Open in app

Prakash Pun

410 Followers · About Follow

Retrofit— A simple Android tutorial

Prakash Pun Dec 24, 2017 · 3 min read

For this tutorial, we are going to see how to use Retrofit HTTP client in your Android application.

Retrofit
A type-safe HTTP client for Android and Java
square.github.io

Retrofit is an awesome type-safe HTTP client for Android and Java built by awesome folks at Square. Retrofit makes it easy to consume JSON or XML data which is parsed into Plain Old Java Objects (POJOs).

If you just want to get the sample project, then you can find it here.

prakashpun/RetrofitTutorial
RetrofitTutorial - A simple android application that uses Retrofit library to read data from REST api
github.com

So, without any further delays, lets get started by first creating a new project in Android Studio.

1. Go to **File ⇒ New Project**. When it prompts you to select the default activity, select **Empty Activity** and proceed.

2. Open **build.gradle** in (Module:app) and add **Retrofit**, **Picasso**, **RecyclerView**, **Gson** dependencies like this.

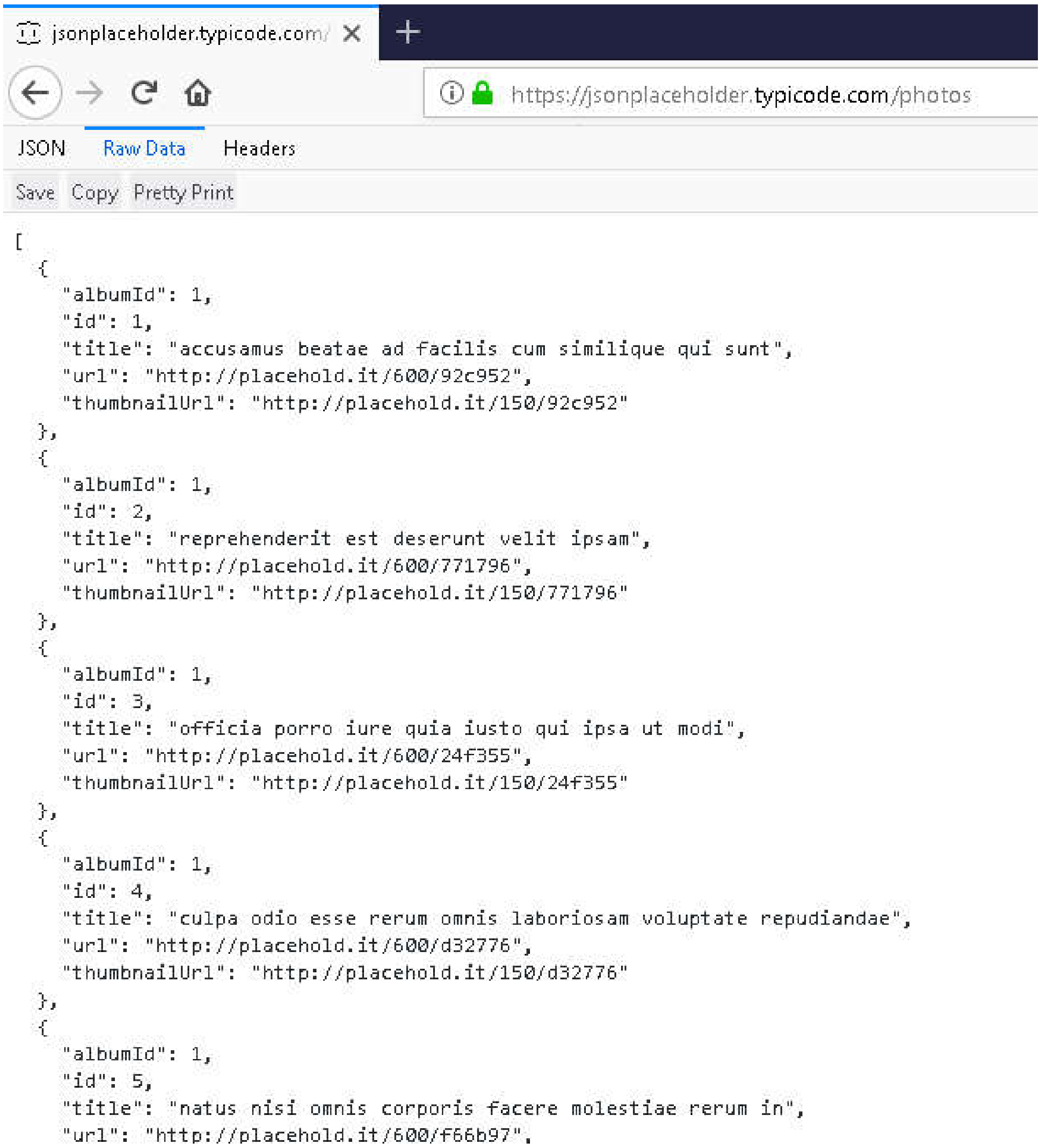
```
dependencies {
    ...

    compile "com.android.support:cardview-v7:26.1.0"
    compile 'com.android.support:recyclerview-v7:26.1.0'
    compile 'com.squareup.picasso:picasso:2.5.2'

    compile 'com.squareup.retrofit2:retrofit:2.3.0'
    compile 'com.squareup.retrofit2:converter-gson:2.3.0'
    compile 'com.jakewharton.picasso:picasso2-okhttp3-downloader:1.1.0'

    ...
}
```

build.gradle hosted with ❤ by **GitHub** **view raw**

3. Don’t forget to add INTERNET permissions in **AndroidManifest.xml** file like this

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.tuts.prakash.retrofittutorial">

    <!--Internet Permission-->
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".activity.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

4. Next, we will create data model to parse our sample JSON data with following structure.


jsonplaceholder.typicode.com/
https://jsonplaceholder.typicode.com/photos
JSON
Raw Data
Headers
Save
Copy
Pretty Print

```
[
  {
    "albumId": 1,
    "id": 1,
    "title": "accusamus beatae ad facilis cum similique qui sunt",
    "url": "http://placehold.it/600/92c952",
    "thumbnailUrl": "http://placehold.it/150/92c952"
  },
  {
    "albumId": 1,
    "id": 2,
    "title": "reprehenderit est deserunt velit ipsam",
    "url": "http://placehold.it/600/771796",
    "thumbnailUrl": "http://placehold.it/150/771796"
  },
  {
    "albumId": 1,
    "id": 3,
    "title": "officia porro iure quia iusto qui ipsa ut modi",
    "url": "http://placehold.it/600/24f355",
    "thumbnailUrl": "http://placehold.it/150/24f355"
  },
  {
    "albumId": 1,
    "id": 4,
    "title": "culpa odio esse rerum omnis laboriosam voluptate repudiandae",
    "url": "http://placehold.it/600/d32776",
    "thumbnailUrl": "http://placehold.it/150/d32776"
  },
  {
    "albumId": 1,
    "id": 5,
    "title": "natus nisi omnis corporis facere molestiae rerum in",
    "url": "http://placehold.it/600/f66b97",
```

Sample JSON data

Create a class named **RetroPhoto.java** under **model** package like this.

5. Create the Retrofit Instance

To issue network requests to a REST API with Retrofit, we need to create an instance using the `Retrofit.Builder` class and configure it with a base URL.

Create a class **RetrofitClientInstance.java** under **network** package. Here **BASE_URL** is the basic URL of our API where we will make a call.

6. Define the Endpoints

The endpoints are defined inside of an interface using special retrofit annotations to encode details about the parameters and request method.

7. Create custom adapter for binding data with RecycleView.

Create a class named **CustomAdapter.java** under **adapter** package like this.

7. Final step

Inside the **onCreate()** method of the **MainActivity.java**, we initialize an instance of the **GetDataService** interface (line 16), the **RecyclerView**, and also the adapter. Finally, we call the **generateDataList**() method.

8. Understanding `enqueue()`

`enqueue()` asynchronously sends the request and notifies your app with a callback when a response comes back. Since this request is asynchronous, Retrofit handles it on a background thread so that the main UI thread isn't blocked or interfered with.

To use `enqueue()`, you have to implement two callback methods:

- `onResponse()`
- `onFailure()`

9. Fire up the app

Finally, you can fire up the app and see a nice list like this.

Final Output

And that's it. Thank you for sticking with me till the end of this tutorial. I hope this was help in some way to you.

If you liked this article make sure to 👏 it below, and follow me on twitter!

Till next time :) Happy Learning.

If you would like to support me, please consider buying me a cuppa

Android Retrofit2 Picasso Tutorial