Practical No 32

X.1

- **Program**
- **XML File:**

  <resources>
  <string name="google_maps_key" templateMergeStrategy="preserve"
  translatable="false">YOUR_KEY_HERE</string>
  </resources>

- **JAVA File:**

  import android.graphics.Color;
  import android.os.AsyncTask;
  import android.support.v4.app.FragmentActivity;
  import android.os.Bundle;
  import android.util.Log;
  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.LatLng;
  import com.google.android.gms.maps.model.MarkerOptions; import
  com.google.android.gms.maps.model.PolylineOptions;import
  org.json.JSONObject;
  import java.io.BufferedReader; import
  java.io.IOException; import
  java.io.InputStream; import
  java.io.InputStreamReader;
  import java.net.HttpURLConnection;
  import java.net.URL;
  import java.util.ArrayList;
  import java.util.HashMap;
  import java.util.List;

```
public class MapsActivity extends FragmentActivity implements
OnMapReadyCallback {
private GoogleMap mMap;
ArrayList markerPoints= new ArrayList();
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_maps);
SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager()
.findFragmentById(R.id.map);
mapFragment.getMapAsync(this);
}
@Override
public void onMapReady(GoogleMap googleMap) {mMap
= googleMap;
LatLng sydney = new LatLng(-34, 151);
mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(sydney,
16));mMap.setOnMapClickListener(new
GoogleMap.OnMapClickListener() {
@Override
public void onMapClick(LatLng latLng) {if
(markerPoints.size() > 1) {
markerPoints.clear();
mMap.clear();
}
markerPoints.add(latLng);
MarkerOptions options = new MarkerOptions();0
options.position(latLng);
if (markerPoints.size() == 1) {
options.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFac
tory.HUE_GREEN));
} else if (markerPoints.size() == 2) {
```

```
options.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFac
tory.HUE_RED));
}
mMap.addMarker(options);
if (markerPoints.size() >= 2) {
LatLng origin = (LatLng) markerPoints.get(0);
LatLng dest = (LatLng) markerPoints.get(1); String
url = getDirectionsUrl(origin, dest);
DownloadTask downloadTask = new DownloadTask();
downloadTask.execute(url);
}
}
});
}
private class DownloadTask extends AsyncTask<String, Void, String>
{@Override
protected String doInBackground(String... url) {String
data = "";
try {
data = downloadUrl(url[0]);
} catch (Exception e) { Log.d("Background
Task", e.toString());
}
return data;
}
@Override
protected void onPostExecute(String result) {
super.onPostExecute(result);
ParserTask parserTask = new ParserTask();
parserTask.execute(result);
}
}
private class ParserTask extends AsyncTask<String, Integer,
List<List<HashMap<String, String>>>> {@Override
```

```
protected List<List<HashMap<String, String>>> doInBackground(String...
jsonData) {JSONObject
jObject;
List<List<HashMap<String, String>>> routes = null;try {
jObject = new JSONObject(jsonData[0]); DirectionsJSONParser
parser = new DirectionsJSONParser();routes =
parser.parse(jObject);
} catch (Exception e) {
e.printStackTrace();
return routes;
}
@Override
protected void onPostExecute(List<List<HashMap<String, String>>>
result) {
ArrayList points = null;
PolylineOptions lineOptions = null;
MarkerOptions markerOptions = new MarkerOptions();for
(int i = 0; i < result.size(); i++) {
points = new ArrayList(); lineOptions =
new PolylineOptions();
List<HashMap<String, String>> path = result.get(i);for
(int j = 0; j < path.size(); j++) {
HashMap<String, String> point = path.get(j); double lat =
Double.parseDouble(point.get("lat")); double lng =
Double.parseDouble(point.get("lng"));LatLng position =
new LatLng(lat, lng); points.add(position);
}
lineOptions.addAll(points);
lineOptions.width(12);
lineOptions.color(Color.RED);
lineOptions.geodesic(true);
}
mMap.addPolyline(lineOptions);
}
```

```
}
private String getDirectionsUrl(LatLng origin, LatLng dest) {
String str_origin = "origin=" + origin.latitude + "," + origin.longitude;String
str_dest = "destination=" + dest.latitude + "," + dest.longitude; String sensor
=
"sensor=false";
String mode = "mode=driving";
String parameters = str_origin + "&" + str_dest + "&" + sensor + "&" +
mode;String
output = "json";
String url = "https://maps.googleapis.com/maps/api/directions/" + output +
"?" + parameters;return url;
}
private String downloadUrl(String strUrl) throws IOException {String
data = "";
InputStream iStream = null;
HttpURLConnection urlConnection = null;try {
URL url = new URL(strUrl);
urlConnection = (HttpURLConnection) url.openConnection();
urlConnection.connect();
iStream = urlConnection.getInputStream();
BufferedReader br = new BufferedReader(new
InputStreamReader(iStream));StringBuffer sb =
new StringBuffer();
String line = "";
while ((line = br.readLine()) != null) {
sb.append(line);
}
data = sb.toString();
br.close();
} catch (Exception e) {
Log.d("Exception", e.toString());
} finally {
iStream.close();
```

```
urlConnection.disconnect();
}
return data;
}
}
```

- **JAVA File 2:**

```
import android.util.Log;
import com.google.android.gms.maps.model.LatLng;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject; import
java.util.ArrayList; import
java.util.HashMap; import
java.util.List;
public class DirectionsJSONParser {
public List<List<HashMap<String,String>>> parse(JSONObject jObject){
List<List<HashMap<String,
String>>> routes = new ArrayList<List<HashMap<String,String>>>() ;JSONArray
jRoutes = null;
JSONArray jLegs = null;
JSONArray jSteps = null;try {
jRoutes = jObject.getJSONArray("routes");for(int
i=0;i<jRoutes.length();i++){
jLegs = ( (JSONObject)jRoutes.get(i)).getJSONArray("legs");List
path = new ArrayList<HashMap<String, String>>(); for(int
j=0;j<jLegs.length();j++){
jSteps = ( (JSONObject)jLegs.get(j)).getJSONArray("steps");for(int
k=0;k<jSteps.length();k++){
String polyline = "";
polyline =
(String)((JSONObject)((JSONObject)jSteps.get(k)).get("polyline")).get("points");List list
=
```

```
decodePoly(polyline);
for(int l=0;l <list.size();l++){
HashMap<String, String> hm = new HashMap<String, String>();
hm.put("lat", Double.toString(((LatLng)list.get(l)).latitude) );
hm.put("lng", Double.toString(((LatLng)list.get(l)).longitude) );
path.add(hm);
}
}
routes.add(path);
}
}
} catch (JSONException e) {
e.printStackTrace();
}catch (Exception e){
}
return routes;
}
private List decodePoly(String encoded) {List
poly = new ArrayList();
int index = 0, len = encoded.length();int lat
= 0, lng = 0;
while (index < len) {
int b, shift = 0, result = 0;do {
b = encoded.charAt(index++) - 63;result
|= (b & 0x1f) << shift;
shift += 5;
} while (b >= 0x20);
int dlat = ((result & 1) != 0 ? ~(result >> 1) : (result >> 1));lat +=
dlat;
shift = 0;
result = 0;do {
b = encoded.charAt(index++) - 63;result
|= (b & 0x1f) << shift;
shift += 5;
```

```
} while (b >= 0x20);
int dlng = ((result & 1) != 0 ? ~(result >> 1) : (result >> 1));lng +=
dlng;
LatLng p = new LatLng((((double) lat / 1E5)),
(((double) lng / 1E5)));
poly.add(p);
}
return poly;
}
}
```

- **Manifest File:**

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.journaldev.maproutebetweenmarkers">
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<application android:allowBackup="true"
android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
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>
```

- **Output:**