CMPS 312 Mobile Application Development

LAB 11 – NETWORKING

OBJECTIVE

Retrofit is a REST Client for Android. It makes it relatively easy to retrieve and upload JSON, XML data via a REST based webservice. In Retrofit you configure which converter is used for the data serialization. Typically for JSON we use GSon, but you can add custom converters to process XML or other protocols. Retrofit also uses the OkHttp library for HTTP requests. OkHttp is an HTTP client that's efficient by default and it has the following features out of the box.

- HTTP/2 support allows all requests to the same host to share a socket.
- Connection pooling reduces request latency (if HTTP/2 isn't available).
- Transparent GZIP shrinks download sizes.
- Response caching avoids the network completely for repeat requests.

Hence, once you've completed this lab you should have a better understanding about Networking and Parsing of data in Android. You should also know how to use and create Network support classes to GET data from other services on the Internet.

OVERVIEW

You are required to create the Application shown in Figure 1 which downloads user information both in XML and JSON format from the **http://randomuser.me** using Retrofit Library. After downloading the content, you then display the information in a recyclerView as shown in Figure 1.

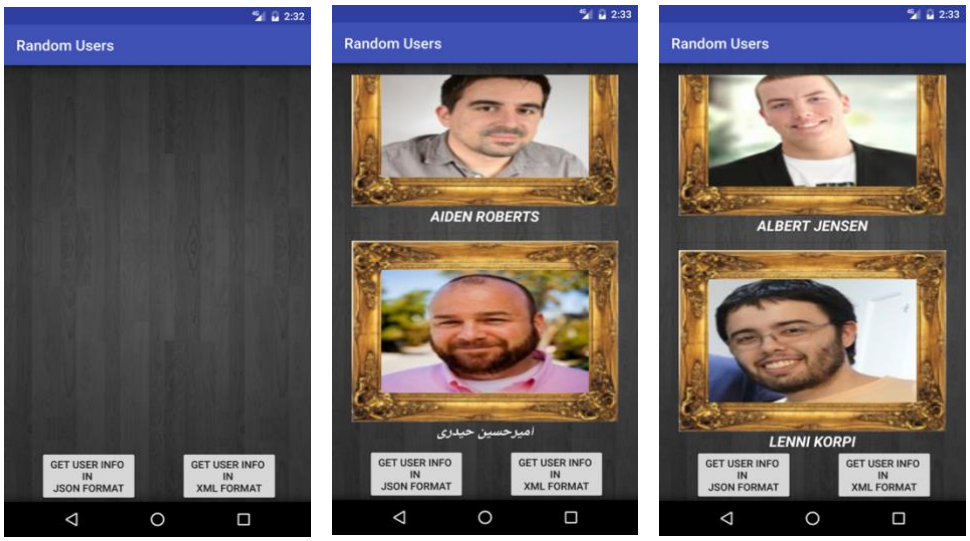

Figure 1 . Application Screen shots

PART A
Implementation Guidelines

1. Download the application skeleton code from the Github and Import it in your IDE. The skeleton code contains the implementation of the **Layouts** , **Adapter, Model Class** and **Activity** that you will need to render the information. However, you are required to do the below tasks that are related to **Networking** and **Parsing**.
2. In your gradle file add the following dependencies

```
//   add the following dependencies
   implementation 'com.squareup.retrofit2:retrofit:2.5.0'
   implementation 'com.squareup.retrofit2:converter-gson:2.5.0'
   implementation 'com.squareup.retrofit2:converter-simplexml:2.1.0'
```

3. Add the following permission to your manifest

```
<uses-permission android:name="android.permission.INTERNET"/>
```

4. In the Main Activity Layout there are two buttons named, "**Get User Info in JSON Format"** and "**GET User Info In XML Format".** Your task is once the user clicks on these buttons.
    1. Download the user information from the following two endpoints that are provided in the skeleton code.

```
String randomeUserSiteJson = "https://randomuser.me/api/?results=5&format=json";
String randomeUserSiteXml = "https://randomuser.me/api/?results=5&format=xml";
```

    For instance, If the user presses on the "**Get User Info in JSON Format",** then you need to pass "**randome_user_site_json**" variable as your request.url **.** Once you download the display it on the screen as a list. The same goes for the "**Get User Info In XML Format" button.** The only thing that you need to change is the site,

5. **Create the Models and interface for possible http operations**

    1. Open the following link in your bowser
    2. https://randomuser.me/api/?results=5&format=json
    3. Study the structure of the JSON data that is returned by the randomuser api
    4. Create a POJO class that can hold this data. As you will see to extract the email, user, login, we will need to first
    5. extract the results array. Then we can extract the email and login.
    6. Create three classes "Login, Result and User". Each should hold the root elements of the JSON returned data.

```
@SerializedName("results")
@Expose
```

    7. Create an interface and name it "**UsersClient**"
    8. Create a single method called "**getUsers**" which returns a Result Object
    9. Annotate the method with GET and also pass the Query Strings

```
public interface UserClient {

    String BASE_URL = "https://randomuser.me";

    @GET("/api/")
    Call<Result> getUsers(
            @Query("results") String results,
            @Query("format") String format
    );

}
```

10. Create the retrofit object

```
Retrofit retrofit = new Builder()
        .baseUrl(UserClient.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();
```

11. Create the client

```
//Create the client
UserClient client = retrofit.create(UserClient.class);
```

12. Make the call

```
//Make the call
Call<Result> call = client.getUsers( results: "5",  format: "json");
```

13. Listen for response

```
//wait for the response using the following call back

call.enqueue(new Callback<Result>() {
    @Override
    public void onResponse(Call<Result> call, Response<Result> response) {
        Toast.makeText( context: MainActivity.this, users.get(0).getLogin().getUsername() ,
                Toast.LENGTH_SHORT).show();
    }

    @Override
    public void onFailure(Call<Result> call, Throwable t) {
        Toast.makeText( context: MainActivity.this, t.getMessage() , Toast.LENGTH_SHORT).show();
    }
});
```

14. display the information into a recyclerView. The Layout and Adapter of the ListView are both provided to you. Hence, all you need is to pass the expected parameters to the adapter which are the "Context and ArrayList of Users as shown below.

```
public MyAdapter(Context context, ArrayList<User> users)
```

15. Download the user images using picaso https://square.github.io/picasso/

16. Add the following Logging Interceptor which logs HTTP request and response data. https://github.com/square/okhttp/tree/master/okhttp-logging-interceptor

```
Retrofit retrofit = new Builder()
        .baseUrl(UserClient.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .client(okHttpClient)
        .build();
```

17. Change the format to XML and Download the same content using the simple xml convertor library.
    You can follow this example https://medium.com/@anirudh03sharma/parsing-xml-using-retrofit-ac43cc0b7628