ANDROID JSON PARSING USING GSON AND RECYCLERVIEW

In this tutorial, we are going to learn how to parse Json object in android application using Gson. The resulting parsed object will be used to populate a RecyclerView.

JSON stands for JavaScript Object Notation and is use for data interchange between android application and a server.

A more detailed definition was taken from ***JSON website*** – "JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition – December 1999".

Gson is a Java library that can be used to convert Java Objects into

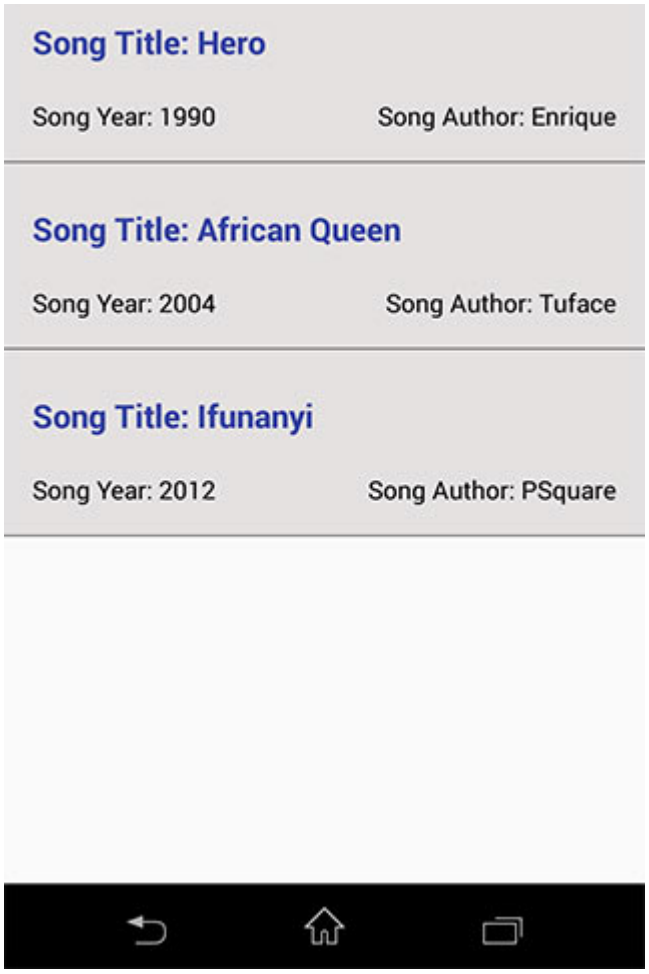
Java Generics. Gson considers both of these as very important design goals.

What we will basically do in this tutorial, is to make a request to a remote server and return a Json string as response, thereafter, the response object will be parsed with Gson library and the resulting data will be bind to android RecycleView.

We will make use of Volley for the network call. Volley is an HTTP library that makes networking for Android apps easier and most importantly, faster. Volley is available through the open AOSP repository.

If you want to a simple tutorial on ***how to parse Json in android using HttpClient,*** I wrote a tutorial on this topic. I will suggest you read it first before you proceed with this.

Before we go deeper into this tutorial, it is important for us to understand what we are planning to achieve. Below is the screen-shot of the application we will be creating.



- Android UI App for Food Ordering and Delivery from Multiple Restaurants

We are going to use android RecycleView. We will create an Adapter for the RecycleView widget. The Adapter will inflate a custom layout file we will create which will hold the UI for the RecycleView items.

This item layout file will contain three TextView widget controls that will hold –

1. Song Title

2. Song Year

3. Song Author

Sony Xperia ZL

Min SDK 14

Target SDK 23

To create a new android application project, follow the steps as stipulated below.

Go to File menu

Click on New menu

Click on Android Application

Enter Project name: AndroidGsonParser

Package: com.inducesmile.androidgsonparser

Select Blank Activity

Keep other default selections

Continue to click on next button until Finish button is active, then click on Finish Button.

Now that we have created our project, we are going to add internet permission in our project **Manifest.xml** file. We need this permission in order to make network calls. Copy and paste this single line of code inside your Manifest.xml file.

```
<uses-permission android:name="android.permission.INTERN
```

Since we are going to use volley library for our network call, RecycleView to display our data to users and Gson library to parse the

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:23.1.1'
    compile 'com.android.support:recyclerview-v7:23.1.1'
    compile 'com.mcxiaoke.volley:library:1.0.19'
    compile 'com.google.code.gson:gson:2.6.1'
}
```

The content of the Json object obtained as the response object is place in a remote .php file. The content of the file is shown below.

```
<?php

echo json_encode(Array(Array( "song_name" => "Hero", "so

?>
```

Lets open the default layout file – **activity_main.xml** located in the **res > layout folder.** This is where we will add the RecycleView. It is a simple layout file. Copy and paste the code below to this layout file.

ACTIVITY_MAIN.XML

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.co
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.inducesmile.androidgsonparser.Mai
    <android.support.v7.widget.RecyclerView
        android:id="@+id/recycler_view"
        android:layout_width="match_parent"
```

We will move on to create the item list layout which our adapter will inflate. Right click on the layout folder and choose ***XML > Layout XML file***. Name the file ***list_item.xml.*** Open this file and add the following code snippet in it.

LIST_ITEM.XML

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.co
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    android:background="@color/colorBackground">
    <TextView
        android:id="@+id/song_title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Text"
        android:layout_marginTop="10dp"
        android:textSize="18dp"
        android:textStyle="bold"
        android:textColor="@color/colorPrimaryDark"/>
    <TextView
        android:id="@+id/song_year"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Text"
        android:textColor="#000"
        android:layout_below="@+id/song_title"
        android:layout_alignLeft="@+id/song_title"
        android:layout_alignStart="@+id/song_title"
        android:layout_marginTop="20dp" />
    <TextView
        android:id="@+id/song_author"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Text"
```

We will create the RecycleView Adapter which will bind our data source to our application list items of the RecycleView.

Right click on the package folder located at **java > package folder,** click on New and select Java file. Name the file ***RecyclerViewAdapter.java.*** This adapter class will extends **RecyclerView.Adapter<RecyclerViewHolders>.** Open this file, copy and paste the code below to the file.

RECYCLERVIEWADAPTER.JAVA

```
import android.content.Context;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import java.util.List;
public class RecyclerViewAdapter extends RecyclerView.Ad
    private List<ItemObject> itemList;
    private Context context;
    public RecyclerViewAdapter(Context context, List<Ite
        this.itemList = itemList;
        this.context = context;
    }
    @Override
    public RecyclerViewHolders onCreateViewHolder(ViewGr
        View layoutView = LayoutInflater.from(parent.get
        RecyclerViewHolders rcv = new RecyclerViewHolder
        return rcv;
    }
    @Override
    public void onBindViewHolder(RecyclerViewHolders hol
        holder.songTitle.setText("Song Title: " + itemLi
        holder.songYear.setText("Song Year: " + itemList
        holder.songAuthor.setText("Song Author: " + item
    }
```

we used to create the previous java file but in this case we will name our java file **RecyclerViewHolders.java.**

Open this class, copy and paste the code below inside this file.

RECYCLERVIEWHOLDERS.JAVA

```
import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.TextView;
public class RecyclerViewHolders extends RecyclerView.Vi
    public TextView songTitle;
    public TextView songYear;
    public TextView songAuthor;
    public RecyclerViewHolders(View itemView) {
        super(itemView);
        itemView.setOnClickListener(this);
        songTitle = (TextView)itemView.findViewById(R.id
        songYear = (TextView)itemView.findViewById(R.id.
        songAuthor = (TextView)itemView.findViewById(R.i
    }
    @Override
    public void onClick(View view) {
    }
}
```

Next, we will create an entity class which will map the properties of the class to the object properties returned from the Json Object. You can see that the ***@SerializedName("")*** annotation was used. This is because the class properties have different names.

Follow the same steps above to create a new java file. We will name the file **ItemObject.java.** Once you have created this file, open the file, copy and paste the code below inside the file.

```
    @SerializedName("artist_name")
    private String songAuthor;
    public ItemObject(String songTitle, String songYear,
        this.songTitle = songTitle;
        this.songYear = songYear;
        this.songAuthor = songAuthor;
    }
    public String getSongTitle() {
        return songTitle;
    }
    public String getSongYear() {
        return songYear;
    }
    public String getSongAuthor() {
        return songAuthor;
    }
}
```

To bring this all together, we will move to the MainActivity file. In this class, we will get an instance of the RecycleView, then, we will create an object of the Adapter and pass it as a parameter to the setAdapter method of the RecycleView class.

Remember that we will also create a Volley string request which will return the Json object that we will use to populate the RecycleView items.

The response Json string object is parse to Java plain object using the Gson library. You can see how all the codes come together.

Open the MainActivity.java file, copy and paste the following code inside the class.

MAINACTIVITY.JAVA

```
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
public class MainActivity extends AppCompatActivity {
    private final String TAG = "MainActivity";
    private RecyclerView recyclerView;
    private LinearLayoutManager layoutManager;
    private RecyclerViewAdapter adapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        recyclerView = (RecyclerView)findViewById(R.id.r
        recyclerView.addItemDecoration(new SimpleDivider
        layoutManager = new LinearLayoutManager(MainActi
        recyclerView.setLayoutManager(layoutManager);
        requestJsonObject();
    }
    private void requestJsonObject(){
        RequestQueue queue = Volley.newRequestQueue(this
        String url ="http://toscanyacademy.com/blog/mp.p
        StringRequest stringRequest = new StringRequest(
                    @Override
                    public void onResponse(String respon
                        Log.d(TAG, "Response " + respons
                        GsonBuilder builder = new GsonBu
                        Gson mGson = builder.create();
                        List<ItemObject> posts = new Arr
                        posts = Arrays.asList(mGson.from
                        adapter = new RecyclerViewAdapte
                        recyclerView.setAdapter(adapter)
                    }
                }, new Response.ErrorListener() {
```

In this code, if you look at this line

```
recyclerView.addItemDecoration(new SimpleDividerItemDeco
```

If you look at the single line of code above, we are using it to add a line separator for the items in the RecycleView. If you like you can omit this part but if you want to add a line separator to the items of the RecycleView, then create a new java file and name it ***SimpleDividerItemDecoration.java.***

Open the file, copy and paste the following code inside this file

SIMPLEDIVIDERITEMDECORATION.JAVA

```
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.drawable.Drawable;
import android.support.v4.content.ContextCompat;
import android.support.v7.widget.RecyclerView;
import android.view.View;
public class SimpleDividerItemDecoration extends Recycle
    private Drawable mDivider;
    public SimpleDividerItemDecoration(Context context)
        mDivider = ContextCompat.getDrawable(context, R.
    }
    @Override
    public void onDrawOver(Canvas c, RecyclerView parent
        int left = parent.getPaddingLeft();
        int right = parent.getWidth() - parent.getPaddin
        int childCount = parent.getChildCount();
        for (int i = 0; i < childCount; i++) {
            View child = parent.getChildAt(i);
            RecyclerView.LayoutParams params = (Recycler
```

Also, we will add this drawable layout file in the drawable folder. Create a new Drawable layout file inside the Drawable folder and name it line_divider.xml. Copy and paste the following code inside the file.

LINE_DIVIDER.XML

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res
    android:shape="rectangle">
    <size
        android:width="1dp"
        android:height="1dp" />
    <solid android:color="@color/colorDivide" />
</shape>
```

Run your application and see for yourself what we have just created.

This brings us to the end of this tutorial, If you find anything confusing kindly contact me with your questions or use the comment box below.

You can download the code for this tutorial below. If you are having hard time downloading the tutorials, kindly contact me.

Android Json Parser With Gson 27.11 MB
Download

Remember to subscribe with your email address so that you will be among the first to receive my new post once it is published.

Please if you love this tutorial, kindly download my new android app – ***Daily Pocket Calculator*** – in Google Play Store and let me know what

How to play YouTube video inside Android WebView using Video Url

Android CardView and Toolbar with Material Design

Android Location API Using Android LocationManager

Android with Yii 2 Restful API Example

Add Header To Android RecyclerView

Android Mobile Food Ordering App For Restaurant - Project Idea Source Code

ABOUT THE AUTHOR

4 COMMENTS

Rafelcf | October 18, 2016 Log in to Reply

Henry seems very good your project, with few lines you get the json, but I have a problem. Ten serious kind make it a little more complicated? as you obtained if the json begins with
{ “Item”: [{ “color”: “red” ……..
Thank you.

Inducesmile

Henry Author | October 19, 2016
Log in to Reply

It is not that difficult. You just have to follow the structure of your Json object. You can as well use Gson with POJO. You can check my tutorial here to see how I achieved it – https://inducesmile.com/android/android-find-distance-and-duration-between-two-points-on-android-map/

s3sx | November 6, 2016 Log in to Reply

Henry Hello,I`m from Kazakhstan I have question

Onclick not working why?
please help me?!!!
public class RecyclerViewHolders extends RecyclerView.ViewHolder

```
songTitle = (TextView)itemView.findViewById(R.id.song_title);
songYear = (TextView)itemView.findViewById(R.id.song_year);
songAuthor = (TextView)itemView.findViewById(R.id.song_author);
}
@Override
public void onClick(View view) {
Toast.makeText(context,"Hello " +
sonYear,Toast.LENGTH.LONG).show;
}
}
FATAL EXCEPTION: main
Process: com.example.abylai.menu, PID: 841
java.lang.NullPointerException
at android.content.ComponentName.(ComponentName.java:77)
at android.content.Intent.(Intent.java:4235)
at
com.example.abylai.menu.RecyclerViewHolders.onClick(RecyclerView
```

Inducesmile

Henry Author | November 7, 2016

Log in to Reply

The error says that you have a NullPointerException, Please check that you are using the real id reference for your view widgets.

ADD A COMMENT

You must be logged in to post a comment.