UMS
UNIVERSITI MALAYSIA SABAH

KP14603 OBJECT ORIENTED PROGRAMMING

SEMESTER II

SESSION 2019/2020

INDIVIDUAL PROJECT REPORT

BASIC CALCULATOR

LECTURER NAME: MADAM SITI HASNAH TANALOL

NAME	MATRIC NUMBER
NURUL AMMAR HANI BINTI AG MARTEN	BI19160333

Table of Contents

INTRODUCTION

Calculator is a device that performs arithmetic operation on numbers. In this project I build a simple calculator that can do only addition, subtraction, multiplication, and division. I've choose to make this kind of project because to improve my understanding the basic function of GUI and to understand object oriented programming.

The main of this project is to provide better user friendly calculator. It is very easy to use with a simple layout to make user didn't feel stress while using the calculator.

OBJECTIVES

1. to perform a number of calculations in response to user supplied input.

2. To make sure that all keys are correctly performing operation on the screen.

3. The databases used by calculators may be very simple.

JAVA CODE

```
//NAME : NURUL AMMAR HANI BINTI AG MARTEN

//NO MATRIC : BI19160333

//PROJECT TITLE : BASIC CALCULATOR


package basiccalculator;


import javax.swing.*;

import java.awt.event.*;

import java.awt.event.ActionListener;


class basiccalculator implements ActionListener

{

        JFrame f;

        JTextField t;

        JButton
b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,bdiv,bmul,bsub,badd,bdec,beq,bdel,bclr;


        static double a=0,b=0,result=0;

        static int operator=0;


        basiccalculator()

        {

                f=new JFrame("Calculator");

                t=new JTextField();
```

```
b1=new JButton("1");
b2=new JButton("2");
b3=new JButton("3");
b4=new JButton("4");
b5=new JButton("5");
b6=new JButton("6");
b7=new JButton("7");
b8=new JButton("8");
b9=new JButton("9");
b0=new JButton("0");
bdiv=new JButton("/");
bmul=new JButton("*");
bsub=new JButton("-");
badd=new JButton("+");
bdec=new JButton(".");
beq=new JButton("=");
bdel=new JButton("Delete");
bclr=new JButton("Clear");

t.setBounds(30,40,280,30);
b7.setBounds(40,100,50,40);
b8.setBounds(110,100,50,40);
b9.setBounds(180,100,50,40);
bdiv.setBounds(250,100,50,40);
```

```
b4.setBounds(40,170,50,40);
b5.setBounds(110,170,50,40);
b6.setBounds(180,170,50,40);
bmul.setBounds(250,170,50,40);

b1.setBounds(40,240,50,40);
b2.setBounds(110,240,50,40);
b3.setBounds(180,240,50,40);
bsub.setBounds(250,240,50,40);

bdec.setBounds(40,310,50,40);
b0.setBounds(110,310,50,40);
beq.setBounds(180,310,50,40);
badd.setBounds(250,310,50,40);

bdel.setBounds(60,380,100,40);
bclr.setBounds(180,380,100,40);

f.add(t);
f.add(b7);
f.add(b8);
f.add(b9);
f.add(bdiv);
```

```
f.add(b4);
f.add(b5);
f.add(b6);
f.add(bmul);
f.add(b1);
f.add(b2);
f.add(b3);
f.add(bsub);
f.add(bdec);
f.add(b0);
f.add(beq);
f.add(badd);
f.add(bdel);
f.add(bclr);

f.setLayout(null);
f.setVisible(true);
f.setSize(350,500);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
f.setResizable(false);

b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
```

```
        b4.addActionListener(this);
        b5.addActionListener(this);
        b6.addActionListener(this);
        b7.addActionListener(this);
        b8.addActionListener(this);
        b9.addActionListener(this);
        b0.addActionListener(this);
        badd.addActionListener(this);
        bdiv.addActionListener(this);
        bmul.addActionListener(this);
        bsub.addActionListener(this);
        bdec.addActionListener(this);
        beq.addActionListener(this);
        bdel.addActionListener(this);
        bclr.addActionListener(this);
}

public void actionPerformed(ActionEvent e)
{
        if(e.getSource()==b1)
                t.setText(t.getText().concat("1"));

        if(e.getSource()==b2)
                t.setText(t.getText().concat("2"));
```

```
if(e.getSource()==b3)
        t.setText(t.getText().concat("3"));

if(e.getSource()==b4)
        t.setText(t.getText().concat("4"));

if(e.getSource()==b5)
        t.setText(t.getText().concat("5"));

if(e.getSource()==b6)
        t.setText(t.getText().concat("6"));

if(e.getSource()==b7)
        t.setText(t.getText().concat("7"));

if(e.getSource()==b8)
        t.setText(t.getText().concat("8"));

if(e.getSource()==b9)
        t.setText(t.getText().concat("9"));

if(e.getSource()==b0)
        t.setText(t.getText().concat("0"));
```

```
if(e.getSource()==bdec)
        t.setText(t.getText().concat("."));

if(e.getSource()==badd)
{
        a=Double.parseDouble(t.getText());
        operator=1;
        t.setText("");
}

if(e.getSource()==bsub)
{
        a=Double.parseDouble(t.getText());
        operator=2;
        t.setText("");
}

if(e.getSource()==bmul)
{
        a=Double.parseDouble(t.getText());
        operator=3;
        t.setText("");
}
```

```
if(e.getSource()==bdiv)
{
        a=Double.parseDouble(t.getText());
        operator=4;
        t.setText("");
}

if(e.getSource()==beq)
{
        b=Double.parseDouble(t.getText());

        switch(operator)
        {
                case 1: result=a+b;
                        break;

                case 2: result=a-b;
                        break;

                case 3: result=a*b;
                        break;

                case 4: result=a/b;
```

```
                                        break;

                                default: result=0;
                        }

                        t.setText(""+result);
                }

                if(e.getSource()==bclr)
                        t.setText("");

                if(e.getSource()==bdel)
                {
                        String s=t.getText();
                        t.setText("");
                        for(int i=0;i<s.length()-1;i++)
                        t.setText(t.getText()+s.charAt(i));
                }
        }
        public static void main(String...s)
        {
                new basiccalculator();
        }
}
```

OBJECT ORIENTED CONCEPT IMPLEMENTATION

1. Encapsulation

Encapsulation allows us to protect the data stored in a class from system-wide access. As its name suggests, it safeguards the internal contents of a class like a real-life capsule.

1. public void actionPerformed

```
public void actionPerformed(ActionEvent e)
{
        if(e.getSource()==b1)
                t.setText(t.getText().concat("1"));

        if(e.getSource()==b2)
                t.setText(t.getText().concat("2"));
```

2. public static void main (string…s)

```
}

public static void main(String...s)
{
        new basiccalculator();
}
```

In above they all the data fields are private. which cannot be accessed directly. These fields can be accessed via public methods only. Fields are made hidden data fields using encapsulation technique of OOPs.

2. Object & Classes

A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type.

1. class basiccalculator

```
class basiccalculator implements ActionListener
{
        JFrame f;
        JTextField t;
        JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,bdiv,bmul,bsub,badd,bdec,beq,bdel,bclr;
```

3. Interface

Inside any implementation class, cannot change the variables declared in interface because by default, they are public, static and final. If there are two or more same methods in two interfaces and a class implements both interfaces, implementation of the method once is enough.

1. class basiccalculator implements ActionListener

```
package basiccalculator;

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

class basiccalculator implements ActionListener
{
        JFrame f;
        JTextField t;
        JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,bdiv,bmul,bsub,badd,bdec,beq,bdel,bclr;
```

4. Inner class

In the above program, there are two nested classes. Processor and RAM inside the outer class CPU. We can declare the inner class as protected. Hence, we have declared the RAM class as protected. Inside the Main class, we first created an instance of an outer class basiccalculator named basiccalculator. Using the instance of the outer class, then created objects of inner classes. use the dot (.) operator to create an instance of the inner class using the outer class.

1. new basiccalculator

```
        }

        public static void main(String...s)
        {
            new basiccalculator();
        }
}
```

5. Abstraction

Abstraction means using simple things to represent complexity. In Java, abstraction means simple things like objects, classes, and variables represent more complex underlying code and data. This is important because it lets avoid repeating the same work multiple times.

```
JFrame f;
JTextField t;
JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,bdiv,bmul,bsub,badd,bdec,beq,bdel,bclr

static double a=0,b=0,result=0;
static int operator=0;
```

READ AND WRITE IMPLEMENTATION

In this project I use JTextField. JTextField is a lightweight component that allows the editing of a single line of text. JTextField is intended to be source-compatible with java.awt.TextField where it is reasonable to do so. This component has capabilities not found in the java.awt.TextField class. The superclass should be consulted for additional capabilities.

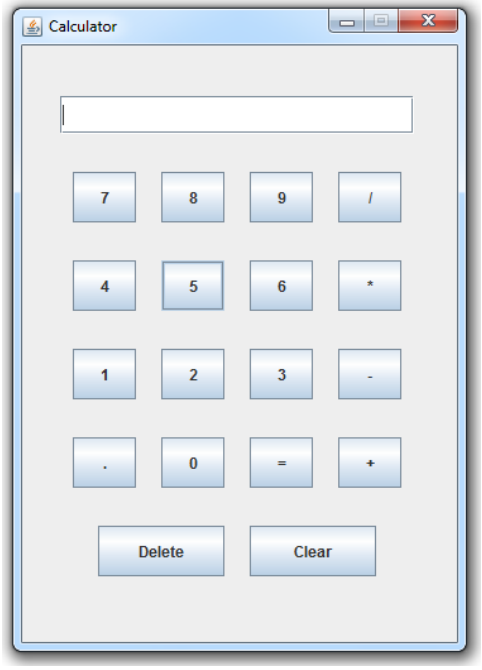
JTextField has a method to establish the string used as the command string for the action event that gets fired. The java.awt.TextField used the text of the field as the command string for the ActionEvent. JTextField will use the command string set with the setActionCommand method if not null, otherwise it will use the text of the field as a compatibility with java.awt.TextField.

```
//NAME : NURUL AMMAR HANI BINTI AG MARTEN
//NO MATRIC : BI19160333
//PROJECT TITLE : BASIC CALCULATOR

package basiccalculator;

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

class basiccalculator implements ActionListener
{
        JFrame f;
        JTextField t;
        JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,bdiv,bmul,bsub,badd,bdec,beq,bdel,bclr;

        static double a=0,b=0,result=0;
        static int operator=0;
```

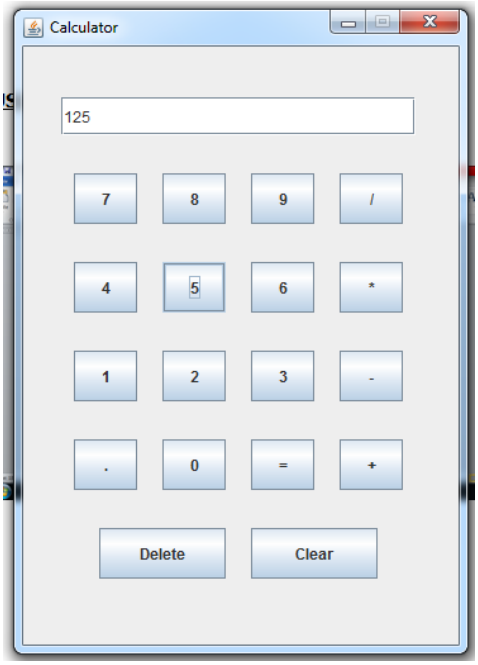
USER MANUAL

Basic calculator project is calculator with a basic operation. For example addition, subtraction, divided and multiplication.

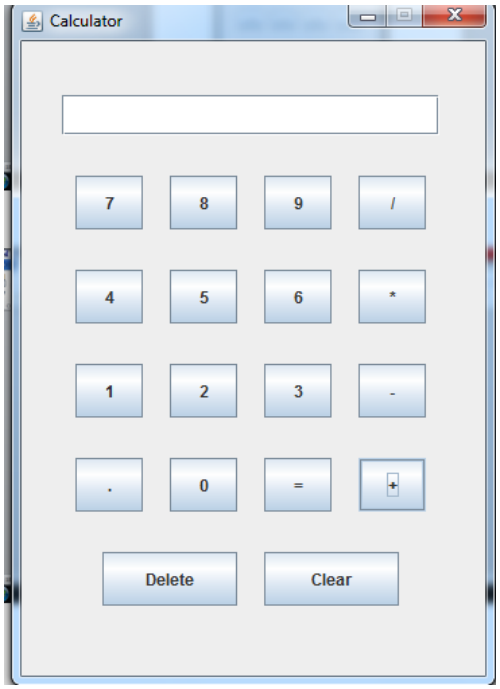
First step, run the project you will see this GUI.



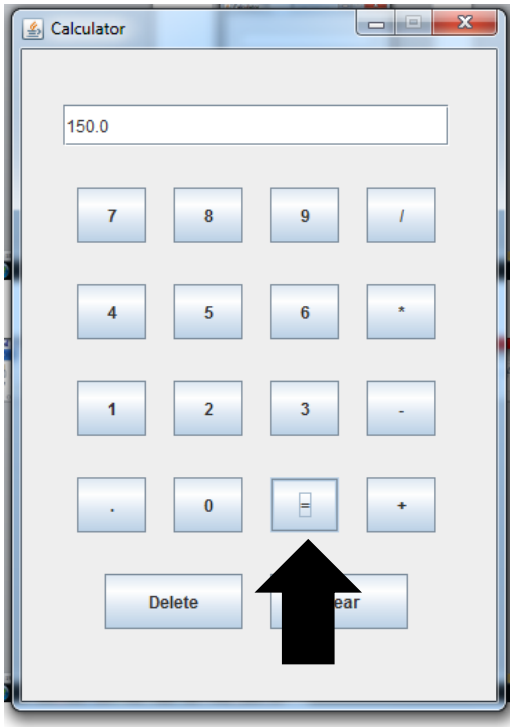
Second, just click the number on the screen.



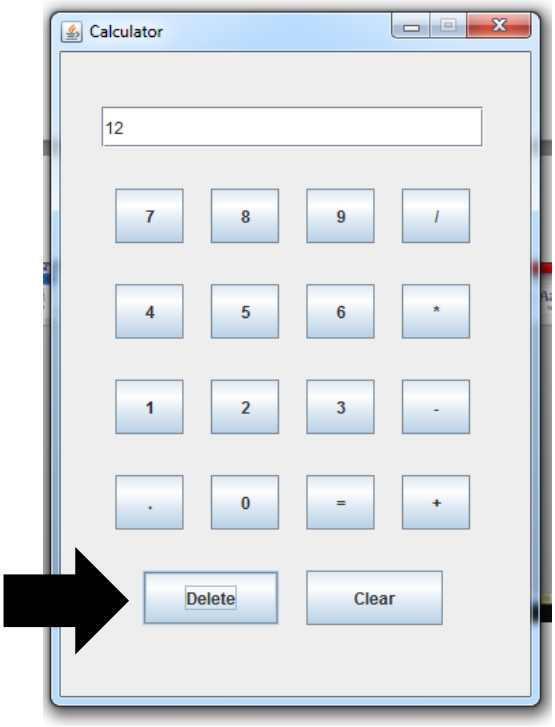
Third, click the operation button.



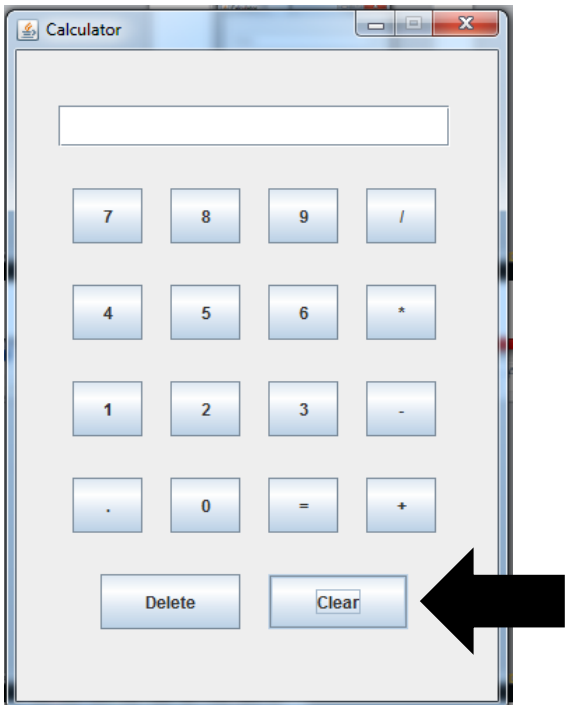
Fourth, just click number and click equals (=) button to get the result.



Fifth, if you want to delete number just click "Delete" button.



Last, just click "Clear" button to finisih it.



CONCLUSION

In conclusion, this is very simple calculator app. The important thing is it help user use this calculator easily. Last but not least, I will keep improve this calculator system if there have any issues from other user.