setSize(400,400);
a.setLayout(null);
a.setVisible(true);
}
}
```

Output:

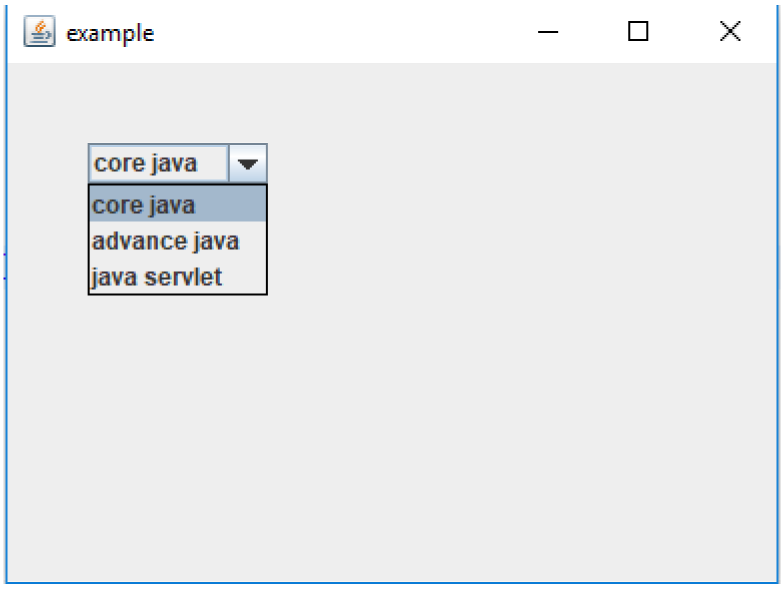


JComboBox Class

It inherits the JComponent class and is used to show pop up menu of choices.

```
import javax.swing.*;
public class Example{
JFrame a;
Example(){
a = new JFrame("example");
string courses[] = { "core java","advance java", "java servlet"};
JComboBox c = new JComboBox(courses);
c.setBounds(40,40,90,20);
a.add(c);
a.setSize(400,400);
a.setLayout(null);
a.setVisible(true);
}
public static void main(String args[])
{
new Example();
}
}
```

Output:


example
core java
core java
advance java
java servlet