# Concrete Code Example of MVP [closed]

Ask Question

Can someone provide a concrete (actual Java code) example of MVP in action?

This would include the following 3 types of classes and how they call each other's methods to achieve the pattern and process/respond to a client-side response:

- **Model** - some kind of value object (VO)
- **View** - represents or generates the UI
- **Presenters** - business logic

java design-patterns mvp

edited May 16 '17 at 9:27
Willi Mentzel
6,084 11 32 58

asked Jul 6 '12 at 17:49
IAmYourFaja
11.6k 123 362 634

**closed** as not constructive by Jim Garrison, kosa, Esko, Hovercraft Full Of Eels, Evan Mulawski Jul 7 '12 at 0:45

As it currently stands, this question is not a good fit for our Q&A format. We expect answers to be supported by facts, references, or expertise, but this question will likely solicit debate, arguments, polling, or extended discussion. If you feel that this question can be improved and possibly reopened, visit the help center for guidance.

If this question can be reworded to fit the rules in the help center, please edit the question.

2 SO doesn't work this way; it is not a discussion or general tutorial forum. Please read the FAQ and How to Ask. Also, your question can be easily answered with a Google search. – Jim Garrison Jul 6 '12 at 17:59

Side stepping here but is MVP yet another name for MVC? – Esko Jul 6 '12 at 18:03

1 @Esko There are differences: blogs.infragistics.com/blogs/todd_snyder/archive/2007/10/17/… – Konrad Reiche Jul 6 '12 at 18:09

1 @Jim - this is a specific question (believe it or not)!! Guess how many concrete, Java-specific code samples of MVP turn up when you do a Google search for it? None! – IAmYourFaja Jul 6 '12 at 18:28

So why not study a *non*-Java solution and then utilize the ideas contained to create your own Java solution? This is mainly an issue of you doing your own grunt work. I don't see the issue here, and I second Jim that this is not really a proper SO question. – Hovercraft Full Of Eels Jul 6 '12 at 21:27

It is clear to me that Java is not capable of supporting MVP frameworks. This is a simple question that, if answerable, would have been answered already. I will explore other architecture solutions. Thanks to everyone that tried to help. – IAmYourFaja Jul 6 '12 at 21:31

Just because you are unwilling or unable to do your own grunt work doesn't mean that Java is incapable. Nice try though. – Hovercraft Full Of Eels Jul 6 '12 at 22:22

Dr. Hovercraft - please see @danLeon's most excellent answer to my question. It will help you implement MVP in the future if you wish to use it :-) – IAmYourFaja Jul 7 '12 at 2:58

@platzhirsch Quickly skimmed through that (will read through it with more thought) but I just have to point out that his MVC solution is incorrect, Controller should have actions such as "get customer's orders" while "load|save customer" is a responsibility of the Model. Maybe it's just my head but for me MVP looks like MVVM which really is just another variation of MVC...gah, I hope someone would make a broad comparison of all these MVwhatevers, just like with anyDD:s. – Esko Jul 7 '12 at 5:25

2 @Esko - I'm no expert but have been out of school for about 3 years now and I'm beginning to realize that a lot of "theoretical" stuff we all learned in our Software Engineering I class is just that...*theoretical*! These PhD's are smart and are great (in fact, its how they make their living!) at pitching 3-letter acronymed patterns in their white papers, but as we see with MVC, MVP, MVVM and who knows what else...when you lift the veil, its all just "Emperor's New Clothes." I bet you MVC/MVP/MVVM all first appear in some PhD's doctoral dissertation...they sound great, but form over substance. – IAmYourFaja Jul 7 '12 at 10:58

Somewhat relevant to the metadiscussion, I just heard the best answer to the question "What is MVC?" - "Marvel Versus Capcom." – Esko Jul 14 '12 at 15:26

## 1 Answer

MVP is my favorite design pattern to create a UI.

The big difference between MVP and MVC is how to handle the view.

- In MVC, the Controller manipulates the view, taking care of how to render in the view parts of the user actions and model. That means that the Controller and View have 'tight coupling'.
- In MVP, the Presenter takes care of the user tasks, the model is shared between the Presenter and View. So the view renders UI according to the model, sometimes the view can have actions to be called from the Presenter. The Presenter and View can have a defined interface contracts to make them 'loose coupling'. For example you can create a View for Java Swing UI and another for JavaFX UI. Or if the connection to the data source changes, then you just need update the

presenter.

There are many styles to program the MVP.

In a formal way, consists in create interfaces for each element of the design pattern.

```
/*-- file: Application.java --*/
import javax.swing.JOptionPane;

/**
 *
 * @author danLeon
 */
interface LoginModel {

    String getUser();

    void setUser(String user);
}

class MyLoginModel implements LoginModel {

    String user;

    @Override
    public String getUser() {
        return user;
    }

    @Override
    public void setUser(String user) {
        this.user = user;
    }
}

interface LoginView {

    LoginPresenter getPresenter();

    void setPresenter(LoginPresenter loginPresenter);

    void updateModelFromView();

    void updateViewFromModel();

    void open();

    void close();

    void userRejected();
}

class MyLoginView extends javax.swing.JFrame implements LoginView {

    private LoginPresenter loginPresenter;

    /**
     * Creates new form MyLoginView
     */
    public MyLoginView() {
        initComponents();
    }


    @SuppressWarnings("unchecked")
    private void initComponents() {
        java.awt.GridBagConstraints gridBagConstraints;

        jLabel1 = new javax.swing.JLabel();
        jTextField1 = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        jButton1 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setBounds(new java.awt.Rectangle(0, 0, 0, 0));
        java.awt.GridBagLayout layout = new java.awt.GridBagLayout();
        layout.columnWidths = new int[] {0, 7, 0};
        layout.rowHeights = new int[] {0, 7, 0, 7, 0};
        getContentPane().setLayout(layout);

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N
        jLabel1.setText("Welcome");
        gridBagConstraints = new java.awt.GridBagConstraints();
        gridBagConstraints.gridx = 0;
        gridBagConstraints.gridy = 0;
        gridBagConstraints.gridwidth = 3;
        getContentPane().add(jLabel1, gridBagConstraints);

        jTextField1.setColumns(13);
        gridBagConstraints = new java.awt.GridBagConstraints();
        gridBagConstraints.gridx = 2;
        gridBagConstraints.gridy = 2;
        gridBagConstraints.fill = java.awt.GridBagConstraints.HORIZONTAL;
        getContentPane().add(jTextField1, gridBagConstraints);
```

```
            jLabel2.setText("User");
            gridBagConstraints = new java.awt.GridBagConstraints();
            gridBagConstraints.gridx = 0;
            gridBagConstraints.gridy = 2;
            getContentPane().add(jLabel2, gridBagConstraints);

            jButton1.setText("Login");
            jButton1.addActionListener(new java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt) {
                    jButton1ActionPerformed(evt);
                }
            });
            gridBagConstraints = new java.awt.GridBagConstraints();
            gridBagConstraints.gridx = 2;
            gridBagConstraints.gridy = 4;
            getContentPane().add(jButton1, gridBagConstraints);

            pack();
        }

        private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
            getPresenter().login();
        }

        private javax.swing.JButton jButton1;
        private javax.swing.JLabel jLabel1;
        private javax.swing.JLabel jLabel2;
        private javax.swing.JTextField jTextField1;


        @Override
        public void updateModelFromView() {
            getPresenter().getModel().setUser(jTextField1.getText());
        }

        @Override
        public void updateViewFromModel() {
            jTextField1.setText(getPresenter().getModel().getUser());
        }

        @Override
        public void open() {
            setVisible(true);
            jTextField1.selectAll();
            jTextField1.requestFocus();
        }

        @Override
        public void close() {
            dispose();
        }

        @Override
        public void userRejected() {
            jLabel1.setText("Try again!");
            jTextField1.selectAll();
            jTextField1.requestFocus();
        }

        @Override
        public LoginPresenter getPresenter() {
            return loginPresenter;
        }

        @Override
        public void setPresenter(LoginPresenter loginPresenter) {
            this.loginPresenter = loginPresenter;
        }
    }

    interface LoginPresenter {

        LoginModel getModel();

        void setModel(LoginModel loginModel);

        LoginView getView();

        void setView(LoginView loginView);

        void setOnLogin(Runnable onLogin);

        void run();

        void login();
    }

    class MyLoginPresenter implements LoginPresenter {

        LoginModel loginModel;
        LoginView loginView;
        private Runnable onLogin;

        @Override
        public LoginModel getModel() {
            return loginModel;
```

```
    }

    @Override
    public void setModel(LoginModel loginModel) {
        this.loginModel = loginModel;
    }

    @Override
    public LoginView getView() {
        return loginView;
    }

    @Override
    public void setView(LoginView loginView) {
        this.loginView = loginView;
    }

    @Override
    public void setOnLogin(Runnable onLogin) {
        this.onLogin = onLogin;
    }

    @Override
    public void run() {
        loginModel.setUser("previousUser");
        loginView.setPresenter(this);
        loginView.updateViewFromModel();
        loginView.open();
    }

    @Override
    public void login() {
        loginView.updateModelFromView();
        if (loginModel.getUser().equalsIgnoreCase("root")) {
            loginView.close();
            loginView.setPresenter(null);// for memory sanity.
            onLogin.run();
        } else {
            loginView.userRejected();
        }
    }
}

public class Application {

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {
                LoginModel loginModel = new MyLoginModel();
                LoginPresenter loginPresenter = new MyLoginPresenter();
                loginPresenter.setModel(loginModel);
                LoginView loginView = new MyLoginView();
                loginPresenter.setView(loginView);
                loginPresenter.setOnLogin(new Runnable() {

                    @Override
                    public void run() {
                        JOptionPane.showMessageDialog(null, "Welcome user!");
                    }
                });
                loginPresenter.run();
            }
        });
    }
}
```

edited Mar 21 '14 at 11:34

fiz
**469** 7 24

answered Jul 6 '12 at 23:47

Daniel De León
**8,291** 3 55 50

3 *That means that the Controller and View have 'tight coupling'.* Thanks. This is one thing I was wondering about with MVC. Also nice of you to post an ever so clear Java example of MVP. – James P. Apr 26 '14 at 1:32

11 i would like to object: getPresenter().getModel().setUser(jTextField1.getText()); is not a line that should be present in a view. – cproinger Jun 22 '15 at 13:29

9 additionally updateModelFromView and updateViewFromModel are not methods one would expect on a view in MVP. the presenter should hold all the presentation-logik. the view should not be able to access the model in any way. – cproinger Jun 22 '15 at 13:31

Is acceptable use models into the View, like JDO – Daniel De León Jun 24 '15 at 17:11

2 I think you've swaped between MVC and MVP, take a look a this one : What are MVP and MVC and what is the difference? – La VloZ Merrill Jun 4 '17 at 21:59