Java Swing MVC Example

Posted by: Ashraf Sarhan in Core Java January 26th, 2016

In this example we are going to demonstrate Java Swing MVC, The MVC pattern is a model of how a user interface can be structured. Therefore it defines the following 3 elements:

- *Model that represents the data for the application.*
- *View that is the visual representation of that data.*
- *Controller that takes user input on the view and translates that to changes in the model.*

1. MVC Components

1.1. Model

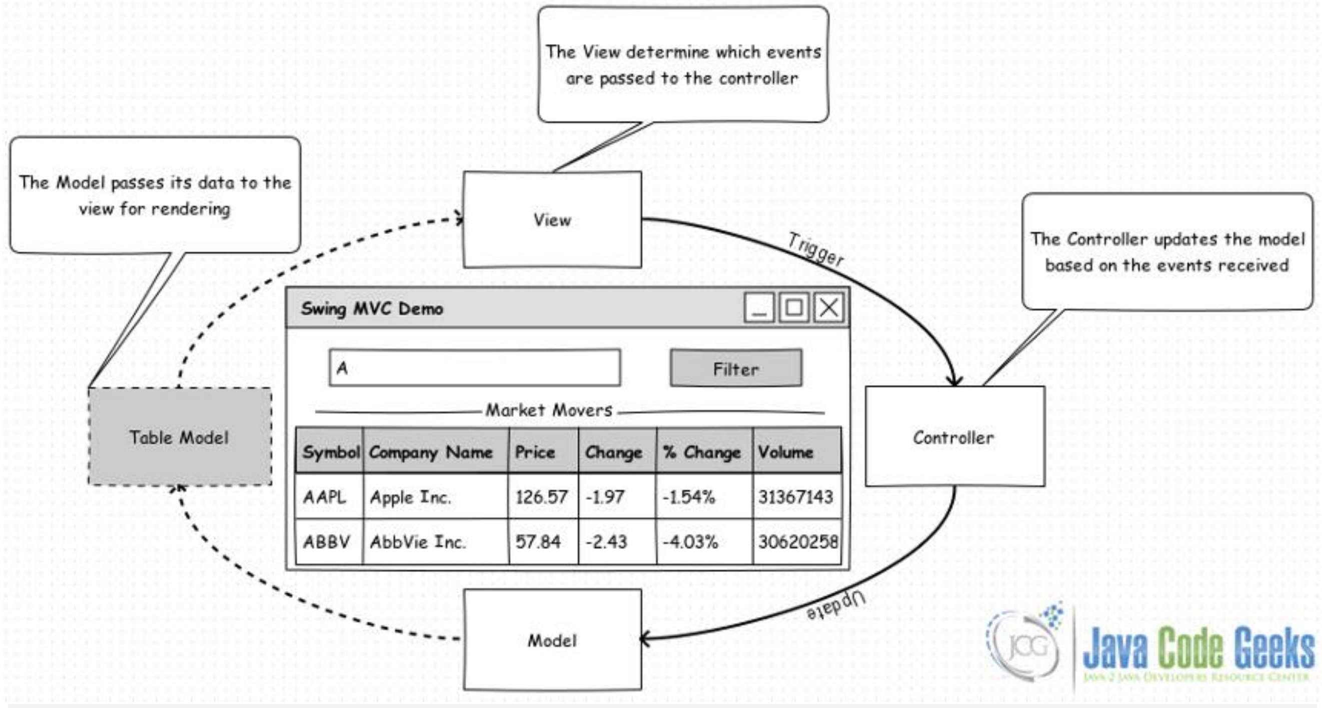
A model is an abstraction of something that is presented to the user. The models provided by Swing fall into two general categories: *GUI-state models* and *application-data models*. *GUI state models* are interfaces that define the visual status of a GUI control, such as whether a button is pressed or armed like ButtonModel. An *application-data model* is an interface that represents some quantifiable data that the UI presents to the user, such as the value of a cell in a table like TableModel.

1.2. View

The view is a UI component that is responsible for presenting data to the user. Thus it is responsible for all UI dependent issues like layout, drawing, etc. JTable is a good example for the view.

1.3. Controller

A controller encapsulates the application code that is executed in order to an user interaction (mouse motion, mouse click, key press, etc.). Controllers might need input for their execution and they produce output. They read their input from models and update models as result of the execution. In swing a controller is normally implemented by an ActionListeneror Action.


The View determine which events are passed to the controller
The Model passes its data to the view for rendering
View
Trigger
The Controller updates the model based on the events received
Swing MVC Demo
A
Filter
Market Movers
Table Model
Controller
Update
Model
Java Code Geeks

Symbol	Company Name	Price	Change	% Change	Volume
AAPL	Apple Inc.	126.57	-1.97	-1.54%	31367143
ABBV	AbbVie Inc.	57.84	-2.43	-4.03%	30620258

Figure 1: Swing MVC Components

Now, lets see our concrete Swing MVC example where we have an application that let you filter stocks. The application UI contains a text field where the user can enter a filter string, button to start the filter and table where the filter results are displayed.

2. Swing MVC Example

2.1. Model

We create `Model.java` class which implements the TableModel interface (or, more likely, subclass the AbstractTableModelclass). The job of this TableModel implementation is to serve as the interface between your data and the JTable as a view component. Also, we add a supplementary `Constants.java` class contains constants used through our code.

Model.java:

```
package com.jcg;

import javax.swing.table.DefaultTableModel;

/**
 * @author ashraf
 *
 */
@SuppressWarnings("serial")
public class Model extends DefaultTableModel {

    public Model() {
        super(Constants.DATA, Constants.TABLE_HEADER);
    }

}
```

Constants.java:

```
package com.jcg;

/**
 * @author ashraf_sarhan
 *
 */
public class Constants {

    public static final Object[] TABLE_HEADER = { "Symbol", "Company Name",
            "Price", "Change", "% Change", "Volume" };

public static final Object[][] DATA = {
        { "BAC", "Bank of America Corporation", 15.98, 0.14, "+0.88%",
                32157250 },
        { "AAPL", "Apple Inc.", 126.57, -1.97, "-1.54%", 31367143 },
        { "ABBV", "AbbVie Inc.", 57.84, -2.43, "-4.03%", 30620258 },
        { "ECA", "Encana Corporation", 11.74, -0.53, "-4.33%", 27317436 },
        { "VALE", "Vale S.A.", 6.55, -0.33, "-4.80%", 19764400 },
        { "FB", "Facebook, Inc.", 81.53, 0.64, "+0.78%", 16909729 },
        { "PBR", "Petróleo Brasileiro S.A. - Petrobras", 6.05, -0.12,
                "-2.02%", 16181759 },
        { "NOK", "Nokia Corporation", 8.06, 0.01, "+0.12%", 13611860 },
        { "PCYC", "Pharmacyclics Inc.", 254.67, 24.19, "+10.50%", 13737834 },
        { "RAD", "Rite Aid Corporation", 7.87, -0.18, "-2.24%", 13606253 } };

}
```

2.2. View

We create View.java class which contains our main UI components, a JTextField where the user can enter a filter string, JButton to start the filter and JTable where the filter results are displayed.

View.java:

```
package com.jcg;

import java.awt.Dimension;

import javax.swing.BorderFactory;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JSplitPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.border.TitledBorder;

/**
 * @author ashraf
 *
 */
public class View {

    public View() {
        // Create views swing UI components
        JTextField searchTermTextField = new JTextField(26);
        JButton filterButton = new JButton("Filter");
        JTable table = new JTable();

        // Create table model
        Model model = new Model();
        table.setModel(model);

        // Create controller
        Controller controller = new Controller(searchTermTextField, model);
        filterButton.addActionListener(controller);

        // Set the view layout
        JPanel ctrlPane = new JPanel();
        ctrlPane.add(searchTermTextField);
        ctrlPane.add(filterButton);

        JScrollPane tableScrollPane = new JScrollPane(table);
        tableScrollPane.setPreferredSize(new Dimension(700, 182));
        tableScrollPane.setBorder(BorderFactory.createTitledBorder(BorderFactory.createEtchedBorder(),"Market Movers",
                TitledBorder.CENTER, TitledBorder.TOP));

        JSplitPane splitPane = new JSplitPane(JSplitPane.VERTICAL_SPLIT, ctrlPane, tableScrollPane);
        splitPane.setDividerLocation(35);
        splitPane.setEnabled(false);

        // Display it all in a scrolling window and make the window appear
        JFrame frame = new JFrame("Swing MVC Demo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.add(splitPane);
        frame.pack();
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
    }

}
```

2.3. Controller

We create `Controller.java` class which implements the ActionListener interface, it will be invoked as a result of a user's action on a view (i.e., it will be invoked because of the filter button click).

Controller.java:

```
package com.jcg;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JOptionPane;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

/**
 * @author ashraf
 *
 */
public class Controller implements ActionListener {

    private JTextField searchTermTextField = new JTextField(26);
    private DefaultTableModel model;

    public Controller(JTextField searchTermTextField, DefaultTableModel model) {
        super();
        this.searchTermTextField = searchTermTextField;
        this.model = model;
    }

    @Override
    public void actionPerformed(ActionEvent e) {

        String searchTerm = searchTermTextField.getText();
        if (searchTerm != null && !"".equals(searchTerm)) {
            Object[][] newData = new Object[Constants.DATA.length][];
            int idx = 0;
            for (Object[] o: Constants.DATA) {
                if ("*".equals(searchTerm.trim())) {
                    newData[idx++] = o;
                } else {
                    if(String.valueOf(o[0]).startsWith(searchTerm.toUpperCase().trim())){
                        newData[idx++] = o;
                    }
                }
            }
            model.setDataVector(newData, Constants.TABLE_HEADER);
        } else {
            JOptionPane.showMessageDialog(null,
                    "Search term is empty", "Error",
                    JOptionPane.ERROR_MESSAGE);
        }
    }

}
```

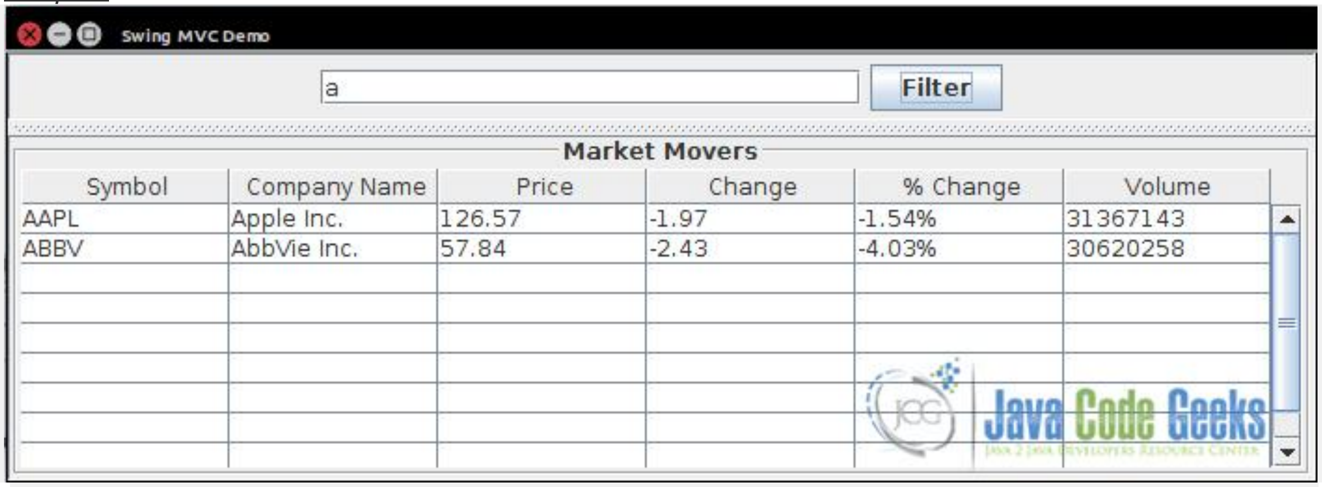
2.4. Running the Swing MVC Example

We create SwingMVCDemo.java class which serve as main class to running our example.

SwingMVCDemo.java:

```
package com.jcg;

import javax.swing.SwingUtilities;

/**
 * @author ashraf_sarhan
 *
 */
public class SwingMVCDemo {

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                try {
                    createAndShowGUI();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    public static void createAndShowGUI() throws Exception {
        new View();
    }
}
```

Output:



Symbol	Company Name	Price	Change	% Change	Volume
AAPL	Apple Inc.	126.57	-1.97	-1.54%	31367143
ABBV	AbbVie Inc.	57.84	-2.43	-4.03%	30620258

Figure 2: Swing MVC Demo