Object - Oriented Programming 2

Week 7. Graphical User Interface (Swing, Swing Components Part-1 and Part-2)

By Elubu Joseph - MSc.IS

Lecturer

Department of Information Technology

Kumi University

Email: josebulinda@gmail.com

jose@kumiuniversity.ac.ug

Agenda

1. Graphical User Interface (Swing,
2. Swing Components Part-1 based on code and
3. Swing components Part-2 based on an IDE)

Java Swing

Java Swing is a GUI Framework that contains a set of classes to provide more powerful and flexible GUI components than **AWT. Swing** provides the look and feel of modern Java GUI. Swing library is an official Java GUI tool kit released by Sun Microsystems. It is used to create graphical user interface with Java.

Swing classes are defined in **javax.swing** package and its sub-packages.

Features of Swing

Java swing has 7 features including; -

1. Platform Independent
2. Customizable
3. Extensible
4. Configurable
5. Lightweight
6. Rich Controls
7. Pluggable Look and Feel

Swing and JFC

Java Foundation classes- JFC are classes which encompass a group of features for building GUI and adding rich graphical functionalities and interactivity to Java applications. **Java Swing** is a part of Java Foundation Classes (JFC).

Features of JFC

1. Swing GUI components.
2. Look and Feel support.
3. Java 2D. is an API for drawing two-dimensional graphics using the Java programming language,(Wikimedia Foundation,2022)

Swing Components Part-1

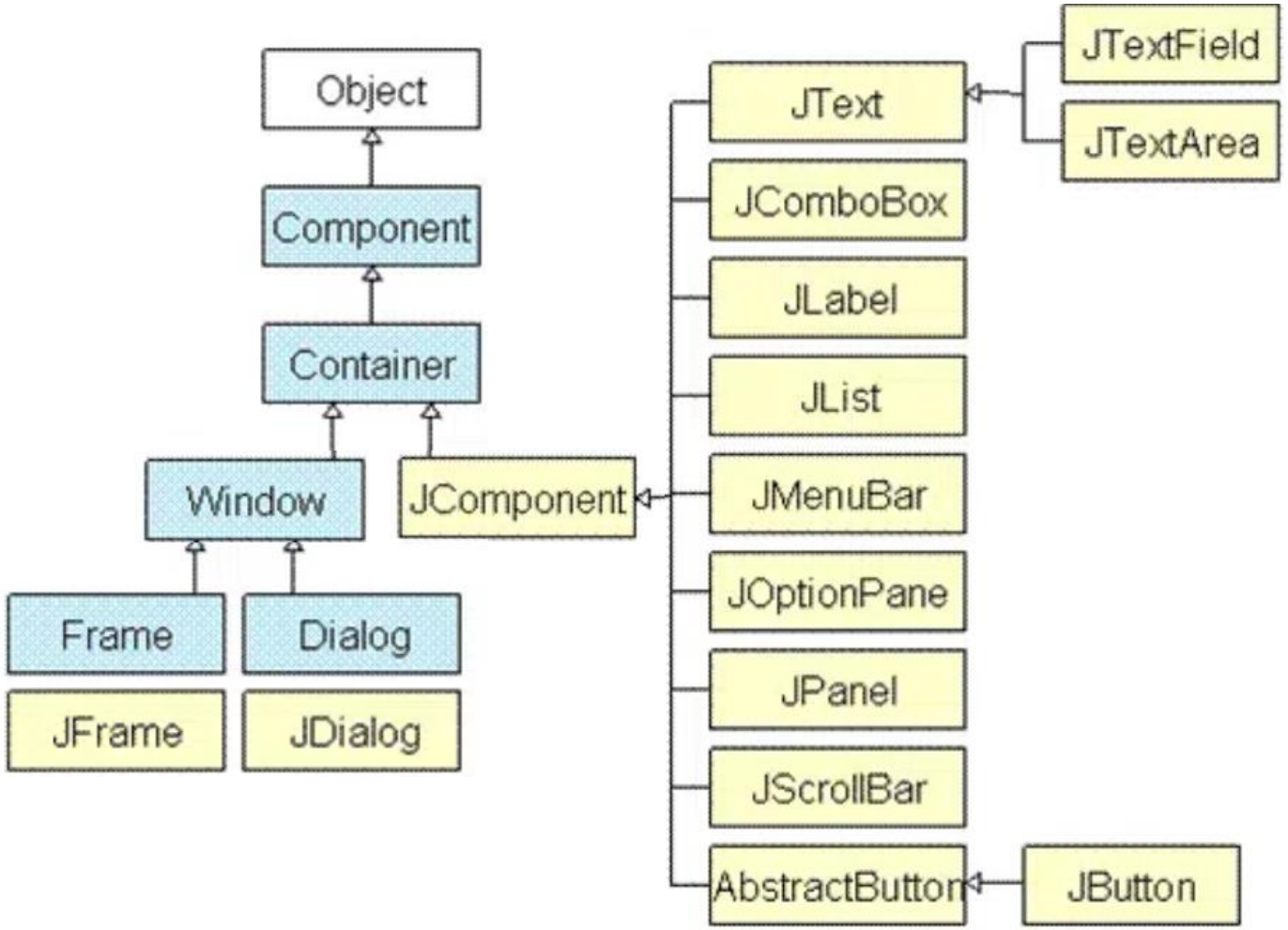
Swing Components Part-1

In this section, we look at the origin of swing components, look at various swing components as we use them to demonstrate and add functionalities on our GUI Applications.

The major concern here is the difference between the Abstract Windowing Toolkit-AWT and Swing packages.

AWT and Swing Hierarchy

Java Swing was build on AWT.



Note! Whatever is having a name starting with letter 'J' is a Swing Classes. While those without J are AWT Classes

(studytonight.com)

A glance inside Swing Classes

1. **JPanel :** JPanel is Swing's version of AWT class Panel and uses the same default layout, FlowLayout. JPanel is descended directly from JComponent.
2. **JFrame :** JFrame is Swing's version of Frame and is descended directly from **Frame** class. The component which is added to the **Frame**, is refered as its Content.
3. **JWindow :** This is Swing's version of Window and has descended directly from **Window** class. Like **Window** it uses BorderLayout by default.
4. **JLabel :** JLabel has descended from JComponent, and is used to create text labels.
5. **JButton :** JButton class provides the functioning of push button. JButton allows an icon, string or both associated with a button.
6. **JTextField :** JTextFields allow editing of a single line of text.

Creating a JFrame

There are Three ways to create a JFrame Window.

1. By instantiating JFrame class.
2. By extending JFrame class.
3. By Using JFrame Form Option of Netbeants IDE

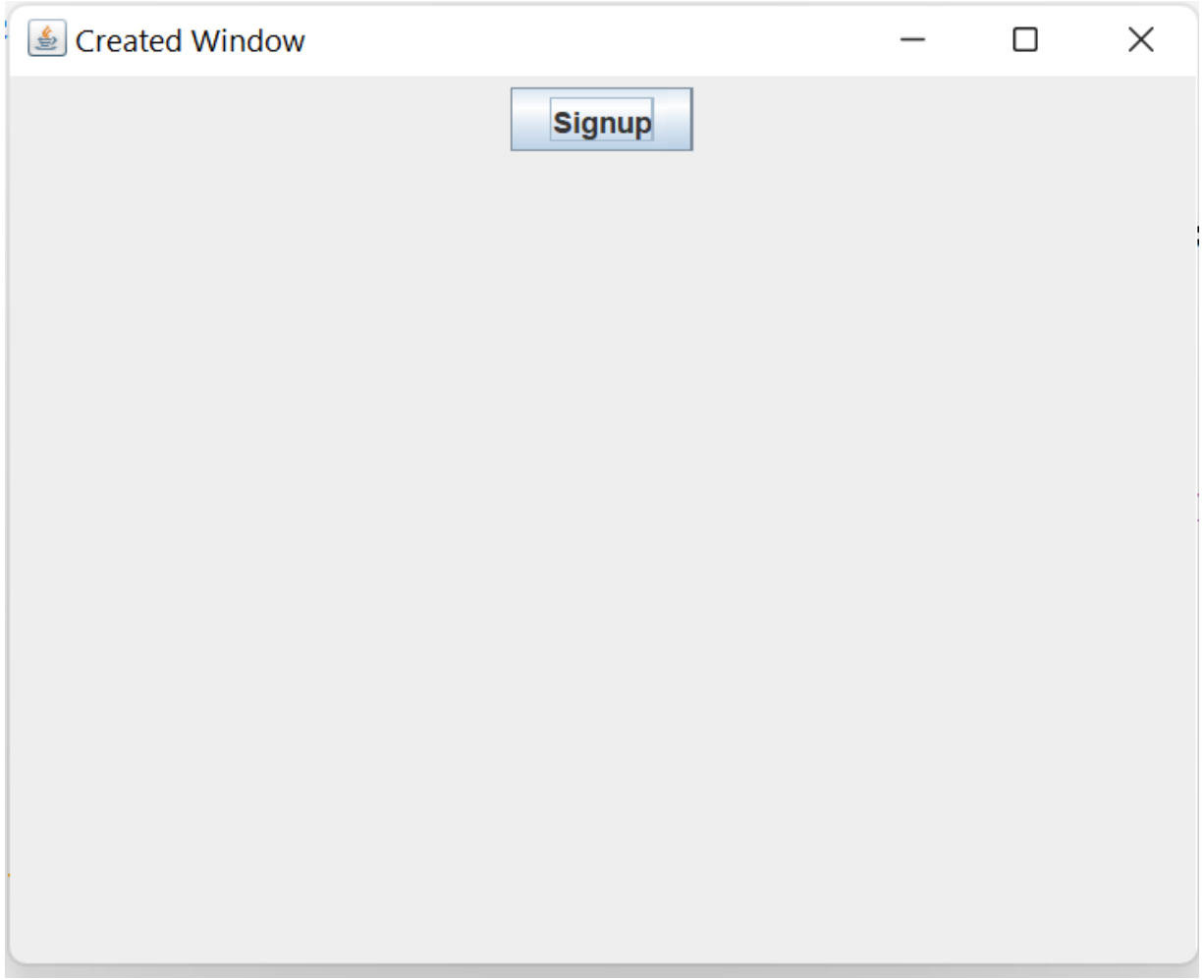
Creating JFrame window by Instantiating JFrame class

```
import javax.swing.*; import java.awt.*;
public class Frame1 {
	JFrame jf;
	public Frame1() {
	jf = new JFrame("Created Window");
	JButton btn = new JButton("Signup");
	jf.add(btn);
	jf.setLayout(new FlowLayout());
	jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	jf.setSize(500, 400); //setting size
	jf.setVisible(true);
	}
	public static void main(String[] args) {
	new Frame1();
	}
}
```

Creating JFrame window by Instantiating JFrame class+Code

```
1   package GUI;
2   import javax.swing.*;
3   import java.awt.*;
4
5   public class Frame1 {
6   JFrame jf;
7           public Frame1() {
8            jf = new JFrame("Created Window");
9           JButton btn = new JButton("Signup");
10          jf.add(btn);
11          jf.setLayout(new FlowLayout());
12          jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
13          jf.setSize(500, 400); //setting size
14          jf.setVisible(true);
15          }
16          public static void main(String[] args) {
            new Frame1();
18          }
19  }
```

Creating JFrame window by Instantiating JFrame class+Output



Creating JFrame window by extending JFrame class-Frame2

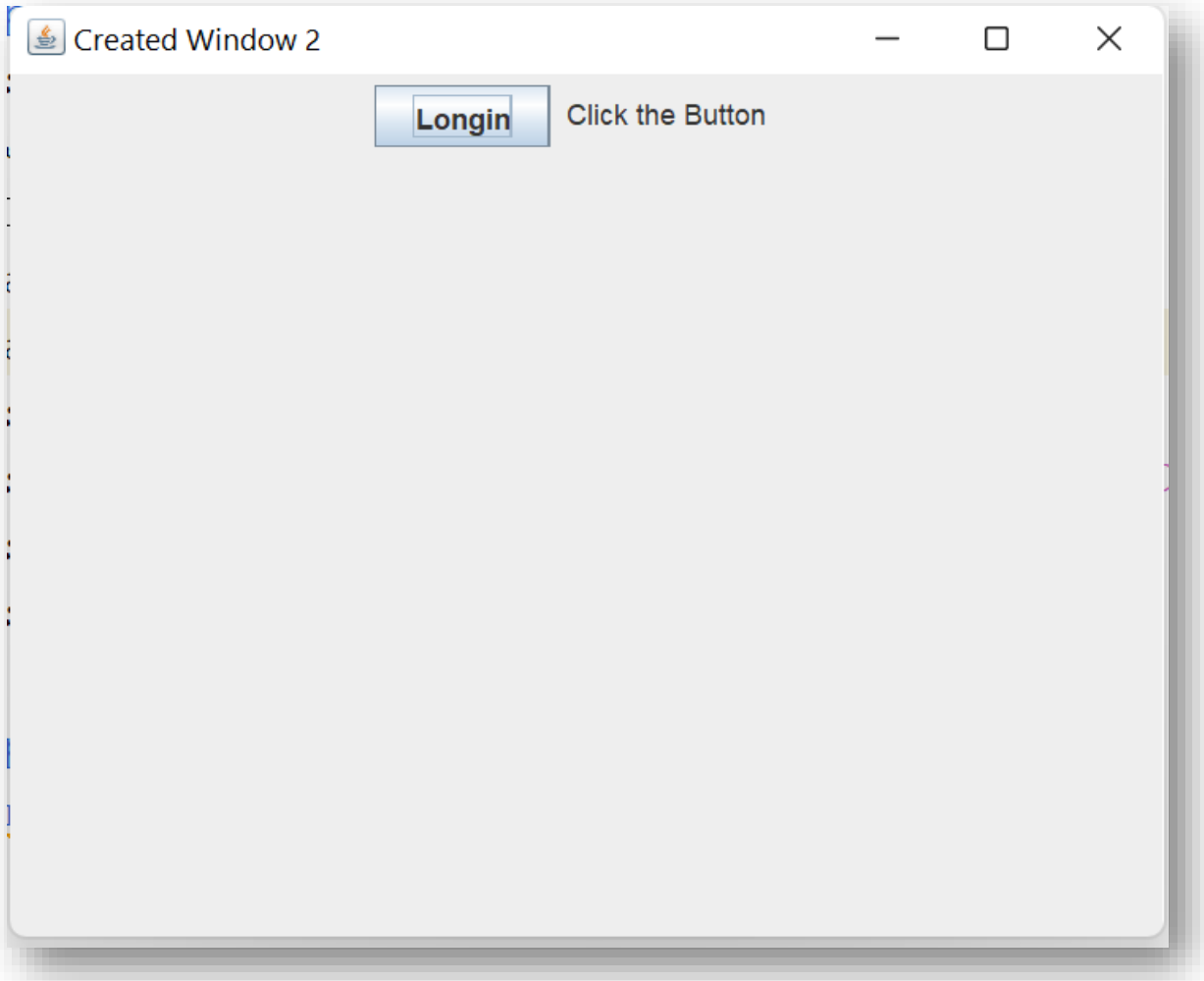
```
import javax.swing.*; import java.awt.*;
public class Frame2 extends JFrame {

	public Frame2() {
		setTitle("Created Window 2");
		JButton btn = new JButton("Longin");
		Label lb = new Label(Click the Button”);
		add(btn); add(lb);
		setLayout(new FlowLayout());
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		setSize(500, 400); //setting size
		setVisible(true);
	}
	public static void main(String[] args) {
		new Frame2();
	}
}
```

Creating JFrame window by extending JFrame class- Frame2 Code

```
1  package GUI;
2  import javax.swing.*; import java.awt.*;
3  public class Frame2 extends JFrame {
4
5      public Frame2() {
6      setTitle("Created Window 2");
7      JButton btn = new JButton("Longin");
8      Label lb = new Label("Click the Button");
9      add(btn);
10     add(lb);
11     setLayout(new FlowLayout());
12     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
13     setSize(500, 400); //setting size
14     setVisible(true);
15     }
16     public static void main(String[] args) {
       new Frame2();
18     }
19 }
```

Creating JFrame window by extending JFrame class- Frame2 output



Remember

While dealing with JFrame, note the following; -

1. Import the javax.swing and java.awt package to use the classes and methods of Swing.
2. While creating a frame (either by instantiating or extending JFrame class), following two attributes are a must for visibility of the frame:

```
setSize(int width, int height);
setVisible(true);
```

Remember

While dealing with JFrame, note the following; -

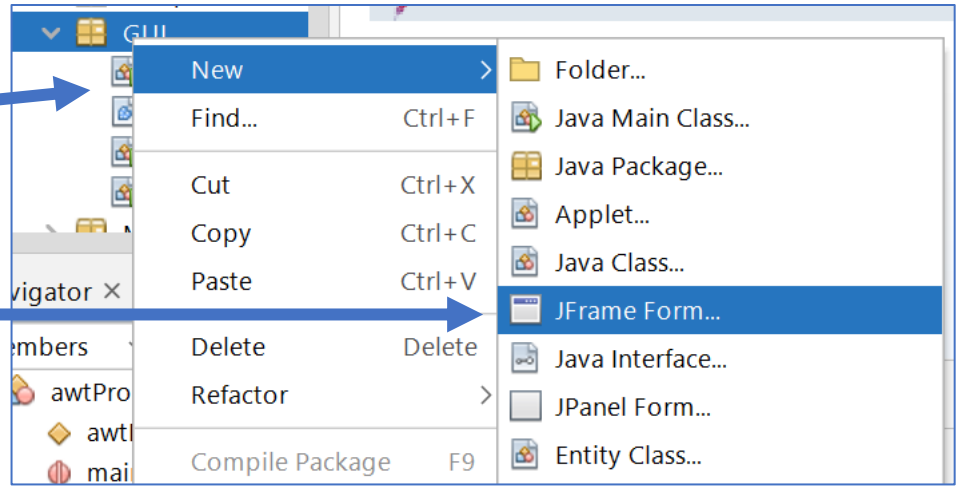
3. When you create objects of other components like Buttons, TextFields, etc. you need to add it to the frame by using the **add(Component's Object)** method
4. You can add the following method also for resizing the frame -

   setResizable(true);

Creating Frame/window Using JFrame Form Option of NetBeans IDE

The easiest way to create a Frame/window is by Using the JFrame Form Option of Netbeants IDE. Follow the following steps to create your Frame in Netbeans IDE.

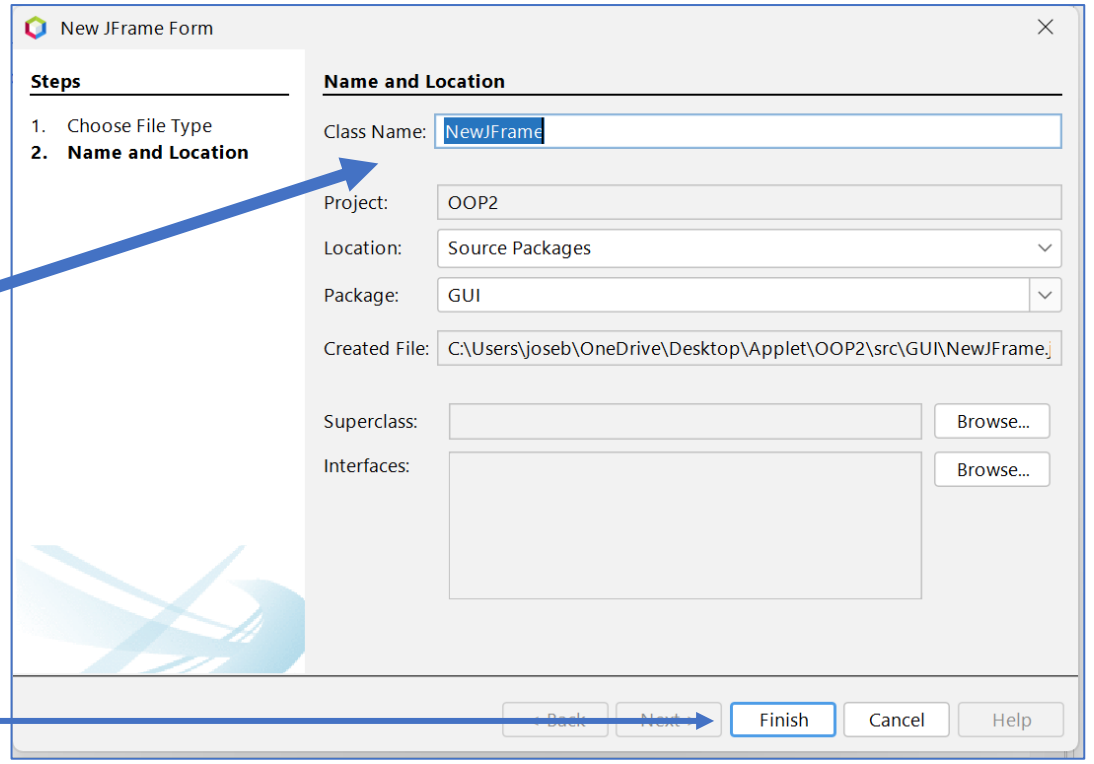
1. **Right** click your Package
2. Point at **New**
3. Click JFrame Form



Creating JFrame Using JFrame Forms Option of NetBeans IDE

The easiest way to create a Frame/window is by Using the JFrame Form Option of Netbeants IDE. Follow the following steps to create your Frame in Netbeans IDE.

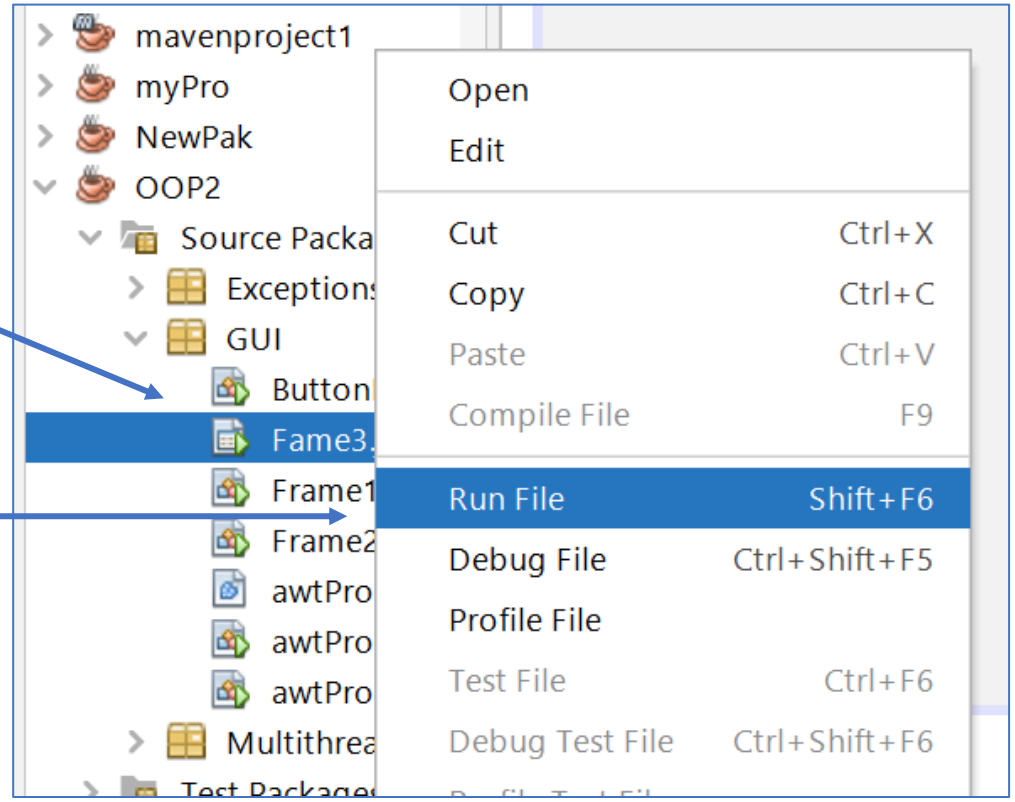
4. Enter Frame name in the Class Name TextBox **e.g.** Fame3
5. Click Finish



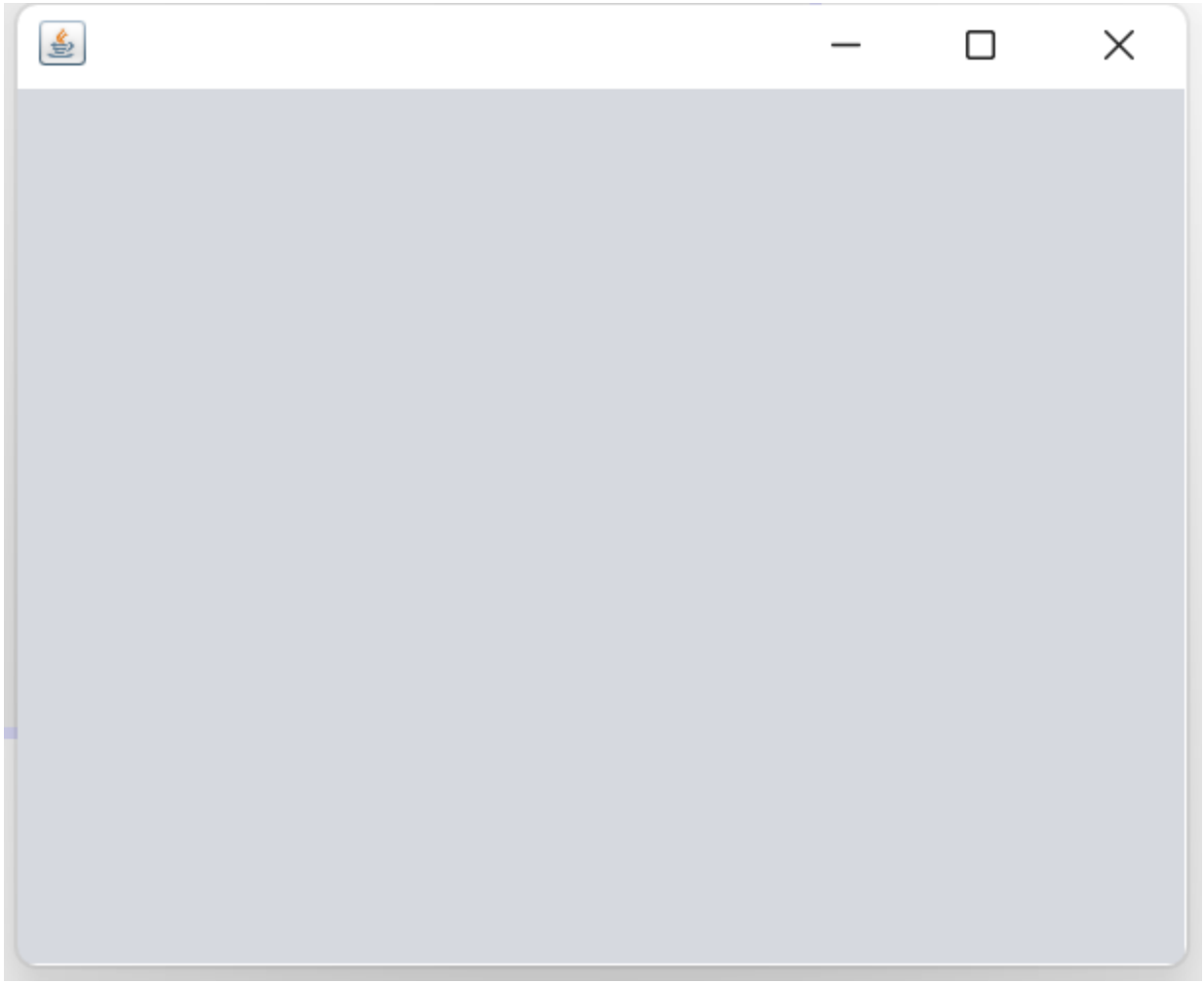
Running the Fame3 program

To run your program, follow the steps below; -

1. Right click on the From you created
2. Click Run File



Running the Fame3 Program+Output



Java Swing Components and Containers

A component is an independent visual control. and Java Swing Framework contains a large set of these components which provide rich functionalities and allow high level of customization. They all are derived from JComponent class. All these components are lightweight components. This class provides some common functionality like pluggable look and feel, support for accessibility, drag and drop, layout, etc.

Java Swing Components and Containers+

A container holds a group of components. It provides a space where a component can be managed and displayed. Containers are of two types:

1 Top level Containers

- inherits Component and Container of AWT.
- cannot be contained within other containers.
- Are heavyweight.
- Example: JFrame, JDialog, JApplet

2 Lightweight Containers

- inherits JComponent class.
- is a general purpose container.
- can be used to organize related components together.
- Example: JPanel

Swing JButton

JButton class provides functionality of a button. It is used to create button component.

Example of Jbutton

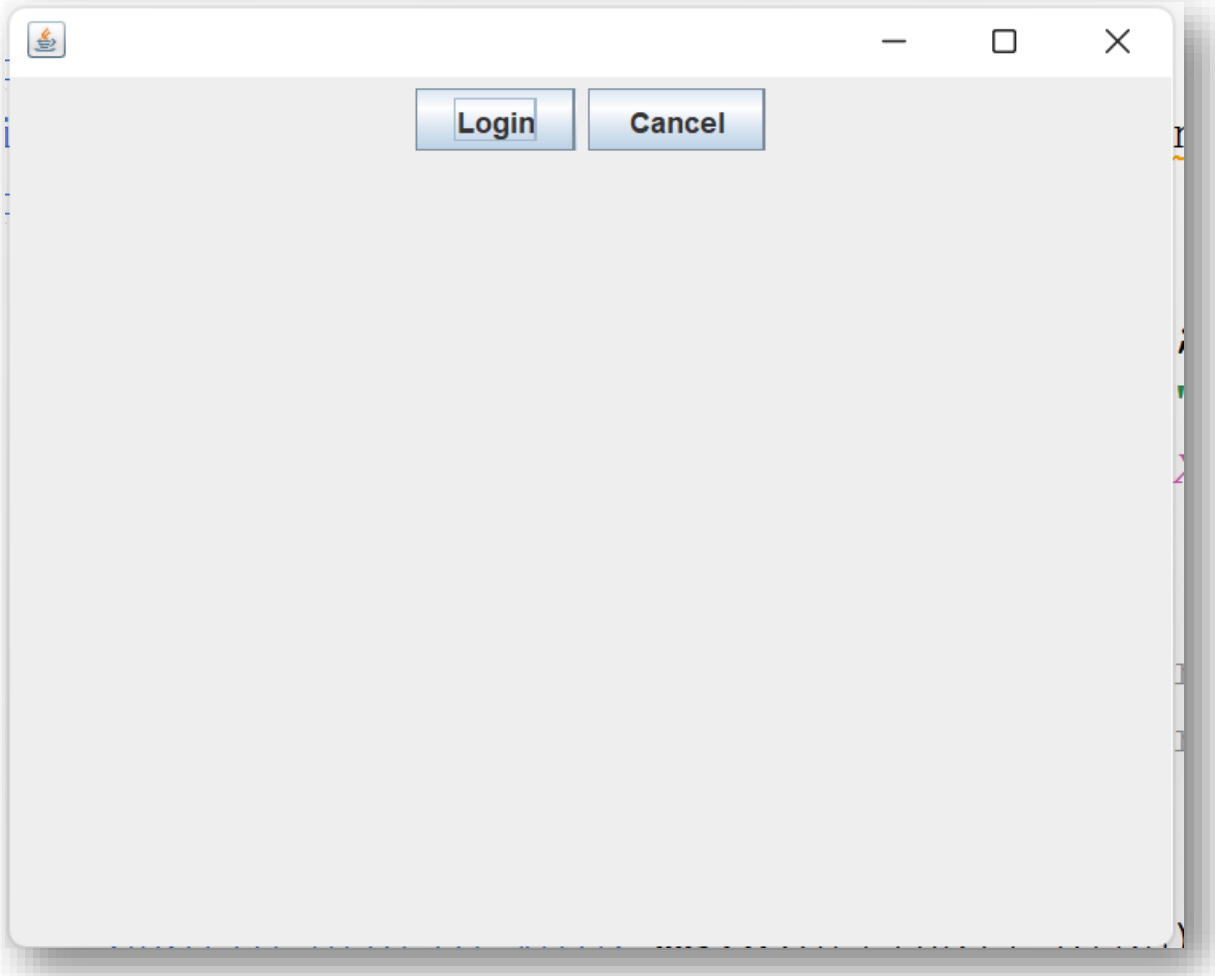
In this example, we are creating two buttons using Jbutton class and adding them into Jframe container.

```
import javax.swing.*; import java.awt.event.*; import java.awt.*;
public class bTest extends JFrame{
        bTest(){
        JButton ob = new JButton("Login");
        JButton ob1 = new JButton("Cancel"); //Creating a No Button.
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)
        setLayout(new FlowLayout());
        setSize(500, 400);
        add(ob); //adding Yes button to frame.
        add(ob1); //adding No button to frame.
        setVisible(true);
        }
        public static void main(String[] args) {
        new bTest();
} }
```

Example of Jbutton+Code

```
2   package GUI;
3   import javax.swing.*; import java.awt.event.*; import java.awt.*;
4   public class bTest extends JFrame{
5       bTest(){
6           JButton ob = new JButton("Login");
7           JButton ob1 = new JButton("Cancel"); //Creating a No Button.
8           setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9           setLayout(new FlowLayout());
10          setSize(500, 400);
11          add(ob); //adding Yes button to frame.
12          add(ob1); //adding No button to frame.
13          setVisible(true);
14      }
15      public static void main(String[] args) {
16          new bTest();
17      }
18  }
```

Example of Jbutton+Output



JTextField

JTextField is used for taking input of single line of text. It is most widely used text component. It has three constructors.

```
JTextField(int cols)
JTextField(String str, int cols)
JTextField(String str)
```

cols represent the number of columns in text field.,

Creating JTextField

In this example, we are creating TextField using JTextField class and added it into JFrame container.

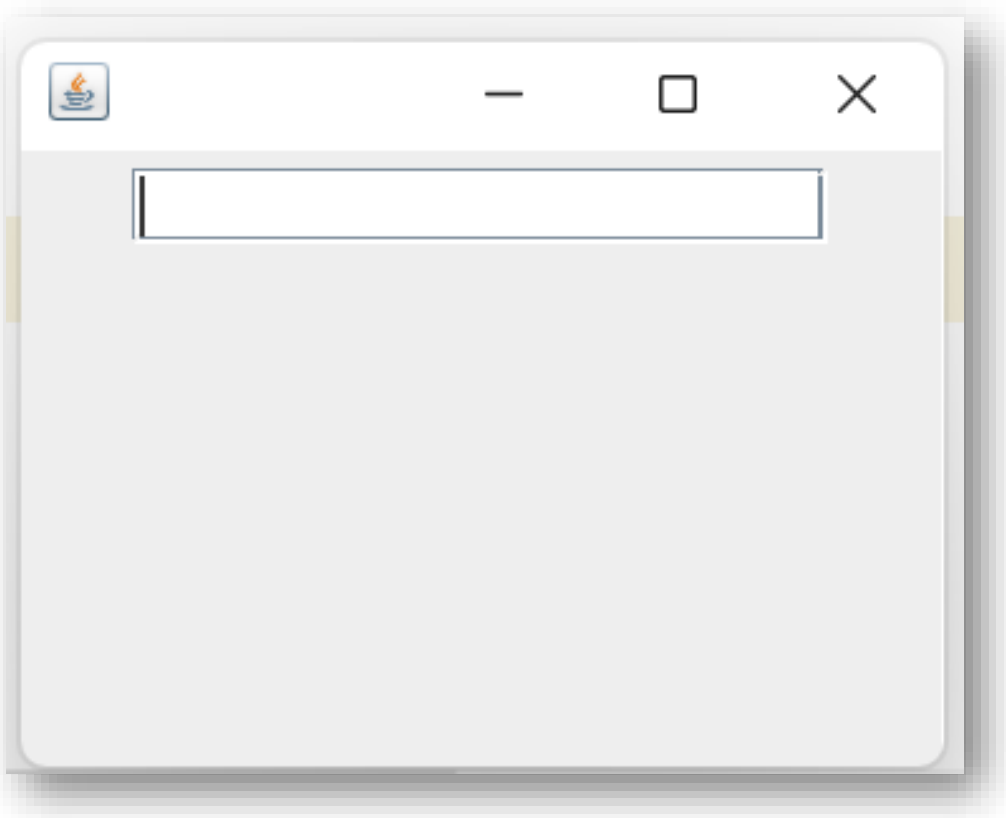
```
import javax.swing.*; import java.awt.event.*; import java.awt.*;
public class JTF extends JFrame{
      public JTF() {
      JTextField ob = new JTextField(20); //creating JTextField.
      add(ob); //adding JTextField to frame.
      setLayout(new FlowLayout());
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setSize(259, 200);
      setVisible(true);
      }
      public static void main(String[] args) {
      new JTF();
      }
}
```

Creating JTextField+Code and Output

```
1   package GUI;
2   import javax.swing.*; import java.awt.event.*;
3   import java.awt.*;
4   public class JTF extends JFrame{
5           public JTF() {
6           JTextField ob = new JTextField(20); //creating JTextField.
7           add(ob); //adding JTextField to frame.
8           setLayout(new FlowLayout());
9           setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10          setSize(259, 200);
11          setVisible(true);
12          }
13          public static void main(String[] args) {
14          new JTF();
15          }
16  }
```

Code

Output



JCheckBox Creation

The JCheckBox class is used to create chekbox in swing framework. In this example, we are creating three checkboxes to get user response.

```
JCheckBox(String str)
```

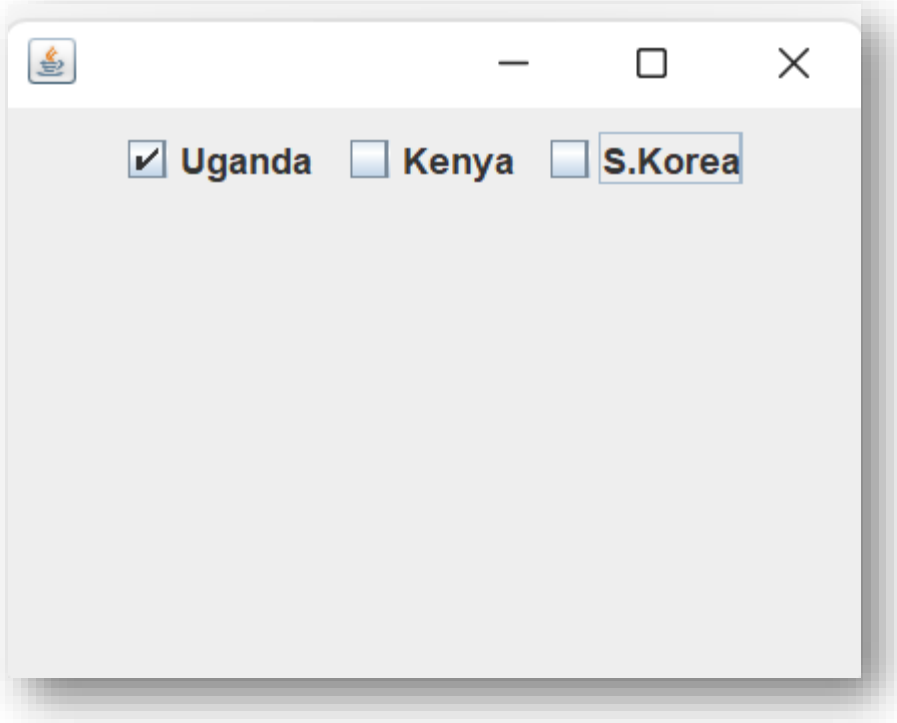
```
import javax.swing.*; import java.awt.event.*; import java.awt.*;
public class JCB extends JFrame {
        public JCB() {
        JCheckBox ob = new JCheckBox("Uganda"); //creating JCheckBox.
        add(ob); //adding JCheckBox to frame.
        ob = new JCheckBox("Kenya"); //creating JCheckBox.
        add(ob); //adding JCheckBox to frame.
        ob = new JCheckBox("S.Korea"); //creating JCheckBox.
        add(ob); //adding JCheckBox to frame.
        setLayout(new FlowLayout());
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 250); setVisible(true);
        }
        public static void main(String[] args) {
        new JCB();
        }
}
```

JCheckBox Creation Code+Output

```
1  package GUI;
2  import javax.swing.*; import java.awt.event.*; import java.awt.*;
3  public class JCB extends JFrame {
4          public JCB() {
5          JCheckBox ob = new JCheckBox("Uganda"); //creating JCheckBox.
6          add(ob); //adding JCheckBox to frame.
7          ob = new JCheckBox("Kenya"); //creating JCheckBox.
8          add(ob); //adding JCheckBox to frame.
9          ob = new JCheckBox("S.Korea"); //creating JCheckBox.
10         add(ob); //adding JCheckBox to frame.
11         setLayout(new FlowLayout());
12         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
13         setSize(300, 250);
14         setVisible(true);
15         }
16         public static void main(String[] args) {
           new JCB();
18         }
19 }
```

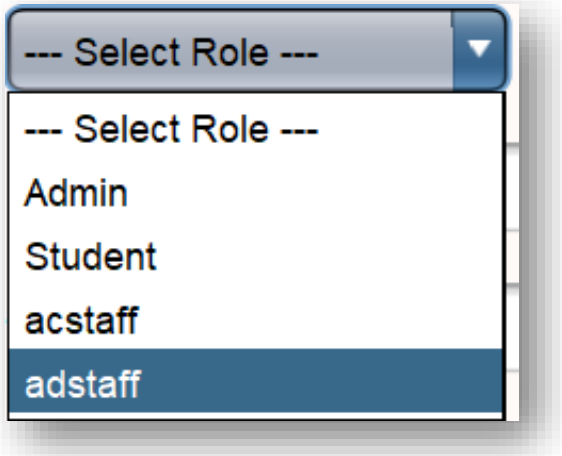
Code

Code



JComboBox

Combo box is a combination of text fields and drop-down list.



JComboBox component is used to create a combo box in Swing. Following is the constructor for JComboBox,

```
JComboBox(String a[])
```

Creating JComboBox

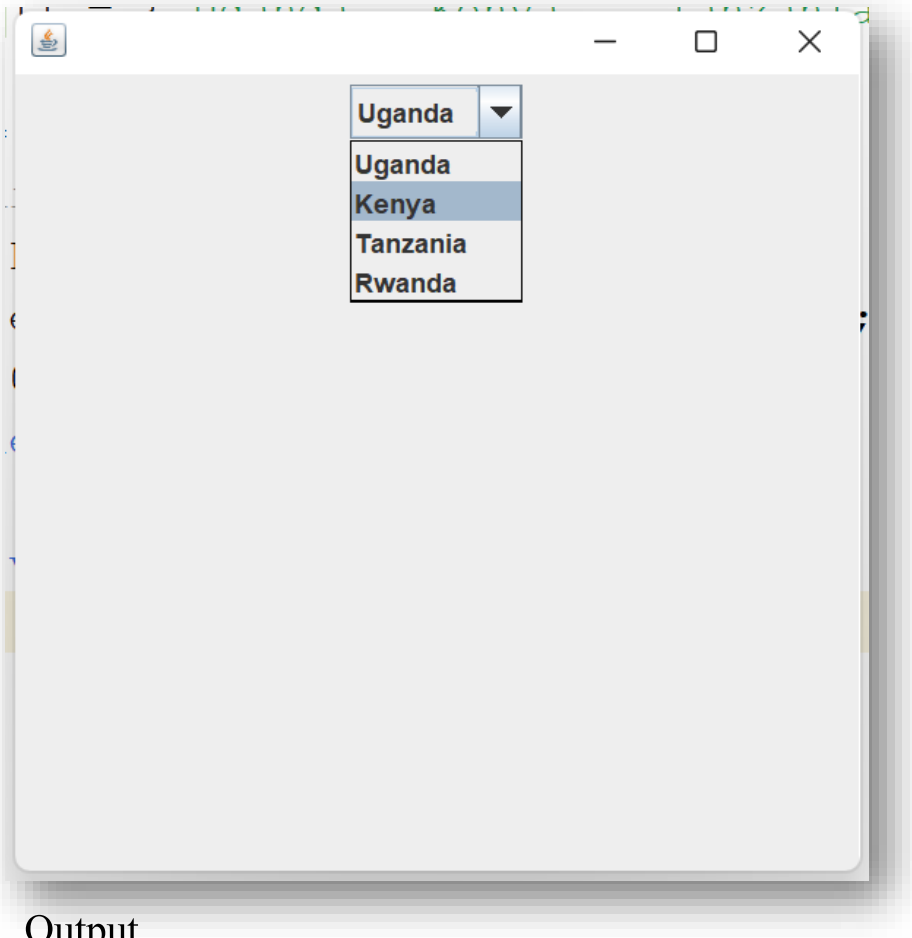
Lets create an example to add combobox to the JFrame . Combobox is used to create a drop-down menu. See the example below.

```
import javax.swing.*; import java.awt.event.*; import java.awt.*;
public class CB extends JFrame {
      String country[] = {"Uganda","Kenya", "Tanzania", "Rwanda"};
      public CB() {
      JComboBox ob = new JComboBox(country);
      add(ob); //adding JComboBox to frame.
      setLayout(new FlowLayout());
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setSize(400, 400);
      setVisible(true);
      }
      public static void main(String[] args) {
      new CB();
      }
}
```

Creating ComboBox Code and Output

```
1   package GUI;
2   import javax.swing.*; import java.awt.event.*; import java.awt.*;
3   public class CB extends JFrame {
4           String country[] = {"Uganda","Kenya", "Tanzania", "Rwanda"};
5           public CB() {
6           JComboBox ob = new JComboBox(country);
7           add(ob); //adding JComboBox to frame.
8           setLayout(new FlowLayout());
9           setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10          setSize(400, 400);
11          setVisible(true);
12          }
13          public static void main(String[] args) {
14          new CB();
15          }
16  }
```

Code



Output

JRadioButton

Radio button is a group of related buttons in which only one can be selected. JRadioButton class is used to create a Radio Button in Frames. Following is the constructor for JRadioButton,

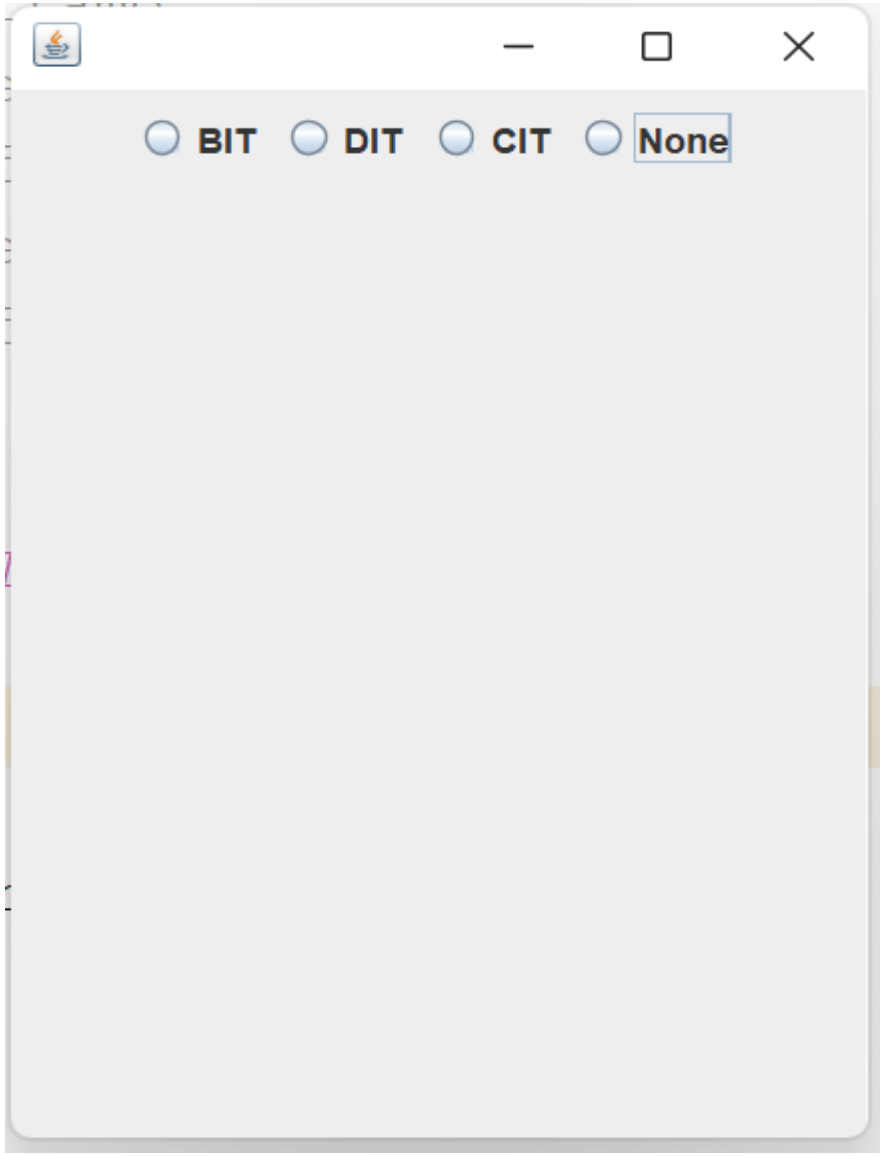
```
JRadioButton(String n)
```

Creating JRadioButton

To create radio button in swing, we use JRadioButton class which is used to get single user response at a time.

```
import javax.swing.*; import java.awt.event.*; import java.awt.*;
public class JRB extends JFrame {
      public JRB() {
      JRadioButton rb = new JRadioButton("BIT");
      add(rb); //adding JRadioButton to frame.
      rb = new JRadioButton("DIT"); //creating JRadioButton.
      add(rb); //adding JRadioButton to frame.
      rb = new JRadioButton("CIT"); //creating JRadioButton.
      add(rb); //adding JRadioButton to frame.
      rb = new JRadioButton("None");
      add(rb); setLayout(new FlowLayout());
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setSize(300, 400); setVisible(true);
      }
      public static void main(String[] args) {
      new JRB();
      } }
```

Output



JLabel

In Java, Swing toolkit contains a JLabel Class. It is under package javax.swing.JLabel class. It is used for placing text in a box. Only Single line text is allowed and the text can not be changed directly.

Declaration

public class JLabel extends JComponent implements SwingConstants, Accessible

The JLabel Contains 4 constructors. They are as follows:

1. JLabel()

2. JLabel(String s)

3. JLabel(Icon i)

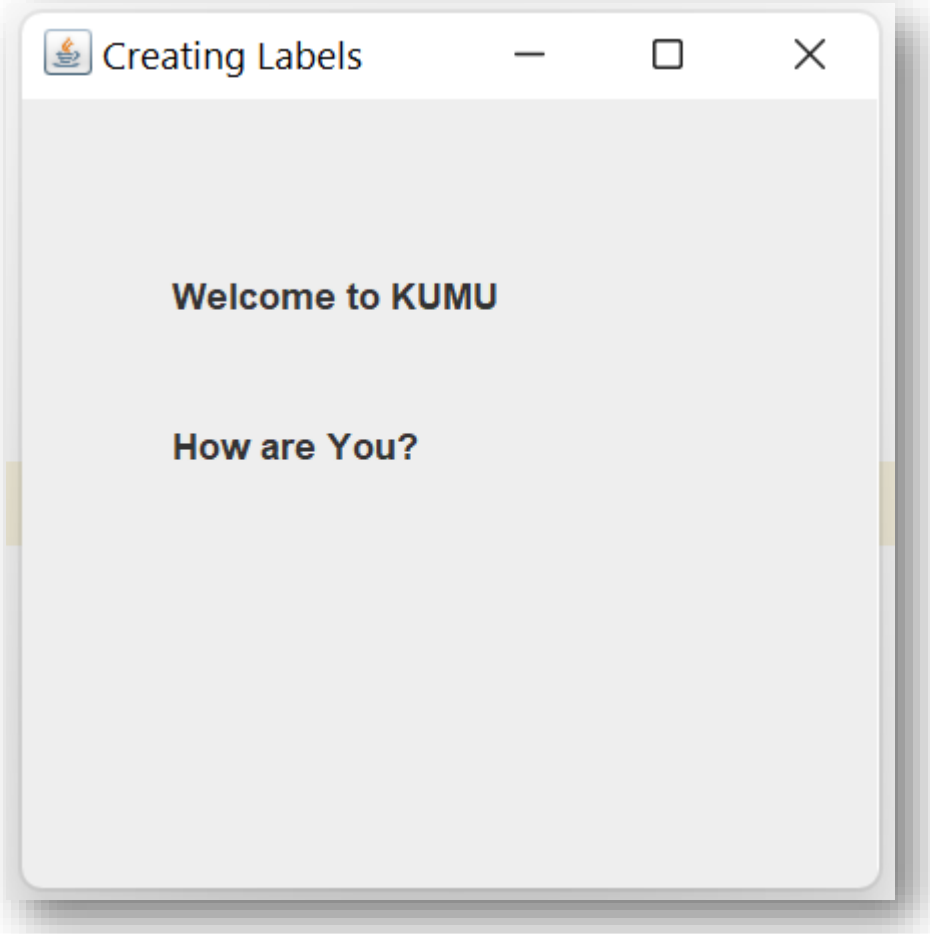
4. JLabel(String s, Icon i, int horizontalAlignment)

Creating a Label using JLabel

```
import javax.swing.*;
class JLPro {
      public static void main(String args[]) {
      JFrame jf= new JFrame("Creating Labels");
      JLabel l1, l2;
      l1=new JLabel("Welcome to KUMU");
      l1.setBounds(50,50, 200,30);
      l2=new JLabel("How are You?");
      l2.setBounds(50,100, 200,30);
      jf.add(l1);
      jf.add(l2);
      jf.setSize(300,300);
      jf.setLayout(null);
      jf.setVisible(true);
      }
}
```

Code and Output

```
1   package GUI;
2   import javax.swing.*;
3   class JLPro {
4           public static void main(String args[]) {
5           JFrame jf= new JFrame("Creating Labels");
6           JLabel l1, l2;
7           l1=new JLabel("Welcome to KUMU");
8           l1.setBounds(50,50, 200,30);
9           l2=new JLabel("How are You?");
10          l2.setBounds(50,100, 200,30);
11          jf.add(l1);
12          jf.add(l2);
13          jf.setSize(300,300);
14          jf.setLayout(null);
15          jf.setVisible(true);
16          }
17  }
```



JTextArea

In Java, Swing toolkit contains a JTextArea Class. It is under package javax.swing.JTextArea class. It is used for displaying multiple-line text.

Declaration

public class JTextArea extends JTextComponent

The JTextArea Contains 4 constructors. They are as following:

1. JTextArea()

2. JTextArea(String s)

3. JTextArea(int row, int column)

4. JTextArea(String s, int row, int column)

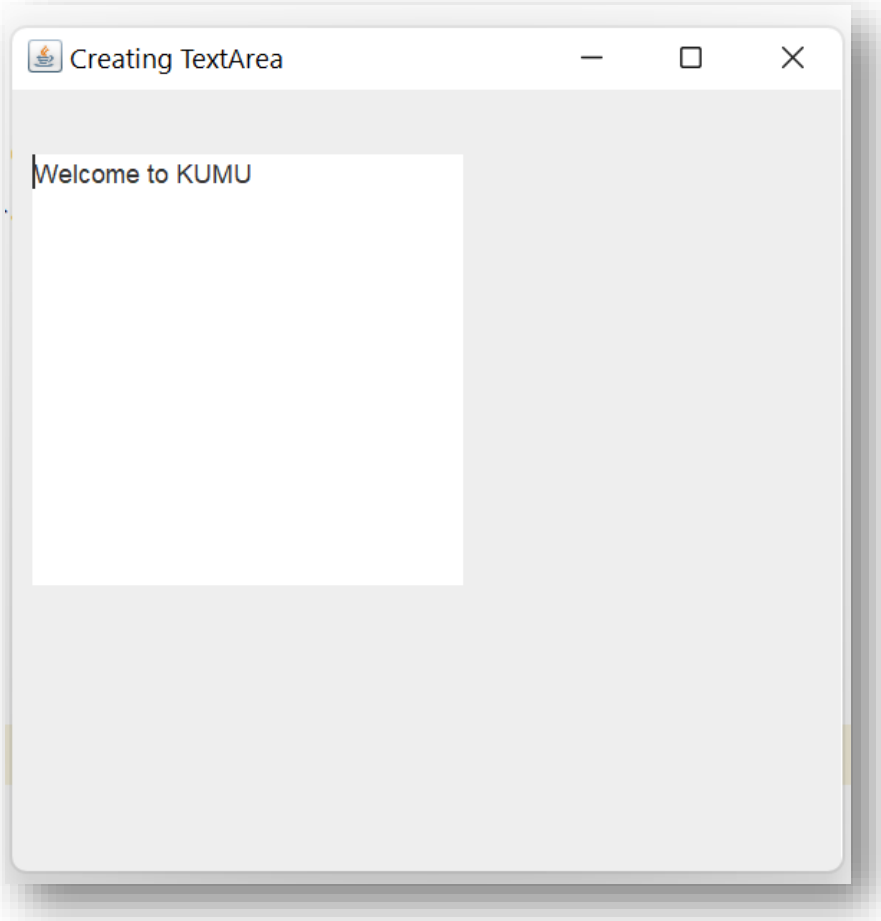
Creating a Text Area using JTextArea

Lets take an example to create text area in swing. We are using JtextArea class to create text area and adding to Jframe container.

```
import javax.swing.*;
public class TAPro{
      TAPro(){
      JFrame taf= new JFrame("Creating TextArea");
      JTextArea ob=new JTextArea("Welcome to KUMU");
      ob.setBounds(10,30, 200,200);
      taf.add(ob);
      taf.setSize(400,400);
      taf.setLayout(null);
      taf.setVisible(true);
      }
      public static void main(String args[]) {
      new TAPro();
      }
}
```

Code and output

```
1  package GUI;
2  import javax.swing.*;
3  public class TAPro{
4          TAPro(){
5          JFrame taf= new JFrame("Creating TextArea");
6          JTextArea ob=new JTextArea("Welcome to KUMU");
7          ob.setBounds(10,30, 200,200);
8          taf.add(ob);
9          taf.setSize(400,400);
10         taf.setLayout(null);
11         taf.setVisible(true);
12         }
13         public static void main(String args[]) {
           new TAPro();
15         }
16 }
```



JPasswordField

In Java, Swing toolkit contains a JPasswordField Class. It is under package javax.swing.JPasswordField class. It is specifically used for password and it can be edited.

Declaration

public class JPasswordField extends JTextField

The JPasswordFieldContains 4 constructors. They are as following:

1. JPasswordField()
2. JPasswordField(int columns)
3. JPasswordField(String text)
4. JPasswordField(String text, int columns)

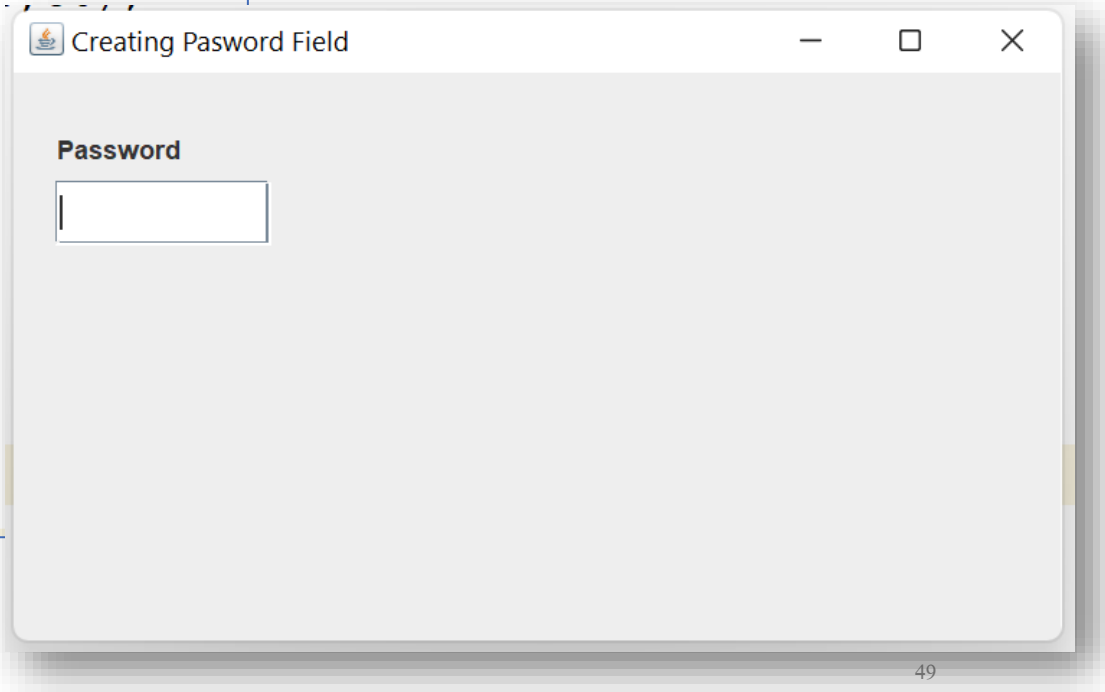
Creating Password Filed using JPasswordField

To generate a password component, swing provides Jpasswordfield that takes user input in encrypted format.

```
import javax.swing.*;
public class JPF extends JFrame{
        public static void main(String[] args) {
        JFrame jf = new JFrame("Creating Pasword Field");
        JPasswordField pob= new JPasswordField();
        JLabel pl=new JLabel("Password ");
        pl.setBounds(20,20,100,30);
        pob.setBounds(20,50, 100,30);
        add(pob);
        add(pl);
        setSize(300,300);
        setLayout(null);
        setVisible(true);
        }
}
```

Creating Password Field using JPasswordField code and Ouptput

```
1   package GUI;
2   import javax.swing.*;
3   public class JPF extends JFrame{
4           public static void main(String[] args) {
5           JFrame jf = new JFrame("Creating Pasword Field");
6           JPasswordField pob= new JPasswordField();
7           JLabel pl=new JLabel("Password ");
8           pob.setBounds(20,50, 100,30);
9           pl.setBounds(20,20,100,30);
10          jf.add(pl);
11          jf.add(pob);
12          jf.setSize(500,300);
13          jf.setLayout(null);
14          jf.setVisible(true);
15          }
16  }
```



JTable Class

In Java, Swing toolkit contains a JTable Class. It is under package javax.swing.JTable class. JTable class used to draw a table to display data.

The JTableContains 2 constructors. They are as following:

1. JTable()

2. JTable(Object[][] rows, Object[] columns)

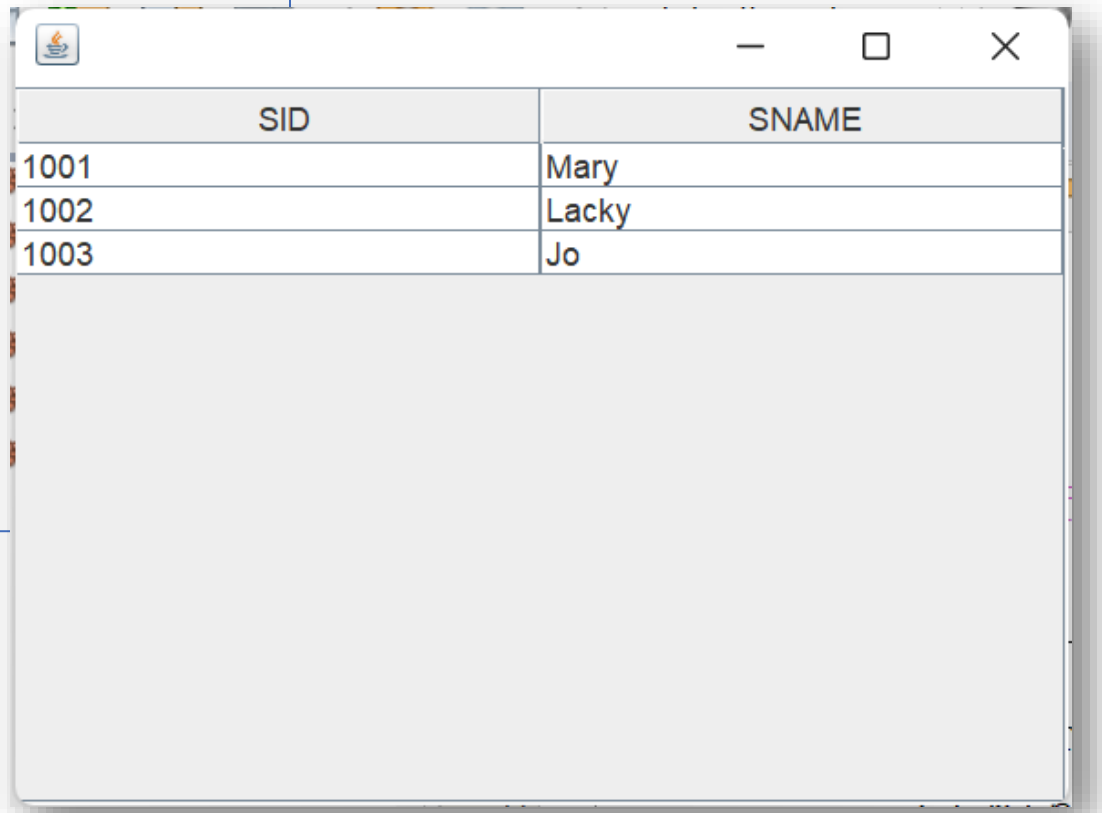
Creating a Table using JTable

We are creating an example to create a table using Jtable class and then add it to the Jframe container.

```
import javax.swing.*;
public class JTPro {
        JFrame jf;
        JTPro(){
        jf=new JFrame();
        String trow [][]={ {"1001","Mary"}, {"1002","Lacky"}, {"1003","Jo"}};
        String tcol[]={"SID","SNAME"};
        JTable jt=new JTable(trow,tcol);
        jf.setBounds(30,40,200,300);
        JScrollPane table_sp=new JScrollPane(jt);
        jf.add(table_sp);
        jf.setSize(400,300);
        jf.setVisible(true);
        }
        public static void main(String[] args) {
        new JTPro();
} }
```

Creating A table using JTable class code and output

```
1   package GUI;
2   import javax.swing.*;
3   public class JTPro {
4       JFrame jf;
5       JTPro(){
6       jf=new JFrame();
7       String trow [][]={ {"1001","Mary"}, {"1002","Lacky"}, {"1003","Jo"}};
8           String tcol[]={"SID","SNAME"};
9           JTable jt=new JTable(trow,tcol);
10          jf.setBounds(30,40,200,300);
11          JScrollPane table_sp=new JScrollPane(jt);
12          jf.add(table_sp);
13          jf.setSize(400,300);
14          jf.setVisible(true);
15      }
16          public static void main(String[] args) {
            new JTPro();
18  } }
```



JList Class

In Java, Swing toolkit contains a JList Class. It is under package javax.swing.JList class. It is used to represent a list of items together. One or more items can be selected from the list.

Declaration

public class JList extends JComponent implements Scrollable, Accessible

The JListContains 3 constructors. They are as following:

1. JList()

2. JList(ary[] listData)

3. JList(ListModel<ary> dataModel)

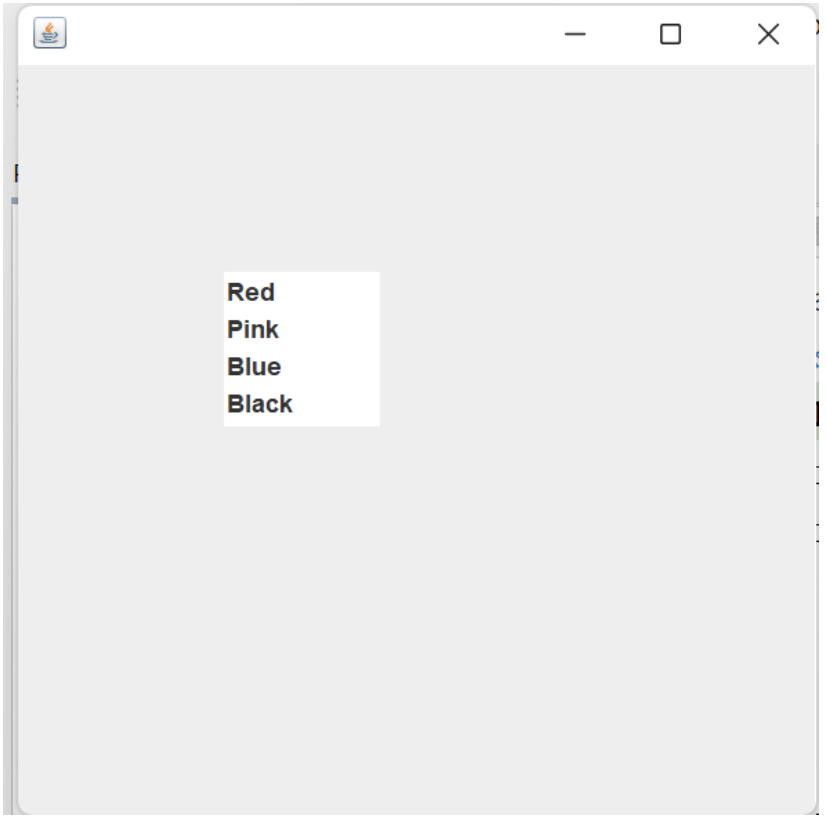
Creating a list Using JList class

In this example, we are creating a list of items using Jlist class. this list is used to show the items in a list format and get user input from the list of items. See the below example.

```
import javax.swing.*;
public class JLPro1 {
        JLPro1() {
        JFrame jf= new JFrame();
        DefaultListModel<String> dl= new DefaultListModel<>();
        dl.addElement("Red");
        dl.addElement("Pink");
        dl.addElement("Blue");
        dl.addElement("Black");
        JList<String> list1 = new JList<>(dl);
        list1.setBounds(100,100, 75,75);
        jf.add(list1);
        jf.setSize(400,400);
        jf.setLayout(null);
        jf.setVisible(true);
        }
        public static void main(String args[]) {
        new JLPro1();
} }
```

Creating a list Uing JList class+Code and output

```
3  import javax.swing.*;
4  public class JLPro1 {
5          JLPro1() {
6          JFrame jf= new JFrame();
7          DefaultListModel<String> dl= new DefaultListModel<>();
8          dl.addElement("Red");
9          dl.addElement("Pink");
10         dl.addElement("Blue");
11         dl.addElement("Black");
12         JList<String> list1 = new JList<>(dl);
13         list1.setBounds(100,100, 75,75);
14         jf.add(list1);
15         jf.setSize(400,400);
16         jf.setLayout(null);
17         jf.setVisible(true);
18         }
19         public static void main(String args[]) {
           new JLPro1();
21 } }
```

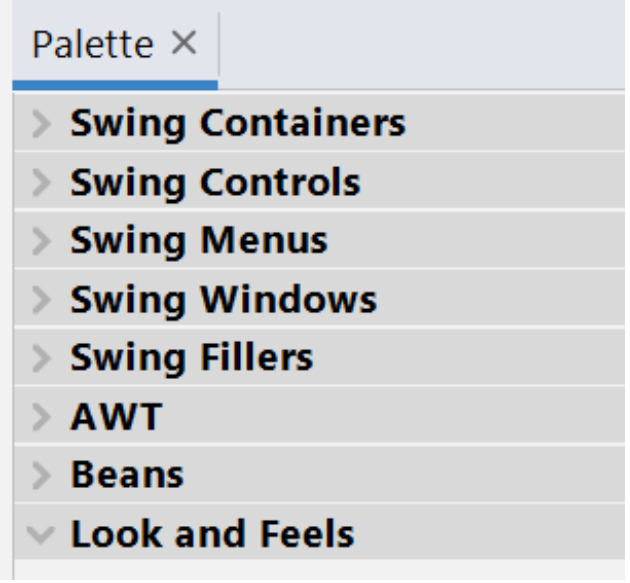


Swing Components Part-2

Accessing Swing Components directly from IDE

Although we can create Swing componenst by coding, we can also create them by drag and drop technique as long as you are using a feature rich IDE like NetBeans.

The Swing components in the IDE are found under Palette panel and they are categorized into Swing Containers, Swing Controls, Swing Menus, Swing Windows and Swing Filters as seen below.



Swing Components Part-2

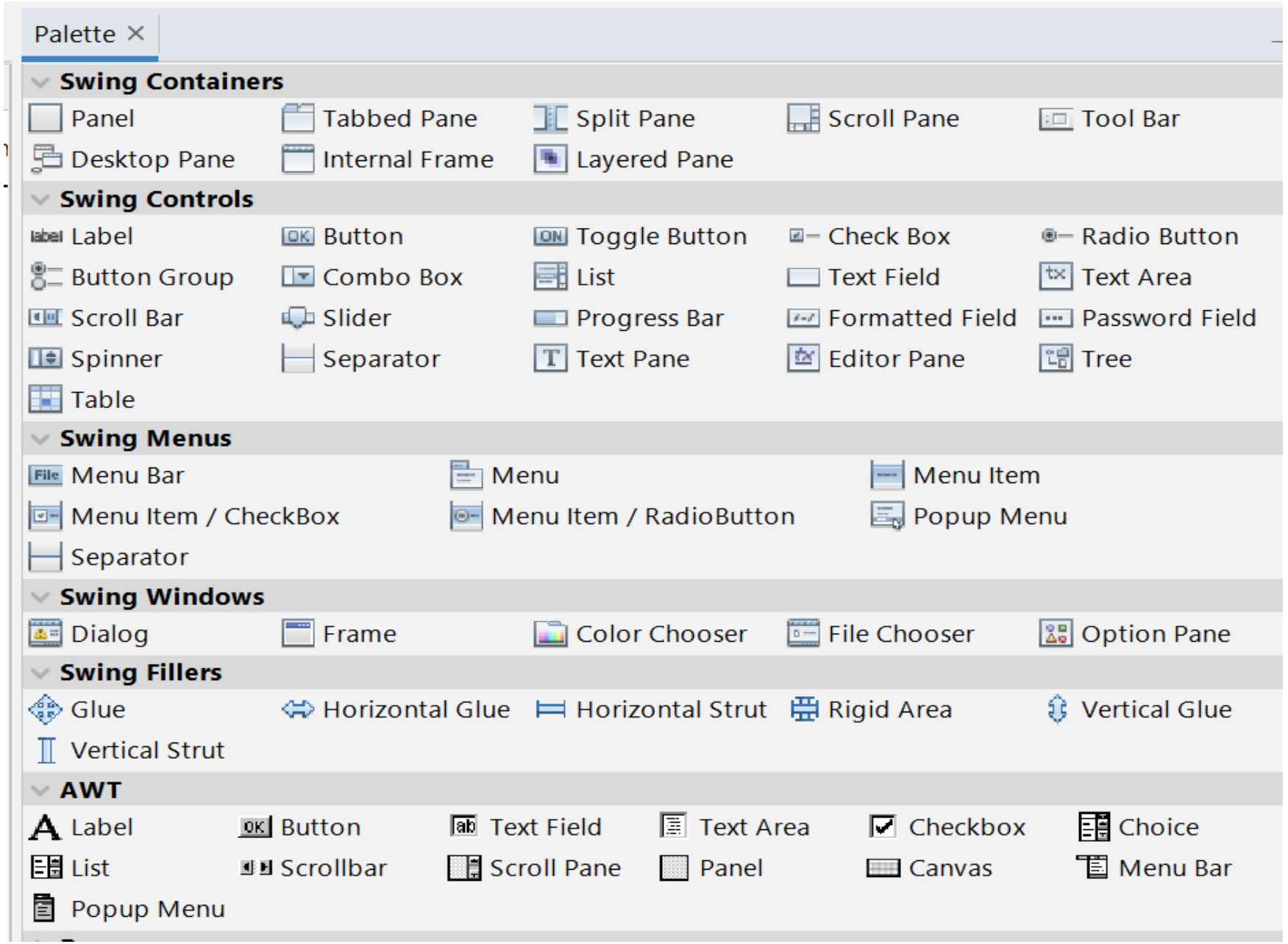
This section helps us to explore swing componests directly using the pallete panel of NetBeans IDE. We will be able to create Various components by drag and drop.

Accessing Swing Components directly from IDE+Pallete

Before you access Swing components you will need to create a JFrame Form java class using the procedure explained on slide 17 and 18.

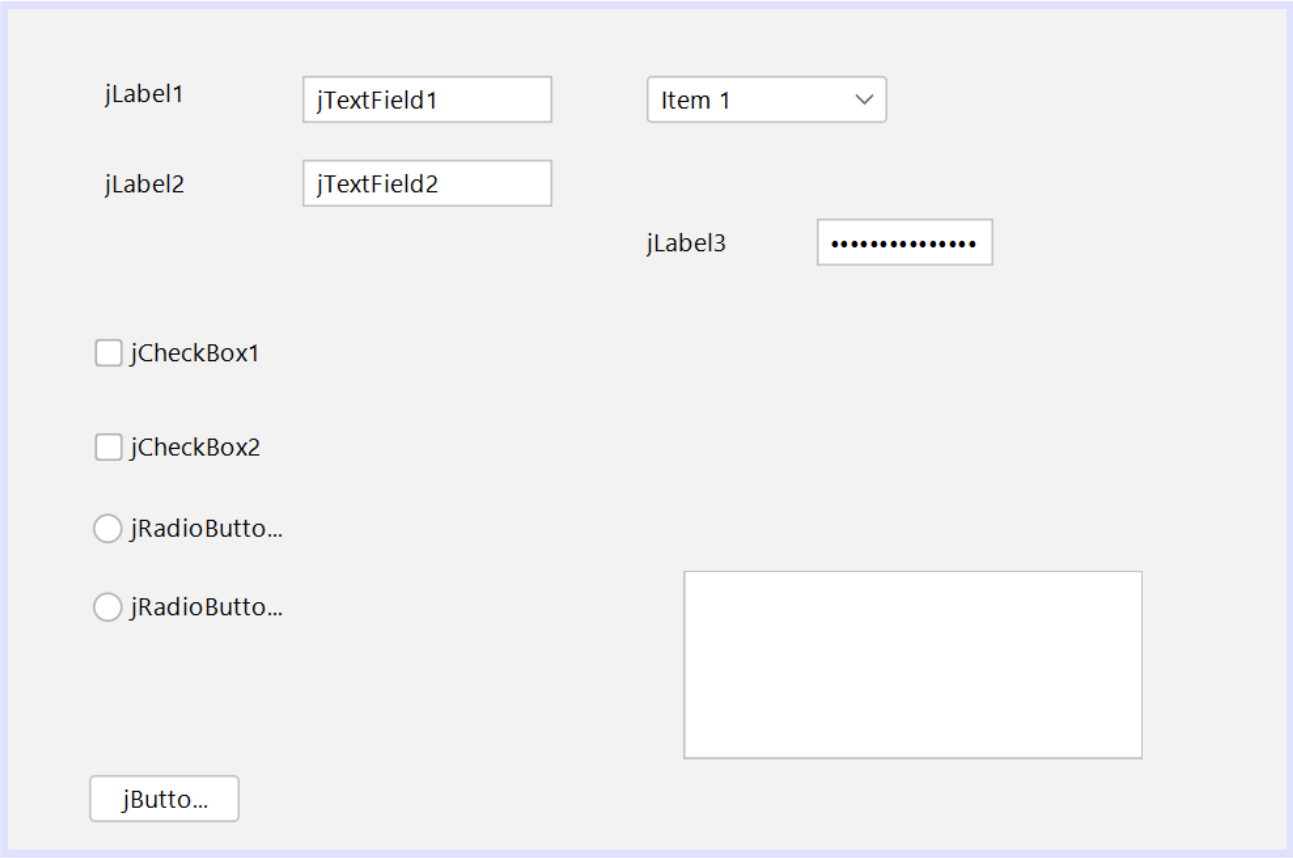
After you have created a JFrame From, say called SwingComponents, then you would be able to see the following components under pallete section of your IDE.

Pallete Components



Utilizing Swing components

Having seen the components above, we can now go a head and add some onto our JFrame window created above, by drag and drop technique



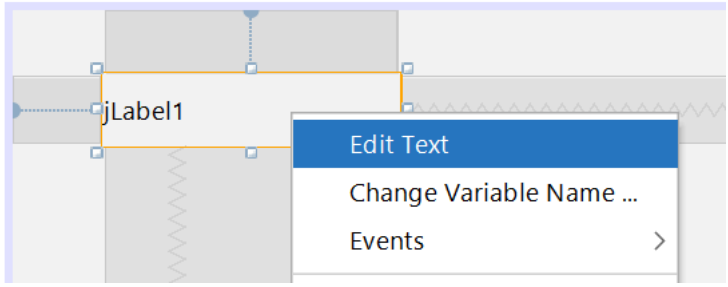
As you can see we have dragged and dropped Seven types of components; - Jlabel, JTextField, JComboBox, JPasswordField, JCheckBox, JRadioButton, JTextArea and JButton Now all we need to do is resize and rename them

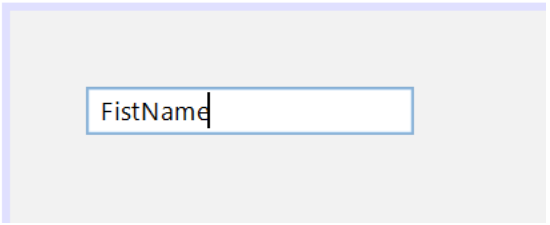
Rename the Items

While renaming items we need to know that each item has two possible names; Display name(what the user sees) and logical/variable name(used for programing or adding functionality to a given item).

To rename Display name of any Item, follow the steps below.

1. Right Click on the item
2. Click Edit Text to rename display name
3. Enter the name
4. Press Enter
5. Click outside the item

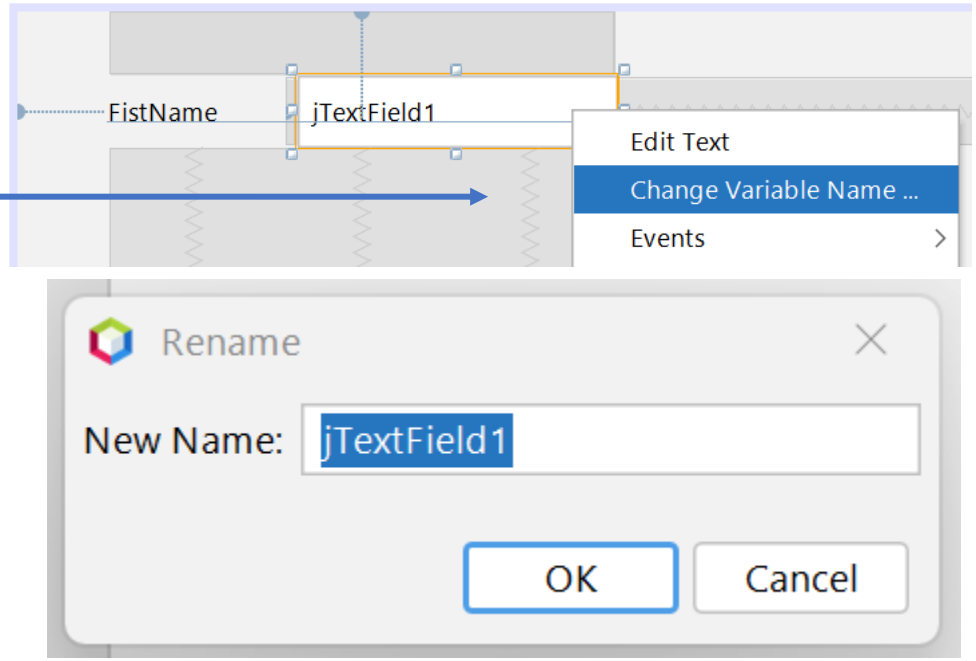




Rename the Items

Renaming Variable Name of an item. In this case, we will be renaming a JTextField1.

1. Right click on the Item
2. Click change Variable Name
3. Enter Name of your choice
4. Click OK



Remove the default display name of the JTextField

1. Right click on the item
2. Click Edit Text
3. Delete the default name
4. Press Enter

Viewing item Variable declarations

1. Double click on any item on the Jframe or click Source tab
2. Scroll to the bottom of the source code page
3. Your should be seeing a list of items as below

```
// Variables declaration - do not modify
private javax.swing.JTextField fname;
private javax.swing.JLabel jLabel1;
// End of variables declaration
```

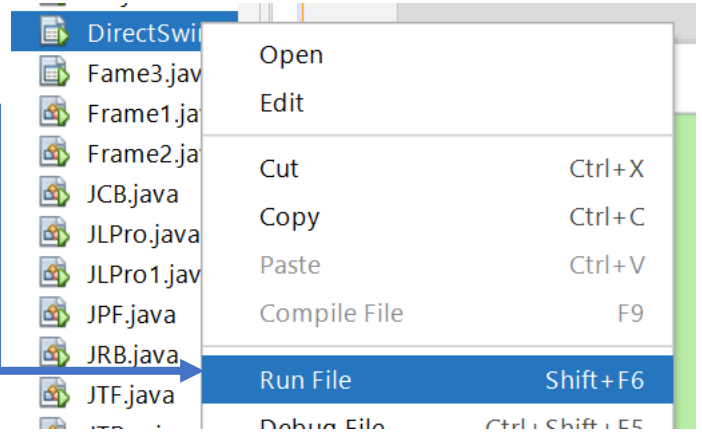
Variable name for the **JTextField** =fname

And Variable name for **JLabel1** = jLabel1

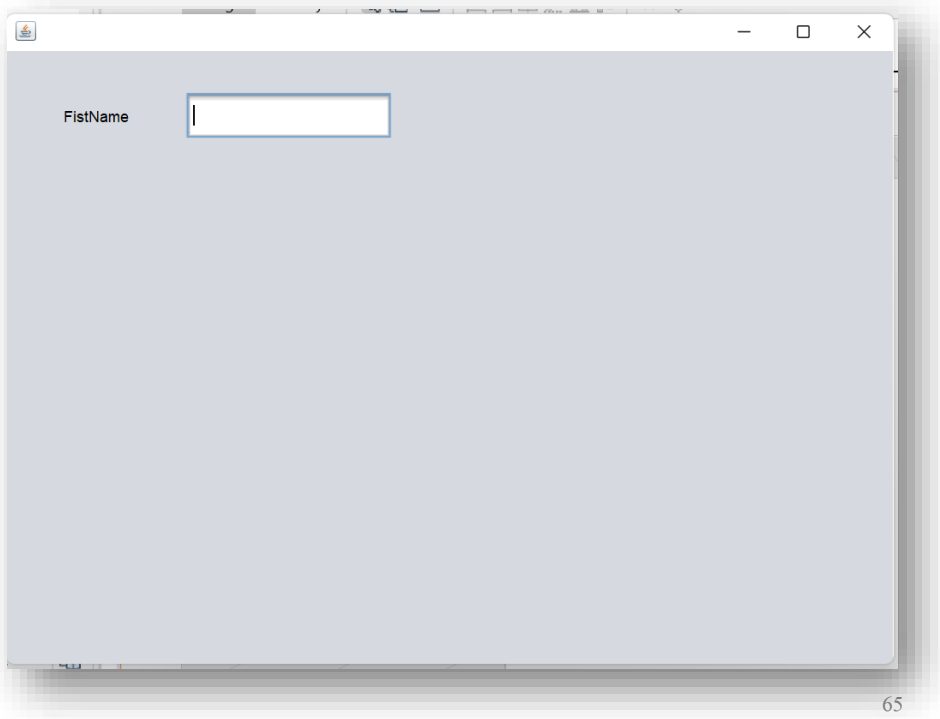
Running the Application

1. Right click on the name of the JFrame Form you created found under your package

2. Click Run File

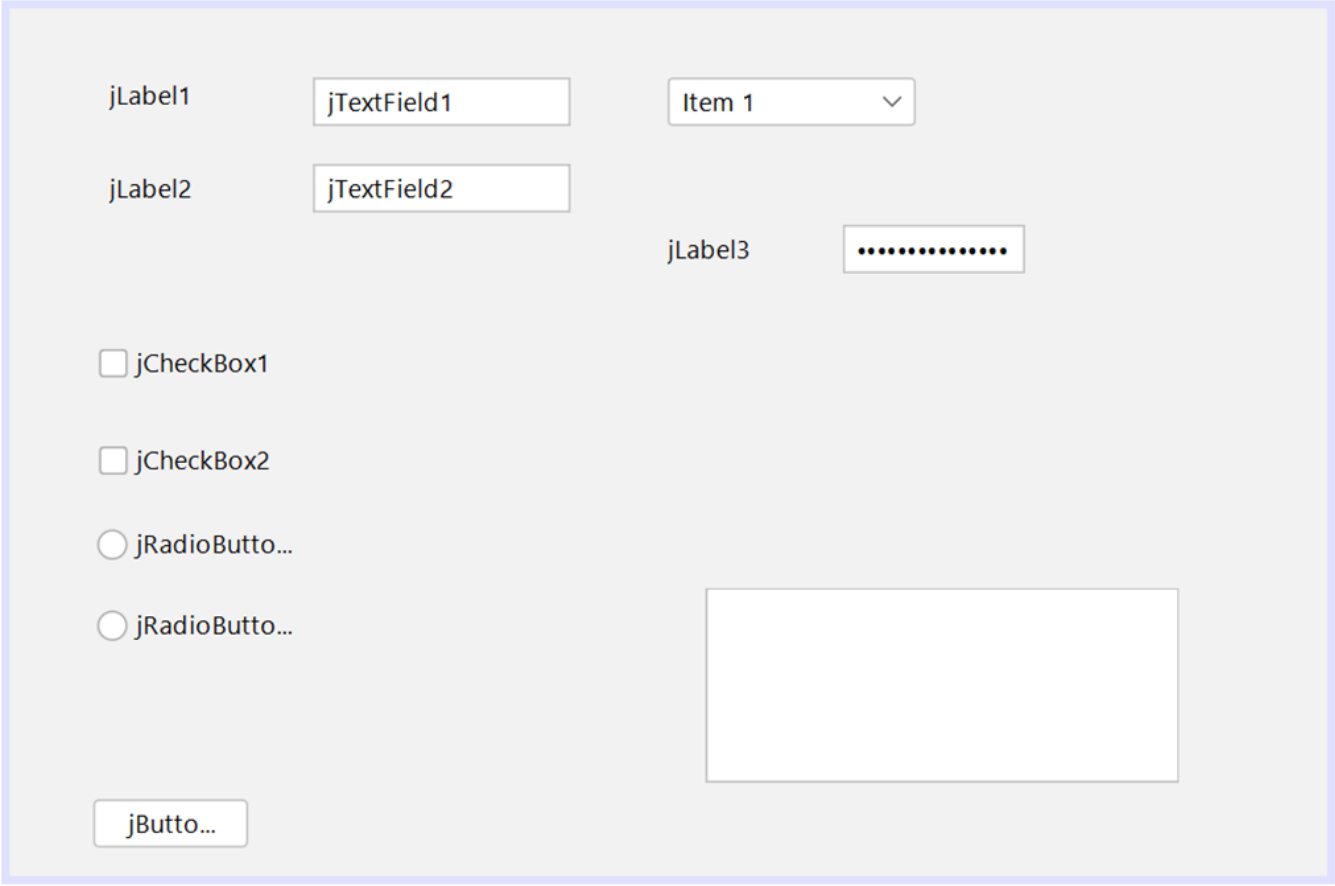


On successful run, you should be seeing



Assignment

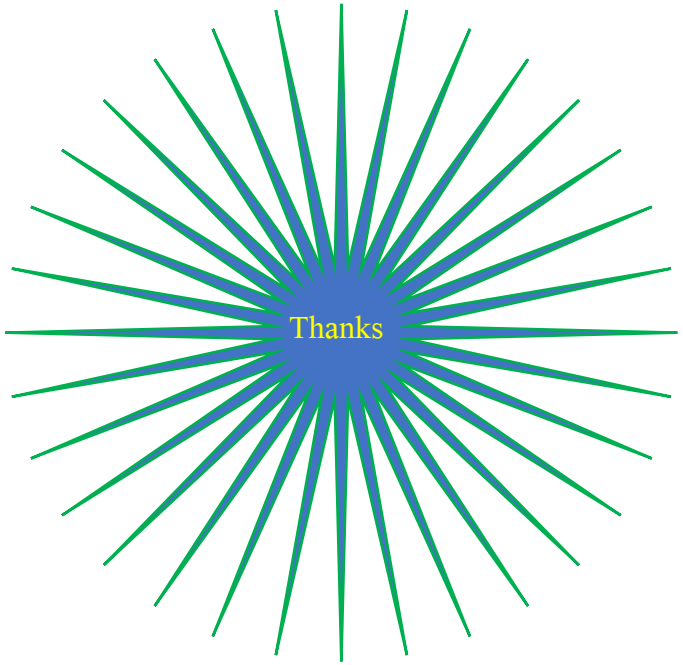
Create a JFrame and drop items as below



Summary

1. GUI(Swing-difination, Fearures of swing, AWT and Swing Hierarchy etc.
2. Swing Components including; Jbutton, JTextField, JPasswordField, JTextArea etc.)
3. Utilization of Swing components based on an IDE

Thank you for Listening



References

Wikimedia Foundation. (2022, January 10). *Java 2d*. Wikipedia. Retrieved October 20, 2022, from https://en.wikipedia.org/wiki/Java_2D

Java swing components and containers. Studytonight.com. (n.d.). Retrieved October 26, 2022, from https://www.studytonight.com/java/java-swing-components.php