**Tutorial No. 4**

**Problem statement:**

Extend Calculator program to implement MVC architecture. Design JFrame to create view, use basic calculator as model and create appropriate controller.

**Design Assumptions:**

Model–view–controller (MVC) is an [architectural pattern](https://en.wikipedia.org/wiki/Architectural_pattern) commonly used for developing [user interfaces](https://en.wikipedia.org/wiki/User_interface) that divides an application into three interconnected parts. This is done to separate internal representations of information from the ways information is presented to and accepted from the user. The MVC design pattern decouples these major components allowing for efficient [code reuse](https://en.wikipedia.org/wiki/Code_reuse) and parallel development.

Traditionally used for desktop [graphical user interfaces](https://en.wikipedia.org/wiki/Graphical_user_interface) (GUIs), this architecture has become popular for designing [web applications](https://en.wikipedia.org/wiki/Web_application) and even mobile, desktop and other clients. Popular programming languages like [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)), [PHP](https://en.wikipedia.org/wiki/PHP) and others have popular MVC frameworks that are currently being used in web application development straight [out of the box](https://en.wikipedia.org/wiki/Out_of_the_box_(feature)).

**Components**

The model is the central component of the pattern. It expresses the application's behavior in terms of the [problem domain](https://en.wikipedia.org/wiki/Problem_domain), independent of the user interface. It directly manages the data, logic and rules of the application.

* A view can be any output representation of information, such as a chart or a diagram. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.
* The third part or section, the controller, accepts input and converts it to commands for the model or view.

**Interactions**

In addition to dividing the application into three kinds of components, the model–view–controller design defines the interactions between them.

* The model is responsible for managing the data of the application. It receives user input from the controller.
* The view means presentation of the model in a particular format.
* The controller is responsible for responding to the user input and perform interactions on the data model objects. The controller receives the input, optionally validates the input and then passes the input to the model.

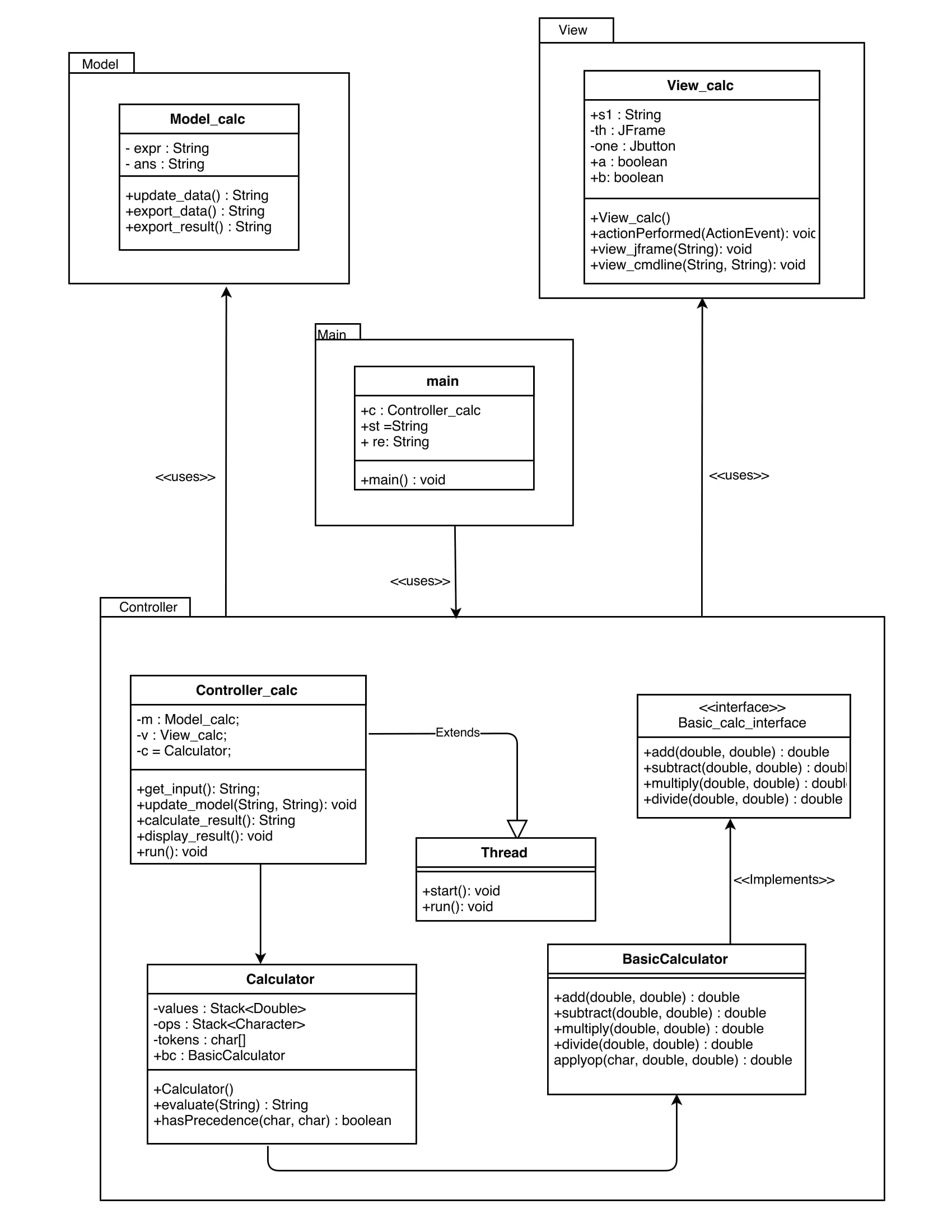
**Advantages**

* Simultaneous development – Multiple developers can work simultaneously on the model, controller and views.
* [High cohesion](https://en.wikipedia.org/wiki/Cohesion_(computer_science)) – MVC enables logical grouping of related actions on a controller together. The views for a specific model are also grouped together.
* [Low coupling](https://en.wikipedia.org/wiki/Loose_coupling) – The very nature of the MVC framework is such that there is low coupling among models, views or controllers
* Ease of modification – Because of the separation of responsibilities, future development or modification is easier
* Multiple views for a model – Models can have multiple views

**Disadvantages**

* Code navigability – The framework navigation can be complex because it introduces new layers of abstraction and requires users to adapt to the decomposition criteria of MVC.
* Multi-artifact consistency – Decomposing a feature into three artifacts causes scattering. Thus, requiring developers to maintain the consistency of multiple representations at once.
* Pronounced learning curve – Knowledge on multiple technologies becomes the norm. Developers using MVC need to be skilled in multiple technologies.

**Design Diagrams:**



**Code:**

**Controller\_calc.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Controller;

import Model.Model\_calc;

import View.View\_calc;

import calculator.\*;

import java.util.Scanner;

/\*

\* @author Jatin

\*/

public class Controller\_calc extends Thread {

private Model\_calc m;

private View\_calc v;

private Calculator c;

private volatile String c\_input = "";

private Scanner scn = new Scanner(System.in);

public Controller\_calc(Model\_calc m, View\_calc v){

this.m = m;

this.v = v;

c = new Calculator();

}

public String get\_input(){

while(v.a!=true)

{

if(!"".equals(c\_input)){

String to\_send = c\_input;

c\_input="";

return to\_send;

}

}

if(v.a==true){

v.a = false;

return v.s1;

}

return null; //will never execute

}

public void update\_model(String st, String ans){

m.update\_data(st,ans);

}

public String calculate\_result(){

return c.evaluate(m.export\_data());

}

public void display\_result(){

v.view\_jframe(m.export\_result());

v.view\_cmdline(m.export\_data(),m.export\_result());

}

@Override

public void run() {

c\_input = scn.next();

v.b=true;

}

}

**main.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Main;

import Model.Model\_calc;

import View.View\_calc;

import Controller.Controller\_calc;

/\*\*

\*

\* @author Jatin

\*/

public class Main {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

Controller\_calc c = new Controller\_calc(new Model\_calc(), new View\_calc());

c.start();

String st ="";

st = c.get\_input();

while(true){

String re = "0";

c.update\_model(st, re);

re = c.calculate\_result();

c.update\_model(st, re);

c.display\_result();

st = c.get\_input();

}

}

}

**Model\_calc.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Model;

/\*\*

\*

\* @author Jatin

\*/

public class Model\_calc {

private static String expr;

private static String ans;

public void update\_data(String st, String result){

expr = st;

ans = result;

}

public String export\_data(){

return expr;

}

public String export\_result(){

return ans;

}

}

**View\_calc.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package View;

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

/\*\*

\*

\* @author Jatin

\*/

public class View\_calc implements ActionListener{

JTextField t;

JButton dec;

JButton equal;

JButton zero;

JButton one;

JButton two;

JButton three;

JButton four;

JButton five;

JButton six;

JButton seven;

JButton eight;

JButton nine;

JButton plus;

JButton minus;

JButton multiply;

JButton divide;

JButton clear;

private JFrame th;

public boolean a=false;

public boolean b=true;

public String s1;

public View\_calc() {

th = new JFrame("Basic Math calculator");

t = new JTextField();

t.setBounds(45,45,245,40);

one = new JButton("1");

one.setBounds(45,110,50,50);

four = new JButton("4");

four.setBounds(45,175,50,50);

seven = new JButton("7");

seven.setBounds(45,240,50,50);

dec = new JButton(".");

dec.setBounds(45,305,50,50);

two = new JButton("2");

two.setBounds(110,110,50,50);

five = new JButton("5");

five.setBounds(110,175,50,50);

eight = new JButton("8");

eight.setBounds(110,240,50,50);

zero = new JButton("0");

zero.setBounds(110,305,50,50);

three = new JButton("3");

three.setBounds(175,110,50,50);

six = new JButton("6");

six.setBounds(175,175,50,50);

nine = new JButton("9");

nine.setBounds(175,240,50,50);

equal = new JButton("=");

equal.setBounds(175,305,50,50);

plus = new JButton("+");

plus.setBounds(240,110,50,50);

minus = new JButton("-");

minus.setBounds(240,175,50,50);

multiply = new JButton("\*");

multiply.setBounds(240,240,50,50);

divide = new JButton("/");

divide.setBounds(240,305,50,50);

clear = new JButton("clear");

clear.setBounds(110,370,100,50);

th.setSize(350, 470);

th.add(one);

th.add(two);

th.add(three);

th.add(four);

th.add(five);

th.add(six);

th.add(seven);

th.add(eight);

th.add(nine);

th.add(dec);

th.add(zero);

th.add(equal);

th.add(plus);

th.add(multiply);

th.add(divide);

th.add(minus);

th.add(t);

th.add(clear);

th.setLayout(null);

th.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

th.setVisible(true);

one.addActionListener(this);

two.addActionListener(this);

three.addActionListener(this);

four.addActionListener(this);

five.addActionListener(this);

six.addActionListener(this);

seven.addActionListener(this);

eight.addActionListener(this);

nine.addActionListener(this);

zero.addActionListener(this);

equal.addActionListener(this);

plus.addActionListener(this);

multiply.addActionListener(this);

divide.addActionListener(this);

minus.addActionListener(this);

dec.addActionListener(this);

clear.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

if(e.getSource()==one){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("1"));

}

if(e.getSource()==two){

{

if(this.b==true){

this.b=false;

t.setText("");

}t.setText(t.getText().concat("2"));

}

}

if(e.getSource()==three){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("3"));

}

if(e.getSource()==four){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("4"));

}

if(e.getSource()==five){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("5"));

}

if(e.getSource()==six){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("6"));

}

if(e.getSource()==seven){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("7"));

}

if(e.getSource()==eight){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("8"));

}

if(e.getSource()==nine){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("9"));

}

if(e.getSource()==zero){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("0"));

}

if(e.getSource()==dec){

if(this.b==true){

this.b=false;

t.setText("");

}

t.setText(t.getText().concat("."));

}

if(e.getSource()==plus)

t.setText(t.getText().concat("+"));

if(e.getSource()==multiply)

t.setText(t.getText().concat("\*"));

if(e.getSource()==divide)

t.setText(t.getText().concat("/"));

if(e.getSource()==minus)

t.setText(t.getText().concat("-"));

if(e.getSource()==clear)

{

t.setText("");

a=false;

}

if(e.getSource()==equal)

{

this.s1=t.getText();

if("".equals(s1))

{

JOptionPane.showMessageDialog(th, "Please enter expression");

}

else {

this.a=true;

this.b=true;

}

}

}

public void view\_jframe(String ans){

t.setText(ans);

}

public void view\_cmdline(String exp, String ans){

System.out.println(exp+" = "+ans);

}

}

**Output:**

**Observation:**