Design Architectures Report

Rana Hossam

MVP

Packages:

1) **View**

   Contains a package for the user interface which itself contains the main activity.

2) **Repository**

   Should handle the data source, in this case it has the model, which is depicted in the Data class.

3) **Presenter**

   Has the presenter class and a listener interface to handle fetching data from the database.

4) **Contract**

   Contains presenter interface and view interface to expose their functionalities and define the communication between them.

Main Activity (view):

Implements the IView interface and has an object of the presenter, on clicking the button, changeValue function is called from the presenter. The main activity defines the structure for the interface methods such as, incrementCount which sets the text with the updated value and on failure which shall be called in case of database failure.

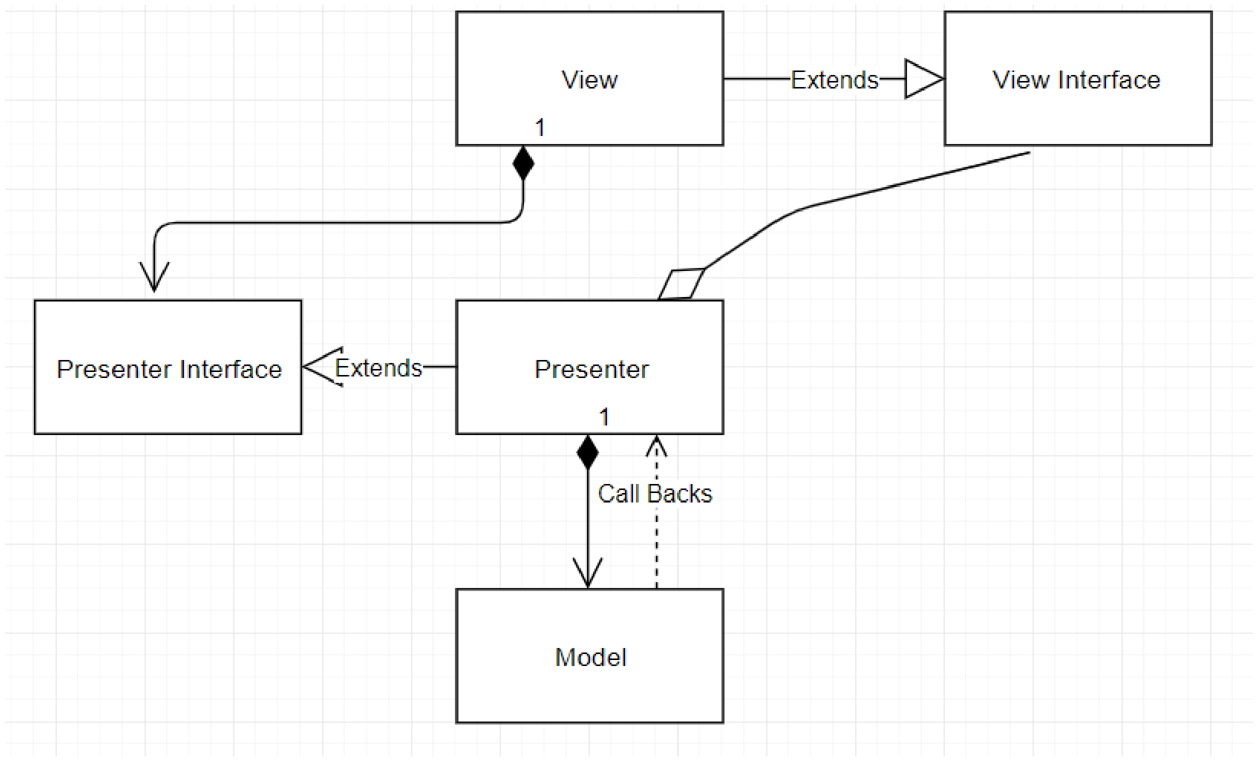
SimplePresenter:

Implements the IPresenter interface, takes the view in its constructor and defines a model. Implements changeValue function which calls incrementCount function from the model and passes to it an anonymous inner class implementing an interface (SuccessListener) that has two functions: onSuccess, which updates the view with a model value, and onFailure which also updates the view with a failure message.

Model:

Implemented with the singleton pattern, has the incrementCount method which the presenter calls, upon its invocation, the success method is invoked, which calls getCount from the model, getCount return the count value after a delay of 1 second to simulate a database.

UML:


View
Extends
View Interface
1
Presenter Interface
Extends
Presenter
1
Call Backs
Model

MVVM

Packages:

1) **View**
   Contains a package for the user interface which itself contains the main activity.
2) **ViewModel**
   Contains the ViewModel class.
3) **Repository**
   Contains the model and a repository class, this class should define the method to fetch data, either locally or from a server, in this case the database is depicted in the DataModel class and so the repository has only one means to fetch the data.

DataModel:

Implemented using singleton design pattern, has the count as a mutable live data. Contains incrementCount method which changes the count value after a delay of 1 second, and getCount method, which is used for observation later in the Repository class and in the View.

Repository:

Contains an object of the model from the sharedInstance, also has 2 methods incrementCount and getCount, both of them use the equivalent functions in the DataModel class.

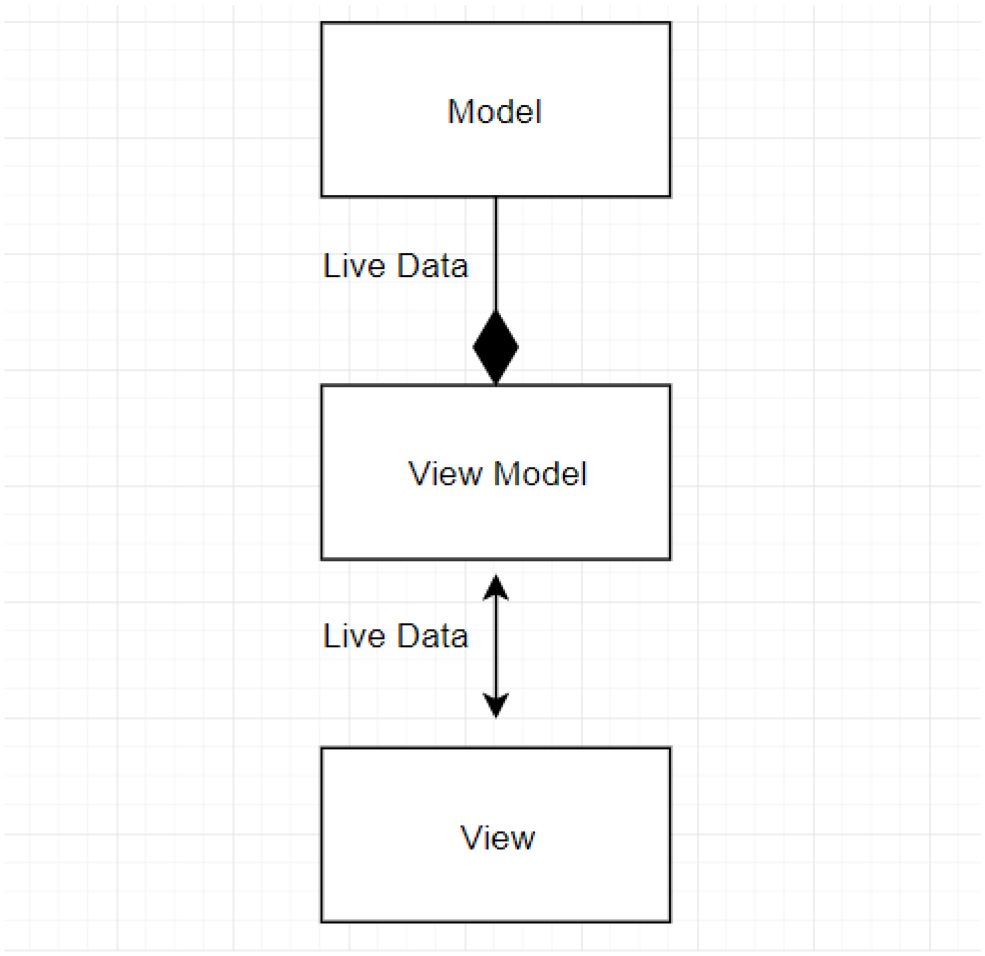
ViewModel:

Contains an object of the repository, and two functions getCount which returns LiveData and incrementCount which is triggered by the button click.

MainActivity:

Contains an object of the ViewModel. On button click, calls incrementCount method from the ViewModel. And observes on the ViewModel.getCount, which is the count value that is made live, onChanged method sets the text with the new value.

UML:


Model
Live Data
View Model
Live Data
View