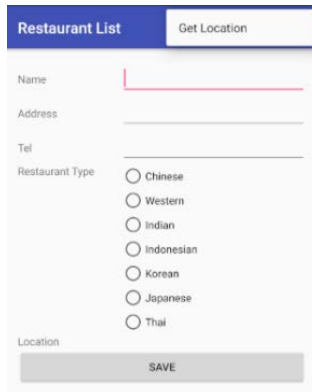


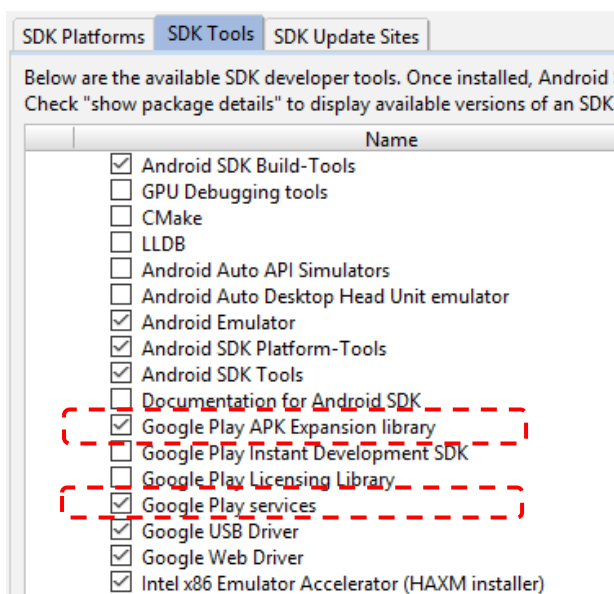
Practical 6: GPS Location & Map View

At the end of this session, you will learn how to get a GPS coordinate for a restaurant and display the location on Google map with a marker

Part I – Getting GPS Location

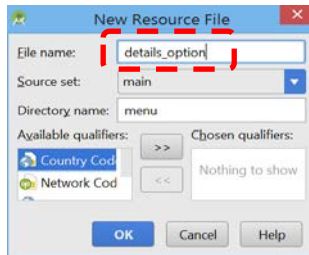


1. It might be good if the Restaurant location can be shown on a map. In this exercise, the GPS coordinates of the restaurant will be saved in the database for later use. In order to incorporate Google Map into Restaurant List app, the following SDK Tools will be needed. At top menu bar, **select Tools > Android > SDK Manager** and check if these tools are installed or updated

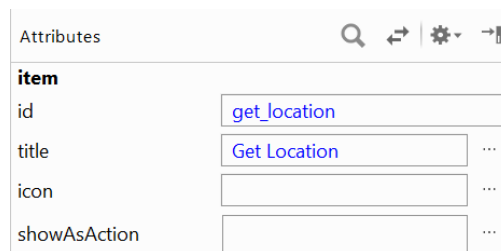
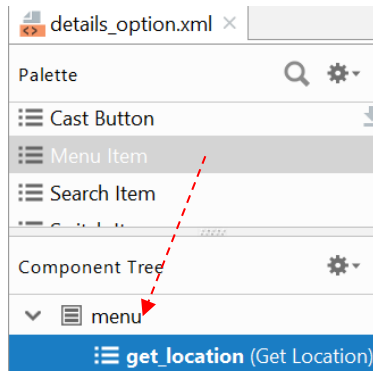


2. Create a new project with the following information
 - **Application Name** : *Restaurant List*
 - **Company Domain** : *sp.com*
 - **Project Location** : *C:\MAD\AndroidStudioProjects\Lab6a or D:\MAD\AndroidStudioProjects\Lab6a*
 - **Minimum SDK** : *Android 5.0 (Lollipop)*
 - **Empty Activity name** : *RestaurantList*
 - **Layout name** : *main*
3. **Close RestaurantList.java and main.xml files** in the **Editor** pane
4. Open your **Windows File Explorer** and navigate to your **Android Studio** workspace where all you projects are created.
5. Copy *AndroidManifest.xml* file, **java** and **res** folders from **Lab5b\app\src\main** project folder and paste into **Lab6a\app\src\main** folder to overwrite the existing file and folder

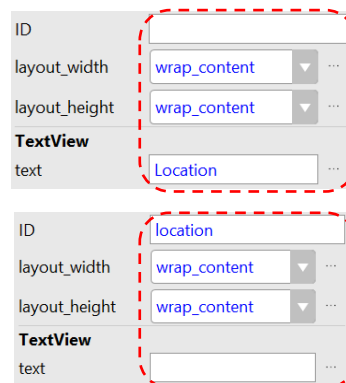
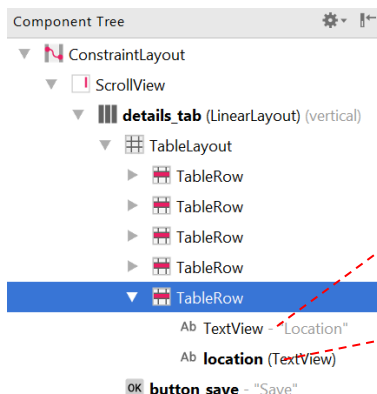
6. At Android Studio Project Pane, right click on **res > menu** folder, select **New > Menu resource file** to create a new menu named **details_option**



7. At Design Editor, drag and drop a Menu Item widget into **menu** widget under Component Tree. Update the item's attributes with the following properties



8. With an option 'Get Location' added, the next widget you need is to show the GPS coordinates on UI display.
9. Open the **detail_form** layout from **res > layout**. Drag and drop one **TableRow** and two **TextView** widgets into the **TableLayout** widget as shown. Update the attributes of the two TextView widgets with the following content



10. Due to the additional GPS location, these information (latitude and longitude) will need be saved as part of the record into the data model of **restaurants_table** in **RestaurantHelper**.
11. Open the **RestaurantHelper.java** to update the **restaurants_table** with two extra fields (latitude and longitude) to store the Restaurant Location information and the location processing methods in the helper

```
1. package com.sp.restaurantlist;
2.
3. import android.content.ContentValues;
4. import android.content.Context;
5. import android.database.Cursor;
6. import android.database.sqlite.SQLiteDatabase;
7. import android.database.sqlite.SQLiteOpenHelper;
8.
9. class RestaurantHelper extends SQLiteOpenHelper {
10.     private static final String DATABASE_NAME = "restaurantlist.db";
11.     private static final int SCHEMA_VERSION = 1;
```

```
12.
13.     public RestaurantHelper(Context context) {
14.         super(context, DATABASE_NAME, null, SCHEMA_VERSION);
15.     }
16.
17.     @Override
18.     public void onCreate(SQLiteDatabase db) {
19.         // Will be called once when the database is not created
20.         db.execSQL("CREATE TABLE restaurants_table (_id INTEGER PRIMARY KEY
AUTOINCREMENT, restaurantName TEXT, restaurantAddress TEXT, restaurantTel TEXT,
restaurantType TEXT, lat REAL, lon REAL);");
21.     }
22.
23.     @Override
24.     public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
25.         // Will not be called until 2nd schema i.e. when SCHEMA_VERSION
26.         // increases
27.     }
28.
29.     /* Read all records from restaurants_table */
30.     public Cursor getAll(String orderBy) {
31.         return (getReadableDatabase().rawQuery("SELECT _id, restaurantName,
restaurantAddress, restaurantTel, restaurantType, lat, lon FROM restaurants_table
ORDER BY " + orderBy, null));
32.     }
33.
34.     public Cursor getById(String id) {
35.         String[] args = {id};
36.
37.         return (getReadableDatabase().rawQuery("SELECT _id, restaurantName,
restaurantAddress, restaurantTel, restaurantType, lat, lon FROM restaurants_table
WHERE _ID=?", args));
38.     }
39.
40.     /* Write a record into restaurants_table */
41.     public void insert(String restaurantName, String restaurantAddress, String
restaurantTel, String restaurantType, double lat, double lon) {
42.         ContentValues cv = new ContentValues();
43.
44.         cv.put("restaurantName", restaurantName);
45.         cv.put("restaurantAddress", restaurantAddress);
46.         cv.put("restaurantTel", restaurantTel);
47.         cv.put("restaurantType", restaurantType);
48.         cv.put("lat", lat);
49.         cv.put("lon", lon);
50.
51.         getWritableDatabase().insert("restaurants_table", "restaurantName", cv);
52.     }
53.
54.     public void update(String id, String restaurantName, String restaurantAddress,
String restaurantTel, String restaurantType, double lat, double lon) {
55.         ContentValues cv = new ContentValues();
56.         String[] args = {id};
57.         cv.put("restaurantName", restaurantName);
58.         cv.put("restaurantAddress", restaurantAddress);
59.         cv.put("restaurantTel", restaurantTel);
60.         cv.put("restaurantType", restaurantType);
61.         cv.put("lat", lat);
62.         cv.put("lon", lon);
63.
64.         getWritableDatabase().update("restaurants_table", cv, "_ID=?", args);
65.     }
66.
67.     public String getID(Cursor c) {
68.         return (c.getString(0));
69.     }
70.
71.     public String getRestaurantName(Cursor c) {
72.         return (c.getString(1));
73.     }
74.
75.     public String getRestaurantAddress(Cursor c) {
```

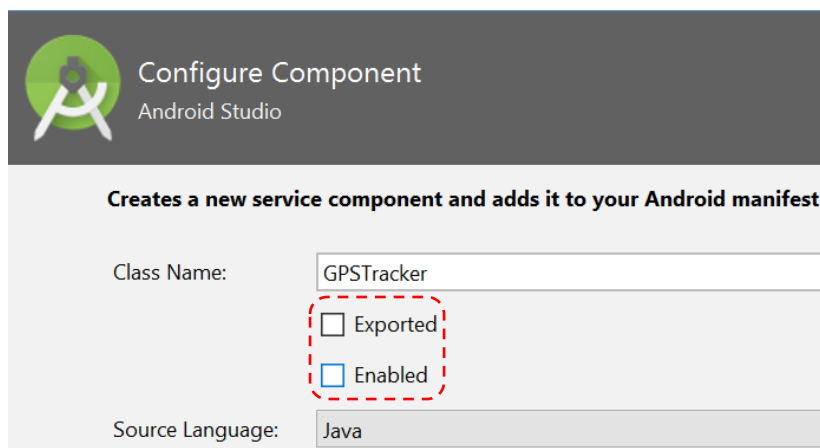
```
76.         return (c.getString(2));
77.     }
78.
79.     public String getRestaurantTel(Cursor c) {
80.         return (c.getString(3));
81.     }
82.
83.     public String getRestaurantType(Cursor c) {
84.         return (c.getString(4));
85.     }
86.
87.     public double getLatitude(Cursor c) {
88.         return (c.getDouble(5));
89.     }
90.
91.     public double getLongitude(Cursor c) {
92.         return (c.getDouble(6));
93.     }
94. }
```

12. You have so far updated the Detail Form UI with an extra **TextView** widget for displaying latitude & longitude information, and update the SQLite Database Helper by introducing two extra fields ("lat" and "lon") to allow you to save the location latitude & longitude as part of a restaurant record in *restaurants_table*.
13. The next step is to create a Service to tell phone to get the GPS information in the background. **GPSTracker** is a subclass of **Service** created to access phone GPS and returns GPS coordinates values whenever *getLatitude()* or *getLongitude()* method is called.
14. ***IMPORTANT**
Due to security feature of Android, when an app requires to access phone GPS, **Permissions** must be set in the *AndroidManifest* file. Open and edit *AndroidManifest.xml* as shown.

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

    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
    <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"/>
```

15. To create the *GPSTracker* service, right click on the **java > com.sp.restaurantlist** folder and select **New > Service > Service**. Enter the file name as 'GPSTracker'. **Uncheck both Exported and Enabled options.**



16. Update the file with the following content

```
1. package com.sp.restaurantlist;
```

```
2.
3. import android.Manifest;
4. import android.app.Service;
5. import android.content.Context;
6. import android.content.DialogInterface;
7. import android.content.Intent;
8. import android.content.pm.PackageManager;
9. import android.location.Location;
10. import android.location.LocationListener;
11. import android.location.LocationManager;
12. import android.net.Uri;
13. import android.os.Bundle;
14. import android.os.IBinder;
15. import android.provider.Settings;
16. import android.support.v4.app.ActivityCompat;
17. import android.support.v7.app.AlertDialog;
18.
19. public class GPSTracker extends Service implements LocationListener {
20.     private Context mContext = null;
21.     boolean isGPSEnabled = false; // flag for GPS status
22.     boolean isNetworkEnabled = false; // flag for network status
23.     boolean canGetLocation = false; // flag for GPS status
24.
25.     Location location; // location
26.     double latitude; // latitude
27.     double longitude; // longitude
28.
29.     // The min distance to change Updates in meters
30.     private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 10; // 10 meters
31.     // The minimum time between updates in milliseconds
32.     private static final long MIN_TIME_BW_UPDATES = 1000 * 60 * 1; // 1 minute
33.     // Declaring a Location Manager
34.     protected LocationManager locationManager;
35.
36.     public GPSTracker() {
37.         checkGPSPermissions();
38.     }
39.
40.     public GPSTracker(Context context) {
41.         this.mContext = context;
42.         checkGPSPermissions();
43.     }
44.
45.     public Location getLocation() {
46.         this.canGetLocation = false;
47.         try {
48.             locationManager = (LocationManager)
mContext.getSystemService(LOCATION_SERVICE);
49.             // getting GPS status
50.             isGPSEnabled =
locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER);
51.             // getting network status
52.             isNetworkEnabled =
locationManager.isProviderEnabled(LocationManager.NETWORK_PROVIDER);
53.             if (!isGPSEnabled && !isNetworkEnabled) {
54.                 // no network provider is enabled
55.                 // prompt user to enable location services
56.                 showEnableLocationAlert();
57.             } else {
58.                 this.canGetLocation = true;
59.                 if (isNetworkEnabled) {
60.                     //Permission granted
61.                     locationManager.requestLocationUpdates(
62.                         LocationManager.NETWORK_PROVIDER,
63.                         MIN_TIME_BW_UPDATES,
64.                         MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
65.                     if (locationManager != null) {
66.                         location =
locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
67.                         if (location != null) {
68.                             latitude = location.getLatitude();
69.                             longitude = location.getLongitude();
```

```
70.         }
71.     }
72. }
73. // if GPS Enabled get lat/long using GPS Services
74. if (isGPSEnabled) {
75.     if (location == null) {
76.         locationManager.requestLocationUpdates(
77.             locationManager.GPS_PROVIDER,
78.             MIN_TIME_BW_UPDATES,
79.             MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
80.         if (locationManager != null) {
81.             location =
locationManager.getLastKnownLocation(locationManager.GPS_PROVIDER);
82.             if (location != null) {
83.                 latitude = location.getLatitude();
84.                 longitude = location.getLongitude();
85.             }
86.         }
87.     }
88. }
89. }
90. } catch (SecurityException e) {
91.     e.printStackTrace();
92. }
93. return location;
94. }
95. /**
96.  * Function to get latitude
97.  */
98. public double getLatitude() {
99.     if (location != null) {
100.         latitude = location.getLatitude();
101.     }
102.     return latitude;
103. }
104.
105. /**
106.  * Function to get longitude
107.  */
108. public double getLongitude() {
109.     if (location != null) {
110.         longitude = location.getLongitude();
111.     }
112.     return longitude;
113. }
114.
115. /**
116.  * Stop using GPS listener
117.  * Calling this function will stop using GPS in your app
118.  */
119.
120. public void stopUsingGPS() {
121.     if (locationManager != null) {
122.         locationManager.removeUpdates(GPSTracker.this);
123.     }
124. }
125.
126. /**
127.  * Function to check GPS/wifi enabled
128.  *
129.  * @return Boolean
130.  */
131. public boolean canGetLocation() {
132.     checkGPSPermissions();
133.     return canGetLocation;
134. }
135.
136. // check for location permission
137. public void checkGPSPermissions() {
138.     int permissionState1 = ActivityCompat.checkSelfPermission(mContext,
139.         android.Manifest.permission.ACCESS_FINE_LOCATION);
140.     int permissionState2 = ActivityCompat.checkSelfPermission(mContext,
```

```

141. Manifest.permission.ACCESS_COARSE_LOCATION);
142. if (permissionStatel == PackageManager.PERMISSION_GRANTED &&
    permissionState2 == PackageManager.PERMISSION_GRANTED) {
143.     // Permission granted, get GPS location
144.     getLocation();
145. }
146. else {
147.     //Prompt user to enable location permission
148.     showEnablePermissionAlert();
149. }
150. }
151.
152. /**
153.  * Function to show settings alert dialog On pressing Settings button will
    launch Settings Options
154.  */
155. public void showEnablePermissionAlert() {
156.     AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);
157.     // Setting Dialog Title
158.     alertDialog.setTitle("Location Permission Settings");
159.     // Setting Dialog Message
160.     alertDialog.setMessage("Restaurant List Location Permission is not
    enabled. Do you want to go to settings menu?");
161.     // On pressing Settings button
162.     alertDialog.setPositiveButton("Settings", new
    DialogInterface.OnClickListener() {
163.         public void onClick(DialogInterface dialog, int which) {
164.             Intent intent = new Intent();
165.             // Goto Application Setting
166.             intent.setAction(
                Settings.ACTION_APPLICATION_DETAILS_SETTINGS);
167.             Uri uri = Uri.fromParts("package",
                BuildConfig.APPLICATION_ID, null);
168.             intent.setData(uri);
169.             intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
170.             mContext.startActivity(intent);
171.         }
172.     });
173. }
174. }
175.
176. // on pressing cancel button
177. alertDialog.setNegativeButton("Cancel", new
    DialogInterface.OnClickListener() {
178.         public void onClick(DialogInterface dialog, int which) {
179.             dialog.cancel();
180.         }
181.     });
182. // Showing Alert Message
183. alertDialog.show();
184. }
185.
186. public void showEnableLocationAlert() {
187.     AlertDialog.Builder alertDialogBuilder = new
    AlertDialog.Builder(mContext);
188.     // Setting Dialog Title
189.     alertDialogBuilder.setTitle("Location Service Settings");
190.     alertDialogBuilder.setMessage("Location service is disabled in your
    device. Would you like to enable it?")
191.     .setCancelable(false)
192.     .setPositiveButton("Goto Settings Page To Enable Location
    Service",
        new DialogInterface.OnClickListener() {
193.             public void onClick(DialogInterface dialog, int id)
194.             {
195.                 //Goto location service setting
196.                 Intent callGPSSettingIntent = new Intent(
197.                     android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);
198.                 mContext.startActivity(callGPSSettingIntent);
199.             }
200.         });
201.     alertDialogBuilder.setNegativeButton("Cancel",
202.         new DialogInterface.OnClickListener() {

```

```
203.         public void onClick(DialogInterface dialog, int id) {
204.             dialog.cancel();
205.         }
206.     });
207.     AlertDialog alert = alertDialogBuilder.create();
208.     alert.show();
209. }
210.
211. @Override
212. public void onLocationChanged(Location location) {
213.     getLocation();
214. }
215.
216. @Override
217. public void onProviderDisabled(String provider) {
218. }
219.
220. @Override
221. public void onProviderEnabled(String provider) {
222. }
223.
224. @Override
225. public void onStatusChanged(String provider, int status, Bundle extras) {
226. }
227.
228. @Override
229. public IBinder onBind(Intent intent) {
230.     // TODO: Return the communication channel to the service.
231.     throw new UnsupportedOperationException("Not yet implemented");
232. }
233.
234. }
```

17. After the GPSTracker has been created, add it into *DetailForm.java* file. Whenever the **Get Location** menu item is selected, the GPSTracker (**Service** subclass) will get GPS location information and display into the new *TextView* introduced in the 'Details' form UI view
18. Open the *DetailForm.java* file and update with the content

```
1. package com.sp.restaurantlist;
2.
3. import android.database.Cursor;
4. import android.support.v7.app.AppCompatActivity;
5. import android.os.Bundle;
6. import android.view.Menu;
7. import android.view.MenuInflater;
8. import android.view.MenuItem;
9. import android.view.View;
10. import android.widget.Button;
11. import android.widget.EditText;
12. import android.widget.RadioGroup;
13. import android.widget.TextView;
14. import android.widget.Toast;
15.
16. public class DetailForm extends AppCompatActivity {
17.     private EditText restaurantName;
18.     private EditText restaurantAddress;
19.     private EditText restaurantTel;
20.     private RadioGroup restaurantTypes;
21.     private Button buttonSave;
22.
23.     private RestaurantHelper helper = null;
24.     private String restaurantID = "";
25.     private TextView location = null;
26.     private GPSTracker gpsTracker;
27.     private double latitude = 0.0d;
28.     private double longitude = 0.0d;
29.
30.
31.     @Override
```


SINGAPORE POLYTECHNIC
School of Electrical & Electronic Engineering

```
32.     protected void onCreate(Bundle savedInstanceState) {
33.         super.onCreate(savedInstanceState);
34.         setContentView(R.layout.detail_form);
35.         restaurantName = (EditText) findViewById(R.id.restaurant_name);
36.         restaurantAddress = (EditText) findViewById(R.id.restaurant_address);
37.         restaurantTel = (EditText) findViewById(R.id.restaurant_tel);
38.         restaurantTypes = (RadioGroup) findViewById(R.id.restaurant_types);
39.
40.         buttonSave = (Button) findViewById(R.id.button_save);
41.         buttonSave.setOnClickListener(onSave);
42.
43.         helper = new RestaurantHelper(this);
44.
45.         location = (TextView) findViewById(R.id.location);
46.         gpsTracker = new GPSTracker(DetailForm.this);
47.
48.         restaurantID = getIntent().getStringExtra("ID");
49.         if (restaurantID != null) {
50.             load();
51.         }
52.
53.     }
54.
55.     private void load() {
56.         Cursor c = helper.getIdByRestaurantID(restaurantID);
57.         c.moveToFirst();
58.         restaurantName.setText(helper.getRestaurantName(c));
59.         restaurantAddress.setText(helper.getRestaurantAddress(c));
60.         restaurantTel.setText(helper.getRestaurantTel(c));
61.
62.         if (helper.getRestaurantType(c).equals("Chinese")) {
63.             restaurantTypes.check(R.id.chinese);
64.         } else if (helper.getRestaurantType(c).equals("Western")) {
65.             restaurantTypes.check(R.id.western);
66.         } else if (helper.getRestaurantType(c).equals("Indian")) {
67.             restaurantTypes.check(R.id.indian);
68.         } else if (helper.getRestaurantType(c).equals("Indonesian")) {
69.             restaurantTypes.check(R.id.indonesian);
70.         } else if (helper.getRestaurantType(c).equals("Korean")) {
71.             restaurantTypes.check(R.id.korean);
72.         } else if (helper.getRestaurantType(c).equals("Japanese")) {
73.             restaurantTypes.check(R.id.japanese);
74.         } else {
75.             restaurantTypes.check(R.id.thai);
76.         }
77.         latitude = helper.getLatitude(c);
78.         longitude = helper.getLongitude(c);
79.         location.setText(String.valueOf(latitude) + ", " + String.valueOf(longitude));
80.     }
81.
82.     @Override
83.     protected void onDestroy() {
84.         super.onDestroy();
85.         helper.close();
86.         gpsTracker.stopUsingGPS();
87.     }
88.
89.     @Override
90.     public boolean onCreateOptionsMenu(Menu menu) {
91.         new MenuInflater(this).inflate(R.menu.details_option, menu);
92.         return super.onCreateOptionsMenu(menu);
93.     }
94.
95.     @Override
96.     public boolean onOptionsItemSelected(MenuItem item) {
97.         if (item.getItemId() == R.id.get_location) {
98.             if (gpsTracker.canGetLocation()) {
99.                 latitude = gpsTracker.getLatitude();
100.                 longitude = gpsTracker.getLongitude();
101.                 location.setText(String.valueOf(latitude) + ", " +
String.valueOf(longitude));
102.                 // \n is for new line
```

```
103.                Toast.makeText(getApplicationContext(), "Your Location is -
                \nLat: " + latitude + "\nLong: " + longitude, Toast.LENGTH_LONG).show();
104.            }
105.            return (true);
106.        }
107.        return super.onOptionsItemSelected(item);
108.    }
109.
110.
111.    View.OnClickListener onSave = new View.OnClickListener() {
112.        @Override
113.        public void onClick(View v) {
114.            // To read date from EditText
115.            String nameStr = restaurantName.getText().toString();
116.            String addrStr = restaurantAddress.getText().toString();
117.            String telStr = restaurantTel.getText().toString();
118.            String restType = "";
119.            //To read selection of restaurantTypes RadioGroup
120.            switch (restaurantTypes.getCheckedRadioButtonId()) {
121.                case R.id.chinese:
122.                    restType = "Chinese";
123.                    break;
124.                case R.id.western:
125.                    restType = "Western";
126.                    break;
127.                case R.id.indian:
128.                    restType = "Indian";
129.                    break;
130.                case R.id.indonesian:
131.                    restType = "Indonesian";
132.                    break;
133.                case R.id.korean:
134.                    restType = "Korean";
135.                    break;
136.                case R.id.japanese:
137.                    restType = "Japanese";
138.                    break;
139.                case R.id.thai:
140.                    restType = "Thai";
141.                    break;
142.            }
143.
144.            if (restaurantID == null) {
145.                helper.insert(nameStr, addrStr, telStr, restType, latitude,
                longitude);
146.            } else {
147.                helper.update(restaurantID, nameStr, addrStr, telStr, restType,
                latitude, longitude);
148.            }
149.            //To close current Activity class and exit
150.            finish();
151.        }
152.    };
153. }
```

19. Check that AndroidManifest.xml has the setting as shown.

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3.     package="com.sp.restaurantlist">
4.
5.     <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
6.     <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
7.     <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
8.     <uses-permission android:name="android.permission.INTERNET" />
9.
10.    <application
11.        android:allowBackup="true"
12.        android:icon="@mipmap/ic_launcher"
13.        android:label="@string/app_name"
14.        android:roundIcon="@mipmap/ic_launcher_round"
15.        android:supportRtl="true"
16.        android:theme="@style/AppTheme">
17.        <activity android:name=".RestaurantList">
18.            <intent-filter>
19.                <action android:name="android.intent.action.MAIN" />
20.
21.                <category android:name="android.intent.category.LAUNCHER" />
22.            </intent-filter>
23.        </activity>
24.        <activity android:name=".DetailForm" />
25.        <activity android:name=".EditPreferences" />
26.        <activity android:name=".AlarmActivity" />
27.
28.        <receiver
29.            android:name=".OnBootReceiver"
30.            android:enabled="false"
31.            android:exported="false">
32.            <intent-filter>
33.                <action android:name="android.intent.action.BOOT_COMPLETED" />
34.            </intent-filter>
35.        </receiver>
36.        <receiver
37.            android:name=".OnAlarmReceiver"
38.            android:enabled="true"
39.            android:exported="false" />
40.
41.        <service
42.            android:name=".GPSTracker"
43.            android:enabled="false"
44.            android:exported="false" />
45.    </application>
46.
47. </manifest>
```

20. To test *Lab6a* project you will need to make use of actual mobile phone

[Note Due to the change of the *restaurants_table* data structure, you need to **uninstall the Restaurant List app in the phone before running**. Otherwise, you will encounter fatal error]

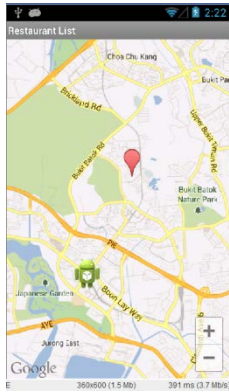
21. At 'Detail' form, click the **MENU** button and select **Get Location** option. A **Toast** messages **GPS Coordinates Found** will be shown and displayed.

[Note Make sure that the phone GPS is enabled and there is at least WiFi connection]

22. Show the result to your lecturer when you have completed this section

Lecturer Signature : _____

Part II – Using Android Google Map



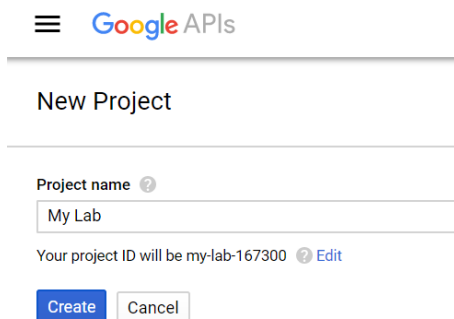
23. With the GPS coordinates for the restaurant, it might be useful to show the location on Google map.
24. Create a new project with the following information
 - **Application Name** : *Restaurant List*
 - **Company Domain** : *sp.com*
 - **Project Location** : *C:\MAD\AndroidStudioProjects\Lab6b or D:\MAD\AndroidStudioProjects\Lab6b*
 - **Minimum SDK** : *Android 5.0 (Lollipop)*
 - **Empty Activity name** : *RestaurantList*
 - **Layout name** : *main*
25. Close *RestaurantList.java* and *main.xml* files in the **Editor** pane
26. Open your **Windows File Explorer** and navigate to your **Android Studio** workspace where all you projects are created.
27. Copy *AndroidManifest.xml* file, **java** and **res** folders from **Lab6a\app\src\main** project folder and paste into **Lab6b\app\src\main** folder to overwrite the existing file and folder to Windows Explorer, copy the following files and folders from *Lab6a* project and save into *Lab6b* project

Creating Android Google Map Key

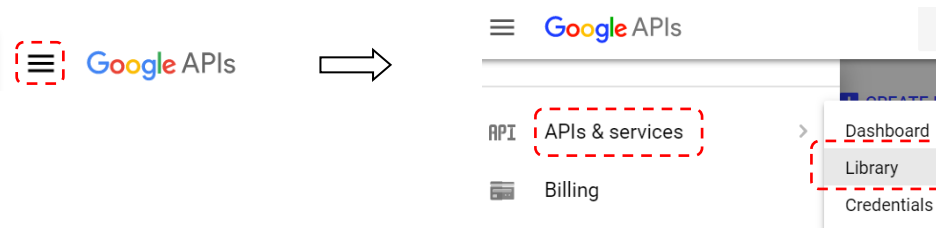
28. To use the Android Google Map v2 services, you will need to register for an API key at the following site
<https://console.developers.google.com/cloud-resource-manager>

[Note: You need a Gmail account]

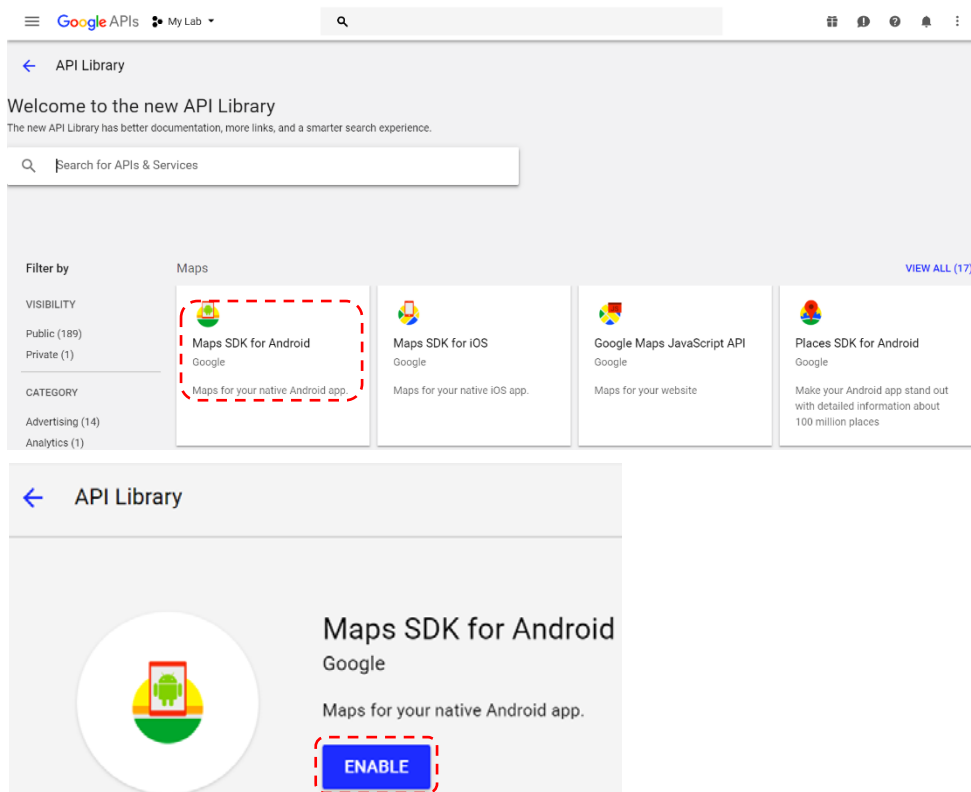
29. After you have login to your Gmail account, click on the **“CREATE PROJECT”** to create a project with name **‘My Lab’**



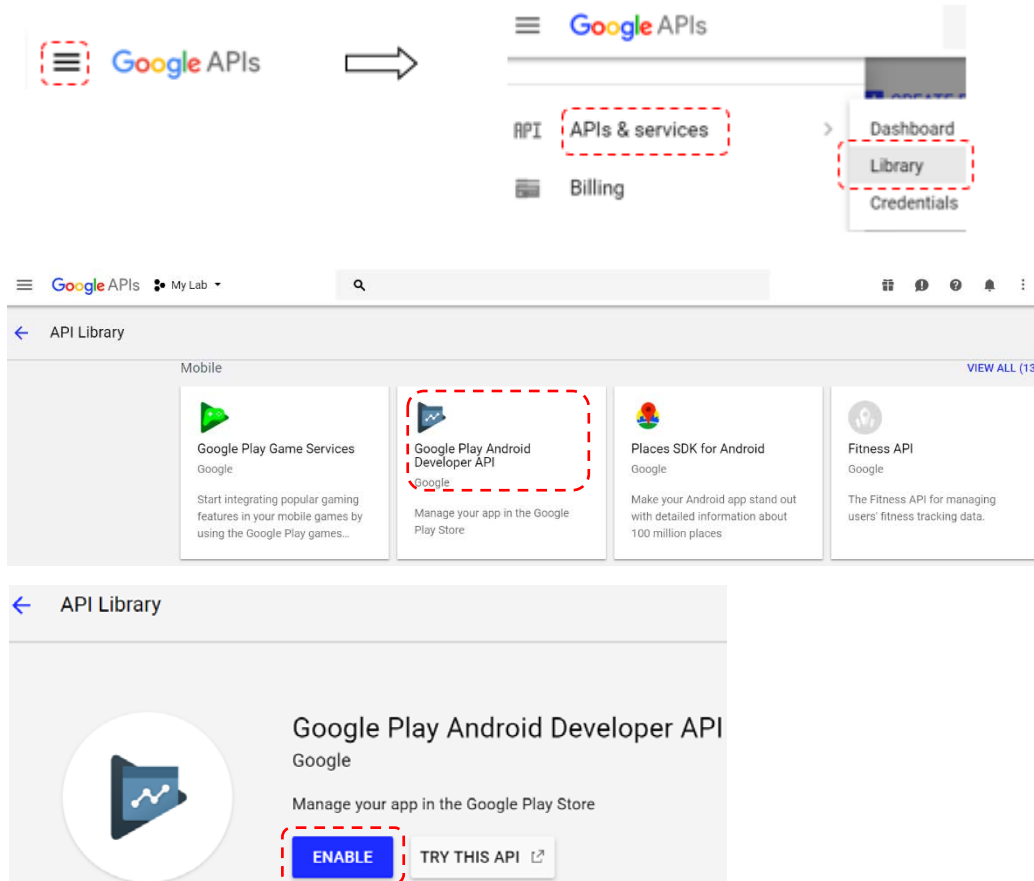
30. After creating My Lab project, at Google Developer Console, click on the **Products & services** menu and select **APIs & services > Library** option



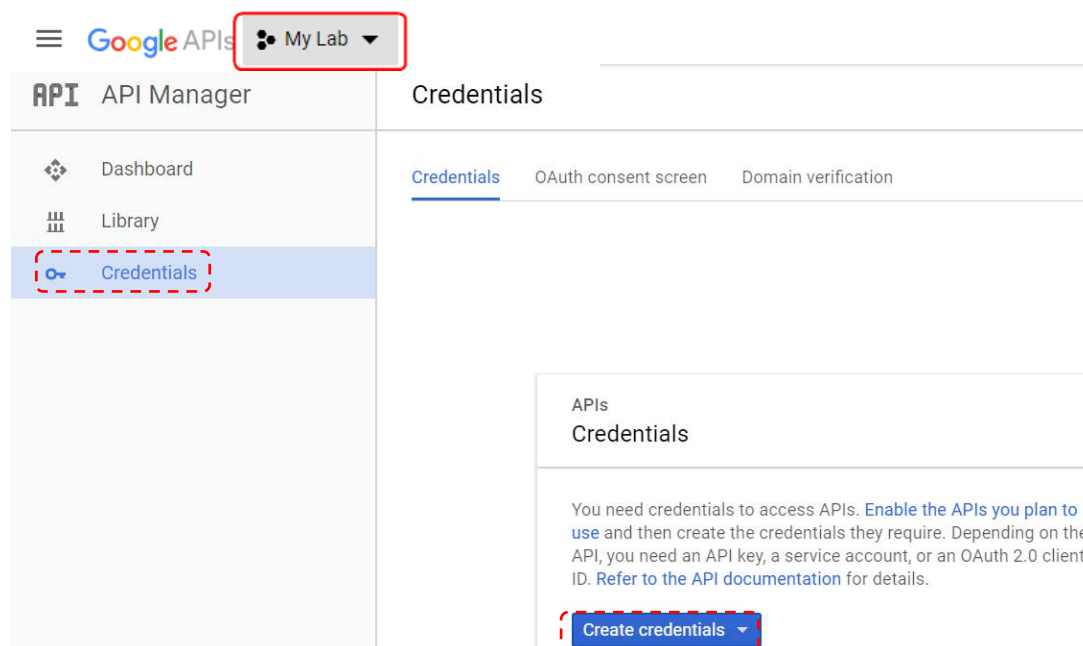
31. Select **Maps SDK for Android** under **Maps** category. Click on the **ENABLE** button to enable the **Map API** for **My Lab** project



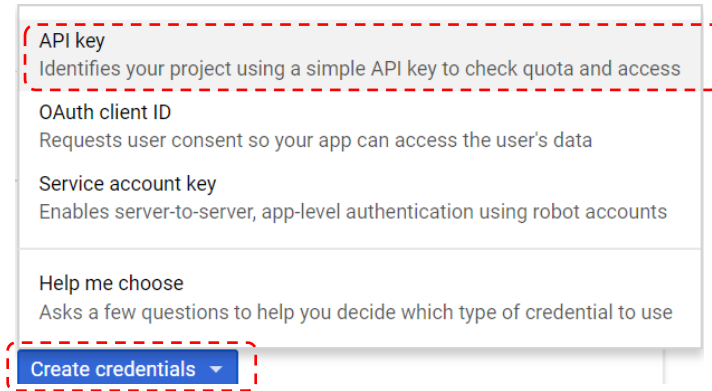
32. Go back to APL Library, under Mobile category (you may need to scroll your browser downward), select Google Play Android Developer API. Click on the **ENABLE** button to enable the **Map API** for **My Lab** project



33. After enabling the necessary APIs, at Google developers Console pane, click **Credentials** option. You need to create an Android **API key** for your Restaurant List app to access Google Map. Make sure that “My Lab” is selected. Click “Create credentials”.



34. Select the **API Key** option under **Create credentials** list



35. At the pop-up screen, you will see the API Key created for My Lab project. Click on the **CLOSE**

API key created

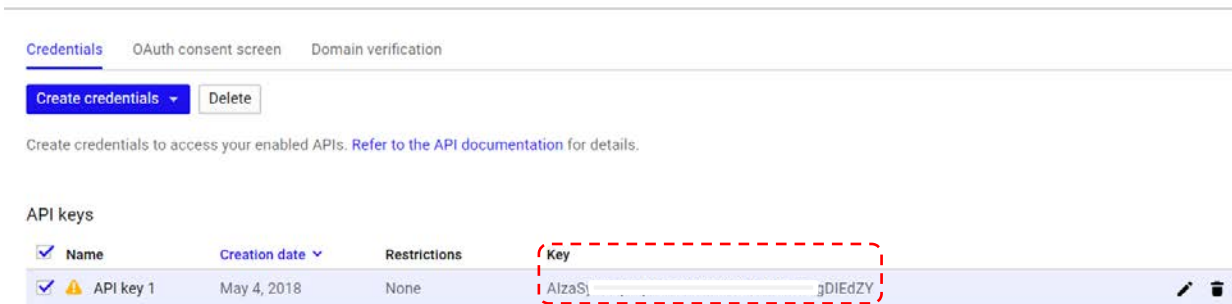
Use this key in your application by passing it with the `key=API_KEY` parameter.

Your API key
AIzaSyC8BMY2uI1gJGq6IYAI6JWNWfpMmXoKLU

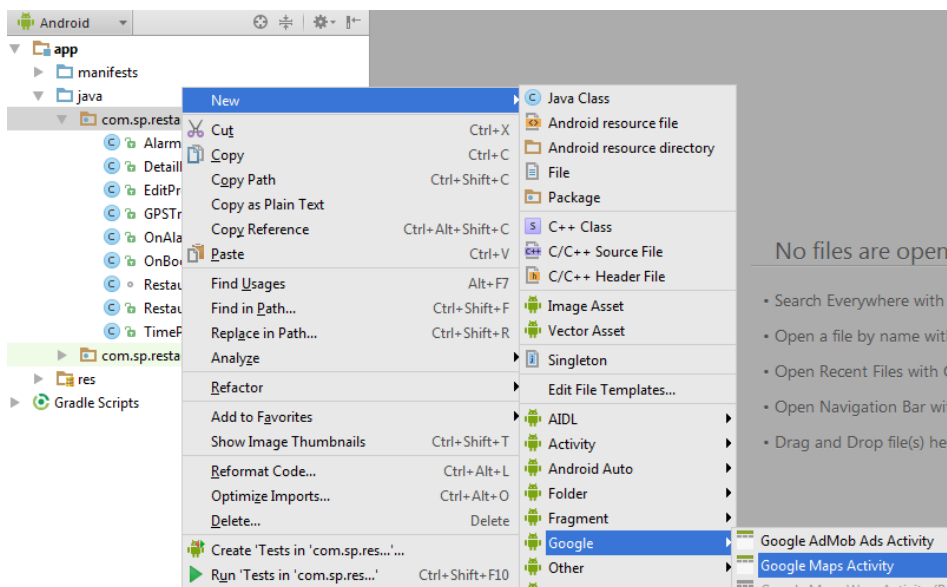
⚠ Restrict your key to prevent unauthorized use in production.

CLOSE **RESTRICT KEY**

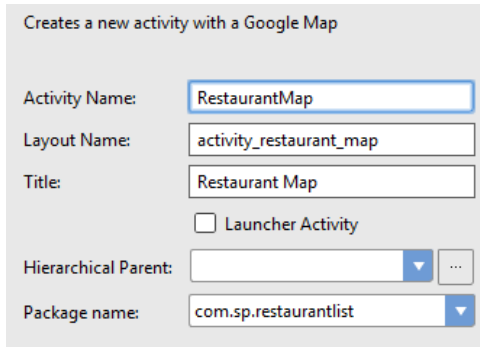
Credentials



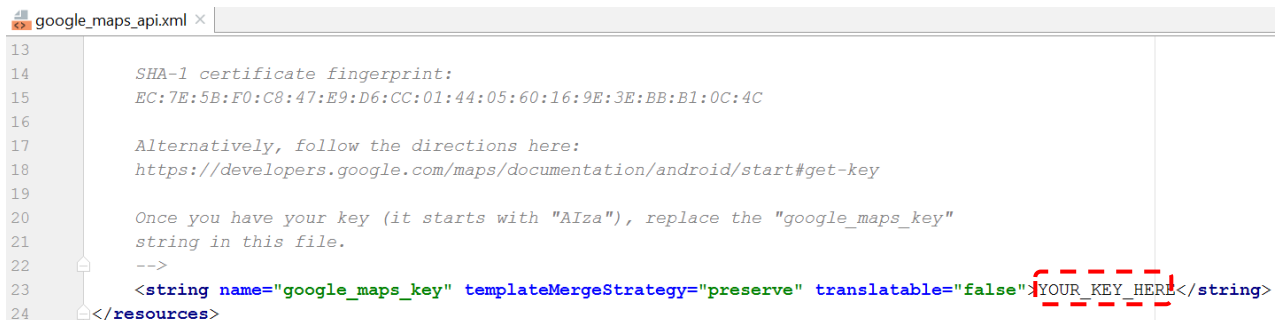
36. At Android Studio, right click on **com.sp.restaurant** folder. Select **New > Google > Google Maps Activity**



37. Enter the activity's name as **RestaurantMap** and the rest of information

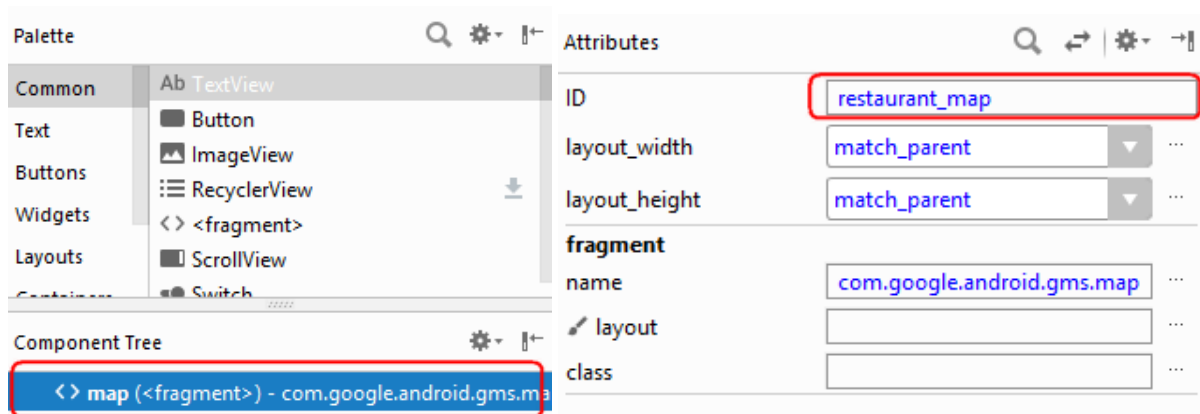


38. A **google_maps_api.xml** will be created. Copy the Android API Key (**STEP 35**) generated and **overwrite YOUR_KEY_HERE** text in the file



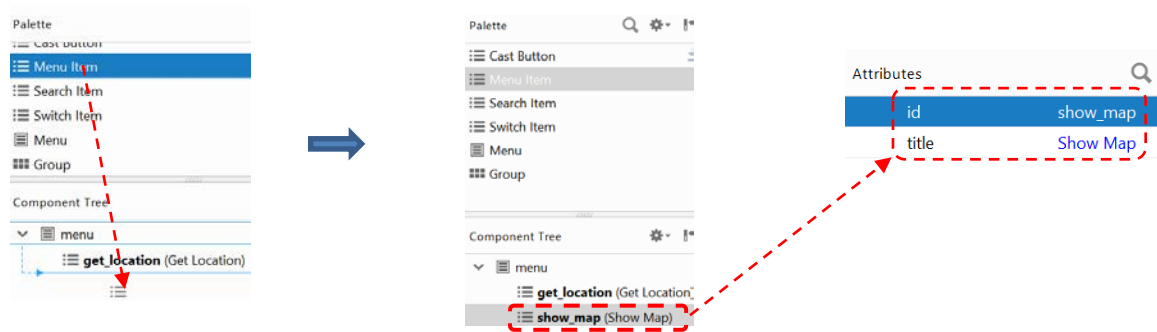
```
13
14     SHA-1 certificate fingerprint:
15     EC: 7E:5B:F0:C8:47:E9:D6:CC:01:44:05:60:16:9E:3E:BB:B1:0C:4C
16
17     Alternatively, follow the directions here:
18     https://developers.google.com/maps/documentation/android/start#get-key
19
20     Once you have your key (it starts with "AIza"), replace the "google_maps_key"
21     string in this file.
22     -->
23     <string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">YOUR_KEY_HERE</string>
24 </resources>
```

39. At **res/layout** folder, double click on **activity_restaurant_map.xml** to open the UI view file for Google Map. In the “Component Tree” select “map” and in the “Attributes” change “ID” to “restaurant_map”



40. Download **marker.png** and **marker_me.png** image icons from Blackboard under **Learning Resources > Practicals > Image Icons** folder and save to **res/drawable** folder.

41. Open the *details_option.xml* file in **res/menu** folder to add in the new **Show Map** option item to the MENU. Update the item **id** to **"show_map"** and **title** to **"Show Map"** widget properties as shown



42. Update new Java class *RestaurantMap.java*, a subclass of **FragmentActivity**.

```

1.  package com.sp.restaurantlist;
2.
3.  import android.os.Bundle;
4.  import android.support.v4.app.FragmentActivity;
5.
6.  import com.google.android.gms.maps.CameraUpdateFactory;
7.  import com.google.android.gms.maps.GoogleMap;
8.  import com.google.android.gms.maps.OnMapReadyCallback;
9.  import com.google.android.gms.maps.SupportMapFragment;
10. import com.google.android.gms.maps.model.BitmapDescriptorFactory;
11. import com.google.android.gms.maps.model.LatLng;
12. import com.google.android.gms.maps.model.Marker;
13. import com.google.android.gms.maps.model.MarkerOptions;
14.
15. public class RestaurantMap extends FragmentActivity implements OnMapReadyCallback {
16.     private GoogleMap mMap;
17.
18.     private double lat;
19.     private double lon;
20.     private String restaurantName;
21.     private double myLat;
22.     private double myLon;
23.     private LatLng RESTAURANT;
24.     private LatLng ME;
25.
26.     @Override
27.     protected void onCreate(Bundle savedInstanceState) {
28.         super.onCreate(savedInstanceState);
29.         setContentView(R.layout.activity_restaurant_map);
30.
31.         lat = getIntent().getDoubleExtra("LATITUDE", 0);
32.         lon = getIntent().getDoubleExtra("LONGITUDE", 0);
33.         restaurantName = getIntent().getStringExtra("NAME");
34.         myLat = getIntent().getDoubleExtra("MYLATITUDE", 0);
35.         myLon = getIntent().getDoubleExtra("MYLONGITUDE", 0);
36.
37.         // Obtain the SupportMapFragment and get notified when the map is ready to be
        used.
38.         SupportMapFragment mapFragment = (SupportMapFragment)
        getSupportFragmentManager().findFragmentById(R.id.restaurant_map);
39.         mapFragment.getMapAsync(this);
40.     }
41.
42.     /**
43.      * Manipulates the map once available.
44.      * This callback is triggered when the map is ready to be used.
45.      * This is where we can add markers or lines, add listeners or move the camera.
        In this case,
46.      * we just add a marker near Sydney, Australia.
47.      * If Google Play services is not installed on the device, the user will be
        prompted to install
48.      * it inside the SupportMapFragment. This method will only be triggered once the
        user has
    
```

```
49.     * installed Google Play services and returned to the app.
50.     */
51.     @Override
52.     public void onMapReady(GoogleMap googleMap) {
53.         mMap = googleMap;
54.
55.         RESTAURANT = new LatLng(lat, lon);
56.         ME = new LatLng(myLat, myLon);
57.
58.         Marker restaurant = mMap.addMarker(new
MarkerOptions().position(RESTAURANT).title(restaurantName));
59.         Marker me = mMap.addMarker(new MarkerOptions().position(ME).title("ME")
.snippet("My location")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.marker_me)));
60.
61.         // Move the camera instantly to restaurant with a zoom of 15.
62.         mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(RESTAURANT, 15));
63.     }
64. }
```

43. Open the *DetailForm.java* file and update the MENU option for Show on Map option

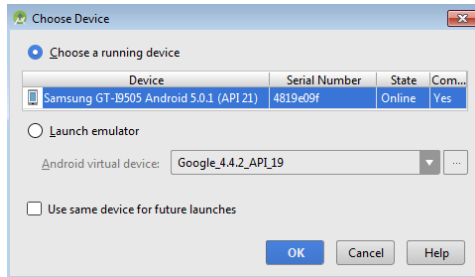
```
1. package com.sp.restaurantlist;
2.
3. import android.content.Intent;
4. import android.database.Cursor;
5. import android.os.Bundle;
6. import android.support.v7.app.AppCompatActivity;
7. import android.view.Menu;
8. import android.view.MenuInflater;
9. import android.view.MenuItem;
10. import android.view.View;
11. import android.widget.Button;
12. import android.widget.EditText;
13. import android.widget.RadioGroup;
14. import android.widget.TextView;
15. import android.widget.Toast;
16.
17. public class DetailForm extends AppCompatActivity {
18.     private EditText restaurantName;
19.     private EditText restaurantAddress;
20.     private EditText restaurantTel;
21.     private RadioGroup restaurantTypes;
22.     private Button buttonSave;
23.     private RestaurantHelper helper = null;
24.     private String restaurantID = "";
25.     private TextView location = null;
26.     private GPSTracker gpsTracker;
27.     private double latitude = 0.0d;
28.     private double longitude = 0.0d;
29.     private double myLatitude = 0.0d;
30.     private double myLongitude = 0.0d;
31.
32.     @Override
33.     protected void onCreate(Bundle savedInstanceState) {
34.         super.onCreate(savedInstanceState);
35.         setContentView(R.layout.detail_form);
36.
37.         restaurantName = (EditText) findViewById(R.id.restaurant_name);
38.         restaurantAddress = (EditText) findViewById(R.id.restaurant_address);
39.         restaurantTel = (EditText) findViewById(R.id.restaurant_tel);
40.         restaurantTypes = (RadioGroup) findViewById(R.id.restaurant_types);
41.
42.         buttonSave = (Button) findViewById(R.id.button_save);
43.         buttonSave.setOnClickListener(onSave);
44.
45.         helper = new RestaurantHelper(this);
46.
47.         location = (TextView) findViewById(R.id.location);
48.         gpsTracker = new GPSTracker(DetailForm.this);
49.
50.         restaurantID = getIntent().getStringExtra("ID");
```

```
51.         if (restaurantID != null) {
52.             load();
53.         }
54.     }
55.
56.     private void load() {
57.         Cursor c = helper.getId(restaurantID);
58.         c.moveToFirst();
59.         restaurantName.setText(helper.getRestaurantName(c));
60.         restaurantAddress.setText(helper.getRestaurantAddress(c));
61.         restaurantTel.setText(helper.getRestaurantTel(c));
62.
63.         if (helper.getRestaurantType(c).equals("Chinese")) {
64.             restaurantTypes.check(R.id.chinese);
65.         } else if (helper.getRestaurantType(c).equals("Western")) {
66.             restaurantTypes.check(R.id.western);
67.         } else if (helper.getRestaurantType(c).equals("Indian")) {
68.             restaurantTypes.check(R.id.indian);
69.         } else if (helper.getRestaurantType(c).equals("Indonesia")) {
70.             restaurantTypes.check(R.id.indonesian);
71.         } else if (helper.getRestaurantType(c).equals("Korean")) {
72.             restaurantTypes.check(R.id.korean);
73.         } else if (helper.getRestaurantType(c).equals("Japanese")) {
74.             restaurantTypes.check(R.id.japanese);
75.         } else {
76.             restaurantTypes.check(R.id.thai);
77.         }
78.
79.         latitude = helper.getLatitude(c);
80.         longitude = helper.getLongitude(c);
81.         location.setText(String.valueOf(latitude) + ", " +
String.valueOf(longitude));
82.     }
83.
84.     @Override
85.     protected void onDestroy() {
86.         super.onDestroy();
87.         helper.close();
88.         gpsTracker.stopSelf();
89.     }
90.
91.     @Override
92.     public boolean onCreateOptionsMenu(Menu menu) {
93.         new MenuInflater(this).inflate(R.menu.details_option, menu);
94.         return super.onCreateOptionsMenu(menu);
95.     }
96.
97.     @Override
98.     public boolean onOptionsItemSelected(MenuItem item) {
99.         if (item.getItemId() == R.id.get_location) {
100.             if (gpsTracker.canGetLocation()) {
101.                 latitude = gpsTracker.getLatitude();
102.                 longitude = gpsTracker.getLongitude();
103.
104.                 location.setText(String.valueOf(latitude) + ", " +
String.valueOf(longitude));
105.
106.                 // \n is for new line
107.                 Toast.makeText(getApplicationContext(), "Your Location is - \nLat: "
+ latitude + "\nLong: " + longitude, Toast.LENGTH_LONG).show();
108.             } else {
109.                 // can't get location. GPS or Network is not enabled
110.                 // Ask user to enable GPS/network in settings
111.                 gpsTracker.showSettingsAlert();
112.                 return (true);
113.             } else if (item.getItemId() == R.id.show_map) {
114.                 //Get my current location
115.                 myLatitude = gpsTracker.getLatitude();
116.                 myLongitude = gpsTracker.getLongitude();
117.
118.                 Intent intent = new Intent(this, RestaurantMap.class);
```

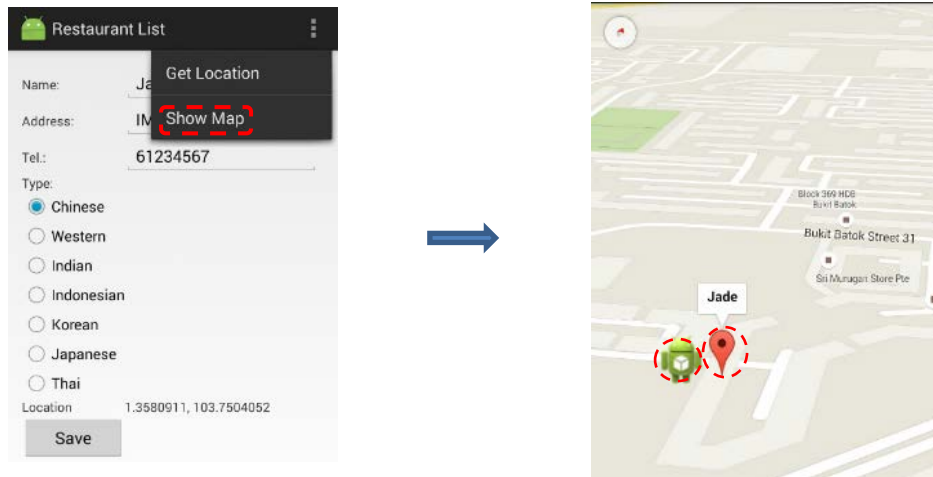
```
119.         intent.putExtra("LATITUDE", latitude);
120.         intent.putExtra("LONGITUDE", longitude);
121.         intent.putExtra("MYLATITUDE", myLatitude);
122.         intent.putExtra("MYLONGITUDE", myLongitude);
123.         intent.putExtra("NAME", restaurantName.getText().toString());
124.         startActivity(intent);
125.         return (true);
126.     }
127.
128.     return super.onOptionsItemSelected(item);
129. }
130.
131. private View.OnClickListener onSave = new View.OnClickListener() {
132.     @Override
133.     public void onClick(View v) {
134.         // To read date from name EditText
135.         String nameStr = restaurantName.getText().toString();
136.         String addrStr = restaurantAddress.getText().toString();
137.         String telStr = restaurantTel.getText().toString();
138.         String restType = "";
139.         //To read selection of restaurantTypes RadioGroup
140.         switch (restaurantTypes.getCheckedRadioButtonId()) {
141.             case R.id.chinese:
142.                 restType = "Chinese";
143.                 break;
144.             case R.id.western:
145.                 restType = "Western";
146.                 break;
147.             case R.id.indian:
148.                 restType = "Indian";
149.                 break;
150.             case R.id.indonesian:
151.                 restType = "Indonesian";
152.                 break;
153.             case R.id.korean:
154.                 restType = "Korean";
155.                 break;
156.             case R.id.japanese:
157.                 restType = "Japanese";
158.                 break;
159.             case R.id.thai:
160.                 restType = "Thai";
161.                 break;
162.         }
163.
164.         if (restaurantID == null) {
165.             helper.insert(nameStr, addrStr, telStr, restType, latitude,
longitude);
166.         } else {
167.             helper.update(restaurantID, nameStr, addrStr, telStr, restType,
latitude, longitude);
168.         }
169.
170.         //To close current Activity
171.         finish();
172.     }
173. };
174. }
```

44. To test this part of the program, it is good to use an actual Android device. Under your Android device **System Settings > { }Developer options**, check the **USB debugging** checkbox
45. Plug the USB cable to your PC and make sure your device driver has successfully installed. Otherwise, **Android Studio** will not be able to detect your Android device

46. At the Top Menu Bar, click on the Run icon.



47. Select the **Show Map** option from the **MENU**, the restaurant location will be shown on the map with two markers. If you click on the marker, a **Toast** will be shown with the restaurant name or your location
[Note: Make sure you have Internet connection before testing]



48. Show the result to your lecturer if you have completed this section

Lecturer Signature : _____

-END-