Android Studio Manual

1. Scope

This document is a small manual for some features that Android Studio has.

1. Features
   1. **Switch between fragments**
      1. Create different fragments.
      2. In the activity layout where you want to switch between fragments define a FrameLayout.

<FrameLayout  
 android:id="@+id/fragment\_container"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"></FrameLayout>

* + 1. Use the support fragment manager to add or replace fragments in the frame layout.

getSupportFragmentManager().  
 beginTransaction().  
 replace(R.id.*fragment\_container*, fragment).  
 commit();

* 1. **Communication between fragment and activity**

The best way to call an activity method from a fragment is using listeners. Follow these steps to achieve this functionality.

* + 1. Define an interface in the **fragment** with the necessary methods that you need to call in the activity.

public interface OnFragmentInteractionListener {  
 // *TODO: Update argument type and name* void onFragmentInteraction(String parameter);  
}

* + 1. Define a variable with interface type in the **fragment.**

private OnFragmentInteractionListener mListener;

* + 1. In the **fragment,** onAttach method, assign the variable defined in previous step with the context parsed as the interface.

@Override  
public void onAttach(Context context) {  
 super.onAttach(context);  
 if (context instanceof OnFragmentInteractionListener) {  
 mListener = (OnFragmentInteractionListener) context;  
 } else {  
 throw new RuntimeException(context.toString()  
 + " must implement OnFragmentInteractionListener");  
 }  
}

* + 1. In the activity where you want to call the method implement the interface.

public class MainActivity extends FragmentActivity  
 implements ItemsFragment.OnFragmentInteractionListener

* + 1. Implement the methods from the interface

@Override  
public void onFragmentInteraction(String parameter) {  
 //empty for now  
 Log.*d*("Parameter", parameter);  
}

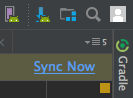
* + 1. Make the call whenever you need it in the **fragment**. This will call the method implemented in the activity.

mListener.onFragmentInteraction(parameter);

* 1. Spinner
  2. Sugar ORM
     1. Inside dependencies in build.grade (Module:app) add sugar ORM reference. In latest android studio versions compile is obsolete, you need to use implementation instead.

compile 'com.github.satyan:sugar:1.3'

* + 1. Syncronize the project using the Sync Now option at the top right hand side of the screen.



* + 1. In AndroidManifest.xml, in the application tag, include the following property:

android:name="com.orm.SugarApp"

* + 1. In AndroidManifest.xml, in the application tag, include the following property. Some project may require to include more properties. You need to verify if it is required when compiling the project.

tools:replace="android:icon"

* + 1. In AndroidManifest.xml, inside the application tag, add the following tags and update the values based on your application. Database is for the database name. Version always 2 the first time. Query log in true to see in the log all events related to the database. Domain package name is the package where the entities for the database are located.

<meta-data android:name="DATABASE" android:value="sugar\_example.db" />

<meta-data android:name="VERSION" android:value="2" />

<meta-data android:name="QUERY\_LOG" android:value="true" />

<meta-data android:name="DOMAIN\_PACKAGE\_NAME" android:value="com.example" />

* + 1. Whenever there is a change in the structure of a table or the database, you need to update the version number in the manifest in order to update the structure of the database.
    2. All entities that you want to store in the database needs to extend from SugarRecord class. If your class has a property id, you need to remove it as it will clash with SugarRecord id (long).

public class Item extends SugarRecord<Item> {

* + 1. All SugarRecord classes need to have its default constructor.

public Item(){  
  
}

* + 1. The first time a method calls database method. Sugar ORM will create the database and tables. In order to add an record use the .save() method from the class.

Item objItem = new Item();  
objItem.setCategoryId(1);  
objItem.setDescription("My android studio test!");  
objItem.setName("Android studio test");  
objItem.setReferencialValue(1000);  
objItem.setStatus("Created");  
objItem.setTradable(true);  
objItem.setUserId(1);  
objItem.save();

* + 1. Instant run option must be disabled until the database is created to avoid problems creating the database. You can disable Instant run in Android Studio Preferences; Build, Execution and Deployment; Instant run.

../../Desktop/Screen%20Shot%202018-05-04%20at%208.15.06%20PM.p

* + 1. When saving a new record, you receive the error: <table> no such table exists, that means that database was created but the table doesn’t. You need to upgrade the version number by 1 value and try again. That should create the table.
    2. In order to get all record from a table you can use the .listAll() method.

List<Item> items = Item.*listAll*(Item.class);

* 1. ReciclerView
  2. Call GET api
     1. Define the URL.

// Create URL  
URL myAPIUrl = new URL("http://www.apis.com/api/ItemsApp");

* + 1. Create the connection. If the url is a https url, use the class HttpsURLConnection.

// Create connection  
HttpURLConnection myConnection =  
 (HttpURLConnection) myAPIUrl.openConnection();

* + 1. Execute the call. Response code 200 means that connection is successful.

if (myConnection.getResponseCode() == 200){

* + 1. Define reading objects.

InputStream responseBody = myConnection.getInputStream();  
  
InputStreamReader responseBodyReader =  
 new InputStreamReader(responseBody, "UTF-8");  
  
JsonReader jsonReader = new JsonReader(responseBodyReader);

* + 1. Use jsonReader to read json response from API

jsonReader.beginArray();  
while(jsonReader.hasNext()){  
 jsonReader.beginObject();  
 int id = 0;  
 String name = "";  
 String description = "";  
 while (jsonReader.hasNext()){  
 String key = jsonReader.nextName();  
 switch (key.toLowerCase()){  
 case "id":  
 id = jsonReader.nextInt();  
 break;  
 case "name":  
 name = jsonReader.nextString();  
 break;  
 case "description":  
 description = jsonReader.nextString();  
 break;  
 default:  
 jsonReader.skipValue();  
 break;  
 }  
 }  
 items.add(new Item(id, name, description));  
 jsonReader.endObject();  
}  
jsonReader.endArray();  
jsonReader.close();

* + 1. Remember that you need to read all element in json response otherwise you will get an error. If you don’t want to read an element, you can use the skipValue() method. Also, if you are reading an object instead an array, the first sentence should be jsonReader.beginObject().
  1. Call POST api

1. References
   1. <https://developer.android.com>
   2. <https://code.tutsplus.com/tutorials/android-from-scratch-using-rest-apis--cms-27117>
   3. <http://satyan.github.io/sugar/index.html>
   4. <https://material.io/guidelines/>