Vanhack Coding Challenge - Wafer Messenger

Felipe Ramos da Silva

25/08/2018

1 Introduction

This little briefing will focus on telling about the solution for the Vanhack coding challenge for the role of Mobile Developer in Wafer Messenger. This challenge has some requirements, let's list them all:

Create an application that:

- 1. Fetches json data through endpoint https://restcountries.eu/rest/v2/all.
- 2. Display json data as listview with following elements parsed from json.
- a - name: this is Country Name.
- b - currencies: this is currency name, if more than 1 currency is present, first currency name is to be displayed.
- c - languages - name: this is language name, if more than 1 language is present first language is to be used.
- 3. On the displayed data, add a custom "Swipe to delete" with the following characteristics:
- Row background color - white.
- Swipe background color - purple.
- Delete icon - (if you cannot see it here - it should be attached -it is a bomb).
- It should have an anchor point for where the swipe just shows the delete icon.
- On a fast swipe past anchor point the row should delete.
- On a slow swipe past anchor point, swiped area should swipe back to anchor point showing the delete icon
- On a slow swipe till anchor point, the swipe should get canceled.
- Clicking/Swiping any other row should cancel the current swipe.
- 4. NO external/3rd party libraries/dependencies are to be used.

2 Dependencies

As we can use only android and Google native dependencies, now it's time to list and explain project dependencies.

- AppCompat.
- Design.
- Gson - for serializing JSON into a java POJO.

- RecyclerView - for displaying the last of countries with swipe interactive.
- Volley - for making HTTP requests.

3 Development Path

3.1 Splash Screen

In order to make a path of all developed code, the first part of the project was to focus on a splash screen, a simply "welcome" screen that makes the application more elegant, fluently and friendly. The splash screen is represented by an activity and a layout xml file. After an implemented runnable timer the splash screen will call the main activity of this application. This section is represented by the project files: SplashScreen.java, splash_screen.xml.

3.2 Main Activity UI

The layout of the main activity(WaferListActivity) is composed by three xml files. The activity_wafer_list.xml is the main one with the Toolbar and the include of the constraint layout(wafer_main.xml) and recycler view that will receive the layout wafer_list_item.xml, which has all the details of the data that will be iterated and displayed in the table.

3.3 Java POJOs

In order to serialize data from JSON to a java object we have to make a java class that will receive it. The project has two java objects, the WaferOriginal.java object that will receive the list of JSON information, where in many cases for each country you have more than one language and currency. Therefore the WaferOriginal will receive all these array of objects of names, languages and currencies, and will parse to a simply POJO called Wafer.java, that has the three requested attributes: Name, Language and Currency.

3.4 Adapter

As we have a list of the object that we want to display in a list, where we can swipe it to delete, we have to create an Adapter whom will manage the RecylerView and ViewHolder to basically iterate each item and manage each index of them. The current adapter WaferListAdapter.java is a class that implements some methods from RecyclerView.Adapter class and has methods to set the text on the screen with the attributes of each item, and remove items from the list that have been swiped by the user. More explanation and details can be found on the source code with comments and description of each step.

As can be found in the application, as a plus I've implemented the undo of the "swipe to delete" of the deleted items. Every time an item from the list is deleted, a Snackbar(A service like a Toast) will pop up with the action to undo the last deleted item. So, as the adapter has the position and index of each deleted item, and the onSwipe method implemented by WaferListActivity has the last deleted one, I can simply insert the item again on the list calling a method from the adapter.

3.5 Helper

Among the classes that we've talked about in this little briefing, the Helper is one of the most important classes, because the RecyclerView will manage the data displayed on the screen, but the Helper will set up interactive methods between the user and the screen, to trigger the swipe to delete functionality.

The RecyclerItemTouchHelper.java which implements some methods from ItemTouchHelper.SimpleCallback, will handle all these interaction with the screen when it's instantiated in the WaferListActivity.

Although this class is handling the interaction in the screen, we have to create a listener. Accordingly to this, I've created an interface that will represent the RecyclerItemTouchHelper method(onSwipe) and will be implemented by the WaferListActivity.Thus now, the main activity can have a "callback" method every time the user swipes the screen.

3.6 RestService

The dependency used to do the HTTP rest call is the google Volley, it's really faster with Gson can help us to serialize the JSON from request body into java POJO. I've implemented two ways of doing the REST call, the first one is creating a HTTP queue and a response listener, which is the faster and shorter way. And also the another way is by using an AsyncTask, which will create a parallel thread from the main one, processing the HTTP request independently.

3.7 WaferListActivity

Finally the activity will make an integration on all the classes an elements talked, it will instantiate all the necessary instances on the method onCreate and it will implement the onSwipe method.