By using **Google Maps** Android API we can integrate google maps in android applications to show the location details on map based on our requirements.

To use google maps in our android applications we need to install **Google Play Services** SDK in our Android Studio because google made **Google Mas API** as a part of Google Play Services SDK.

To install Google Play Services, open **Android Studio** à Go to **Tools** menu à **Android** à click **SDK Manager**, then new window will open in that select **SDK Tools** tab à Select **Google Play Services** à click **OK**

# Android Google Maps API Example

Create a new android application using android studio and give names as **GoogleMapExample**

Now we need to select the form factors which we need for our app. In case if you're not sure what you need, just select **Phone and Tablet** and then click **Next**

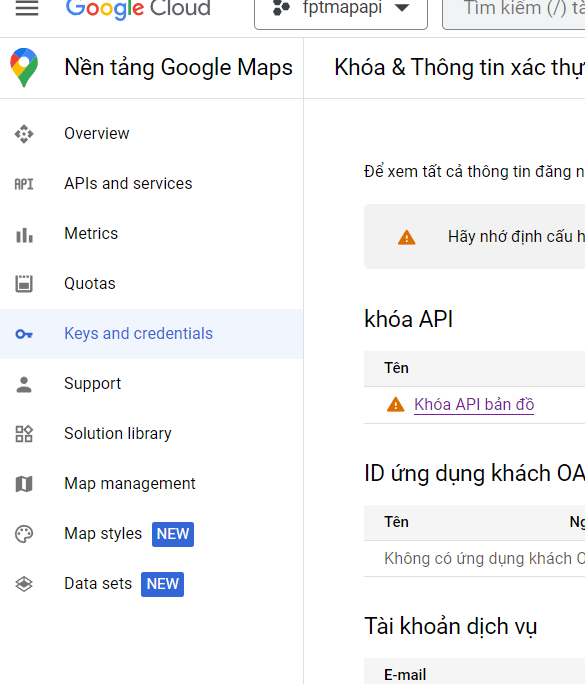
Now select the Google Maps Activity in '**Add an activity to Mobile**' dialog and click **Next**

Customize the activity by entering activity name, layout name and title as prompted. In case if default values are fine, then click **Finish**

Android Studio will open **google\_maps\_api.xml** and **MapsActivity.java** files in the editor. The **google\_maps\_api.xml** file will contains instructions to generate a Google Maps API key to access Google Maps servers

# Create a Project in Google Console

Copy and paste the console URL (<https://console.cloud.google.com/google/maps-apis/> ) in browser and it will take you to **Google API Console** like as shown below. Follow the instructions to create a **new project** on **Google API Console**.



Now copy the API Key, go back to android studio and paste the API key into the **<string>** element in **google\_maps\_api.xml** file like as shown below.

<string name="google\_maps\_key" templateMergeStrategy="preserve" translatable="false">*This is your key API*</string>

## Activity\_maps.xml

By default, the XML file (**activity\_maps.xml**) that defines the app's layout is at **res/layout/** contains the following code.

<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="com.FPTUniversity.googlemapexample.MapsActivity" />

## AndroidManifest.xml

Our application manifest file (**AndroidManifest.xml**) will contain the code like as shown below with required our **Google Maps API Key** and permissions.

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

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

If you observe above code it contains a user permission to access location and Meta tag to get google maps API key.

## MapsActivity.java

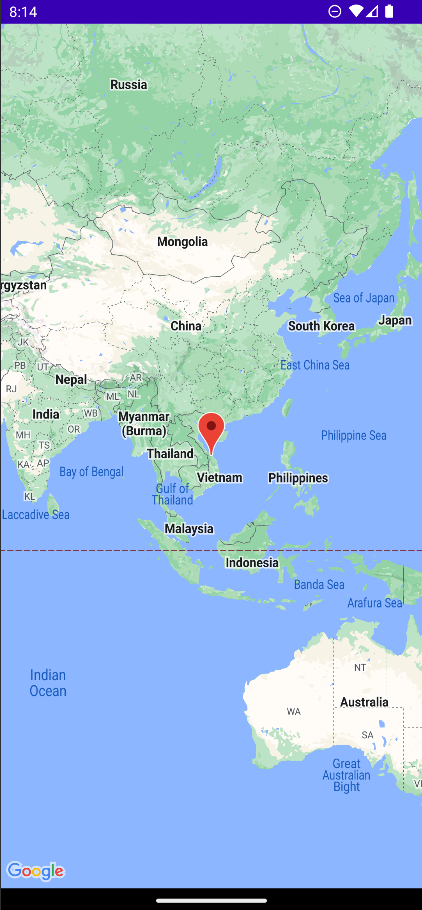
By default, the Java file (**MapsActivity.java**) that defines the maps activity will contain the following code.

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 hyderabad and move the camera  
        LatLng hyderadbad = new LatLng(16, 108);  
        mMap.addMarker(new MarkerOptions().position(hyderadbad).title("Da Nang"));  
        mMap.moveCamera(CameraUpdateFactory.newLatLng(hyderadbad));  
    }  
}

Generally, during the launch of our activity, **onCreate()** callback method will be called by android framework to get the required layout for an activity.

## Output of Android Google Maps Example

When we run the above example using an android virtual device (AVD) we will get a result like as shown below.



Reference: <https://dotnet.edu.vn/ChuyenMuc/Xay-dung-chuong-trinh-Map-APIGPSLocator-tren-DT-Android-361.aspx>