



**National University of Computer & Emerging Sciences,  
Karachi**



**Computer Science Department  
Spring 2022, Lab Manual – 11**

<b>Course Code: CL-217</b>	<b>Course : Object Oriented Programming Lab</b>
<b>Instructor(s) :</b>	<b>Abeer Gauher, Hajra Ahmed, Syed Zain ul Hassan</b>

## **LAB - 11**













### **Java GUI**

# Java Swing

Swing is the principal GUI toolkit for the Java programming language. It is a part of the JFC (Java Foundation Classes), which is an API for providing a graphical user interface for Java programs. It is completely written in Java.

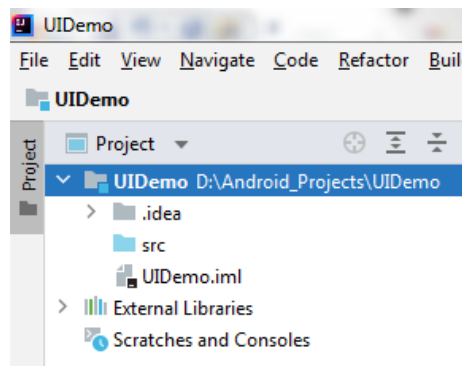
## Java Swing Components

Following are some examples of the Swing components available for use in Java GUI/Forms applications.

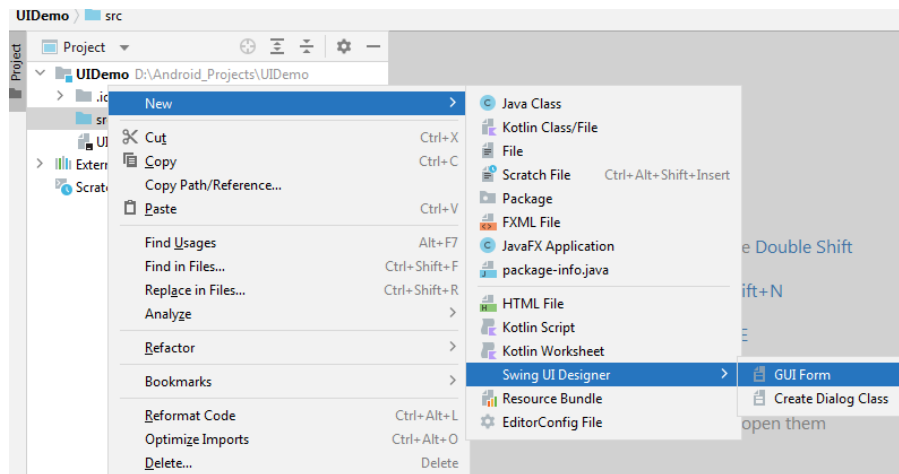
-  HSpacer
-  VSpacer
-  JPanel
-  JScrollPane
-  JRadioButton
-  JButton
-  JCheckBox
-  JLabel
-  JTextField
-  JPasswordField
-  JFormattedTextField
-  JTextArea
-  JTextPane
-  JEditorPane
-  JComboBox
-  JTable

## Creating GUI Application in Java (step-by-step)

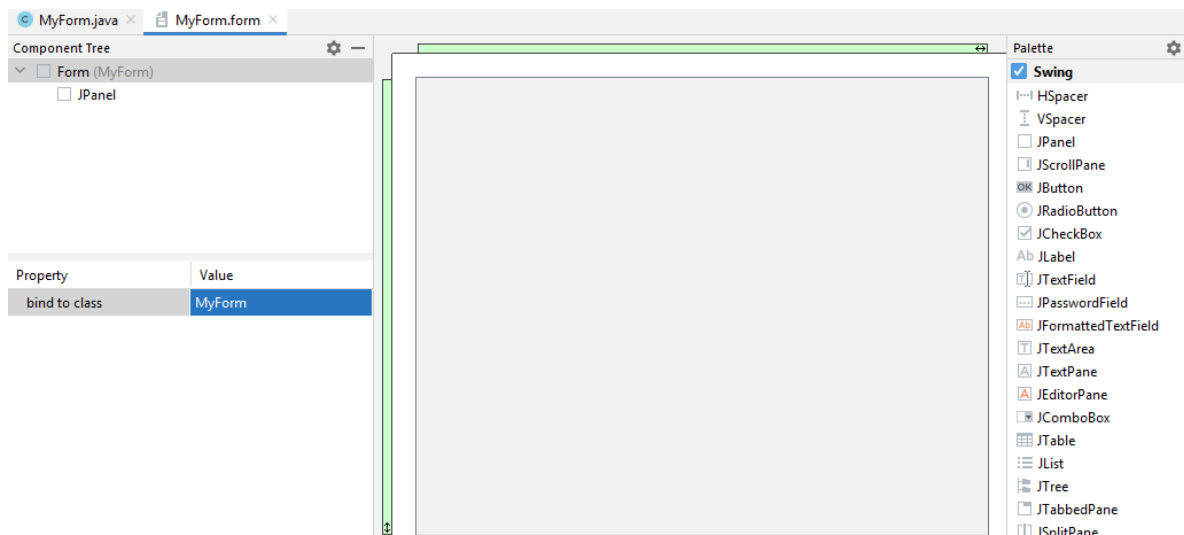
Step 1: To create a GUI Application in Java, open IntelliJ IDE and create a new console application project.



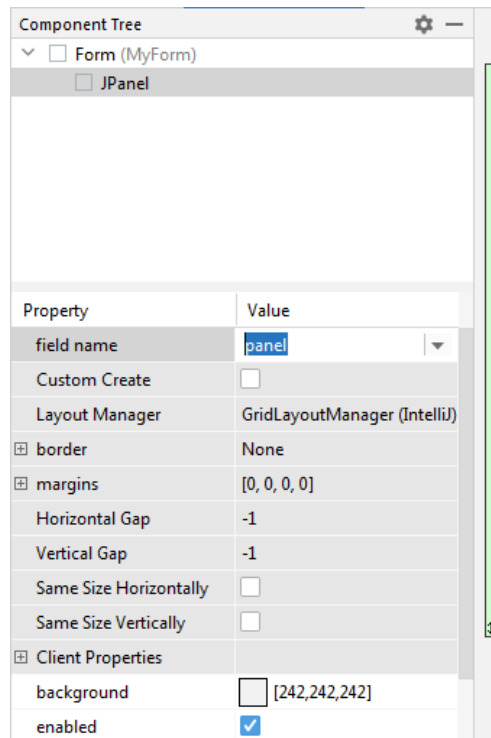
Step 2: Now right-click on the src icon and add a new GUI Form.



Step 3: You should now add the GUI form and rename it. (MyForm.form) in our example. After adding the form, you will view the designer area where all the Swing controls can be added. The JPanel serves as the main area where other controls can be added to create the UI.



Step 4 (Optional): If you want you can rename the controls (JPanel, JButton etc) using the Property Window and set other properties as well.

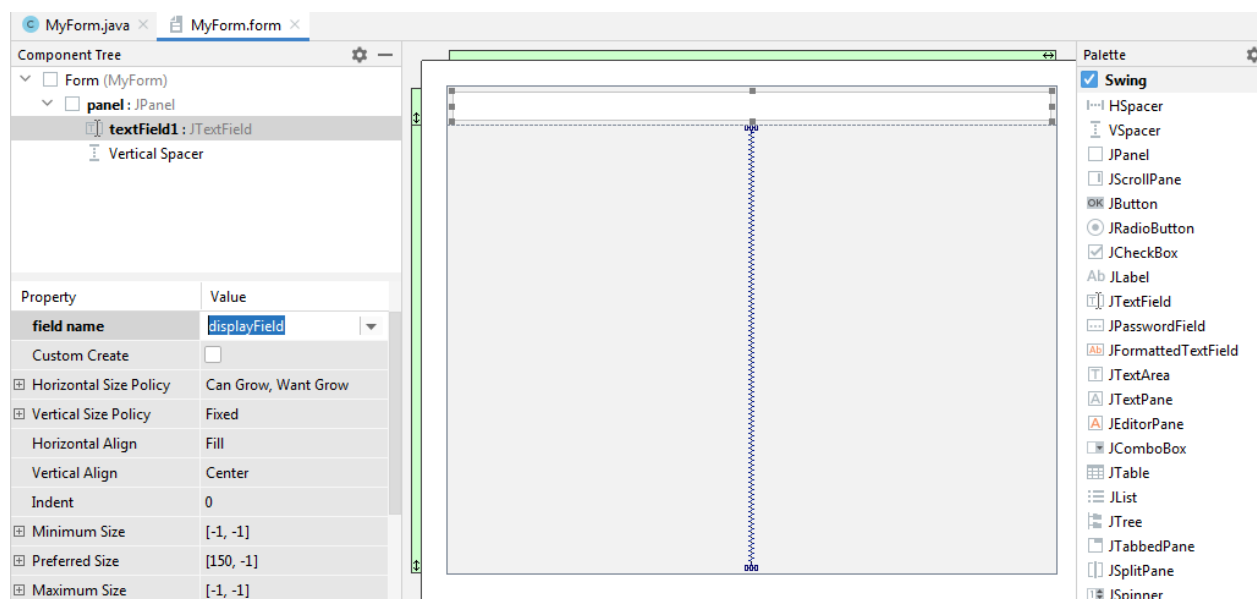


You can also modify other useful properties to change the appearance and behavior of your UI controls such as border, margins etc.

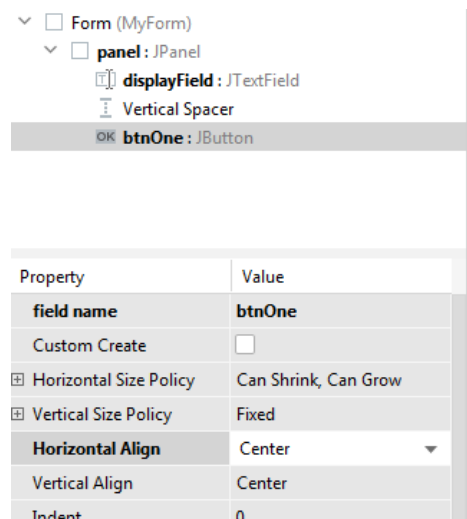
Step 5: To add a new control to the form area. Simply drag and drop it as following:

## Example: GUI based Calculator

1) Drag and drop JTextField in your form area and rename it to displayField (you can choose any other name as well e.g. textField1). This will be field to display input and results of our calculations.

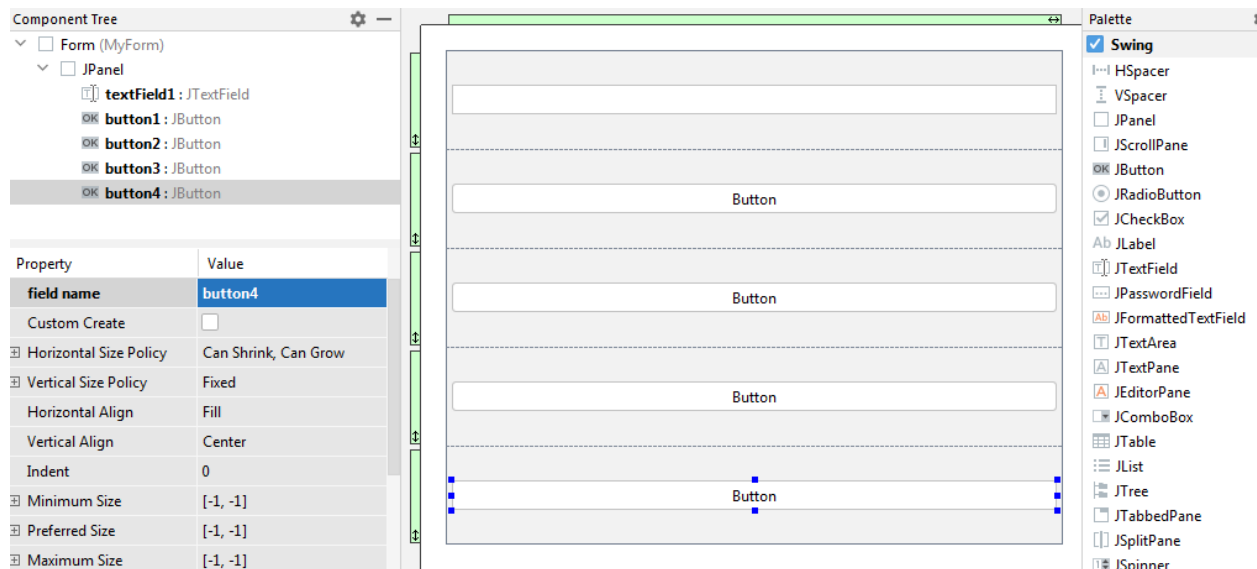


2) In order to give inputs for calculations, we will need buttons. Go ahead and drag-drop JButton to the form area. Then click on that button in either design view or Component Tree and change its properties **field name** to "btnOne" and **horizontal align** to "center".

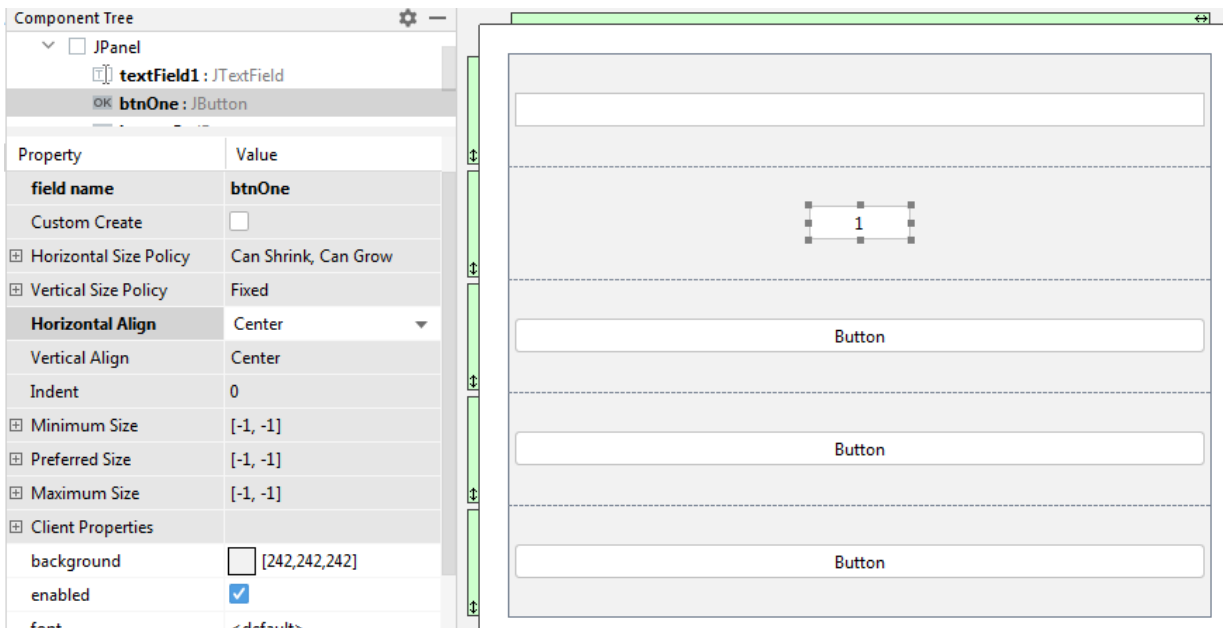


3) Also change its vertical align property to "Top".

4) Now drag and drop JButton from the menu to your form.

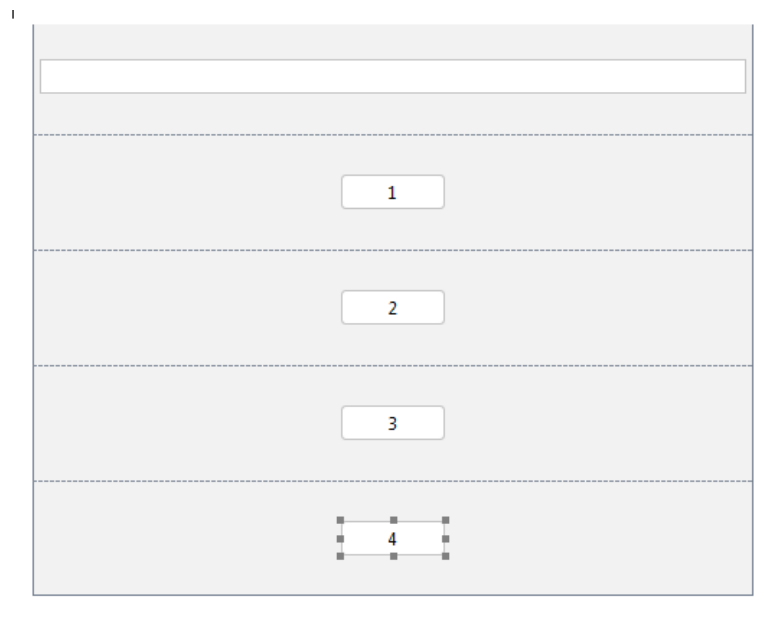


Similarly drag and drop three more buttons. Now change the field name property of the first button to "btnOne" and text property to "1". And also horizontal align property to "center".

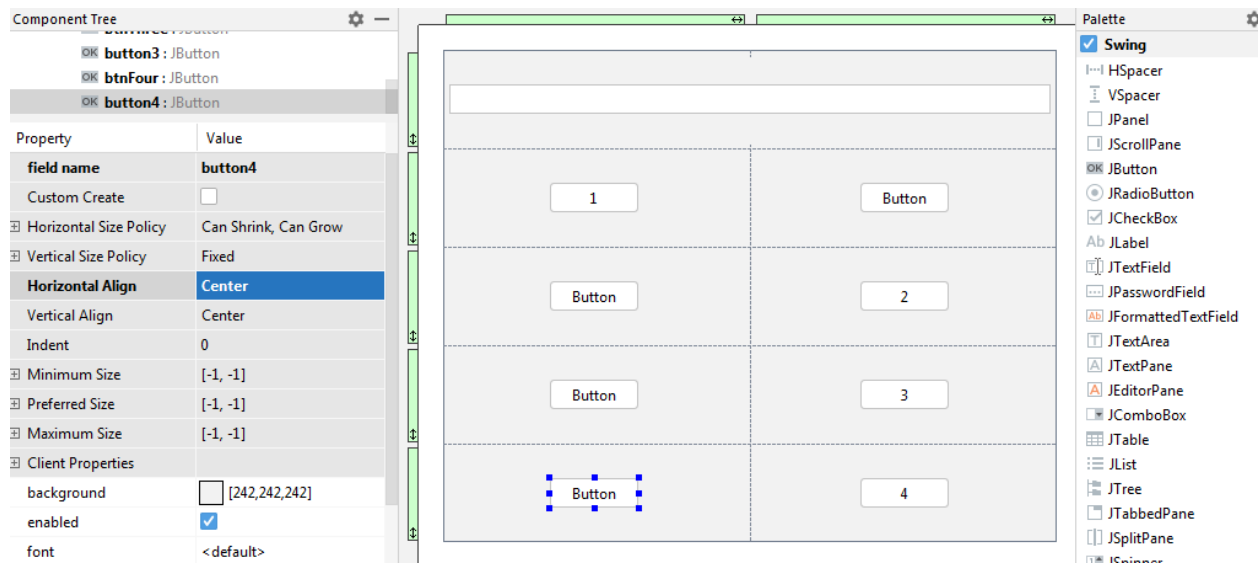


Note: You can also set other properties for this button's appearance as you like.

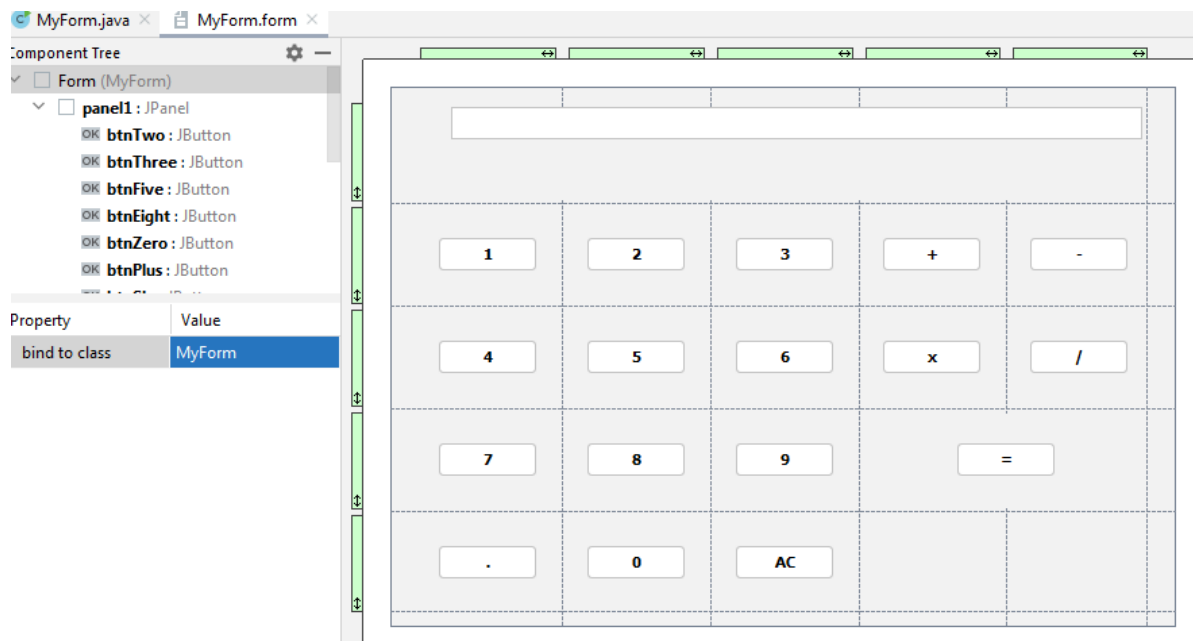
5) Do the same for other buttons such that we have second button's field name property set to "btnTwo" and text set to "2", third button's field name set to "btnThree" and text to "3" and so on.



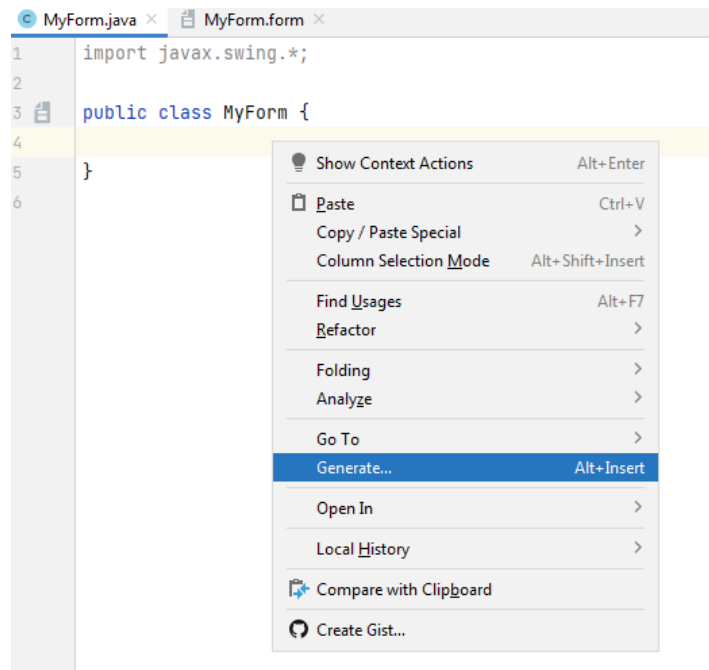
6) Drag and drop some more buttons in the same way.



7) Add more buttons and set the properties in similar manner to create the following final layout.

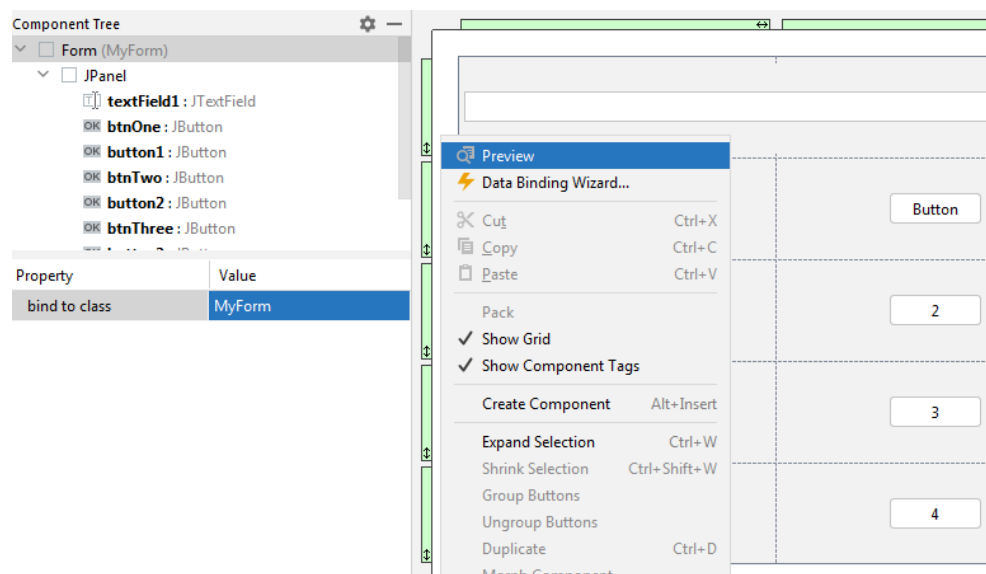


8) Go to **MyForm.java** and right click anywhere in the code area to click on "Generate".



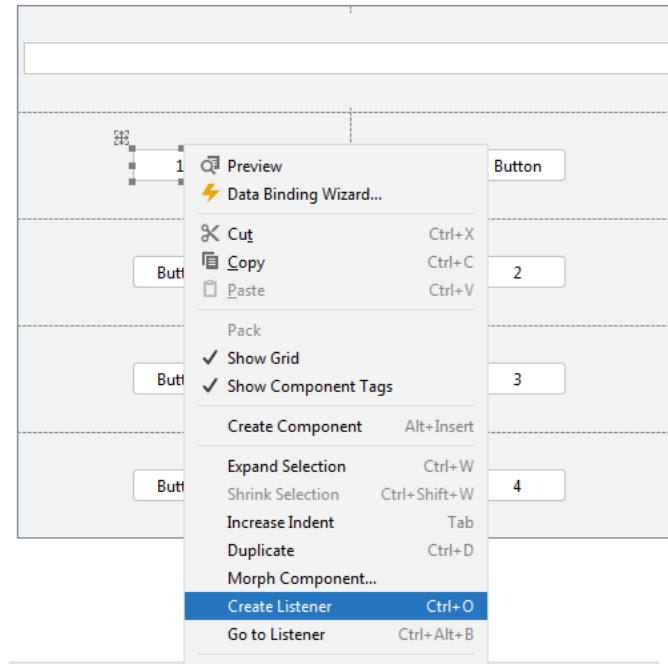
9) After clicking on Generate, click on Form Main( ) to generate main function for your application.

10) To view your layout so far, go to design view and select "Preview".

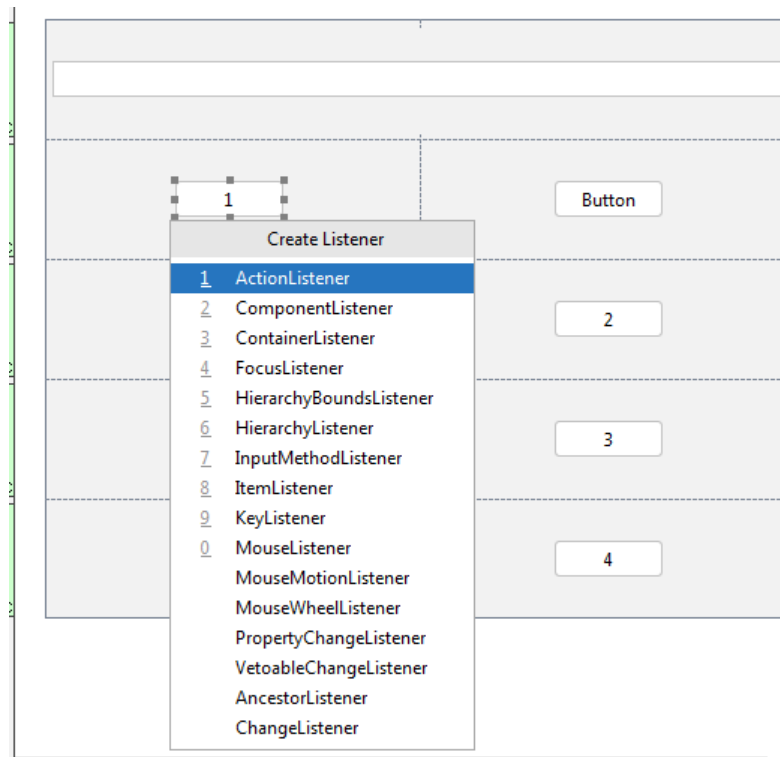


11) Now click on btnOne and select "Create Listener".





12) Now select Action Listener from the options. This will create a function in your code that will be executed each time the button is clicked.



13) Go to MyForm.java and create the following variables to be used for calculations.

```

public class MyForm {
    private double total1 = 0.0;
    private double total2 = 0.0;
    private char operator;

```

14) Add the following code to BtnOne's action performed function (this function will be auto generated after performing Step 12).

```

btnOne.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String btnOneText = textField1.getText() + btnOne.getText();
        textField1.setText(btnOneText);
    }
});

```

The getText( ) and setText( ) methods are used to read and write the text property of the button.

15) Copy the same code (but changing name of button and String local variable) in all of the action performed functions of btnTwo, btnThree,..., btnZero. Since the action is similar for all the digits. For example, code for btnTwo would be.

```

btnTwo.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String btnTwoText = textField1.getText() + btnTwo.getText();
        textField1.setText(btnTwoText);
    }
});

```

16) Add the following code to btnPoint's function.

```

btnPoint.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if(textField1.getText().equals(""))
        {
            textField1.setText("0.");
        }
        else if(textField1.getText().contains("."))
        {
            btnPoint.setEnabled(false);
        }
        else
        {
            String btnPointText = textField1.getText() + btnPoint.getText();
            textField1.setText(btnPointText);
        }
        btnPoint.setEnabled(true);
    }
});

```

The `setEnabled` method takes a boolean argument to indicate whether the button is "enabled" for use or not.

17) Now create a global method in your `MyForm.java` class to check which operator was selected.

```
private void get_operator(String btnText)
{
    operator = btnText.charAt(0);
    total1 = total1 + Double.parseDouble(textField1.getText());
    textField1.setText("");
}
```

The `parseDouble` method converts a given string value to double value.

18) Add the following code to `btnPoint`'s action function.

```
btnPoint.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if(textField1.getText().equals(""))
        {
            textField1.setText("0.");
        }
        else if(textField1.getText().contains("."))
        {
            btnPoint.setEnabled(false);
        }
        else
        {
            String btnPointText = textField1.getText() + btnPoint.getText();
            textField1.setText(btnPointText);
        }
        btnPoint.setEnabled(true);
    }
});
```

The if conditions ensure that no incorrect input (e.g. ..2..5) is given and input is properly formatted.

19) Add the following code to `btnPlus` `actionPerformed` method.

```
btnPlus.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        get_operator(btnPlus.getText());
    }
});
```

Copy the same code for `btnMinus`, `btnDivide` and `btnMultiply` (but make sure to change the button name).

20) Add the following code to btnEquals actionPerformed method. Pressing the btnEquals should produce the result of our arithmetic operations. It also checks which operator is being used.

```
btnEquals.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        switch(operator)  
        {  
            case '+':  
                total2 = total1 + Double.parseDouble(textField1.getText());  
                break;  
            case '-':  
                total2 = total1 - Double.parseDouble(textField1.getText());  
                break;  
            case '/':  
                total2 = total1 / Double.parseDouble(textField1.getText());  
                break;  
            case 'x':  
                total2 = total1 * Double.parseDouble(textField1.getText());  
                break;  
        }  
  
        textField1.setText(Double.toString(total2));  
        total1 = 0.0;  
    }  
});
```

21) Add the following code to btnClear actionPerformed method. This code will allow us to clear the input display area of our form (textField1 in example) and resets it.

```
btnClear.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        total2 = 0.0;  
        textField1.setText("");  
    }  
});
```

22) Run your application and test it.

# Lab Tasks

1. Modify the given calculator GUI application to include exponentiation (x POWER y).
2. Create a GUI based application to take a string value as input and contain buttons to allow users to perform the following operations. (Note: You can display the output using JLabel).
  - i) Finding the no. of most frequent character
  - ii) Reverse the input string
  - iii) Search for a given word in the input string
3. Create the following layout using GUI elements available in Java. (Note: No backend code is required, only front end/design is to be created).

Fee Report

Name of the Student:

Name of the Father:

Roll Number:

Email ID:

Contact Number:

Address:

Gender:

☐ Male

☐ Female

Nationality:

Year of passing 10th

2016

Year of passing 12th

2019

Points Secured in 10th:

☐ SEAS

☐ SLABS

☐ HOSTLER

☐ DAY SCHOLAR

Groups Offered here are:

CSE

CSE(2,50,000)

ECE(2,50,000)

EEE(2,50,000)

MECH(2,50,000)

CIVIL(2,50,000)

2 SHARE(1,50,000)

3 SHARE(1,40,000)

5 SHARE(1,20,000)

8 SHARE(1,10,000)

bus(40,000)

Show

Generate Receipt

Reset

Print