

BeMobile - International Business Men - Test

The following document presents environment, technologies and OS versions, explanation of the architectural decisions made in the app and a short guide for usage.

Environment, Technologies and OS Versions

Below I will list the most important environment, technologies and OS versions that i have used to develop the app.

- Android Studio 3.5.1 (Build #AI-191.8026.42.35.5900203, built on September 25, 2019)
- JRE: 1.8.0_202-release-1483-b49-5587405 x86_64
- JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o
- macOS: 10.13.6

Architecture

The app has been built based on the MVVM architecture. Modules contains classes that have specific responsibilities. I will explain the modules below.

Views

In this module I have defined all the activities of the app. We have only two activities for this application: MainActivity and TransactionActivity. These classes are only responsible of displaying the transactions information. Also if an error occurs this classes will show the alert message indicating an error on the retrieval of transactions or rates (i.e: No internet connection). Data will be displayed with the help of Observables which are a strong base of the MVVM android architecture implementation.

Models

Models as the words says it contains the models/entities of the application. I have encapsulated two classes:

- 1) Transactions: contains SKU, Amount and Currency of a Transaction.
- 2) Rates: contains From, To and the actual Rate value of a Rate.

ViewModels

Here you will find the logic implemented to feed the views module. This could be compared to a controller for an MVC architecture. ViewModels are in charge of retrieving Transactions and Rates. This module is strongly coupled to repositories and service modules because it needs them to retrieve the data from the corresponding endpoints. Coroutines were used to implement the asynchronous behaviour while fetching the data, leaving the UI unblocked.

Repositories

This module encapsulates the logic of Transactions and Rates retrieval. In our case we always go fetch the rates to the server.

Service

Endpoints interface definition and RetrofitClient implementation for network communication.

Helpers

Generally we can find ourselves as using functions that are applied to objects, arrays, strings or integers through all the app.

Adapters

To be able to show custom elements of the Lists (RecyclerViews) we need custom adapters. This module encapsulates TransactionsAdapter and what I have defined as a RelatedTransactionAdapter.

Config

This modules actually contains the constants of the app such as DOMAIN and endpoints used. Here we could also add values as API keys for example. This is very powerful because it will allow us to change the domain easily in case a migration of servers occur just by modifying this parameter.

Running the application

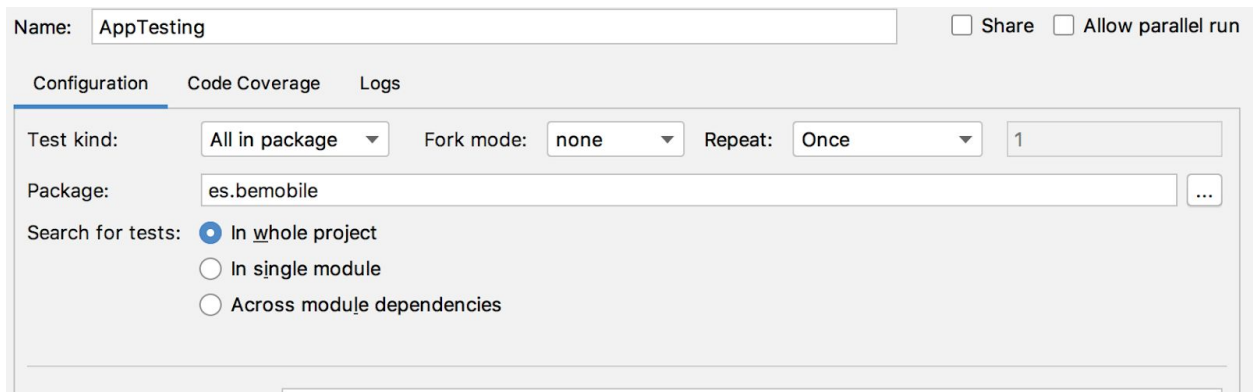
To be able to run the application, you just need to open it in the Android Studio and press “play”. You can plug in your device or just use an emulator for Android Studio.

Unit testing

I have created three classes under es/bemobile (test) which allows to test :

- 1) Round half to even.
- 2) The retrieval of the rateValue from a list of Rates (it may happen there is not a one to one conversion. A recursive function was made for this purpose).
- 3) Calculate the total amount for a given specific transaction.

You will need to add the following Run/Debug configuration to run the Unit tests :



The screenshot shows a 'Run/Debug configuration' dialog box. At the top, the 'Name' field is set to 'AppTesting'. To the right of the name field are two checkboxes: 'Share' and 'Allow parallel run', both of which are unchecked. Below the name field are three tabs: 'Configuration', 'Code Coverage', and 'Logs'. The 'Configuration' tab is selected and highlighted with a blue underline. Inside the 'Configuration' tab, there are several settings: 'Test kind:' is set to 'All in package' with a dropdown arrow; 'Fork mode:' is set to 'none' with a dropdown arrow; 'Repeat:' is set to 'Once' with a dropdown arrow, followed by a text field containing the number '1'. Below these, the 'Package:' field contains 'es.bemobile' and has a small '...' button to its right. At the bottom, the 'Search for tests:' section has three radio button options: 'In whole project' (which is selected with a blue dot), 'In single module', and 'Across module dependencies'.