Android Map Application Tutorial

Version: 1.0 Beta

Prepared by
Biplab Pramanik
Excoflare Software Technologies (P) Ltd.

Revision History

Version	Date	Author	Description
Version 1	09/11/10	Biplab Pramanik	Final release

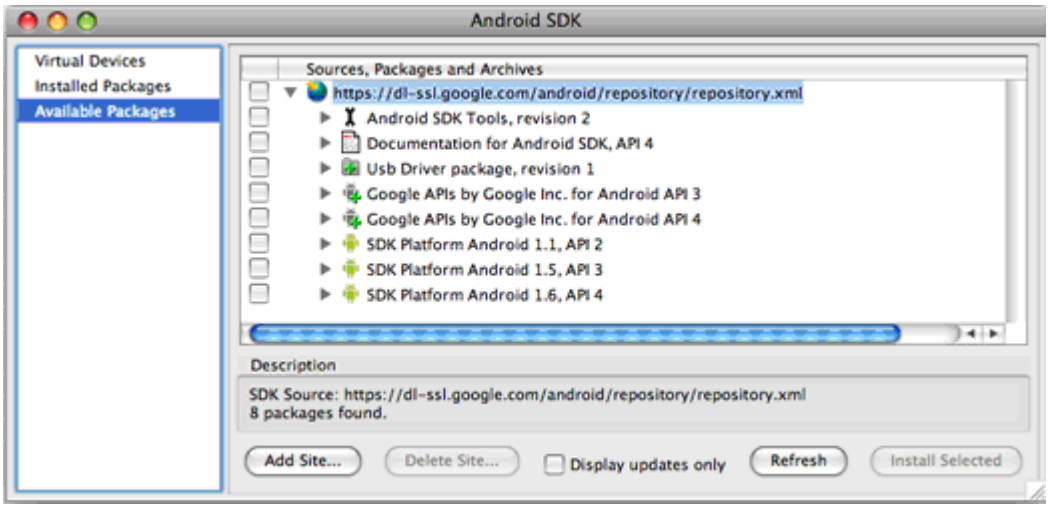
Contents

Google Maps library

This tutorial requires external Google Maps library installed in your SDK environment.

To add external libraries follow the below steps.

1. Open Eclipse
2. Select **Window** > **Android SDK and AVD Manager**.
3. Select **Available Packages** in the left panel. This will reveal all of the components that are currently available for download from the SDK repository.
4. Select Google APIs by Google Inc for Android API 8 and click **Install Selected**.



Obtaining a Maps API Key

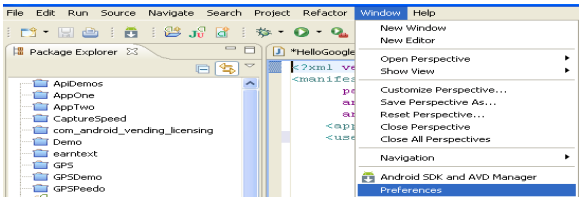
Because MapView gives you access to Google Maps data, you need to register with the Google Maps service and agree to the applicable Terms of Service before your MapView will be able to obtain data from Google Maps.

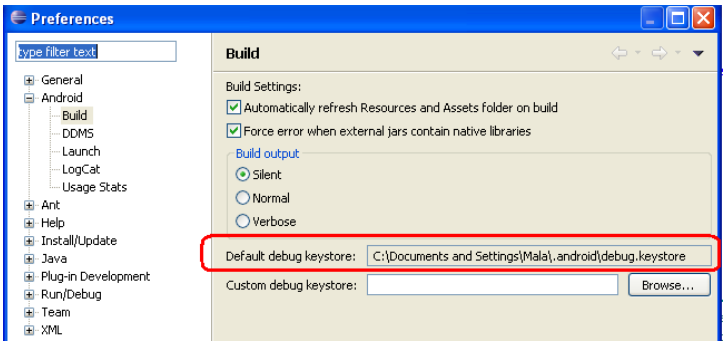
Registering for a Maps API Key is simple, free, and has two parts:

1. Registering the MD5 fingerprint of the certificate.
2. Register the Certificate fingerprint and get the map view along with API key.

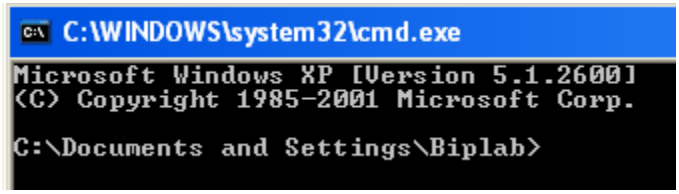
Registering the MD5 fingerprint of the certificate.

1. Open **Eclipse-> Windows>Preferences>Android>Build**

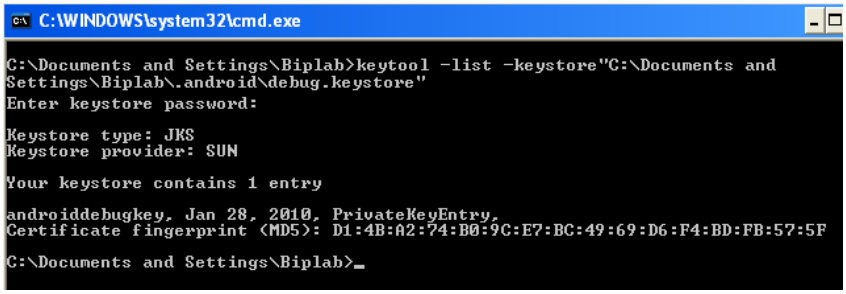




2. Copy the Default debug keystore.
3. Open **Command Prompt**



4. Type keytool -list -keystore "C:\Documents and Settings\Biplab\.android\debug.keystore"



5. Enter password as android

Register the Certificate fingerprint and get the map view along with API key

1. Go To http://code.google.com/android/maps-api-signup.html and place your certification key
2. If it asks Google account information put

   User id : **publish.excoflare@gmail.com**
   Password: **guwahati123**

Sign Up for the Android Maps API

The Android Maps API lets you embed Google Maps in your own Android applications. A single Maps API key is valid f
for more information about application signing. To get a Maps API key for your certificate, you will need to provide its the
Linux or Mac OSX, you would examine your debug keystore like this:

```
$ keytool -list -keystore ~/.android/debug.keystore
...
Certificate fingerprint (MD5): 94:1E:43:49:87:73:BB:E6:A6:88:D7:20:F1:8E:B5:98
```

If you use different keys for signing development builds and release builds, you will need to obtain a separate Maps API
the corresponding certificate.

You also need a Google Account to get a Maps API key, and your API key will be connected to your Google Account.

```
Android Maps APIs Terms of Service

Last Updated: October 13, 2008

Thanks for your interest in the Android Maps APIs.  The Android Maps
APIs are a collection of services (including, but not limited to, the
"com.google.android.maps.MapView" and "android.location.Geocoder"
classes) that allow you to include maps, geocoding, and other content
from Google and its content providers in your Android applications.
The Android Maps APIs explicitly do not include any driving
directions data or local search data that may be owned or licensed by
Google.


1.  Your relationship with Google.
    1.1. Your use of any of the Android Maps APIs (referred to in
```

I have read and agree with the terms and conditions (printable version)

My certificate's MD5 fingerprint: D1:4B:A2:74:B0:9C:E7:BC:49:69:D6:F4:BD:FB:57:5F

Generate API Key

Thank you for signing up for an Android Maps API key!

Your key is:

```
0OJo5yUfd8SycQiuPymqvXLsDQpfzdhxaRHxFRw
```

This key is good for all apps signed with your certificate whose fingerprint is:

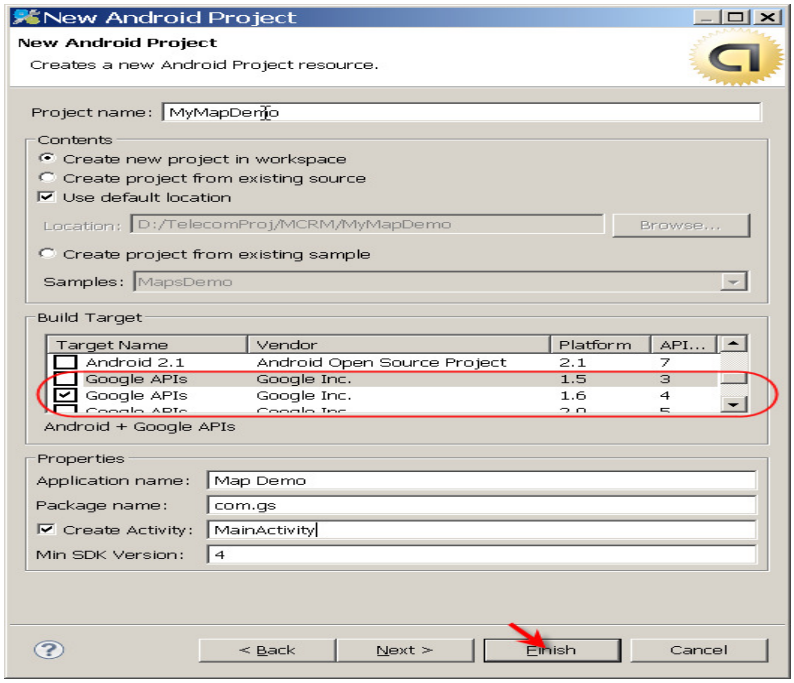
```
D1:4B:A2:74:B0:9C:E7:BC:49:69:D6:F4:BD:FB:57:5F
```

Here is an example xml layout to get you started on your way to mapping glory:

```
<com.google.android.maps.MapView
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:apiKey="0OJo5yUfd8SycQiuPymqvXLsDQpfzdhxaRHxFRw"
   />
```

Create a map based android project

1. **Eclipse > New Project > Android Project > Select Google Map 1.5**



2. Declare the Maps library in the Android Manifest. Open the AndroidManifest.xml file and add the following element:

```
<uses-library android:name="com.google.android.maps" />
```

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.excoflare.map"
      android:versionCode="1"
      android:versionName="1.0">
    <application android:icon="@drawable/icon" android:label="@string/app_name">
    <uses-library android:name="com.google.android.maps" />
```

3. Add Internet permission to the Manifest.

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

4. Add NoTitleBar theme to activity.

```
<activity android:name=".HelloGoogleMaps" android:label="@string/app_name"
      android:theme="@android:style/Theme.NoTitleBar">
```

5. Open the res/layout/main.xml file and add a single com.google.android.maps.MapView as the root node:

```
<?xml version="1.0" encoding="utf-8"?>
<com.google.android.maps.MapView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/mapview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:clickable="true"
    android:apiKey="Your Maps API Key goes here"
/>
```

Here, you need to add the Map API Key which is generated before.

6. Now open the **MainActivity.java** file. For this Activity, extend MapActivity (instead of android.app.Activity):

```
public class MainActivity extends MapActivity {
/** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    }

    @Override
    protected boolean isRouteDisplayed() {
       // TODO Auto-generated method stub
        return false;

    }
}
```

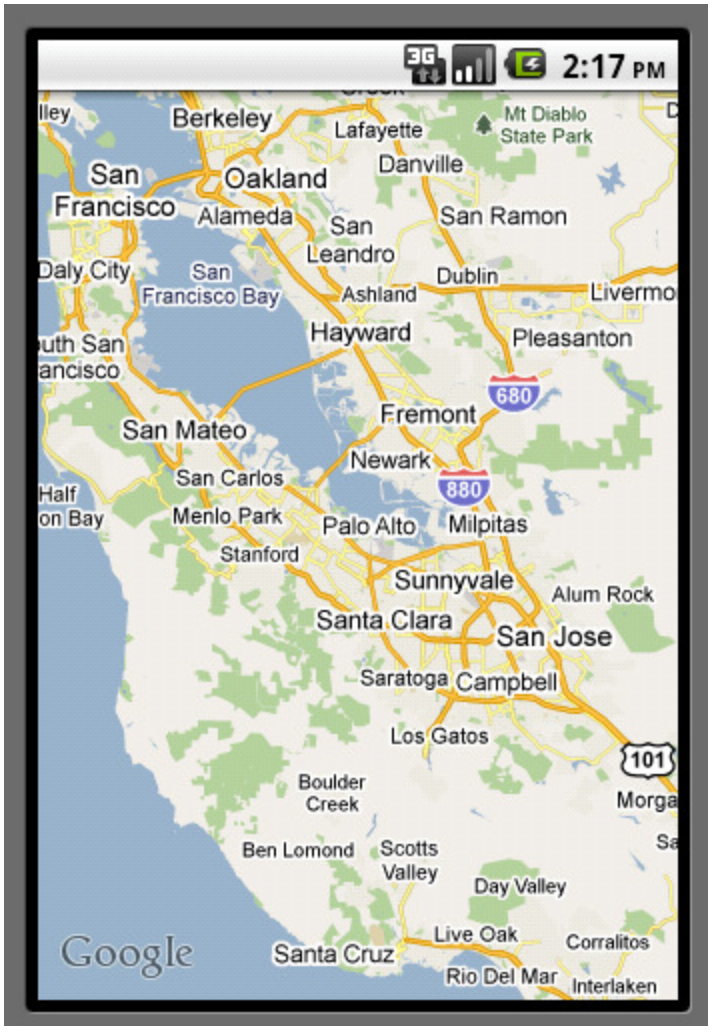
7. Create a GeoPoint, MapView and MapController instance.

```
double latitude = 10.47, longitude = 79.10;
GeoPoint geoPoint = new GeoPoint((int) (latitude * 1000000),
(int) (longitude * 1000000));
MapView myMapView = (MapView) findViewById(R.id.myGMap);
myMapView.setSatellite(false);
myMapView.setBuiltInZoomControls(true);
myMapView.displayZoomControls(true);
MapController myMC = = myMapView.getController();
myMC.setCenter(geoPoint);
myMC.setZoom(15);
```

8. Need to Add the User Permission

```
    </application>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"></uses-permission>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"></uses-permission>
<uses-permission android:name="android.permission.INTERNET"></uses-permission>
```

9. Output:

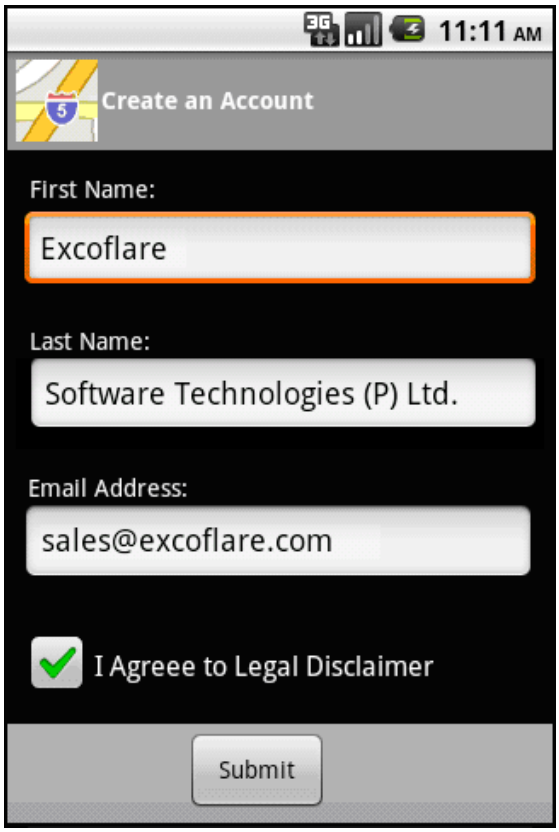


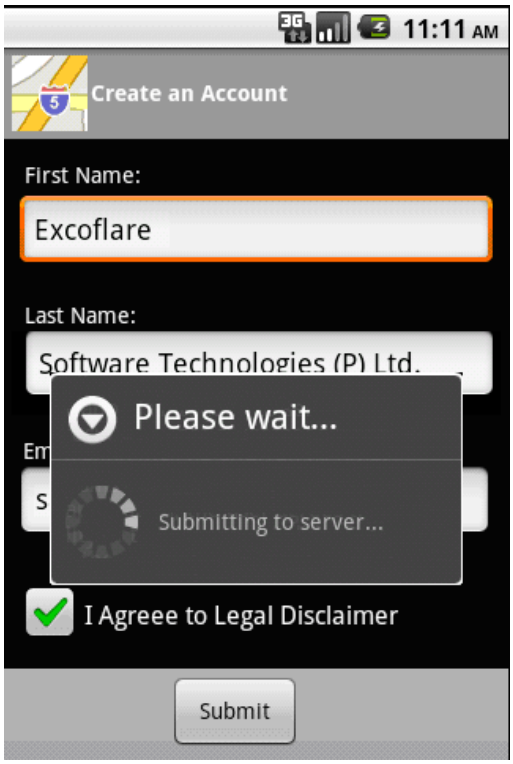
DriveRoute

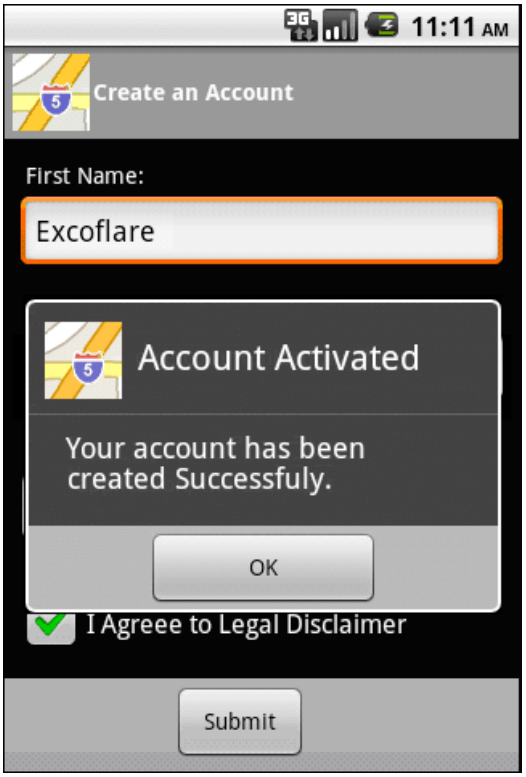
DriveRoute is a safe driving application which directs the path to travel from source location to destination location. This application shows the path in Google Map and also submits information to server.

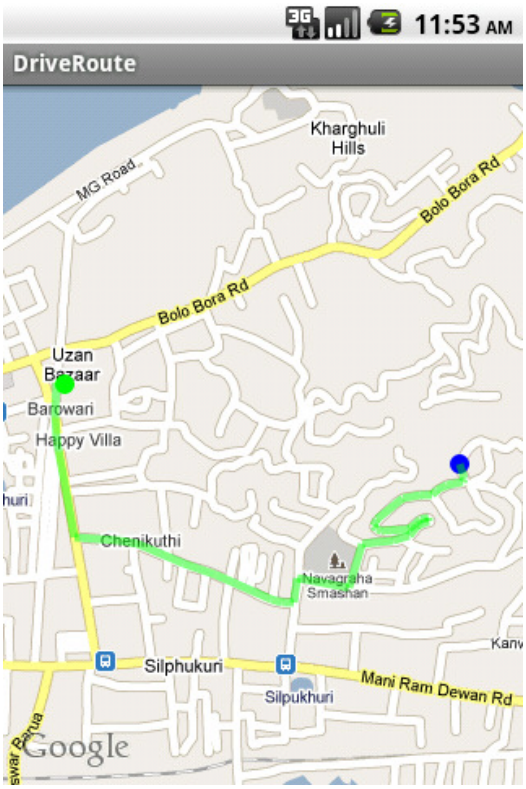
Android features:

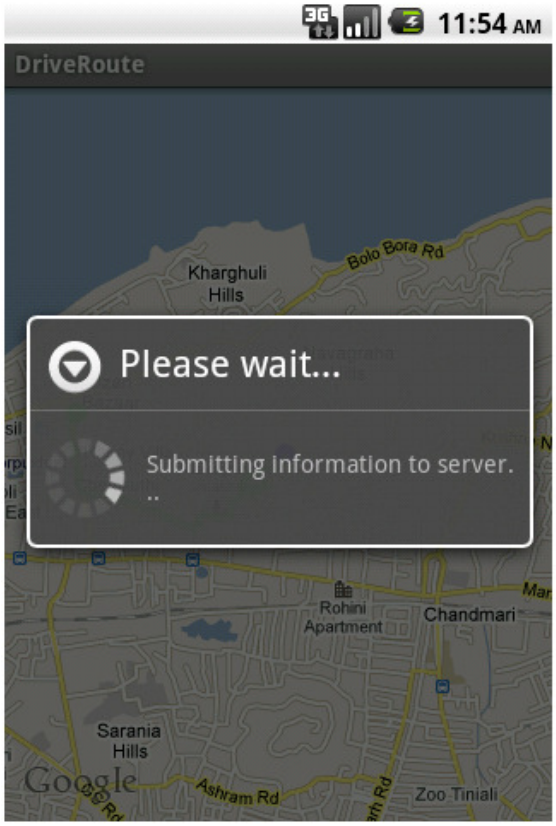
1. Http connection to server to submit account creation details.
2. Google Map API implementation
3. XML parsing
4. Text file reading from SD card
5. Canvas drawing to show path




11:11 AM
Create an Account
First Name:
Excoflare
Last Name:
Software Technologies (P) Ltd.
Please wait...
Submitting to server...
I Agreee to Legal Disclaimer
Submit


11:11 AM
Create an Account
First Name:
Excoflare
Account Activated
Your account has been created Successfuly.
OK
I Agreee to Legal Disclaimer
Submit


11:53 AM
DriveRoute
Kharghuli Hills
MG Road
Bolo Bora Rd
Bolo Bora Rd
Uzan Bazaar
Barowari
Happy Villa
Chenikuthi
Navagraha Smashan
Silphukuri
Silpukhuri
Mani Ram Dewan Rd
Google


11:54 AM
DriveRoute
Kharghuli Hills
Bolo Bora Rd
Please wait...
Submitting information to server. ..
Rohini Apartment
Chandmari
Sarania Hills
Google
Ashram Rd
Zoo Tiniali