

User Interface Design (Applet & Swing)

Dr. Susovan Jana

Associate Professor & Programme In-Charge
Department of Computer Science & Engineering (IoT)
Institute of Engineering & Management, Kolkata, INDIA

Email (O): susovan.jana@iem.edu.in

Email (P): jana.susovan2@gmail.com

Applet in Java

Dr. Susovan Jana

Associate Professor & Assistant HOD

Department of Computer Science & Engineering (IoT)
Institute of Engineering & Management, Kolkata, INDIA

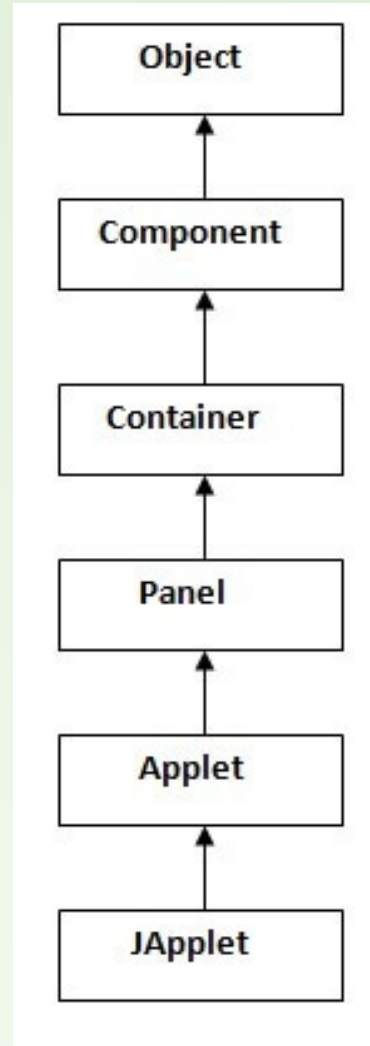
Email (O): susovan.jana@iem.edu.in

Email (P): jana.susovan2@gmail.com

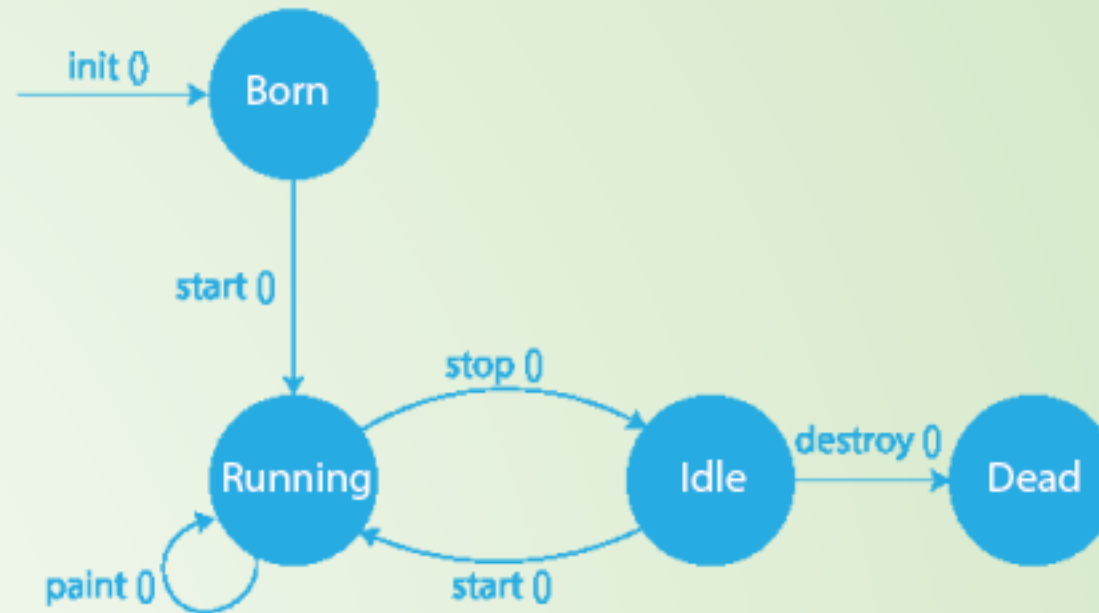
What is Applet?

- An applet is a program written in the Java programming language that can be included in an HTML page, much in the same way an image is included in a page.

Hierarchy of Applet Class



Life Cycle of Applet



Life Cycle of Applet

- ❑ `public void init():` is used to initialize the Applet. It is invoked only once.
- ❑ `public void start():` is invoked after the `init()` method or browser is maximized. It is used to start the Applet.
- ❑ `public void stop():` is used to stop the Applet. It is invoked when Applet is stopped or browser is minimized.
- ❑ `public void destroy():` is used to destroy the Applet. It is invoked only once.
- ❑ `public void paint(Graphics g):` is used to paint the Applet. It provides Graphics class object that can be used for drawing oval, rectangle, arc etc.

Applet Program (JAVA File): First.java

```
import java.applet.Applet;  
import java.awt.Graphics;  
public class First extends Applet{  
    public void paint(Graphics g){  
        g.drawString("welcome",150,150);  
    }  
}
```

Applet Program (HTML File): myapplet.html

```
<html>
```

```
<body>
```

```
<applet code="First.class" width="300" height="300">
```

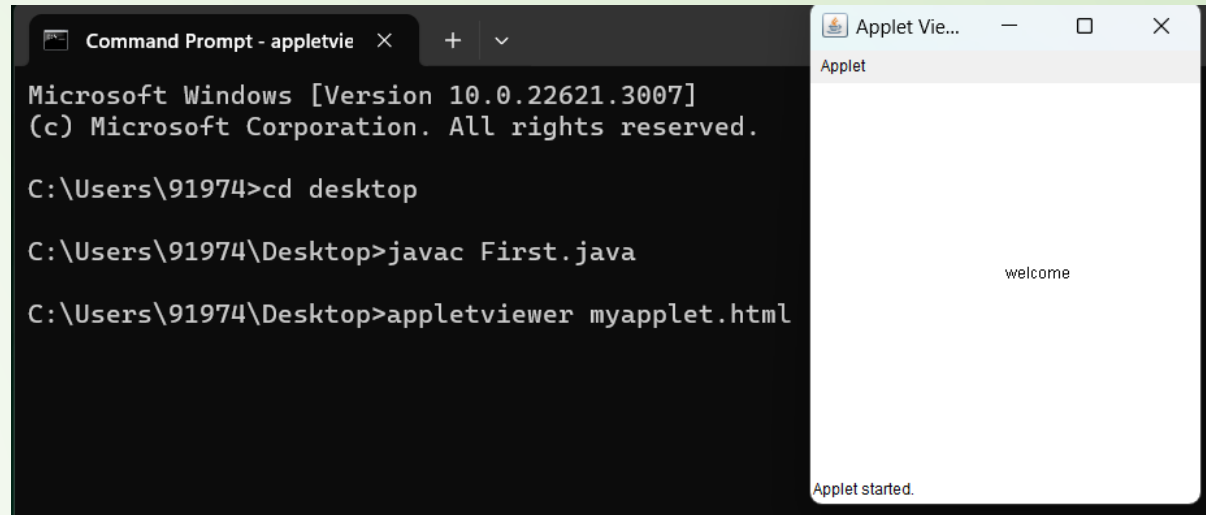
```
</applet>
```

```
</body>
```

```
</html>
```


Execution Steps

- ❑ Step 1: Compile the .java File
 - > javac First.java
- ❑ Step 2: Keep both the .class and .html file in same directory
- ❑ Step 3: Run applet viewer to display applet
 - > appletviewer myapplet.html



The screenshot shows a Windows Command Prompt window titled "Command Prompt - appletvie" and an Applet Viewer window titled "Applet Vie...". The Command Prompt displays the following commands and output:

```
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.

C:\Users\91974>cd desktop

C:\Users\91974\Desktop>javac First.java

C:\Users\91974\Desktop>appletviewer myapplet.html
```

The Applet Viewer window shows a white area with the text "welcome" in the center. At the bottom of the window, it says "Applet started."

Display Graphics in Applet

❑ `import java.awt.Graphics`

Methods under Graphics Class

- ❑ `public abstract void drawString(String str, int x, int y):` is used to draw the specified string.
- ❑ `public void drawRect(int x, int y, int width, int height):` draws a rectangle with the specified width and height.
- ❑ `public abstract void fillRect(int x, int y, int width, int height):` is used to fill rectangle with the default color and specified width and height.
- ❑ `public abstract void drawOval(int x, int y, int width, int height):` is used to draw oval with the specified width and height.
- ❑ `public abstract void fillOval(int x, int y, int width, int height):` is used to fill oval with the default color and specified width and height.

Methods under Graphics Class

- ❑ `public abstract void drawLine(int x1, int y1, int x2, int y2)`: is used to draw line between the points(x1, y1) and (x2, y2).
- ❑ `public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer)`: is used draw the specified image.
- ❑ `public abstract void drawArc(int x, int y, int width, int height, int startAngle, int arcAngle)`: is used draw a circular or elliptical arc.
- ❑ `public abstract void fillArc(int x, int y, int width, int height, int startAngle, int arcAngle)`: is used to fill a circular or elliptical arc.
- ❑ `public abstract void setColor(Color c)`: is used to set the graphics current color to the specified color.
- ❑ `public abstract void setFont(Font font)`: is used to set the graphics current font to the specified font.

GraphicsDemo.java

```
import java.applet.Applet;  
import java.awt.*;  
public class GraphicsDemo extends Applet{  
    public void paint(Graphics g){  
        g.setColor(Color.red);  
        g.drawString("Welcome",50, 50);  
        g.drawLine(20,30,20,300);  
        g.drawRect(70,100,30,30);  
        g.fillRect(170,100,30,30);  
        g.drawOval(70,200,30,30);  
        g.setColor(Color.pink);  
        g.fillOval(170,200,30,30);  
        g.drawArc(90,150,30,30,30,270);  
        g.fillArc(270,150,30,30,0,180);  
    }  
}
```

myapplet.html

```
<html>
```

```
<body>
```

```
<applet code="GraphicsDemo.class" width="300"  
height="300">
```

```
</applet>
```

```
</body>
```

```
</html>
```

Swing in Java

Dr. Susovan Jana

Associate Professor & Assistant HOD

Department of Computer Science & Engineering (IoT)
Institute of Engineering & Management, Kolkata, INDIA

Email (O): susovan.jana@iem.edu.in

Email (P): jana.susovan2@gmail.com

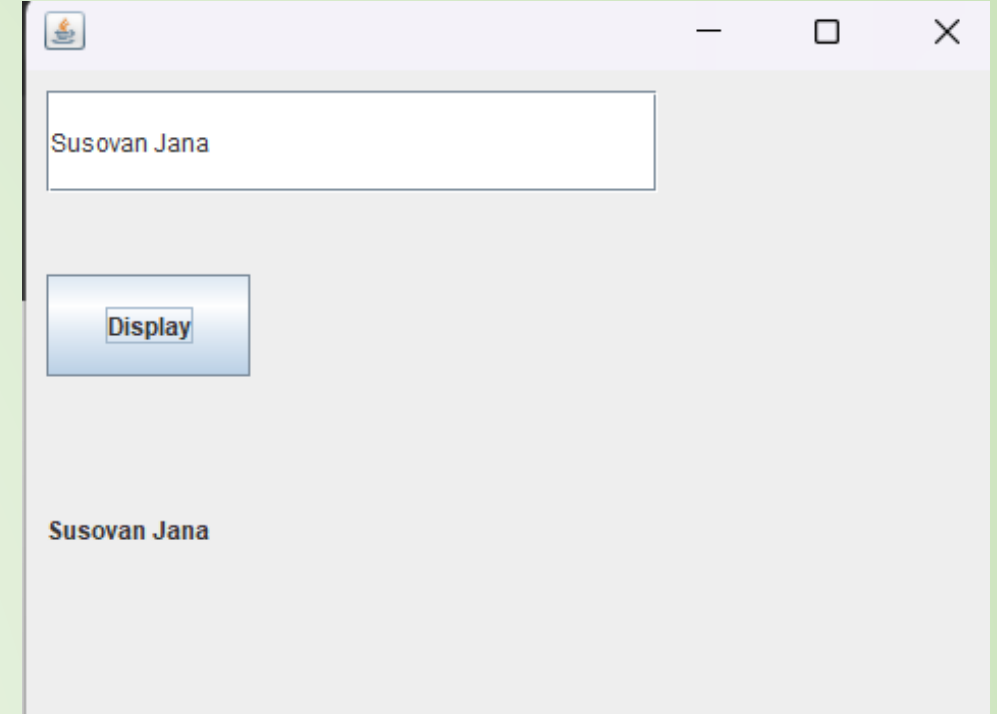
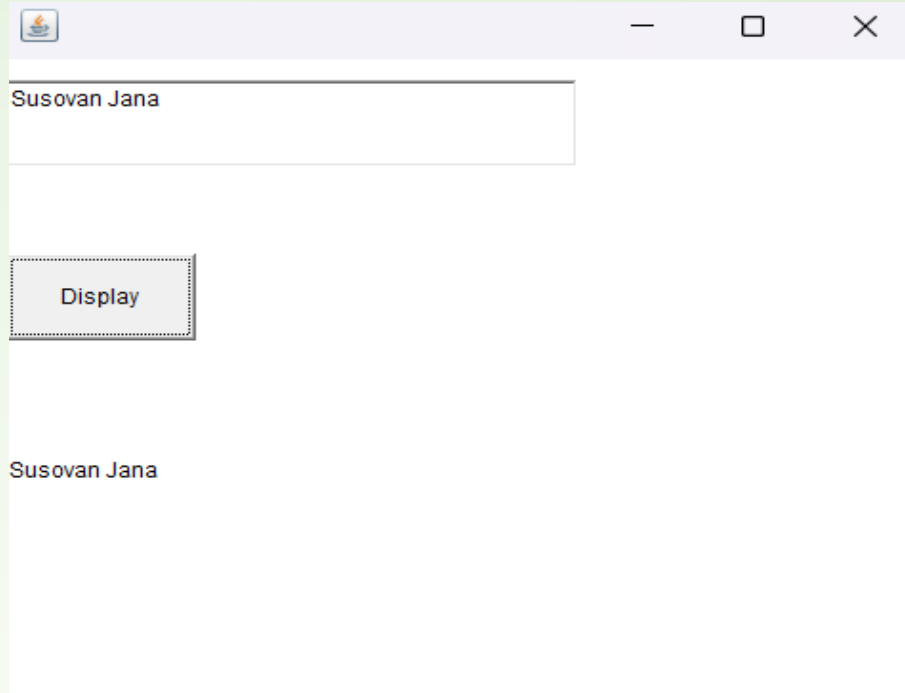
What is swing?

- ❑ It is used to do the GUI.
- ❑ It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.

AWT vs Swing

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 .

AWT vs Swing



Commonly used Methods of Component class

Method	Description
<code>public void add(Component c)</code>	add a component on another component.
<code>public void setSize(int width,int height)</code>	sets size of the component.
<code>public void setLayout(LayoutManager m)</code>	sets the layout manager for the component.
<code>public void setVisible(boolean b)</code>	sets the visibility of the component. It is by default false.

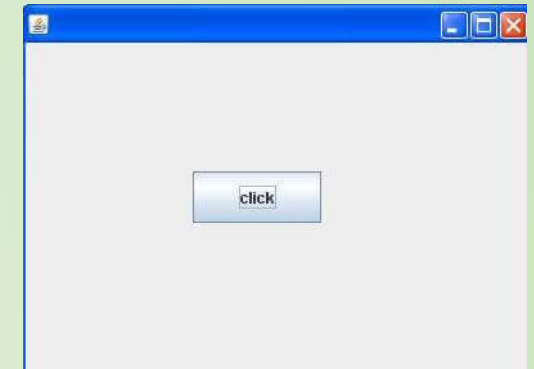
Creating a frame in Java

- Two ways to create a frame
 - By creating the object of Frame class (association)
 - By extending Frame class (inheritance)

Creating object of JFrame class

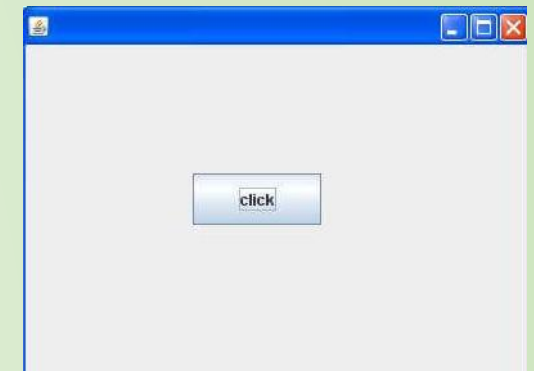
```
import javax.swing.*;

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
    } }
```



Extending Frame class

```
import javax.swing.*;
public class Simple extends JFrame{//inheriting JFrame
Simple(){
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 Simple();
}}
```



JButton

Constructor	Description
JButton()	It creates a button with no text and icon.
JButton(String s)	It creates a button with the specified text.
JButton(Icon i)	It creates a button with the specified icon object.

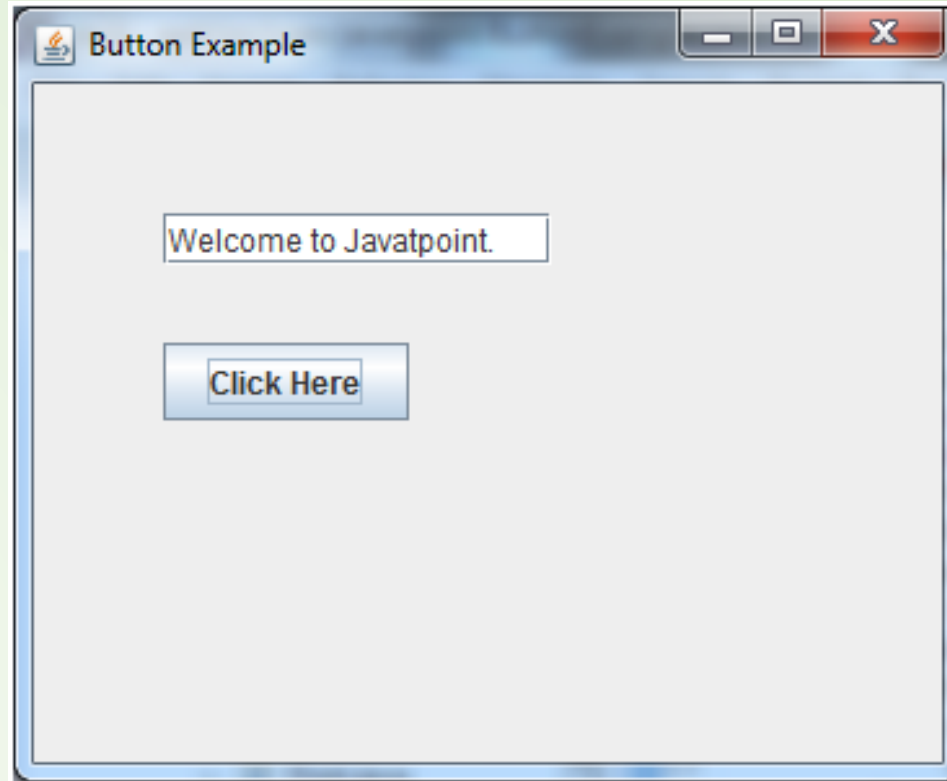
JButton with ActionListener

```
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 Java");
            }
        });
        f.add(b);f.add(tf);
        f.setSize(400,400);
        f.setLayout(null);
        f.setVisible(true);
    } }
```


JButton with ActionListener



Example of displaying image on the 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 & JButton

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

public class Demoswing extends JFrame implements ActionListener{
    JTextField jt;
    JButton jb;
    JLabel jl;

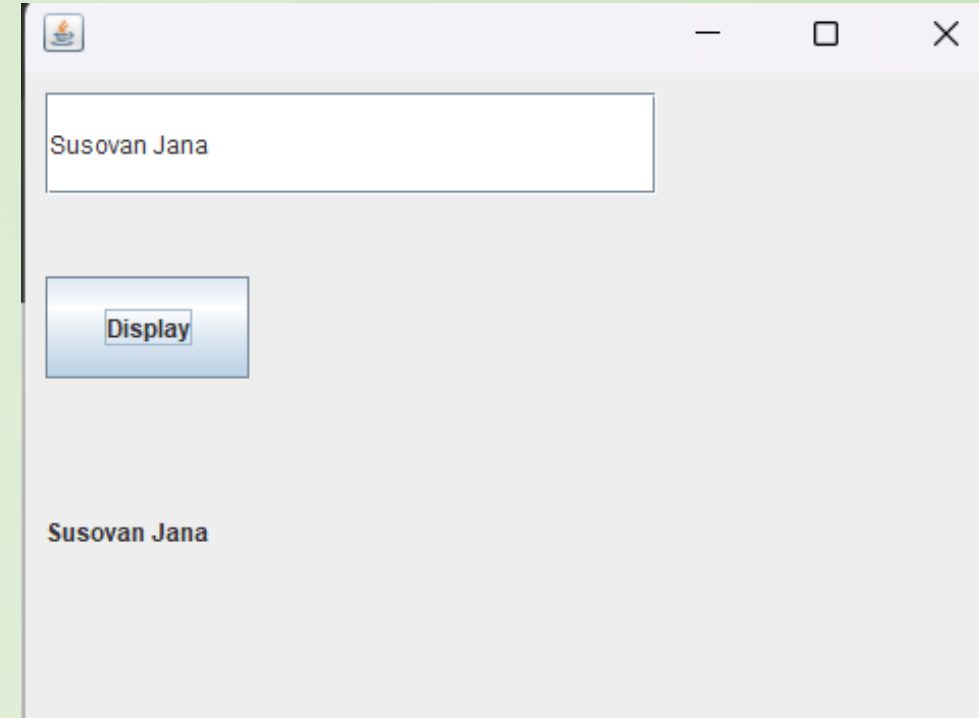
    Demoswing()
    {
        jt=new JTextField();
        jt.setBounds(10,10,300,50);
        jb=new JButton("Display");
        jb.setBounds(10,100,100,50);
        jl=new JLabel();
        jl.setBounds(10,200,300,50);
        add(jt);
        add(jb);
        add(jl);
    }
}
```

JLabel & JButton

```
jb.addActionListener(this);  
setSize(500,500);  
setLayout(null);  
setVisible(true);  
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
}
```

```
public void actionPerformed(ActionEvent e){  
    jl.setText(jt.getText());  
}
```

```
public static void main(String args[])  
{Demoswing ds=new Demoswing();}  
}
```



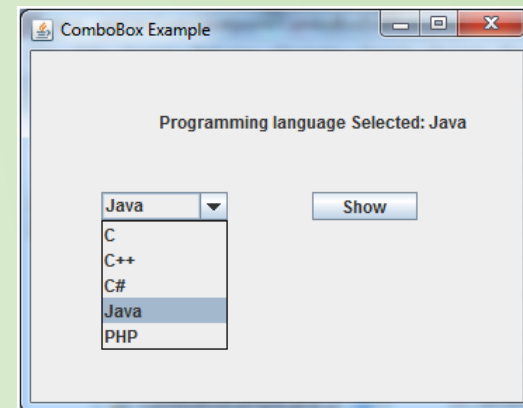
JComboBox with ActionListener

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

public class ComboBoxExample {
    JFrame f;
    ComboBoxExample(){
        f=new JFrame("ComboBox Example");
        final JLabel label = new JLabel();
        label.setHorizontalAlignment(JLabel.CENTER);
        label.setSize(400,100);
        JButton b=new JButton("Show");
        b.setBounds(200,100,75,20);
        String languages[]={"C","C++","C#","Java","PHP"};
        final JComboBox cb=new JComboBox(languages);
        cb.setBounds(50, 100,90,20);
        f.add(cb); f.add(label); f.add(b);
```

```
f.setLayout(null);
f.setSize(350,350);
f.setVisible(true);
b.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String data = "Programming language Selected: "
            + cb.getItemAt(cb.getSelectedIndex());
        label.setText(data);
    }
});

public static void main(String[] args) {
    new ComboBoxExample();
}
```

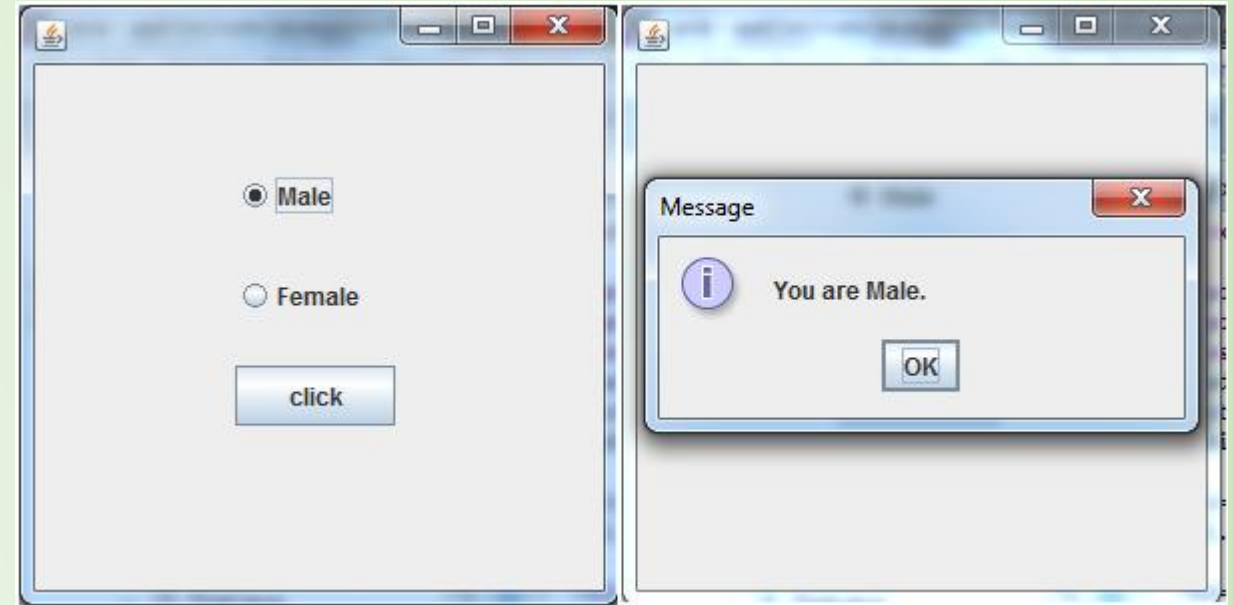


JRadioButton Example

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

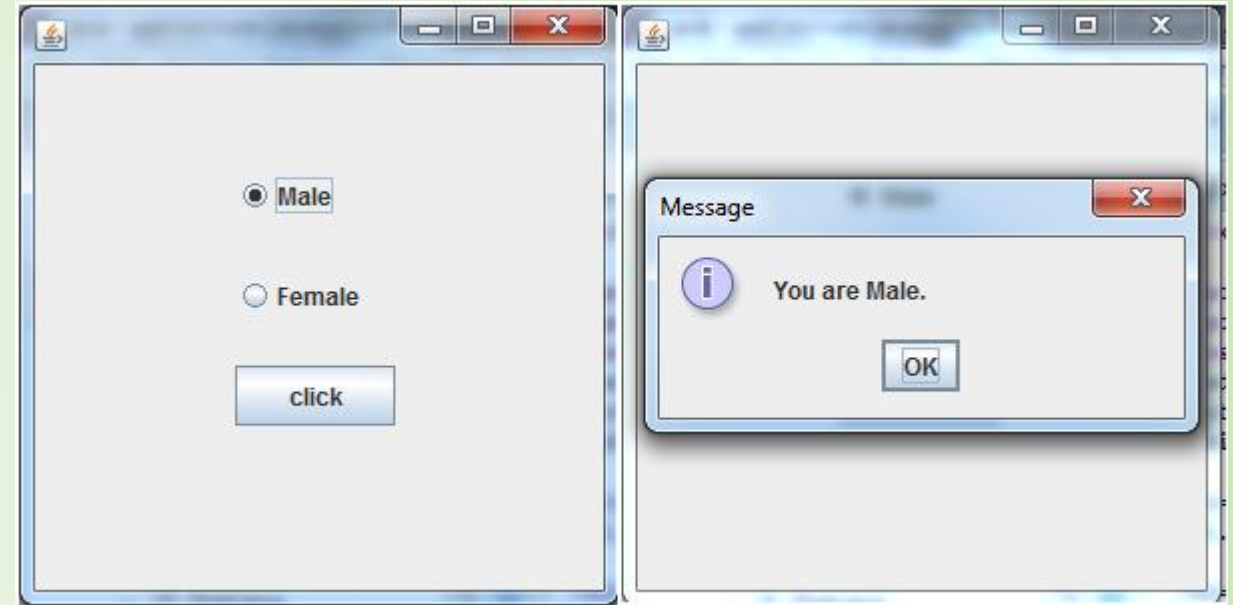
class RadioButtonExample extends JFrame implements ActionListener{
    JRadioButton rb1,rb2;
    JButton b;

    RadioButtonExample(){
        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);
    }
}
```



JRadioButton Example

```
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 RadioButtonExample();  
}}
```



Assignment 1: Basic Calculator

- ❑ **Objective:** Create a simple calculator with a GUI.
- ❑ **Requirements:**
 - Use JFrame, JTextField, JButton, and JLabel.
 - Provide buttons for numbers (0-9) and basic operations (+, -, *, /).
 - Display results in a JTextField.
 - Implement action listeners for button clicks.

Assignment 2: Student Registration Form

- ❑ **Objective:** Design a registration form for students.
- ❑ **Requirements:**
 - Use JLabel, JTextField, JRadioButton (for gender selection), JComboBox (for department selection), and JButton.
 - Add a "Submit" button that displays the entered information in a message dialog (JOptionPane).
 - Validate that no fields are left empty.

Assignment 3: To-Do List Application

❑ **Objective:** Develop a basic to-do list GUI.

❑ **Requirements:**

- Use JTextField for entering tasks.
- Use a JButton to add tasks to a JList.
- Add a "Remove" button to delete selected tasks.
- Store tasks in an ArrayList and update the list dynamically.

Assignment 4: Traffic Light Simulation

- ❑ **Objective:** Simulate a traffic light system using Swing.
- ❑ **Requirements:**
 - Use three JRadioButton controls for "Red," "Yellow," and "Green."
 - When a button is selected, change the background color of a JPanel to match the light.
 - Use ButtonGroup to ensure only one selection at a time.

Assignment 5: Simple Login System

- ❑ **Objective:** Create a GUI-based login system.
- ❑ **Requirements:**
 - Use JLabel, JTextField, JPasswordField, and JButton.
 - Verify username and password (use predefined credentials).
 - Show a message dialog (JOptionPane) with "Login Successful" or "Invalid Credentials."
 - Clear fields if login fails.

