AndroidManifest.xml

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

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

**package="com.example.jevitha.a26\_1\_googlemap\_currentlocation"**>

*<!--*

*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"**>

*<!--*

*The API key for Google Maps-based APIs is defined as a string resource.*

*(See the file "res/values/google\_maps\_api.xml").*

*Note that the API key is linked to the encryption key used to sign the APK.*

*You need a different API key for each encryption key, including the release key that is used to*

*sign the APK for publishing.*

*You can define the keys for the debug and release targets in src/debug/ and src/release/.*

*-->*

<**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**>

Activity\_maps.xml

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

<**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=".MapsActivity"** />

MapsActivity.java

**package** com.example.jevitha.a26\_1\_googlemap\_currentlocation;

**import** android.Manifest;

**import** android.app.AlertDialog;

**import** android.content.Context;

**import** android.content.DialogInterface;

**import** android.content.pm.PackageManager;

**import** android.graphics.Color;

**import** android.location.Location;

**import** android.location.LocationListener;

**import** android.location.LocationManager;

**import** android.support.annotation.NonNull;

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

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

**import** android.os.Bundle;

**import** android.support.v4.content.ContextCompat;

**import** android.widget.Toast;

**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.BitmapDescriptorFactory;

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

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

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

**import** java.util.ArrayList;

**public class** MapsActivity **extends** FragmentActivity **implements**

OnMapReadyCallback,LocationListener {

**protected** LocationManager **locationManager**;

**private** GoogleMap **mMap**;

**private static final int *LOCATION\_PERMISSION\_REQUEST\_CODE*** = 1;

@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**);

**locationManager** = (LocationManager) getSystemService(Context.***LOCATION\_SERVICE***);

**if** (ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_FINE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED*** && ActivityCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_COARSE\_LOCATION***) != PackageManager.***PERMISSION\_GRANTED***) {

**return**;

}

**locationManager**.requestLocationUpdates(LocationManager.***GPS\_PROVIDER***, 1000, 0, **this**);

}

*/\*\**

*\* Manipulates the map once available.*

*\* This callback is triggered when the map is ready to be used.*

*\* This is where we can add markers or lines, add listeners or move the camera. In this case,*

*\* we just add a marker near Sydney, Australia.*

*\* If Google Play services is not installed on the device, the user will be prompted to install*

*\* it inside the SupportMapFragment. This method will only be triggered once the user has*

*\* installed Google Play services and returned to the app.*

*\*/*

@Override

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

**mMap** = googleMap;

Toast.*makeText*(**this**, **"OnMapReady"**, Toast.***LENGTH\_SHORT***).show();

LatLng home = **new** LatLng(11.0168, 76.9558);

**float** zoomLevel = (**float**) 5.0;

**mMap**.addMarker(**new** MarkerOptions().position(home).title(**"You are at here!!!"**));

**mMap**.moveCamera(CameraUpdateFactory.*newLatLngZoom*(home, zoomLevel));

}

@Override

**public void** onLocationChanged(Location location) {

LatLng home = **new** LatLng(location.getLatitude(), location.getLongitude());

**float** zoomLevel = (**float**) 5.0;

**mMap**.addMarker(**new** MarkerOptions().position(home).title(**"You are at here now!!!"**));

**mMap**.moveCamera(CameraUpdateFactory.*newLatLngZoom*(home, zoomLevel));

}

@Override

**public void** onStatusChanged(String s, **int** i, Bundle bundle) {

}

@Override

**public void** onProviderEnabled(String s) {

}

@Override

**public void** onProviderDisabled(String s) {

}

}