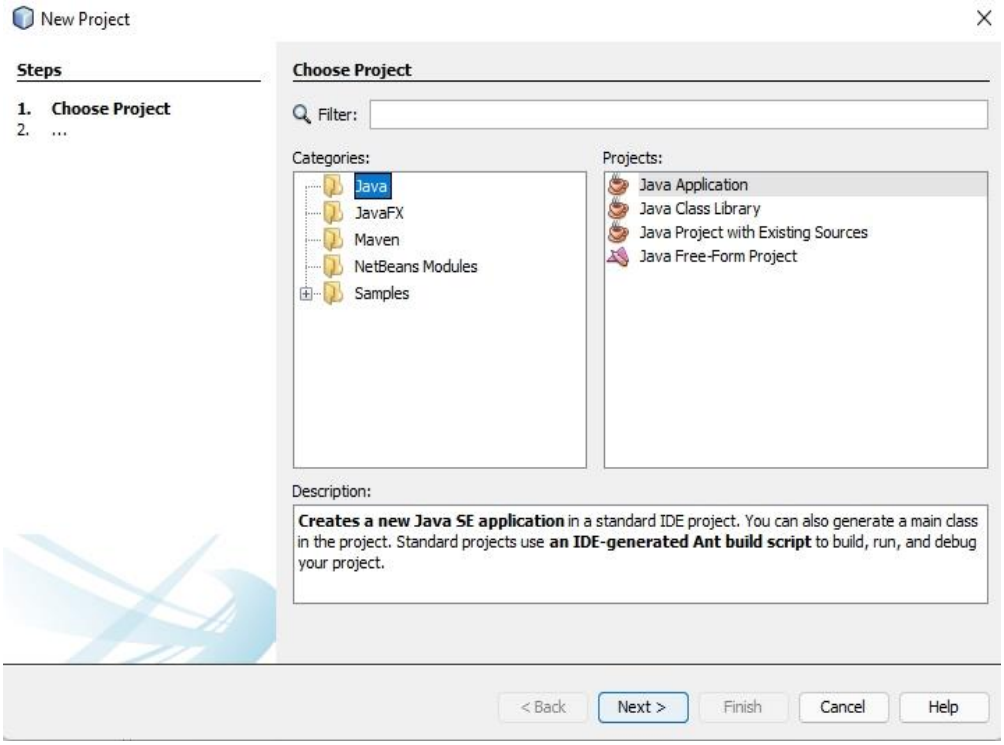
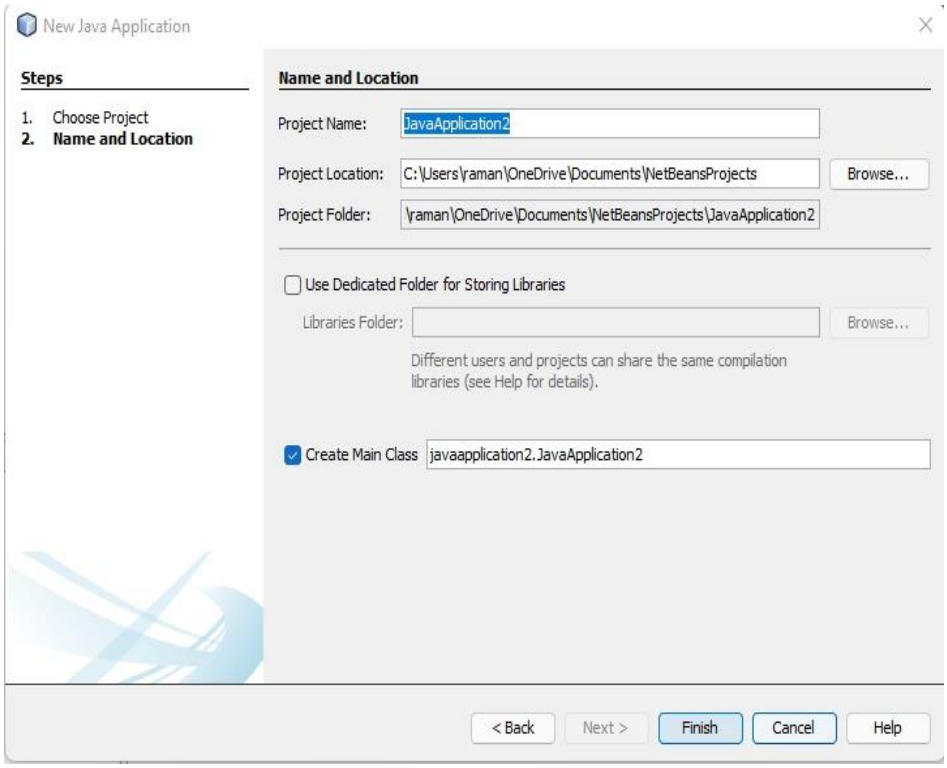
Java Swing Course in Netbeans 8.2

Steps to create a Java Swing Application in Netbeans

1. Select a new project

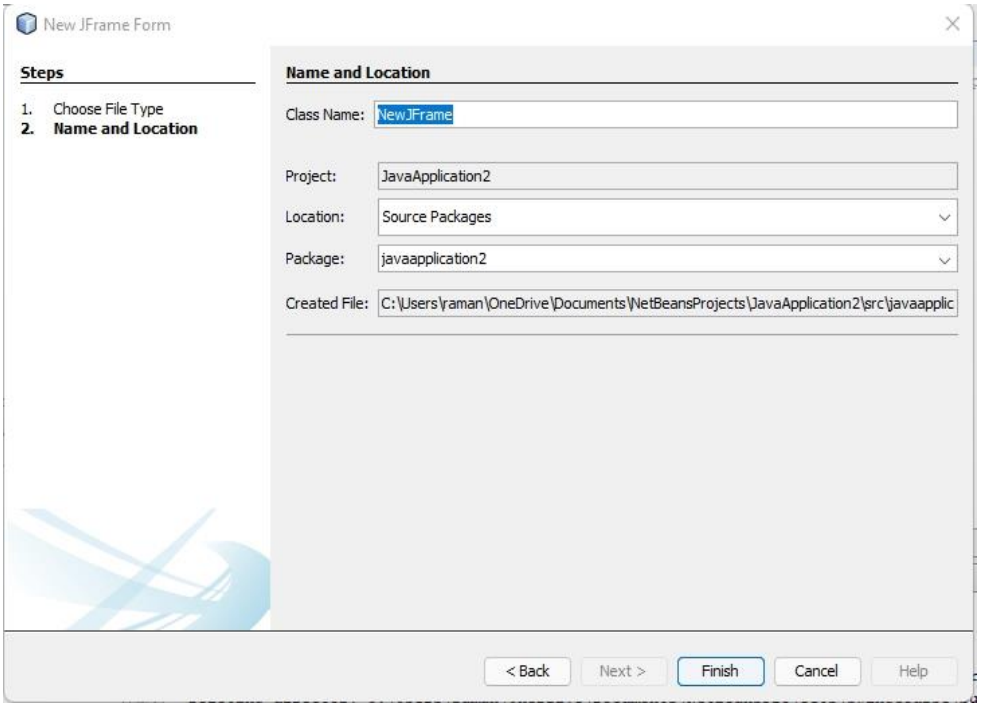


Select Java in Categories and Java Application in Projects
The following window will appear



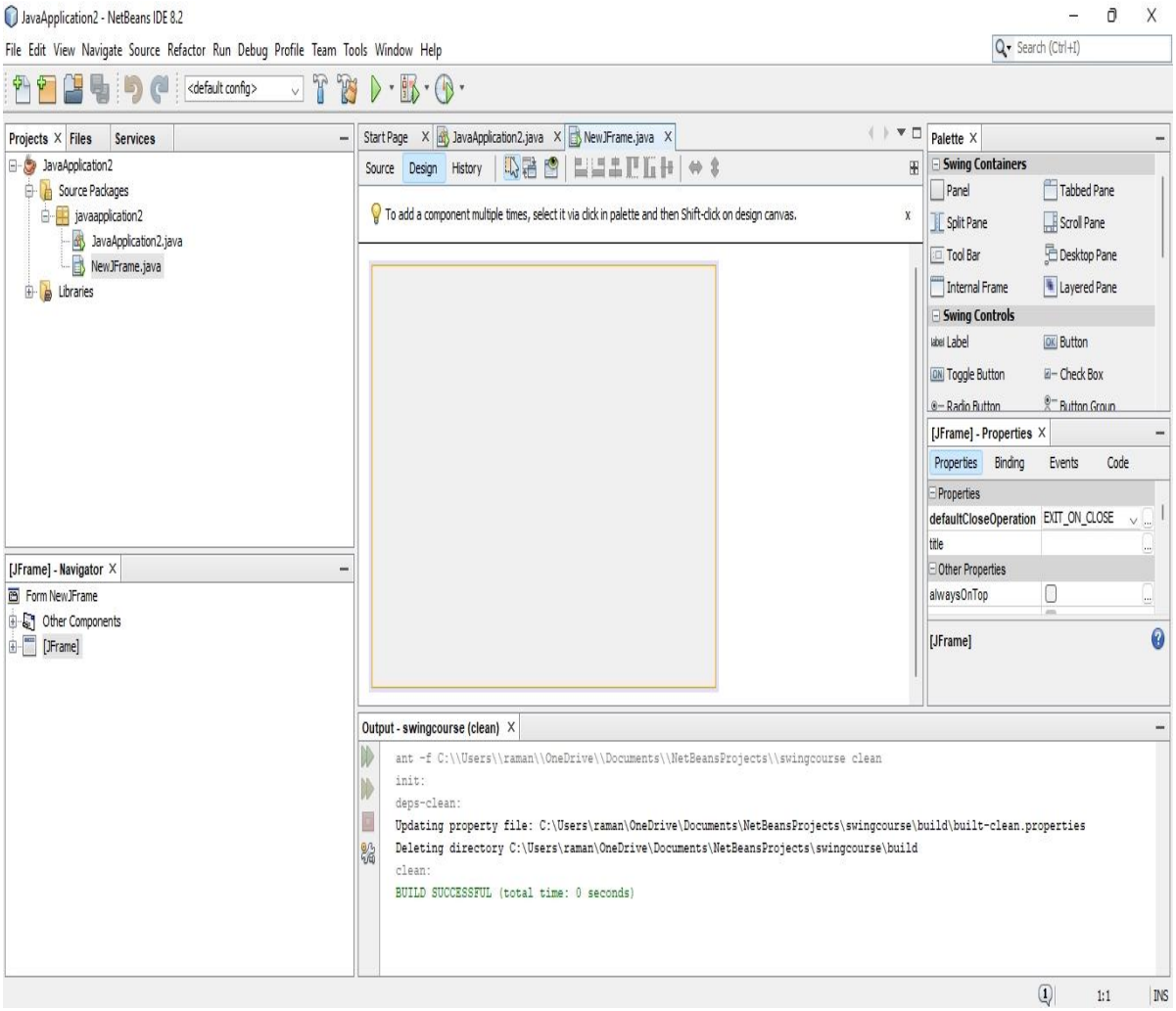
Click Finish

Now Right Click on Icon javaapplication2
and select New Jframe



Click Finish

Following window will appear



Program to enter name of person and display it in a label named as jLabel1

Go to Design View and select Layout of frame as Absolute Layout

Place a Label, TextField , Button and one more Label on the Frame

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
String name="";
name=jTextField1.getText();
jLabel2.setText("Name you entered is " + name);
// TODO add your handling code here:
    }
```

When you click on Button You will get following output
Clicking on a Button will run event jButton1ActionPerformed of the button

Program to add , subtract , multiply, quotient and remainder of two numbers

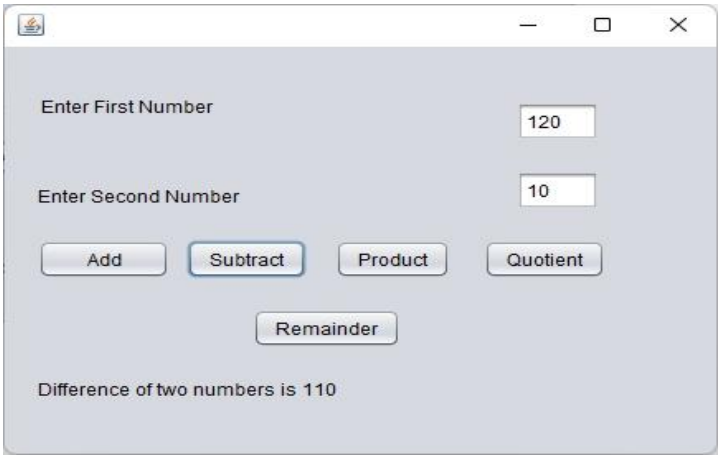
Click on

above button to run jFrame2

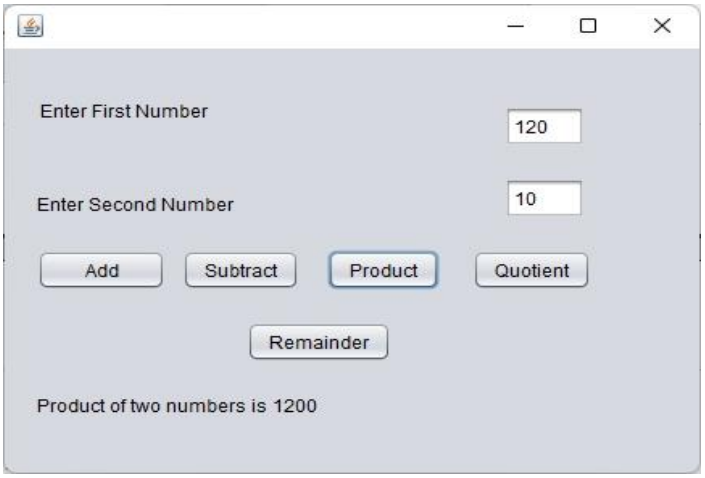
```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
int a,b,sum;
a=Integer.parseInt(jTextField1.getText());
b=Integer.parseInt(jTextField2.getText());
sum=a+b;
jLabel3.setText("Sum of two numbers is " + sum);
// TODO add your handling code here:
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
int a,b,diff;
a=Integer.parseInt(jTextField1.getText());
b=Integer.parseInt(jTextField2.getText());
diff=a-b;
jLabel3.setText("Difference of two numbers is " + diff);      // TODO add your handling code here:
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
int a,b,product;
a=Integer.parseInt(jTextField1.getText());
b=Integer.parseInt(jTextField2.getText());
product=a*b;
jLabel3.setText("Product of two numbers is " + product);          // TODO add your handling code
here:
    }
```

Output Image for jButton4 Action Performed event method is
Output Image for jButton1 Action Performed event method is

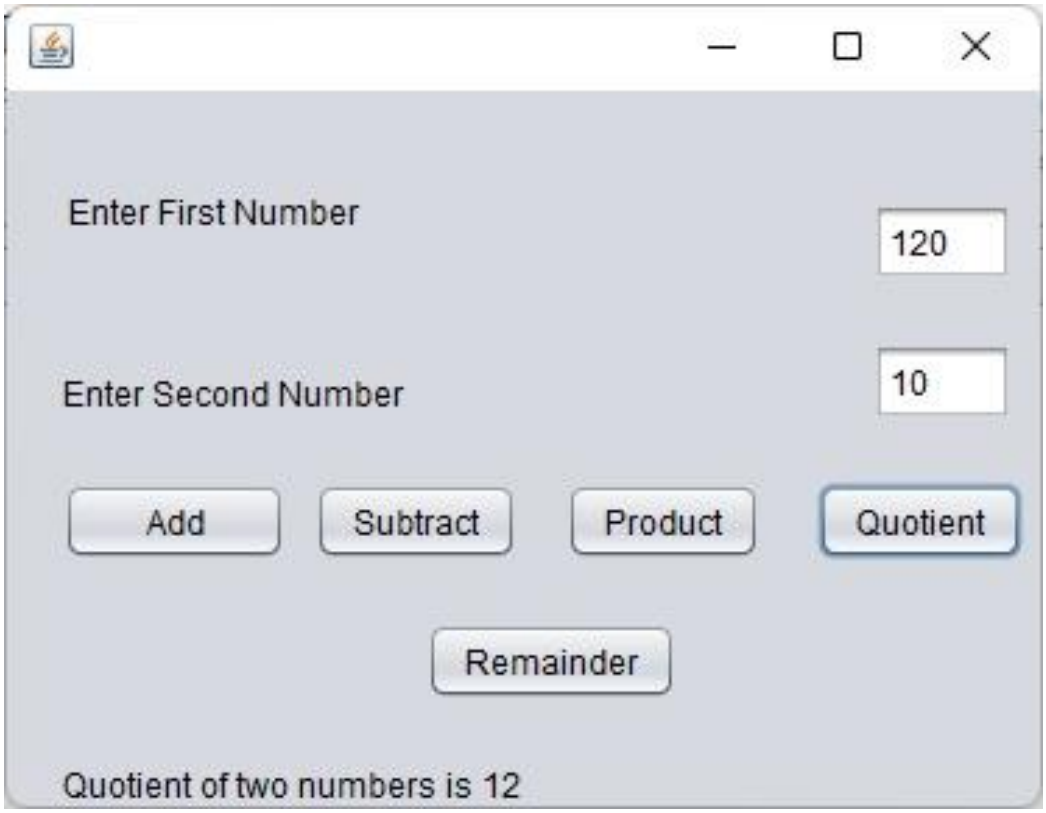
Output Image for jButton2 Action Performed event method is



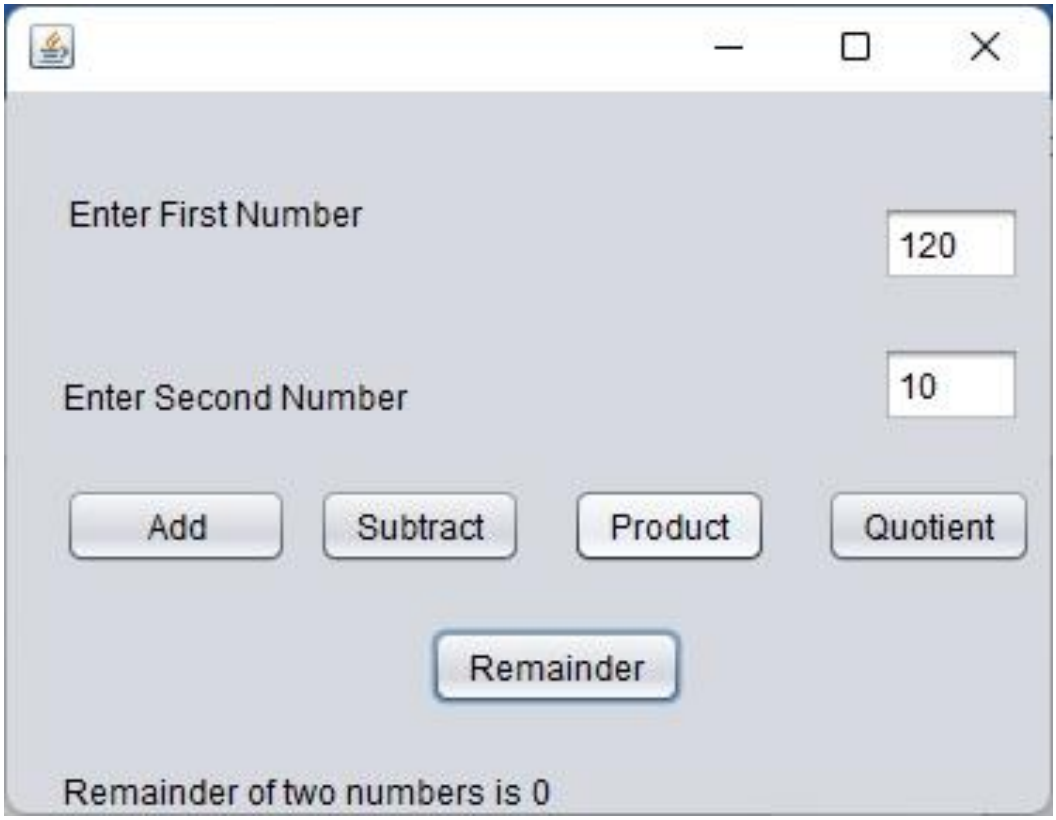
Output Image for jButton3 Action Performed event method is



Output Image for jButton4 Action Performed event method is



Output Image for jButton5 Action Performed event method is



Program to demonstrate ToggleButton in Java Swing

When clicked on ToggleButton ItemEventListener Method is invoked

```
private void jToggleButton1ItemStateChanged(java.awt.event.ItemEvent evt) {
        int state = evt.getStateChange();

                // if selected print selected in console
                if (state == evt.SELECTED) {
                   jLabel1.setText("Selected");
                }
                else {

                   // else print deselected in console
                   jLabel1.setText("Deselected");
                }    // TODO add your handling code here:
    }
```

Output

Program to demonstrate jCheckBox Button

On Clicking on Button jButton1, if jCheckBox1 is Selected , JLabel1 Text will be set as Cricket CheckBox is Selected. If jCheckBox2 is Selected JLabel2 Text will be set as Cricket CheckBox is not Selected.

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
if(jCheckBox1.isSelected()==true)
{
    jLabel1.setText("Cricket CheckBox is Selected");
}
else
{
    jLabel1.setText("Cricket CheckBox is not Selected");
}
// TODO add your handling code here:
    }
```

RadioButtons

Radiobutton is a control to select a single option from multiple options.
To use RadioButton in Java Swing Application, we need ButtonGroup Radio Button
We set the property buttonGroup to buttonGroup1 and we have to add jButtonGroup to Java swing frame. jButton Group is a control but if we place it on a form, buttongroup is not shown on the form. We can also set selected property to true of jRadioButton to set it by default.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
if(jRadioButton1.isSelected()==true)
{
    jLabel1.setText("Male Radio Button is Selected");
}
else
{
    jLabel1.setText("Female Radio Button is Selected");
}
// TODO add your handling code here:
    }
```

When radiobutton1 is selected , jLabel will display “Male Radio Button is Selected” and when radiobutton2 is selected and on click jLable1 will display “Female Radio Button is Selected”.

Combo Box Control

Combo Box is a drop down list box.
Example given below displays the selected item in the Combo Box in jLabel.
Model property of Combo Box is a list of items like a list

- Cricket
- Football
- Table Tennis
- Badminton

Only one option from multiple options can be retrieved from Combo Box on clicking a jButton.

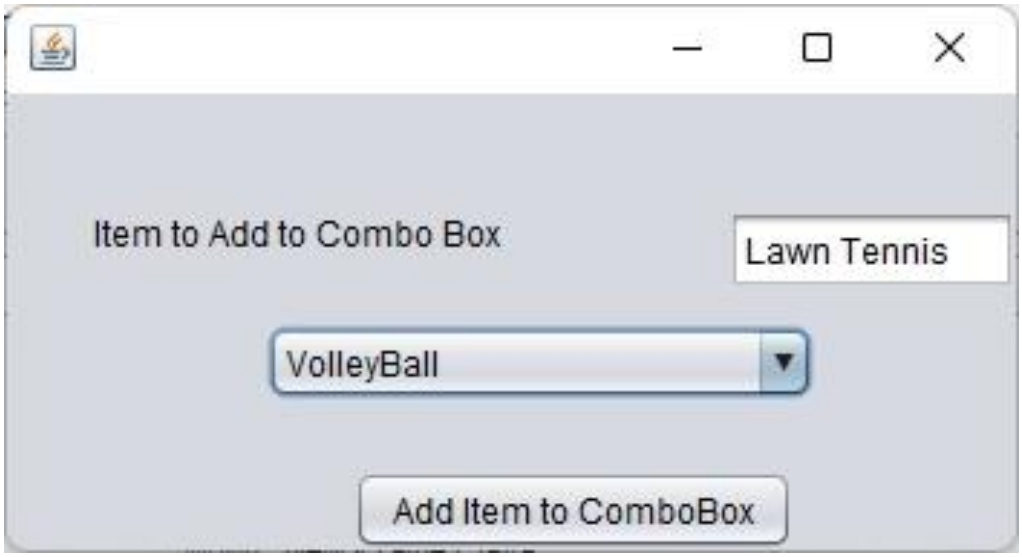
```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
jLabel1.setText(jComboBox1.getSelectedItem().toString());        // TODO add your handling code
here:
    }
```

Output

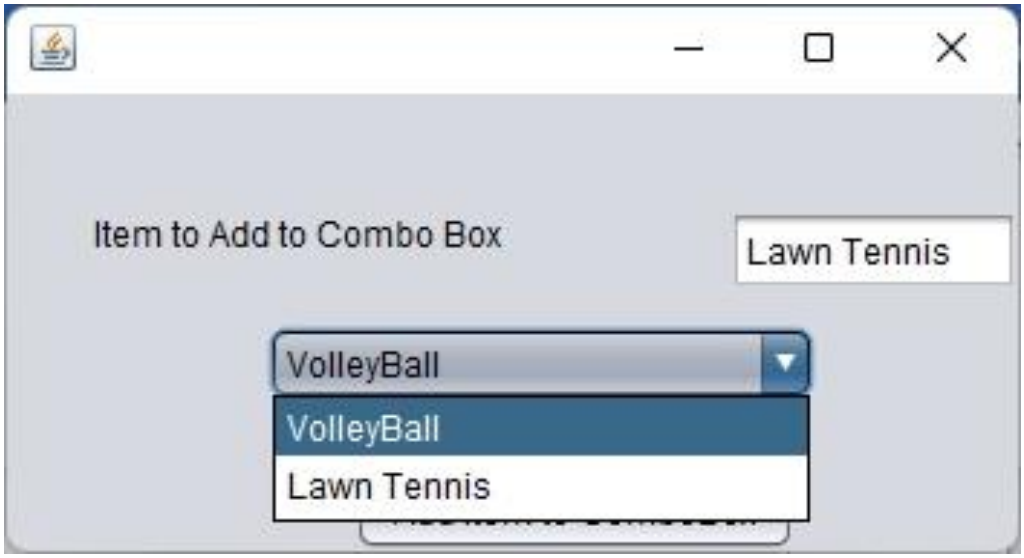
```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
jLabel1.setText(jComboBox1.getSelectedItem().toString());        // TODO add your handling code
here:
    }
```

Following example is to add an item to Combo Box

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
jComboBox1.addItem(jTextField1.getText());        // TODO add your handling code here:
    }
```



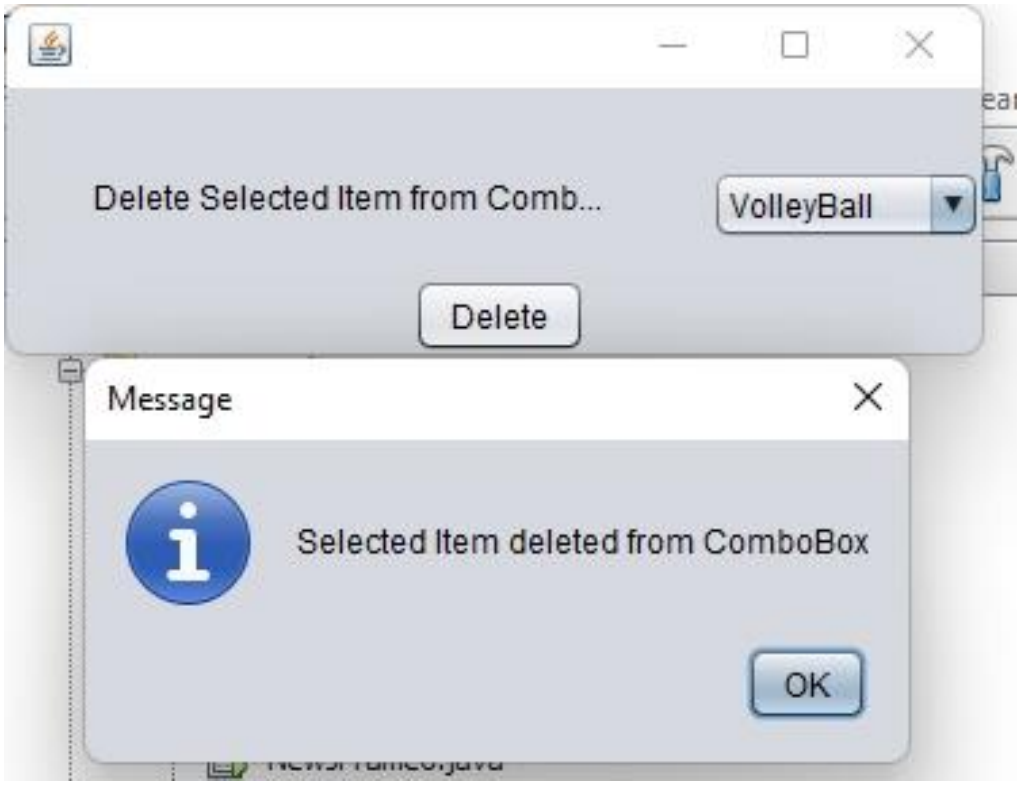
Following example is to show records in a Combo Box



To delete a item from combo box we use object of class DefaultComboBoxModel.

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
DefaultComboBoxModel model = (DefaultComboBoxModel) jComboBox1.getModel();
model.removeElementAt(jComboBox1.getSelectedIndex());
JOptionPane.showMessageDialog(this,"Selected Item deleted from ComboBox");
// TODO add your handling code here:
    }
```

We have to import two classes javax swing package
import javax.swing.DefaultComboBoxModel;
import javax.swing.JOptionPane;



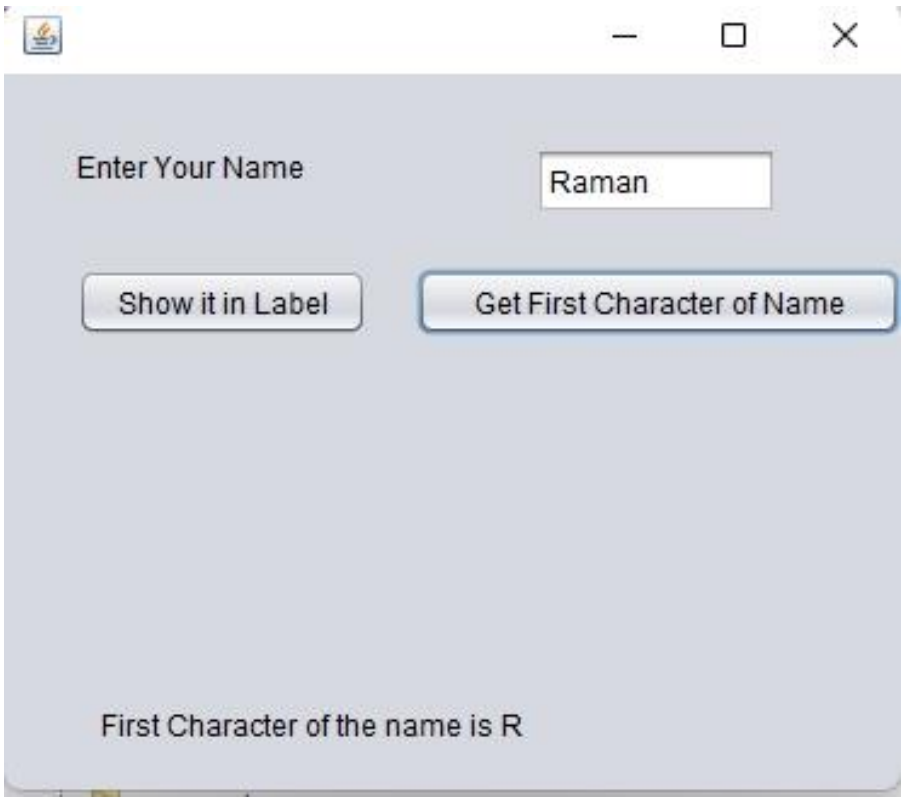
jTextField Class

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
String name="";
name=jTextField1.getText();
jLabel1.setText("Name You Entered is " + name);
// TODO add your handling code here:
    }
```

Following code gives First Character of name in a jLabel Control

```
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
String name="";
char ch;
name=jTextField1.getText();
ch=name.charAt(0);
jLabel2.setText("First Character of the name is " + ch);        // TODO add your handling code here:
    }
```

Output



We use charAt() of String class to get first alphabet or character of String entered in jTextField.

Following code is to reverse a string entered in a TextField
We have used StringBuilder class to reverse a string and output will be shown in jLabel2.

To Set Font Size of the String in jTextField

```
    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
Font font1 = new Font("SansSerif", Font.BOLD, 20);
jTextField1.setFont(font1);        // TODO add your handling code here:
    }
```

We have used a class Font to a new Font Size to the String in jTextField1.
In the above example constructor of class Font takes three Parameters

1. Font Name
2. Font Type like BOLD, ITALIC
3. Font Size which is taken as 20.

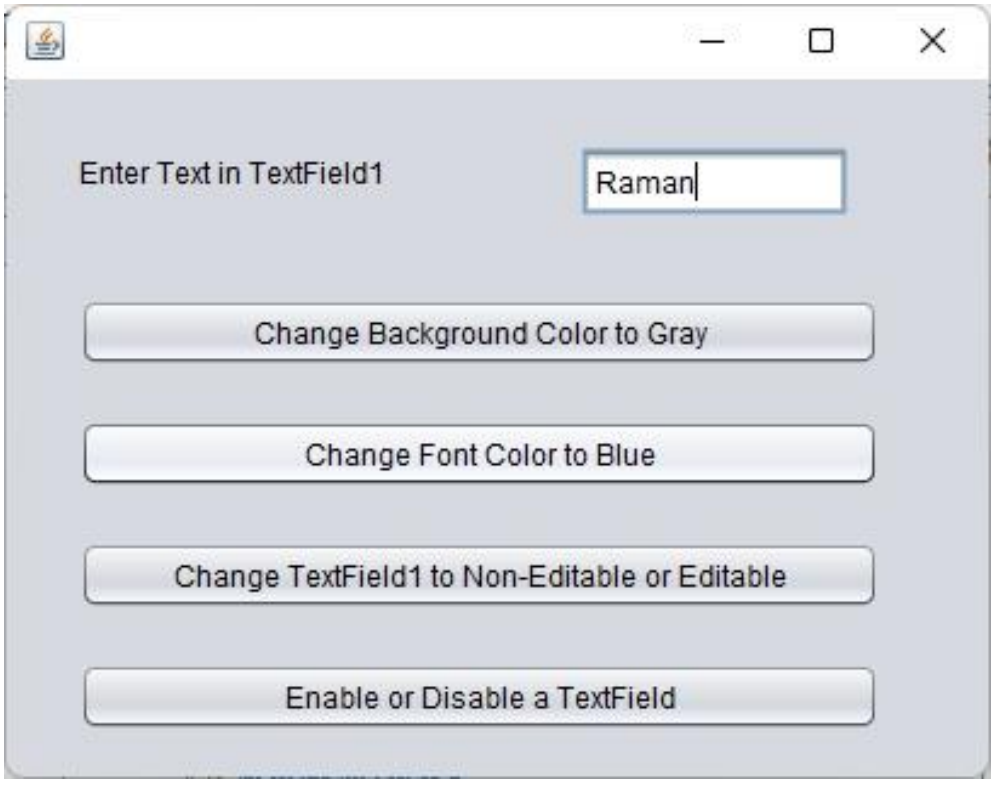
```
    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
Font font1 = new Font("SansSerif", Font.ITALIC, 20);
jTextField1.setFont(font1);        // TODO add your handling code here:
    }
```

Change Background Color to GRAY.
We have used Class Color class to set color of textfield to Gray

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
 Color color = Color.GRAY;
 jTextField1.setBackground(color);        // TODO add your handling code here:
    }
```

Change Foreground Color to BLUE

```
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
  Color color = Color.BLUE;
 jTextField1.setForeground(color);       // TODO add your handling code here:
    }

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
if(jTextField1.isEditable()==true)
{
   jTextField1.setEditable(false);
}
else
{
   jTextField1.setEditable(true);
}// TODO add your handling code here:
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
 if(jTextField1.isEnabled()==true)
{
   jTextField1.setEnabled(false);
}
else
{
   jTextField1.setEnabled(true);
}
 // TODO add your handling code here:
    }
```



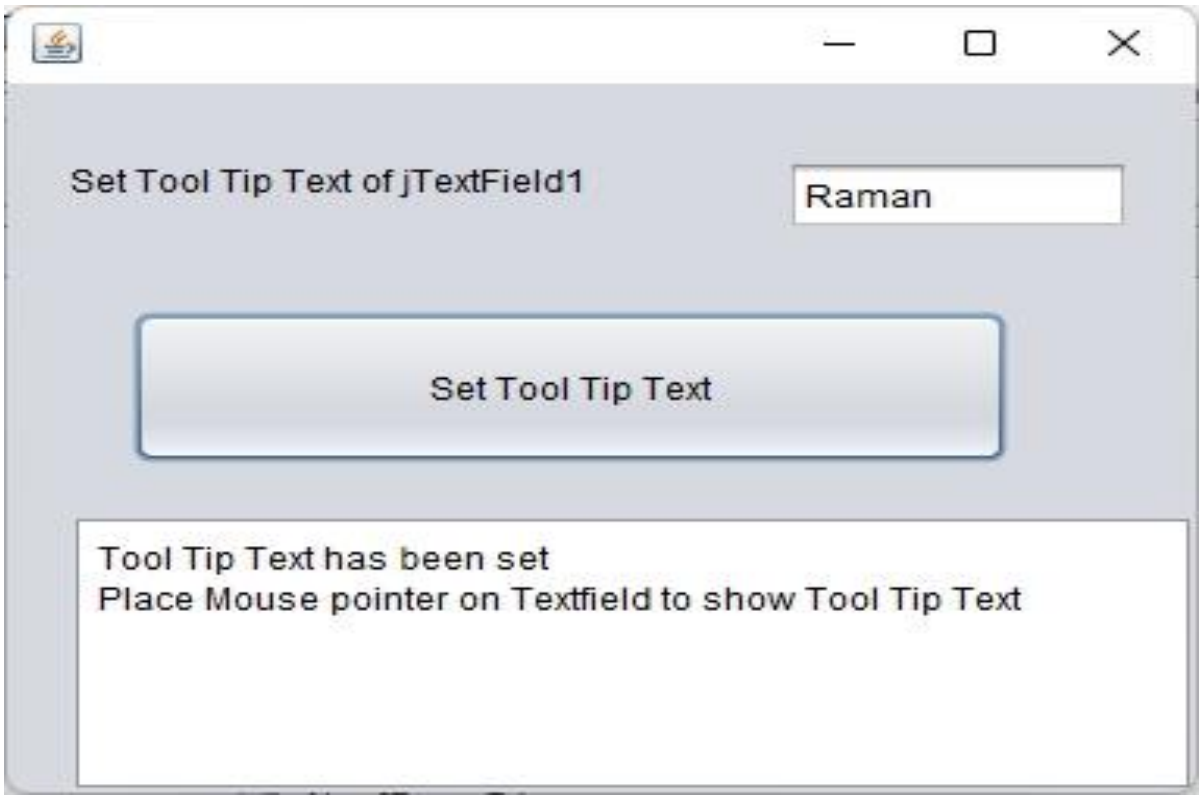
Hide or show a jTextField
Or how to make jTextField as Visible or Invisible

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
if(jTextField1.isVisible()==true)
{
   jTextField1.setVisible(false);
}
else
{
   jTextField1.setVisible(true);
}// TODO add your handling code here:
    }
```

Set Tool Tip Text
What is a Tool Tip Text.
Tool Tip Text is Text which is shown when place a mouse pointer on the jTextField

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
jTextField1.setToolTipText("This is JTextField's ToolTip Text");
jTextArea1.setText("Tool Tip Text has been set \nPlace Mouse pointer on Textfield to show Tool Tip Text");// TODO add your handling code here:
    }
```

Output



jList Control – ListBox Control

DefaultListModel model;
(Add the above statement constructor)

```
    public NewJFrame11() {  //This is a constructor
        initComponents();
        model = new DefaultListModel(); // creating instance of class DefaultListModel
    }
```

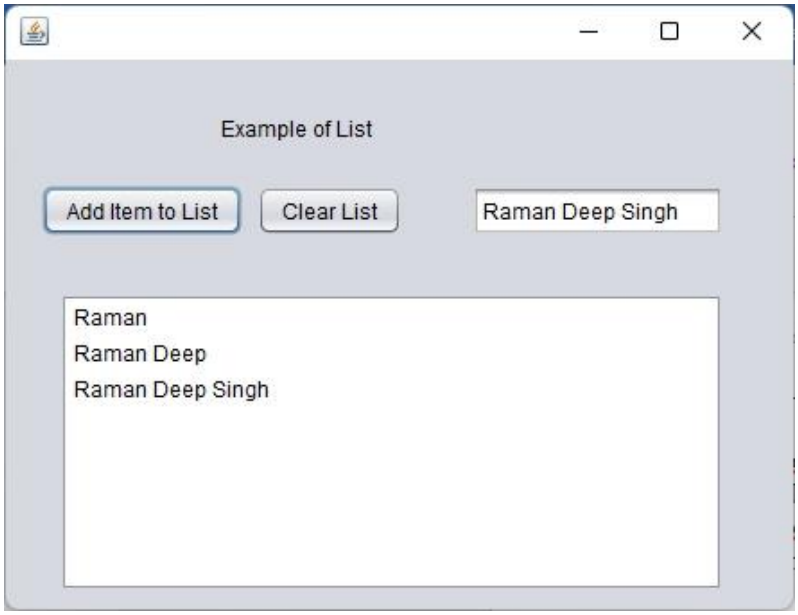
Code to add Element to jList1

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
String a="";
a=jTextField1.getText();
model.addElement(a);
 jList1.setModel(model);

// TODO add your handling code here:
    }
```

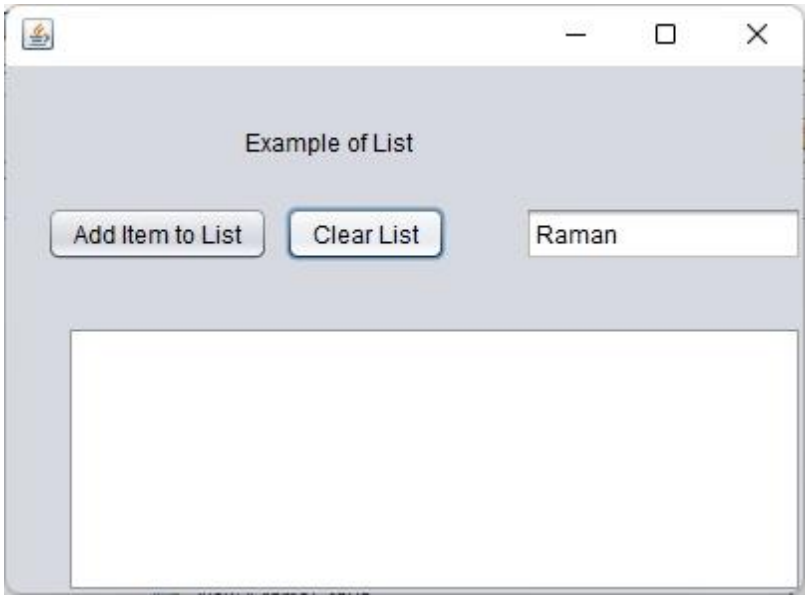
Code to clear the List

```
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
model.clear();
jList1.setModel(model);// TODO add your handling code here:
    }
```

Output of Add Element



Output of Clear List

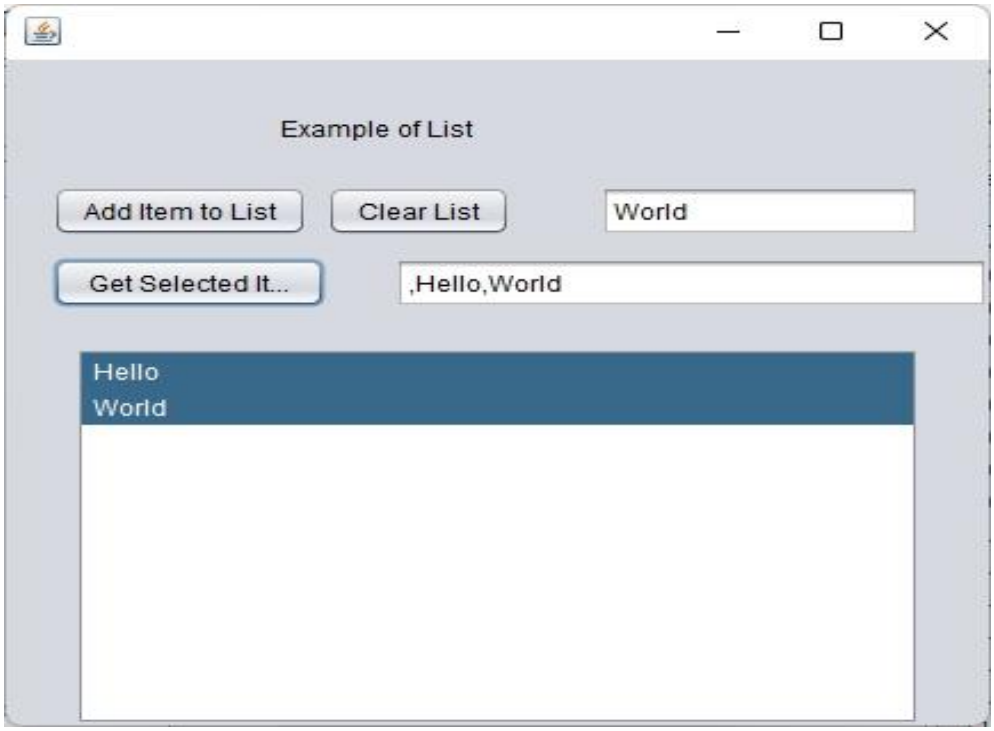


There is a property of jList Control that is selectionMode which means that whether we can select a single item of list or multiple items in List

If selectionMode is set as SINGLE, only a single item can be selected. If selectionMode is set to MULTIPLE_INTERVAL, multiple items can be selected in the list.

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
String a="";
ArrayList arr;
```

```
jTextField2.setText("");
arr=(ArrayList)jList1.getSelectedValuesList();
Iterator itr=arr.iterator();
while(itr.hasNext())
{
   jTextField2.setText(jTextField2.getText() + "," + itr.next());
}

// TODO add your handling code here:
   }
```



Create a PDF File from TextArea

```
try
{
   Document document = new Document(new Rectangle(PageSize.A4));

   PdfWriter writer = PdfWriter.getInstance(document, new
FileOutputStream("c:\\temp\\pdffile1.pdf"));
   document.open();
            document.add(new Paragraph(jTextArea1.getText()));
    document.close();
}
catch(Exception e)
{
   JOptionPane.showMessageDialog(this,e.toString());
}

   private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
String a="";
a=jTextField1.getText();
jTextArea1.append(a);
```

```
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
jTextArea1.setText("");        // TODO add your handling code here:
    }
```

Import packages

```
import com.itextpdf.text.Document;
import com.itextpdf.text.DocumentException;
import com.itextpdf.text.PageSize;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.Rectangle;
import com.itextpdf.text.pdf.Barcode128;
import com.itextpdf.text.pdf.PdfWriter;
import java.io.FileOutputStream;
```