August 25, 2013 10:16 AM

Android working with Google Maps V2

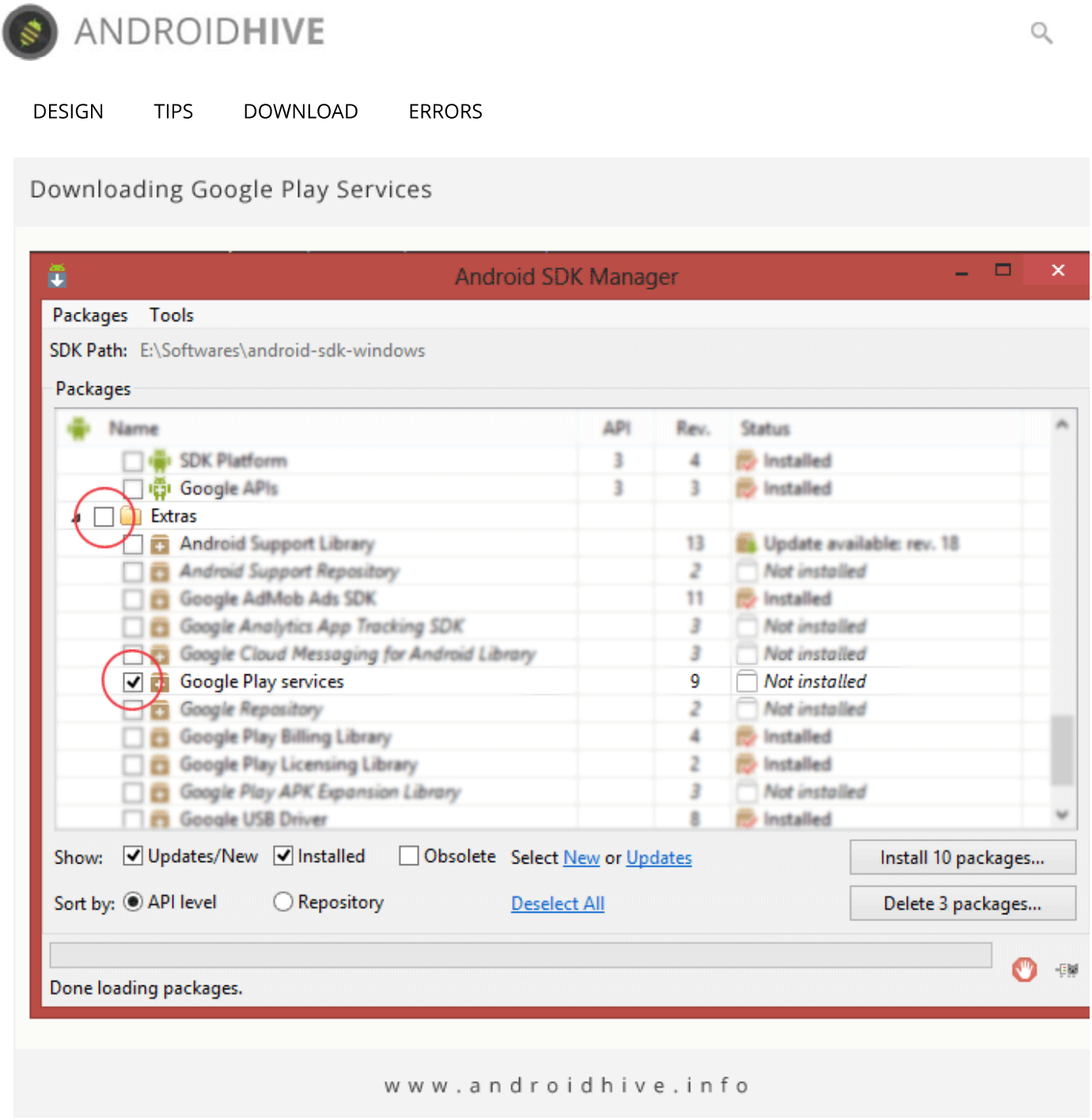
If you have developed any app that contains Google Maps v1, It's time to upgrade it to Google Maps V2 as google maps version 1 deprecated officially on December 3rd, 2012 and it won't work anymore. This article aims to give knowledge about how to implements newer Google Maps into your applications. If you have already worked with V1, implementing V2 is very easy. Refer Google Maps Docs for any topic that is not covered in this tutorial.

DOWNLOAD CODE

Before starting a new project, we need to go through some pre required steps. These steps invol importing required library, generating SHA1 fingerprint and configuring maps in google console.

1. Downloading Google Play Services

Google made new Maps V2 API as a part of Google Play Services SDK. So before we st developing maps we need to download google play services from SDK manger. You can open S manager either from Eclipse or from android sdk folder.

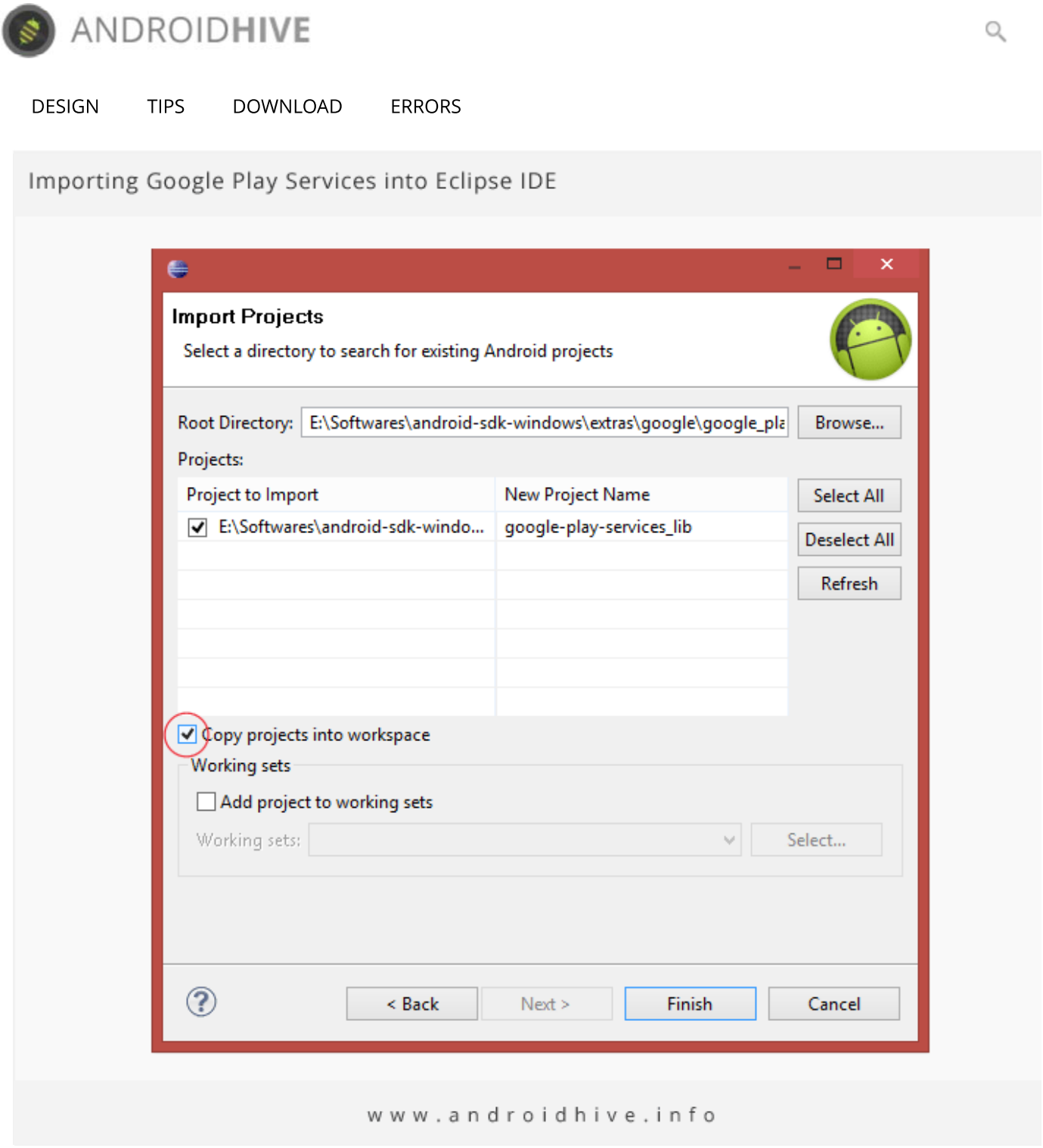


2. Importing Google Play Services into Eclipse

After downloading play services we need to import it to Eclipse which will be used as a library for maps project.

1. In Eclipse goto **File ⇒ Import ⇒ Android ⇒ Existing Android Code Into Workspace**

2. Click on Browse and select Google Play Services project from your android sdk folder. You can loc play services library project from
android-sdk-windows\extras\google\google_play_services\libproject\google-play-services_lib



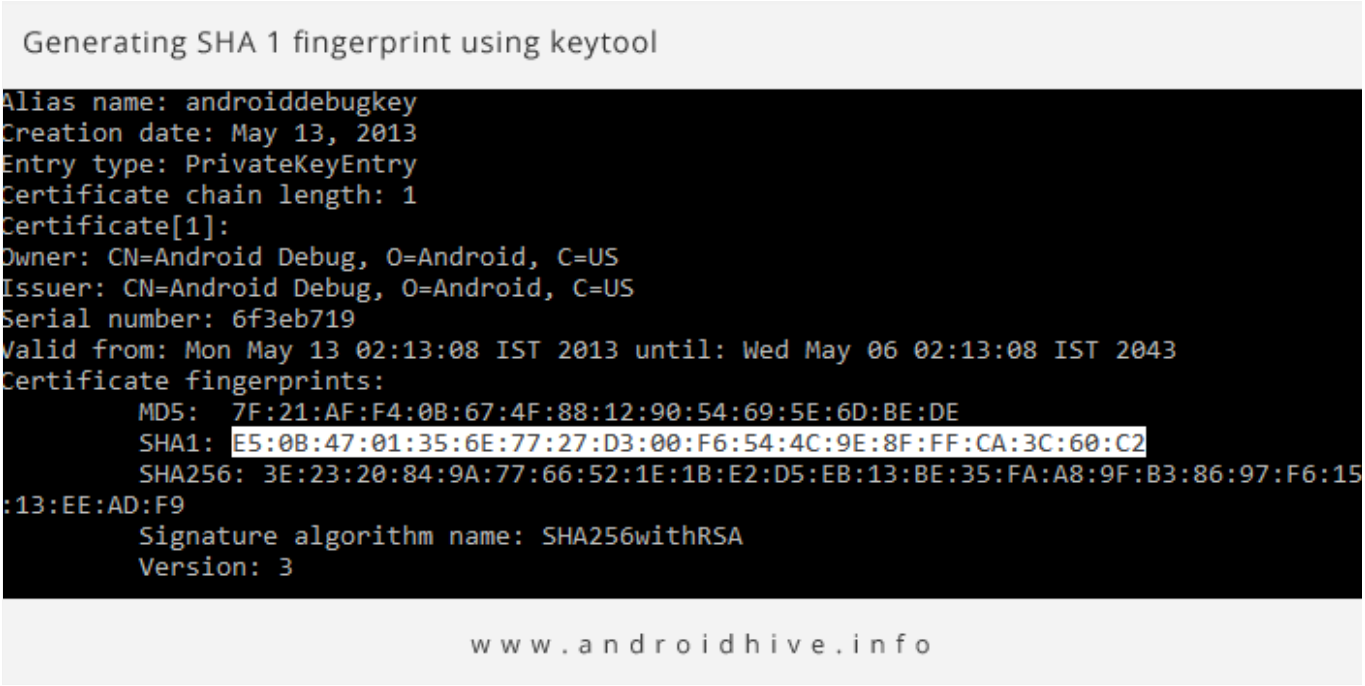
3. Getting the Google Maps API key

1. Same as in maps v1 we need to generate SHA-1 fingerprint using java `keytool`. Open your termi and execute the following command to generate SHA-1 fingerprint.

On Windows

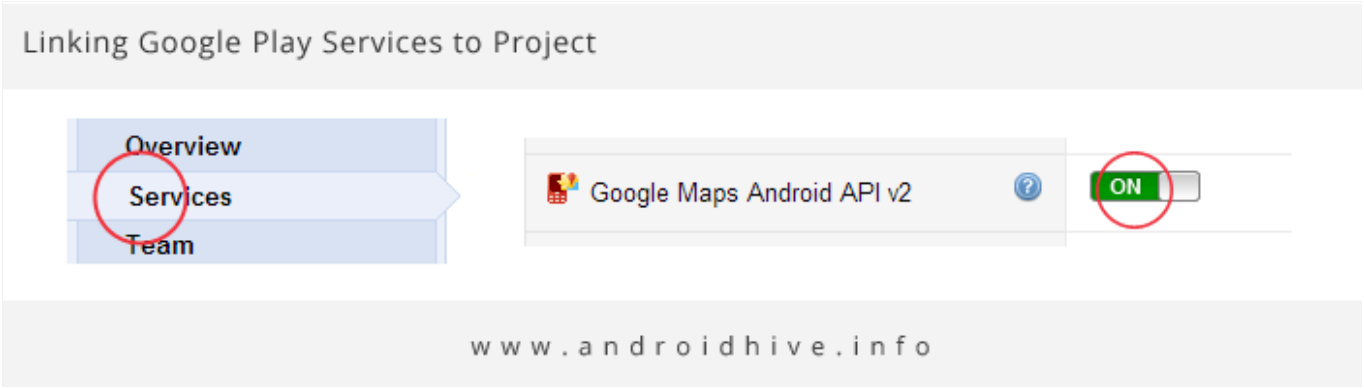
```
keytool -list -v -keystore ~/.android/debug.keystore -alias androiddebugkey -st
```

In the output you can see SHA 1 finger print.

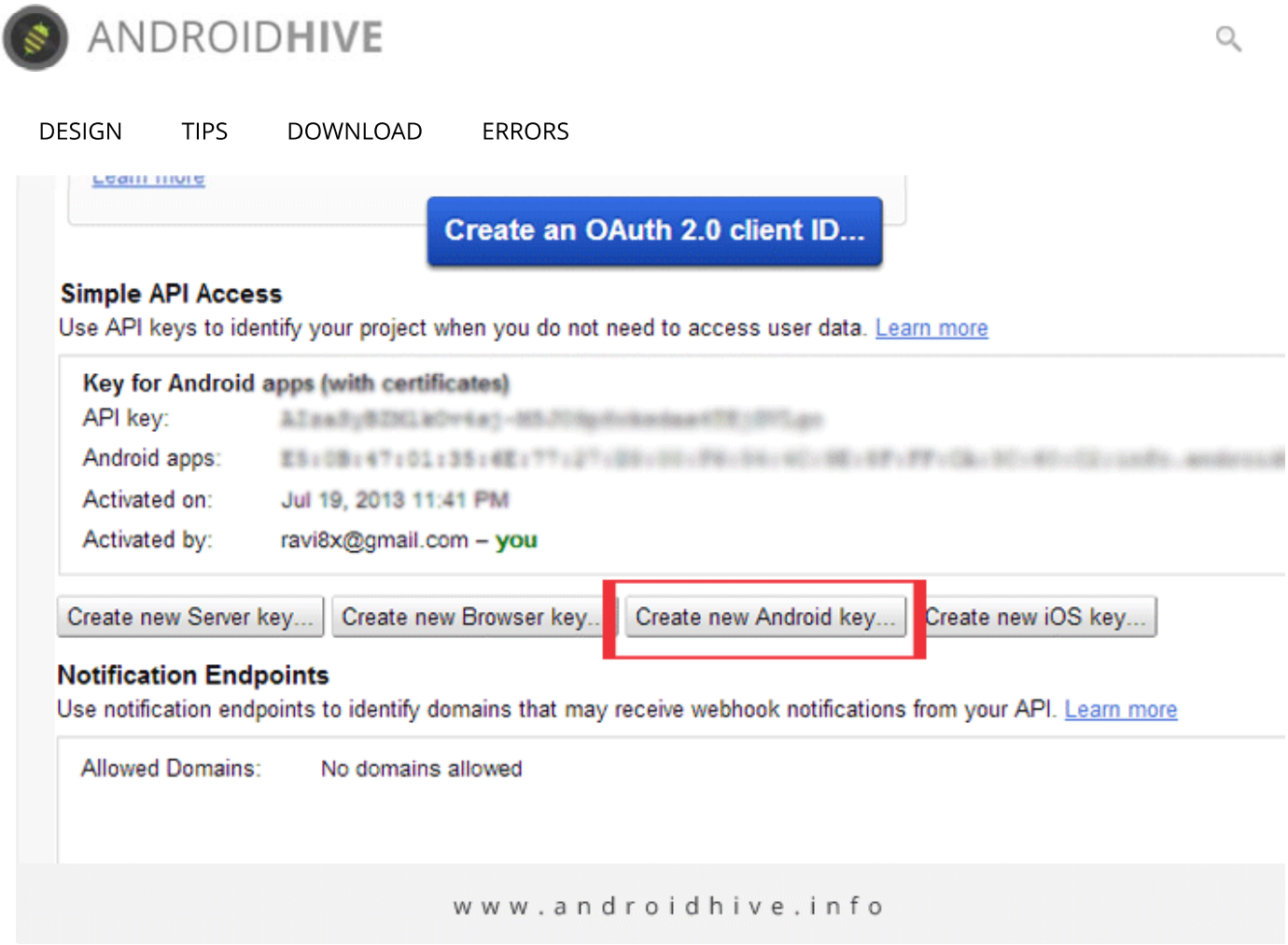


2. Now open Google API Console

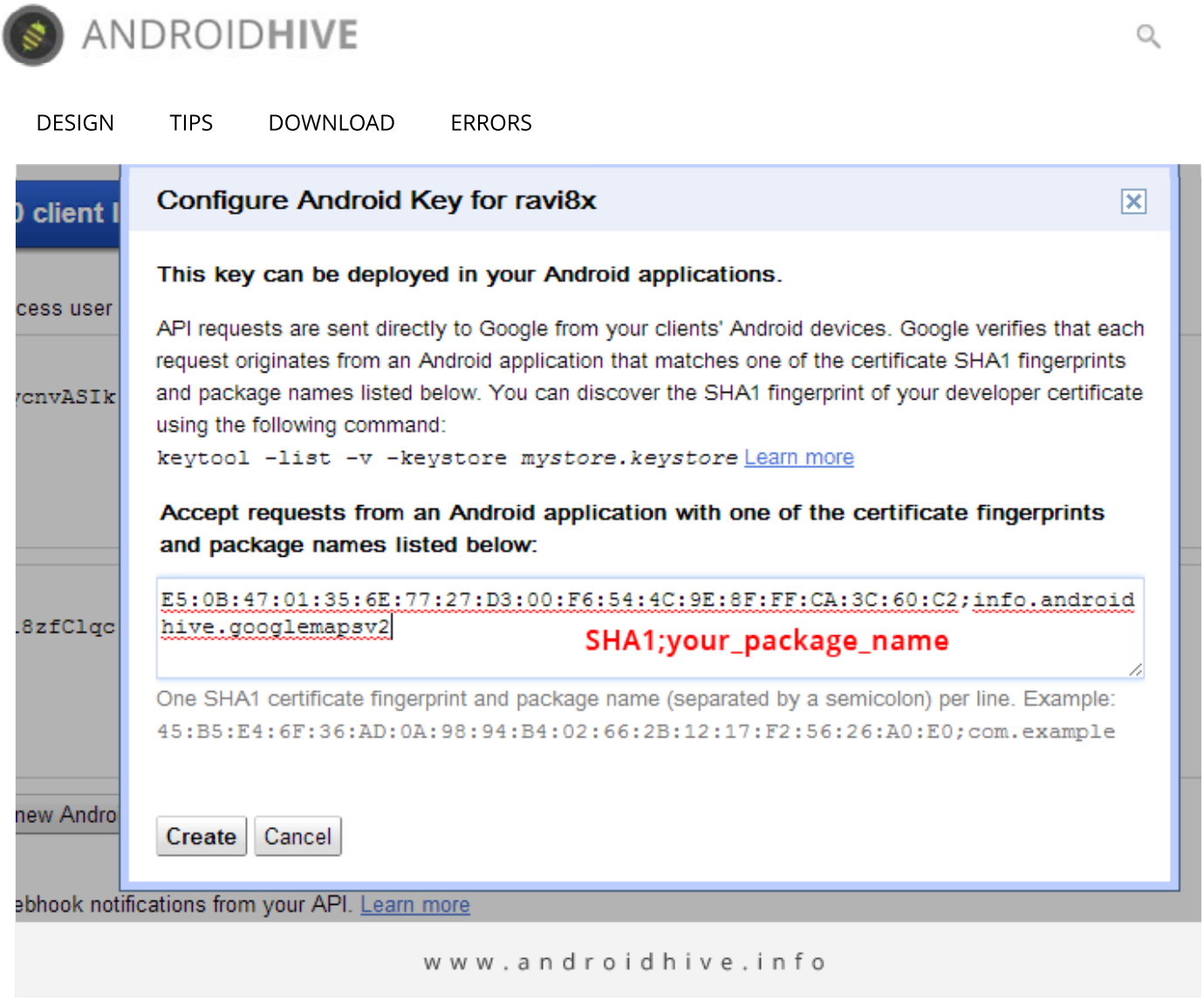
3. Select **Services** on left side and turn on **Google Maps Android API v2**



4. Now select **API Access** on left side and on the right side click on **Create new Android key...**



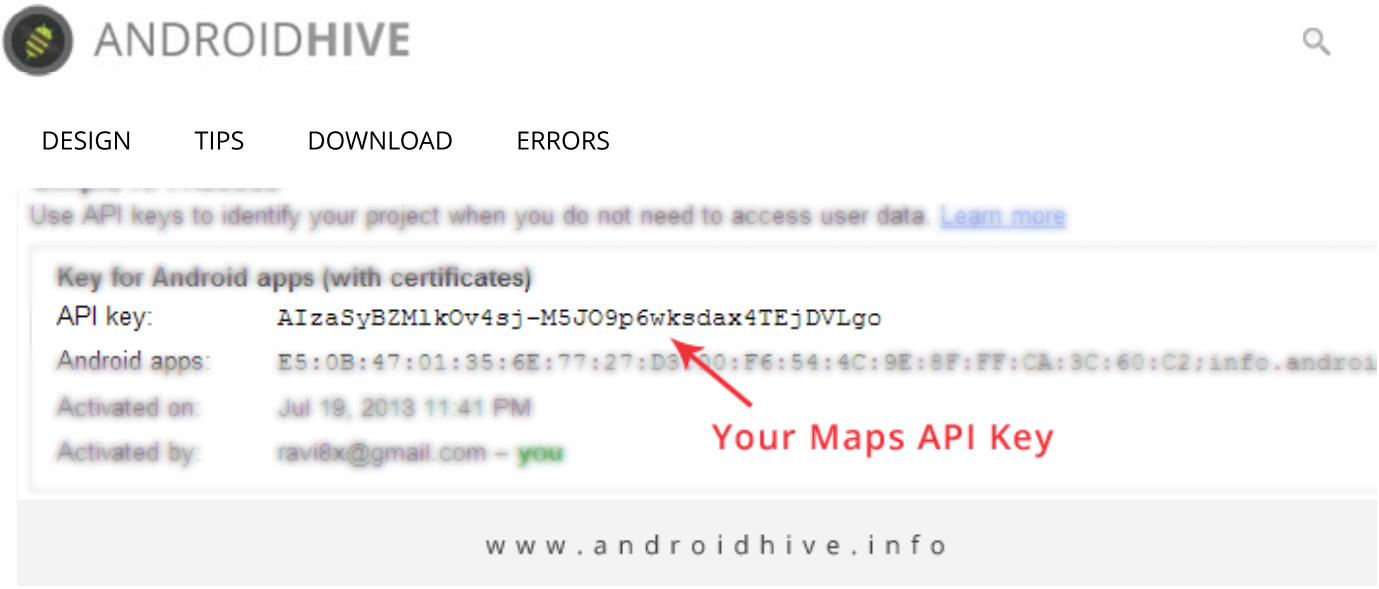
5. It will popup a window asking the SHA1 and package name. Enter your `SHA 1` and your `andro` `project package name` separated by semicolon `;` and click on create.



I have given like below

```
BE:03:E1:44:39:7B:E8:17:02:9F:7F:B7:98:82:EA:DF:84:D0:FB:6A;info.androidhive.go
```

And note down the API key which required later in our project.



4. Creating new Project

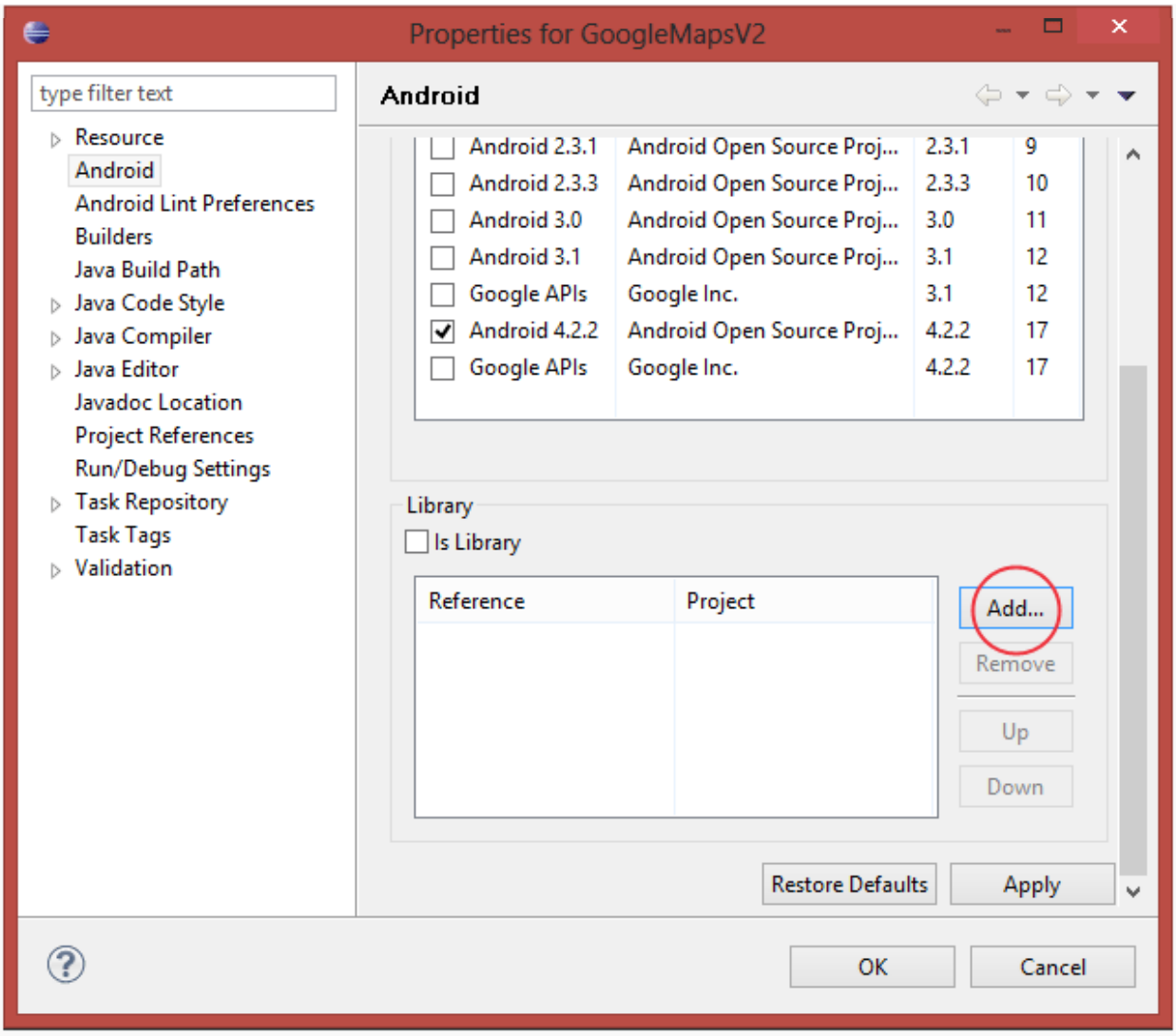
After completing required configuration, It's time to start our project.

1. In Eclipse create a new project by going to **File ⇒ New ⇒ Android Application Project** and required details. I kept my project name as `Google Maps V2` and package name `info.androidhive.info`

2. Now we need to use Google Play Services project as a library to use project. So **right click** on proj and select **properties**. In the properties window on left side select **Android**. On the right you can se **Add** button under library section. Click it and select **google play services** project which we impor previously.

ANDROIDHIVE

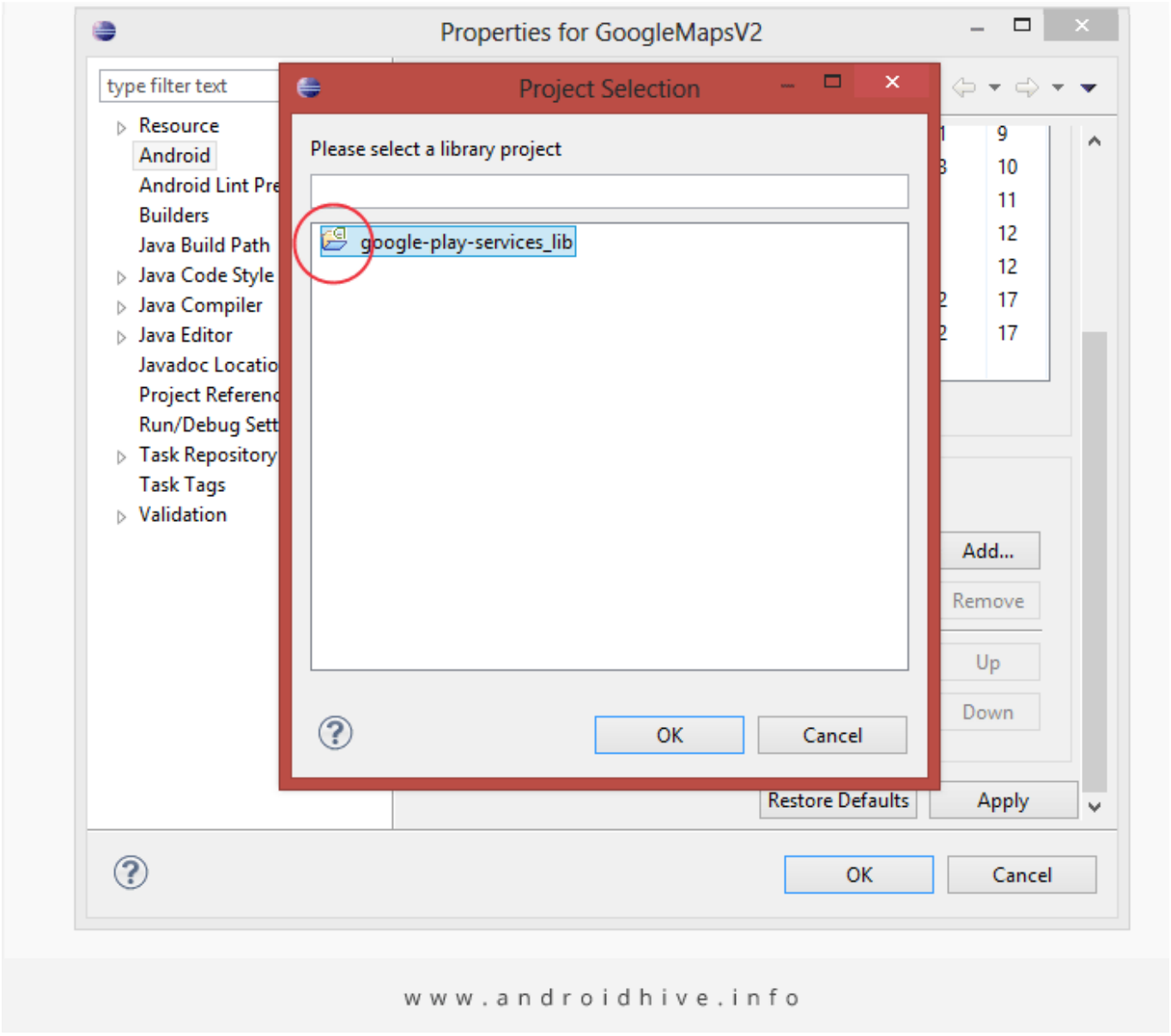
DESIGN TIPS DOWNLOAD ERRORS



www.androidhive.info

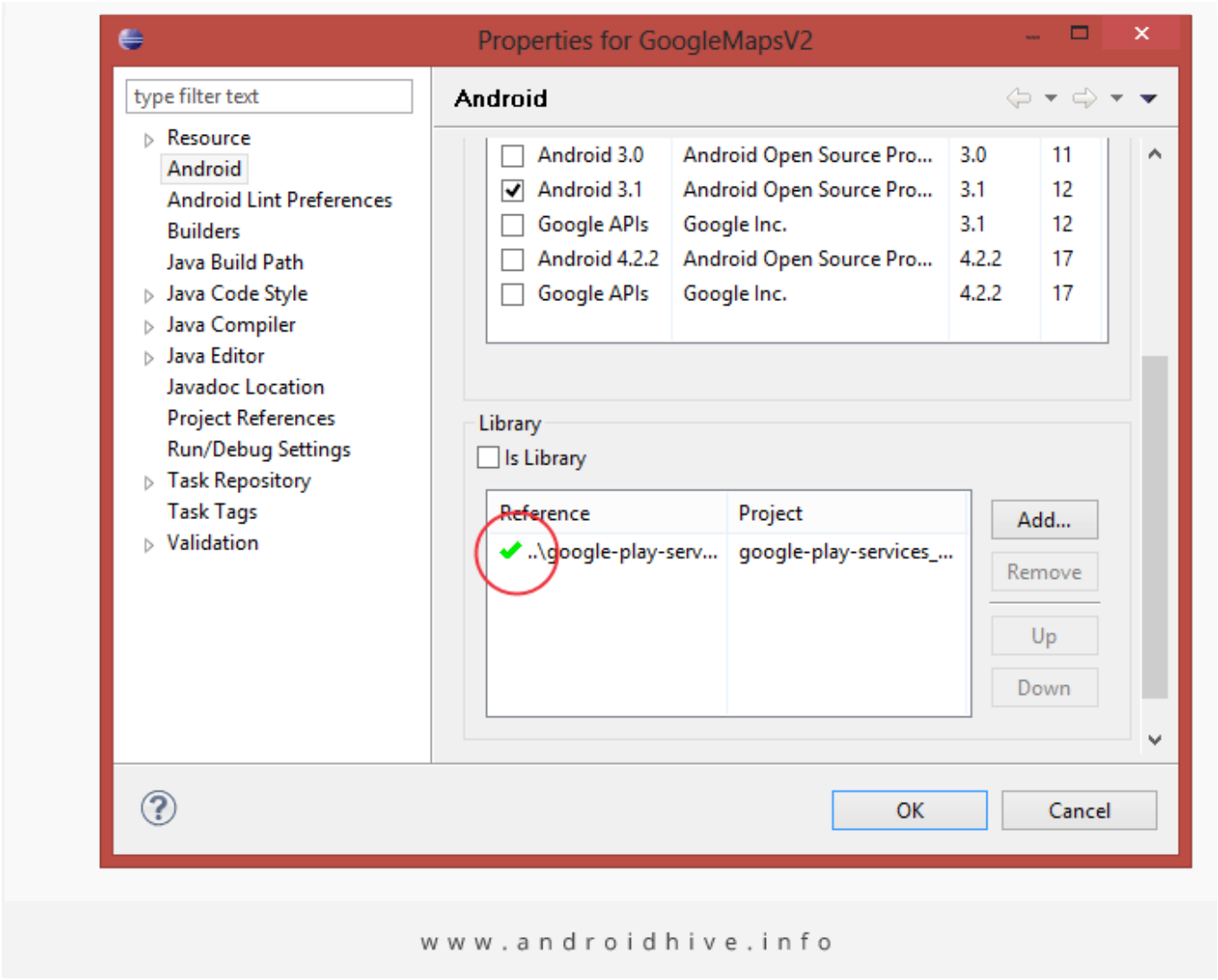
ANDROIDHIVE

DESIGN TIPS DOWNLOAD ERRORS



ANDROIDHIVE

DESIGN TIPS DOWNLOAD ERRORS



3. Add the Map Key in the manifest file. Open `AndroidManifest.xml` file and add the following cc before tag. Replace the `android:value` with your map key which you got from google console.

```
<!-- Goolge Maps API Key -->
<meta-data
     android:name="com.google.android.maps.v2.API_KEY"
     android:value="AIzaSyBZMlkOv4sj-M5JO9p6wksdax4TEjDVLgo" />
```

4. Google maps needs following permissions and features.

`ACCESS_NETWORK_STATE` – To check network state whether data can be downloaded or not
`INTERNET` – To check internet connection status
`WRITE_EXTERNAL_STORAGE` – To write to external storage as google maps store map data external storage
`ACCESS_COARSE_LOCATION` – To determine user's location using WiFi and mobile cell data

package)

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="info.androidhive.googlemapsv2"
    android:versionCode="1"
    android:versionName="1.0" >

    <permission
        android:name="info.androidhive.googlemapsv2.permission.MAPS_RECEIVE"
        android:protectionLevel="signature" />

    <uses-permission android:name="info.androidhive.googlemapsv2.permission.MAP

    <uses-sdk
        android:minSdkVersion="12"
        android:targetSdkVersion="17" />

    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="com.google.android.providers.gsf.permission.
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" /

    <!-- Required to show current location -->
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" /
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

    <!-- Required OpenGL ES 2.0. for Maps V2 -->
    <uses-feature
        android:glEsVersion="0x00020000"
        android:required="true" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name">
        <activity
            android:name="info.androidhive.googlemapsv2.MainActivity"
            android:label="@string/app_name"
            android:theme="@style/AppBaseTheme">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <!-- Goolge API Key -->
        <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="AIzaSyBZMlkOv4sj-M5JO9p6wksdax4TEjDVLgo" />
    </application>
```

class. Open your main activity layout file `activity_main.xml` file and add following code. I us `RelativeLayout` as a parent element. You can remove it and use MapFragment directly.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <fragment
        android:id="@+id/map"
        android:name="com.google.android.gms.maps.MapFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</RelativeLayout>
```

6. Add the following code in your Main Activity java (`MainActivity.java`) class.

MainActivity.java

```
public class MainActivity extends Activity {

    // Google Map
    private GoogleMap googleMap;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        try {
            // Loading map
            initilizeMap();

        } catch (Exception e) {
            e.printStackTrace();
        }

    }

    /**
     * function to load map. If map is not created it will create it for you
     * */
    private void initilizeMap() {
        if (googleMap == null) {
            googleMap = ((MapFragment) getFragmentManager().findFragmentById(
                    R.id.map)).getMap();

            // check if map is created successfully or not
```

```
    }

    @Override
    protected void onResume() {
        super.onResume();
        initilizeMap();
    }

}
```

Run your project and congratulations if you see a map displaying on your device.

ANDROIDHIVE

DESIGN TIPS DOWNLOAD ERRORS

You can place a marker on the map by using following code.

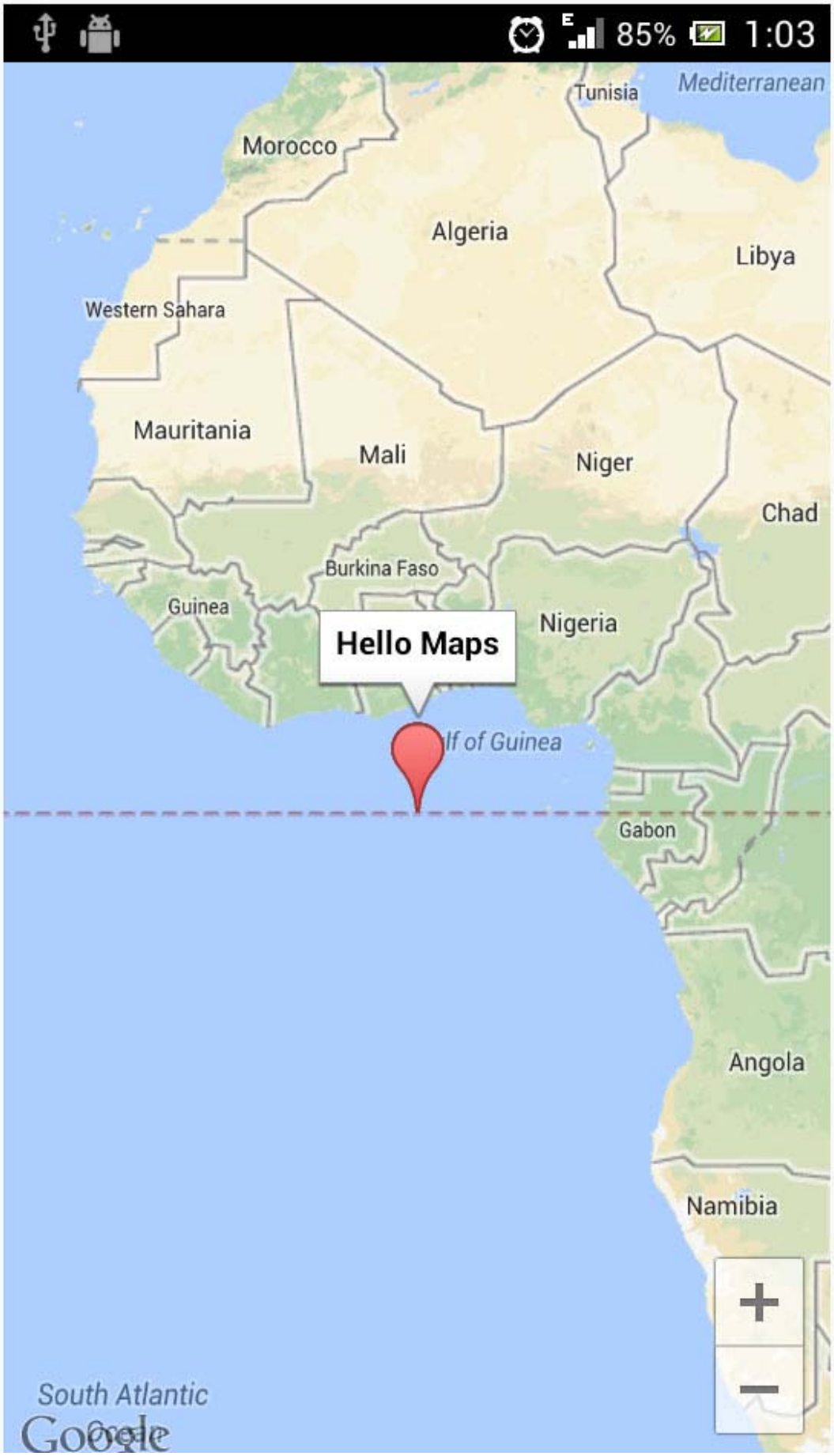
```
// latitude and longitude
double latitude = ;
double longitude = ;

// create marker
MarkerOptions marker = new MarkerOptions().position(new LatLng(latitude, longit

// adding marker
googleMap.addMarker(marker);
```

ANDROIDHIVE

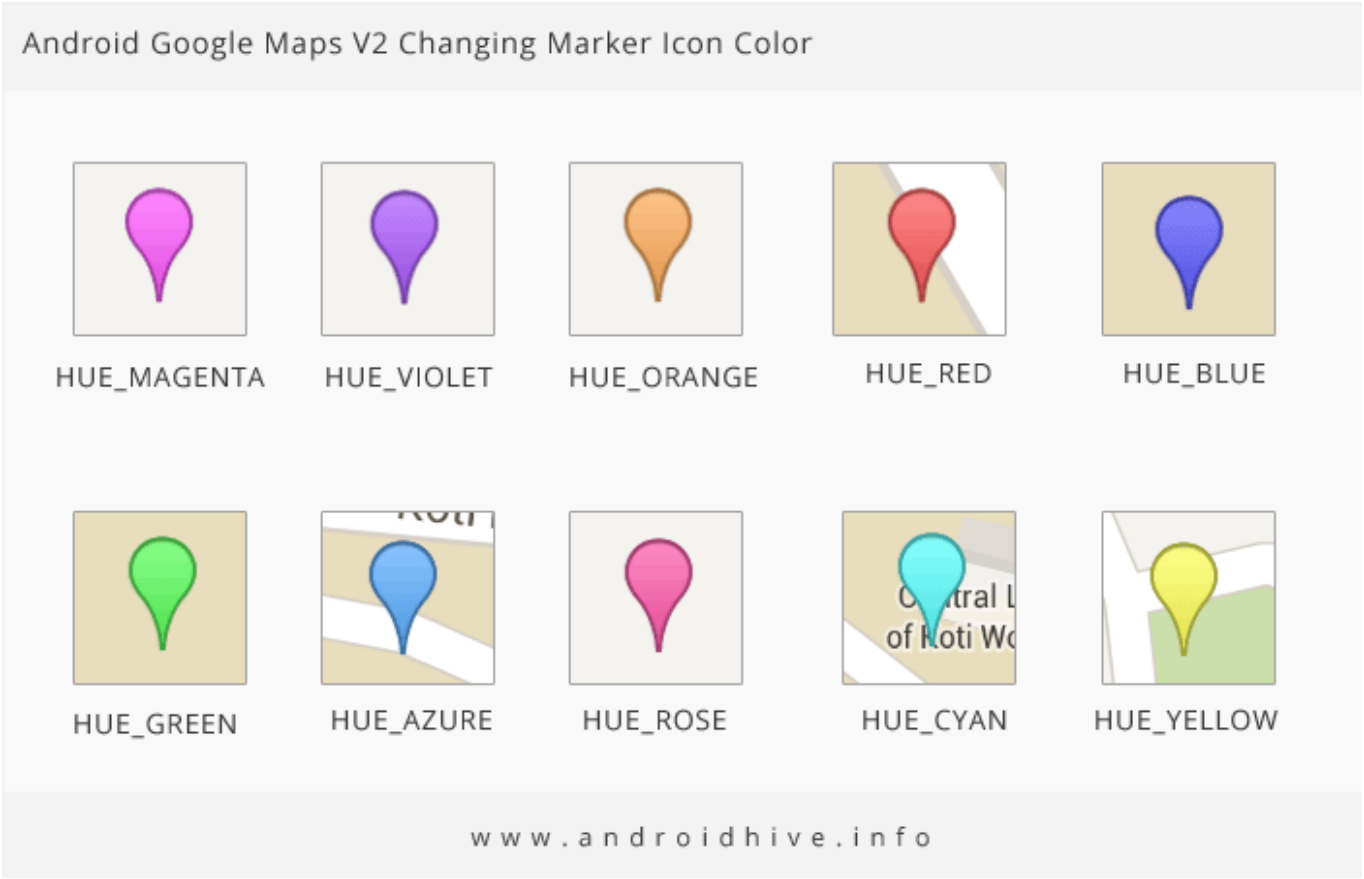
DESIGN TIPS DOWNLOAD ERRORS



By default map marker color will be RED. Google maps provides some set of predefined colored ic for the marker.

```
// ROSE color icon
marker.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_R

// GREEN color icon
marker.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_G
```



Custom Marker Icon

Apart from maps native marker icons, you can use own image to show as a marker. You can load icon from any kind of supported sources.

`fromAsset(String assetName)` – Loading from assets folder

Below I loaded a custom marker icon from **drawable** folder

```
// latitude and longitude
double latitude = 17.385044;
double longitude = 78.486671;

// create marker
MarkerOptions marker = new MarkerOptions().position(new LatLng(latitude, longit

// Changing marker icon
marker.icon(BitmapDescriptorFactory.fromResource(R.drawable.my_marker_icon)));

// adding marker
googleMap.addMarker(marker);
```

Moving Camera to a Location with animation

You may want to move camera to a particular position. Google maps provides set of functions to achi this.

```
CameraPosition cameraPosition = new CameraPosition.Builder().target(
                new LatLng(17.385044, 78.486671)).zoom(12).build();

googleMap.animateCamera(CameraUpdateFactory.newCameraPosition(cameraPosition));
```

Following are enhancements and features that google maps provides. You can utilize these featu which suites to your requirements.

Changing Map Type

Google provides 4 kinds of map types **Normal**, **Hybrid**, **Satellite** and **Terrain**. You can toggle to any k of map using `googleMap.setMapType()` method.

```
googleMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
googleMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
googleMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
googleMap.setMapType(GoogleMap.MAP_TYPE_TERRAIN);
googleMap.setMapType(GoogleMap.MAP_TYPE_NONE);
```

ANDROIDHIVE

DESIGN TIPS DOWNLOAD ERRORS

You can show user's current location on the map by calling `setMyLocationEnabled()`. Pass tru false to enable or disable this feature

```
googleMap.setMyLocationEnabled(true); // false to disable
```

Zooming Buttons

You can call `setZoomControlsEnabled()` function to get rid of those zooming controls on the m By disabling these buttons map zooming functionality still work by pinching gesture.

```
googleMap.getUiSettings().setZoomControlsEnabled(false); // true to enable
```

Zooming Functionality

You can disable zooming gesture functionality by calling `setZoomGesturesEnabled()`

```
googleMap.getUiSettings().setZoomGesturesEnabled(false);
```

Compass Functionality

Compass can be disabled by calling `setCompassEnabled()` function

```
googleMap.getUiSettings().setCompassEnabled(true);
```

My Location Button

My location button will be used to move map to your current location. This button can be show hidden by calling `setMyLocationButtonEnabled()` function

```
googleMap.getUiSettings().setMyLocationButtonEnabled(true);
```

My rotate gesture can be enabled or disabled by calling `setRotateGesturesEnabled()` method

```
googleMap.getUiSettings().setRotateGesturesEnabled(true);
```

Although google maps provides lot of other features, I covered only basic topics in this tutor
Remaining topics seems to be pretty much lengthy, so I'll post them as separate articles.

ANDROIDHIVE

DESIGN TIPS DOWNLOAD ERRORS

ANDROIDHIVE

DESIGN TIPS DOWNLOAD ERRORS

Tag Cloud

Action Bar Adapter Animation Apps
Async Beginner Camera Dashboard
Database facebook GCM Gestures
Google GPS Grid Intermediate json
List View Maps MySQL Navigation Drawer
PHP Pinch Quick Tips sessions Spinner
SQLite Swipe Tab View Twitter UI
Video View Pager xml

ANDROIDHIVE

DESIGN TIPS DOWNLOAD ERRORS

1 Android SQLite Database Tutorial - 559,894 views

2 Android Custom ListView with Image and Text - 457,757 views

3 Android JSON Parsing Tutorial - 439,975 views

4 How to connect Android with PHP, MySQL - 410,455 views

5 Android Push Notifications using Google Cloud Messaging (GCM), PHP and MySQL - 329,651 views

6 Android Tab Layout Tutorial - 307,143 views

7 Android Login and Registration with PHP, MySQL and SQLite - 305,154 views

8 Android XML Parsing Tutorial - 239,803 views

9 Android Login and Registration Screen Design - 227,771 views

10 Android ListView Tutorial - 210,785 views

NETWORK

DESIGN
design.androidhive.info

TIPS
tips.androidhive.info

ERRORS
errors.androidhive.info

DOWNLOAD
download.androidhive.info

115332 subscribers & **922510** downloads

REQUEST TUTORIAL

Email / Name *

DESIGN TIPS DOWNLOAD ERRORS

SEND