**Lab 9: Android Maps**

# **Introduction**

# In addition to the interfaces described above, you must implement the LocationListener interface if you want to receive location updates. The callback methods listed below are provided by this interface and must be implemented in your activity class. The LocationRequest object is used to ask the LocationClient for a quality of service (QoS) for location updates.

**Let’s get Started:**

In this experiment we will develop an Android App to demonstrate the use of Android Maps and Location.

**Step 1: Create a New Project in Android Studio as shown below**

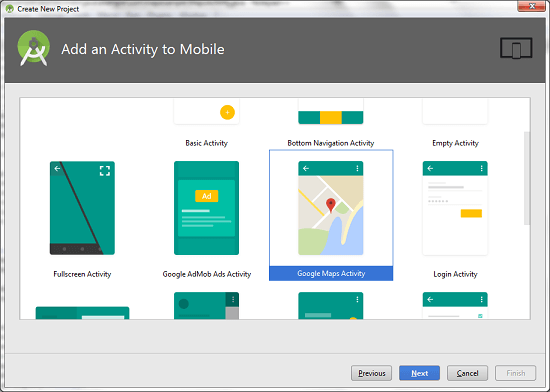
Graphical user interface, text, application

Description automatically generated

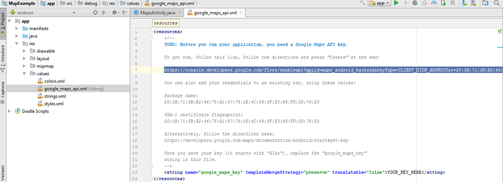
### **Example of Google Map**

Create a Project of Google map integrating within our app. For doing this we select Google Maps Activity.

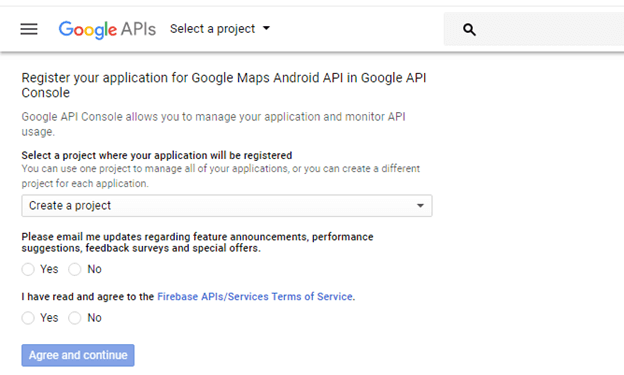
Play Video



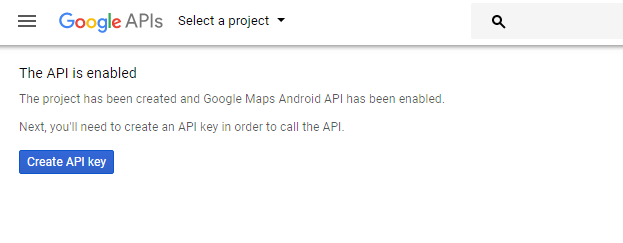
Copy the URL from google\_map\_api.xml file to generate Google map key.



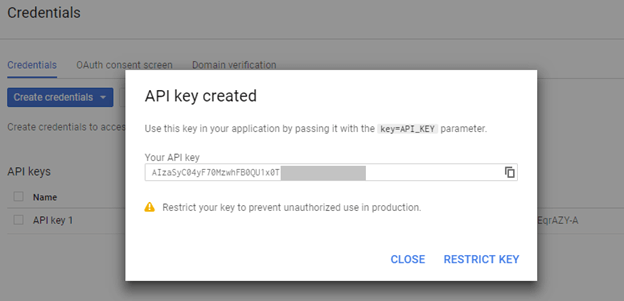
Paste the copied URL at the browser. It will open the following page.



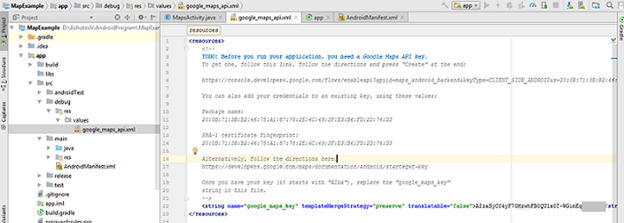
Click on Create API key to generate API key.



After clicking on Create API key, it will generate our API key displaying the following screen.



Copy this generated API key in our google\_map\_api.xml file



**activity\_maps.xml**

**<fragment** xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:map="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:id="@+id/map"

    android:name="com.google.android.gms.maps.SupportMapFragment"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    tools:context="example.com.mapexample.MapsActivity" **/>**

### **MapsActivity.java**

To get the GoogleMap object in our MapsActivity.java class we need to implement the OnMapReadyCallback interface and override the onMapReady() callback method.

**package** example.com.mapexample;

**import** android.support.v4.app.FragmentActivity;

**import** android.os.Bundle;

**import** com.google.android.gms.maps.CameraUpdateFactory;

**import** com.google.android.gms.maps.GoogleMap;

**import** com.google.android.gms.maps.OnMapReadyCallback;

**import** com.google.android.gms.maps.SupportMapFragment;

**import** com.google.android.gms.maps.model.LatLng;

**import** com.google.android.gms.maps.model.MarkerOptions;

**public** **class** MapsActivity **extends** FragmentActivity **implements** OnMapReadyCallback{

**private** GoogleMap mMap;

    @Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_maps);

        // Obtain the SupportMapFragment and get notified when the map is ready to be used.

        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()

                .findFragmentById(R.id.map);

        mapFragment.getMapAsync(**this**);

    }

    @Override

**public** **void** onMapReady(GoogleMap googleMap) {

        mMap = googleMap;

        // Add a marker in Sydney and move the camera

        LatLng sydney = **new** LatLng(-34, 151);

        mMap.addMarker(**new** MarkerOptions().position(sydney).title("Marker in Sydney"));

        mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));

    }

}

### **Required Permission**

Add the following user-permission in AndroidManifest.xml file.

**<uses-permission** android:name="android.permission.ACCESS\_FINE\_LOCATION" **/>**

**<uses-permission** android:name="android.permission.ACCESS\_COARSE\_LOCATION" **/>**

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

### **AndroidManifest.xml**

**<?xml** version="1.0" encoding="utf-8"**?>**

**<manifest** xmlns:android="http://schemas.android.com/apk/res/android"

    package="example.com.mapexample"**>**

    <!--

         The ACCESS\_COARSE/FINE\_LOCATION permissions are not required to use

         Google Maps Android API v2, but you must specify either coarse or fine

         location permissions for the 'MyLocation' functionality.

    --**>**

**<uses-permission** android:name="android.permission.ACCESS\_FINE\_LOCATION" **/>**

**<uses-permission** android:name="android.permission.ACCESS\_COARSE\_LOCATION" **/>**

**<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"**>**

**<meta-data**

            android:name="com.google.android.geo.API\_KEY"

            android:value="@string/google\_maps\_key" **/>**

**<activity**

            android:name=".MapsActivity"

            android:label="@string/title\_activity\_maps"**>**

**<intent-filter>**

**<action** android:name="android.intent.action.MAIN" **/>**

**<category** android:name="android.intent.category.LAUNCHER" **/>**

**</intent-filter>**

**</activity>**

**</application>**

**</manifest>**

### **build.gradel**

Add the following dependencies in build.gradel file.

dependencies {

    implementation fileTree(dir: 'libs', include: ['\*.jar'])

    implementation 'com.android.support:appcompat-v7:26.1.0'

    implementation 'com.google.android.gms:play-services-maps:11.8.0'

    testImplementation 'junit:junit:4.12'

    androidTestImplementation 'com.android.support.test:runner:1.0.1'

    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'

}

**Output**

