Online Market Place Application

Assignment #1: Report

Implementation of MVC design pattern and Java RMI

Course: CSCI 50700 - Object-Oriented Design and Programming

By
Mani Manjusha Kottala
IUPUI, Computer Information Science

Table of Contents

Introduction:

The main job in the given assignment is to build a skeleton for the online market place application using the given requirements. This assignment also aims at creating the domain model based on the requirement analysis and identifying the classes necessary for application from the requirements. An application using (MVC) Model-View-Controller pattern is implemented in the assignment using Java RMI.

Java RMI:

Java Remote Method Invocation is based on the RPC protocol, it is used for building distributed applications in Java[1]. The main aim of RMI is to make the remote objects and remote calls behave local to the application. In the application developed for this assignment, MarketPlace is the remote interface and MarketPlaceController acts as the server application which implements the remote interface MarketPlace. MarketPlaceView acts as the client application where it uses the reference of the remote interface MarketPlace for remote calls. Remote methods of MarketPlaceController like login(), register(), getItems() can be accessed in the MarketPlaceView using that reference.

Model-View-Controller(MVC) Pattern:

Generally, in applications like online market place, there needs to be separation between the user interface, the back-end business logic and database. Model-View-Controller (MVC) pattern provides separation between user interface and the application functionality. Its main aim is to decouple the components into three interconnected parts- Model has the business rules and the application data, View has the user interface like the visual representation of the model and Controller acts as the link between View and Model. Its main principle is to never include the business logic in the UI applications.

In the application developed in assignment, the three components of the MVC pattern are:

- View: MarketPlaceView

  In the MarketPlaceView class, the User Interface of the application should be developed, this class has functions like enterLogin(), browseItems(), registration() and displaProfile() which interacts with the controller for getting therequested data from the Model.
- Model: MarketPlaceModel

  In MarketPlaceModel class, the business logic and the persistent storage should be implemented. Its method can be accessed using the controller and data can be passed to the controller for updations of the view
- Controller: MarketPlaceController

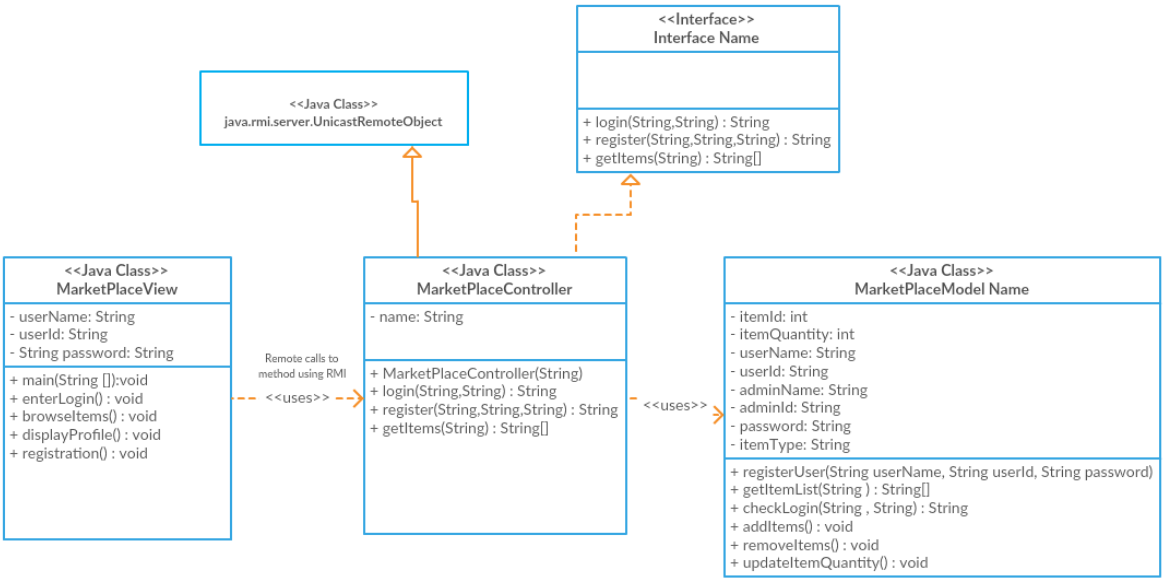
  In the MarketPlaceController class, acts as the interactor between controller and view. The Controller can interact with the Model for accessing its methods like getItems(), checkLogin(), addItems(), removeItems() etc and pass the information received from model to the client

Domain Model [3]:

As per the requirements, for implementing the MVC pattern using Java RMI the following classes and interfaces are required:

- Class - MarketPlaceView
- Class - MarketPlaceController
- Class - MarketPlaceModel

- Interface – MarketPlace
- Class – java.rmi.server.UnicastRemoteObject is extended in the MarketPlaceController for implementing Java RMI



Sample runs:

1. Starting the RMI registry and Running the server application

```
[mkottala@tesla Assignment1]$ rmiregistry 2525&
[1] 168272
[mkottala@tesla Assignment1]$ javac *.java
[mkottala@tesla Assignment1]$  java -Djava.security.policy=policy MarketPlaceController
Creating a Server Connection!
MarketPlaceModel: binding it to name: //tesla.cs.iupui.edu:2525/MarketPlaceServer
Market Place Server is Ready!
```

Or

Using makefile: MakeControllerServer.sh

```
[mkottala@tesla Assignment1]$ sh MakeControllerServer.sh
Creating a Server Connection!
```

2. Running the Client application

```
[mkottala@tesla Assignment1]$ java -Djava.security.policy=policy MarketPlaceView
Enter Action :
1. Login
2. Register User
1
 User login
Enter User Id:
mkottala
Enter Password:
mkottala
Successful login
```

```
[mkottala@tesla Assignment1]$ java -Djava.security.policy=policy MarketPlaceView
Enter Action :
1. Login
2. Register User
2
 Enter User Registration Details:
User Name:
Manjusha
User Id:
manju
Enter Password:
m@nju
User Registration Successful Manjusha
[mkottala@tesla Assignment1]$
```

Or
Using MakeFile: MakeClientView.sh

```
[mkottala@tesla Assignment1]$ sh MakeClientView.sh
Enter Action :
1. Login
2. Register User
1
 User login
Enter User Id:
mkottala
Enter Password:
mkottala
Successful login
```

3. Terminating RMI registry:

```
[mkottala@tesla Assignment1]$ fg
rmiregistry 2525
^C[mkottala@tesla Assignment1]$
```

Conclusion:

Working on this first assignment introduced me to what a software design pattern is and how to build a software following these object-oriented design patterns. Constructing a software following design patterns helps to improve software quality along with providing reliability. Adding to this, I have learnt to design a design model using the given requirements and turning this model into classes. Making use of MVC pattern helps a software to separate graphical interface and logic. Java RMI helps a software to work in a distributed manner hence providing reliable communication for the components server and client. Building a clean skeleton framework is the first step and this helps to further progress constructing other software patterns on top of this framework.

References:

[1] https://www.tutorialspoint.com/java_rmi/index.htm

[2] https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller

[3] https://dzone.com/articles/uml2-class-diagram-java