Location-Based Services

Coventry University

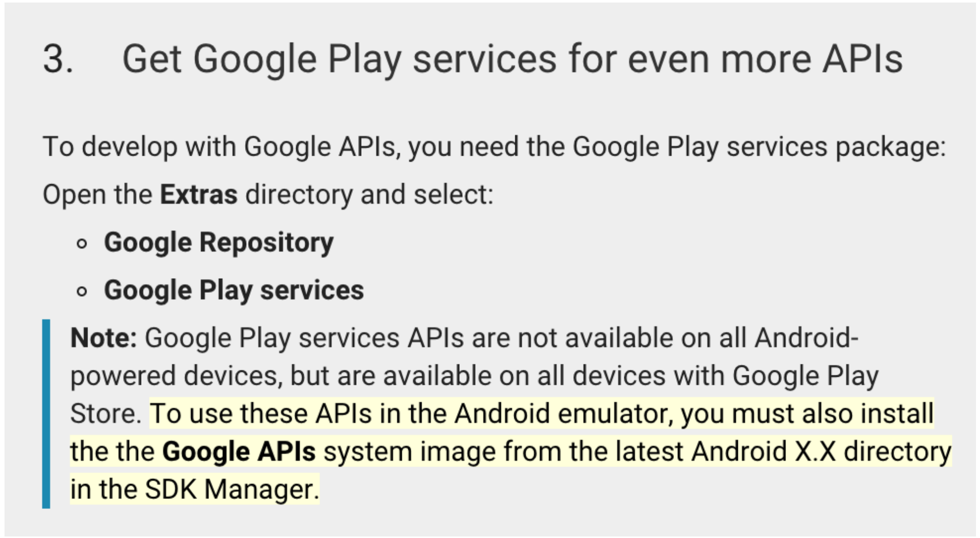
Contents

- Google play services
- Displaying maps

GOOGLE PLAY SERVICES

Setting Up : to develop

To develop an app using the Google Play services APIs, you need to set up your project with the Google Play services SDK


3. Get Google Play services for even more APIs

To develop with Google APIs, you need the Google Play services package:

Open the **Extras** directory and select:

- **Google Repository**
- **Google Play services**

Note: Google Play services APIs are not available on all Android-powered devices, but are available on all devices with Google Play Store. To use these APIs in the Android emulator, you must also install the the **Google APIs** system image from the latest Android X.X directory in the SDK Manager.

http://developer.android.com/sdk/installing/adding-packages.html

Setting Up : to develop

```
apply plugin: 'com.android.application'
...

dependencies {
    compile 'com.android.support:appcompat-v7:21.0.3'
    compile 'com.google.android.gms:play-services:6.5.87'
}
```

```
<meta-data android:name="com.google.android.gms.version"
           android:value="@integer/google_play_services_version" />
```

Setting Up : to debug

- A compatible Android device that runs Android 2.3 or higher and includes Google Play Store.
- The Android emulator with an AVD that runs the Google APIs platform based on Android 4.2.2 or higher.

Get permissions

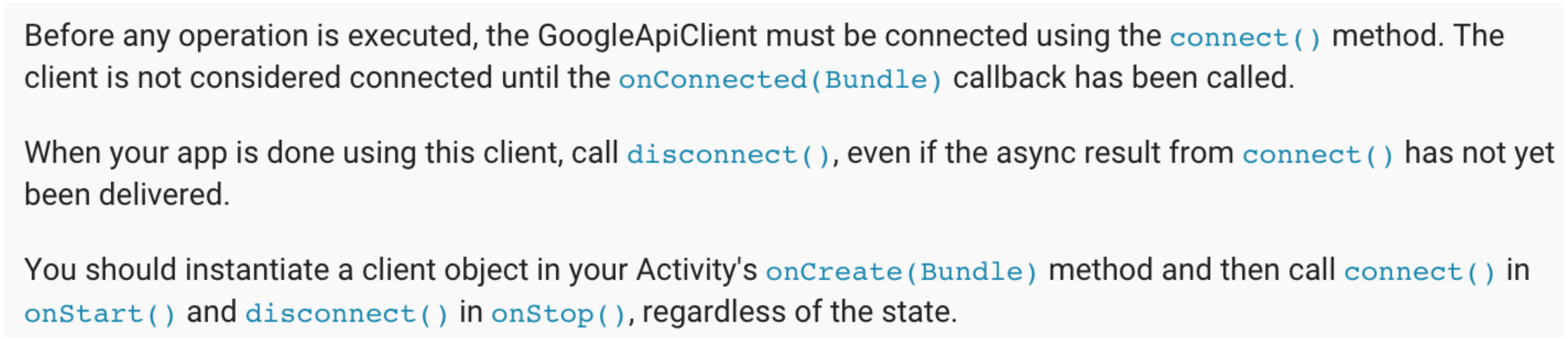
```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.google.android.gms.location.sample.basiclocationsample" >

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

API Client builder

```java
protected synchronized void buildGoogleApiClient() {
    mGoogleApiClient = new GoogleApiClient.Builder(this)
        .addConnectionCallbacks(this)
        .addOnConnectionFailedListener(this)
        .addApi(LocationServices.API)
        .build();
}
```

```java
GoogleApiClient client = new GoogleApiClient.Builder(this)
         .addApi(Plus.API)
         .addScope(Plus.SCOPE_PLUS_LOGIN)
         .setAccountName("users.account.name@gmail.com")
         .build();
client.connect();
```

API Client interface



Before any operation is executed, the GoogleApiClient must be connected using the `connect()` method. The client is not considered connected until the `onConnected(Bundle)` callback has been called.

When your app is done using this client, call `disconnect()`, even if the async result from `connect()` has not yet been delivered.

You should instantiate a client object in your Activity's `onCreate(Bundle)` method and then call `connect()` in `onStart()` and `disconnect()` in `onStop()`, regardless of the state.

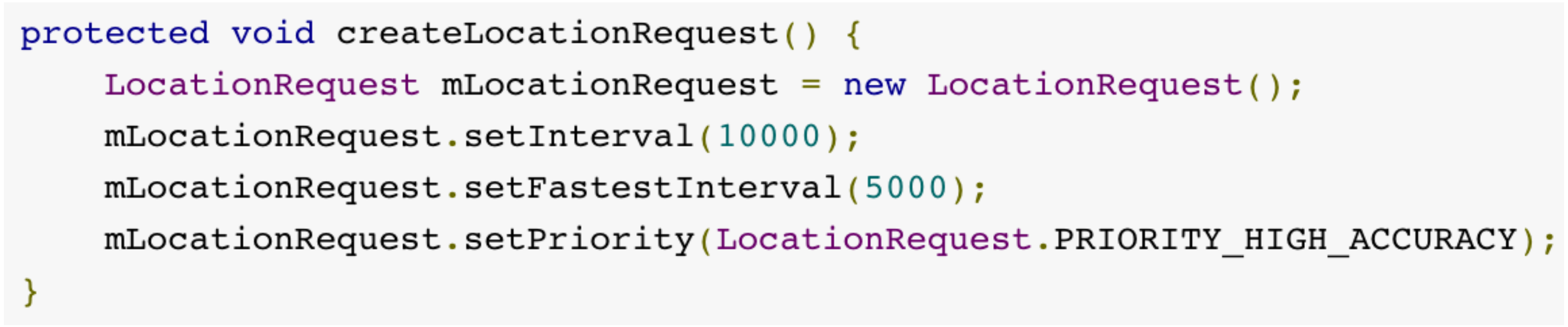
Last location

```
public class MainActivity extends ActionBarActivity implements
        ConnectionCallbacks, OnConnectionFailedListener {
    ...
    @Override
    public void onConnected(Bundle connectionHint) {
        mLastLocation = LocationServices.FusedLocationApi.getLastLocation(
                mGoogleApiClient);
        if (mLastLocation != null) {
            mLatitudeText.setText(String.valueOf(mLastLocation.getLatitude()));
            mLongitudeText.setText(String.valueOf(mLastLocation.getLongitude()));
        }
    }
}
```

Location updates

```
@Override
public void onConnected(Bundle connectionHint) {
    ...
    if (mRequestingLocationUpdates) {
        startLocationUpdates();
    }
}

protected void startLocationUpdates() {
    LocationServices.FusedLocationApi.requestLocationUpdates(
            mGoogleApiClient, mLocationRequest, this);
}
```

Location updates

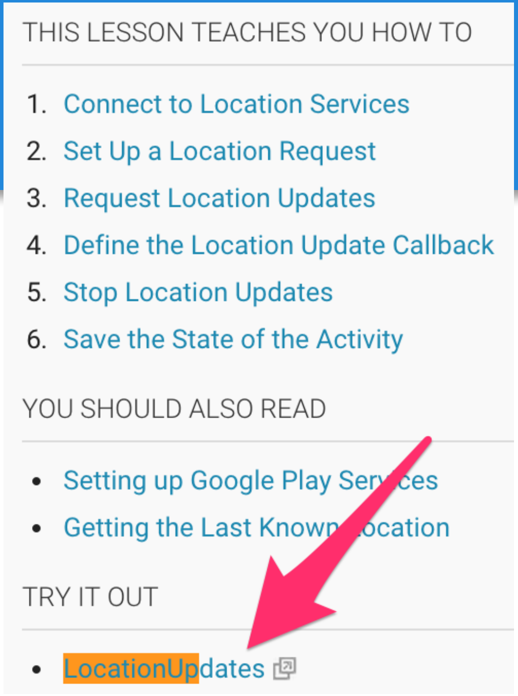


```
protected void createLocationRequest() {
    LocationRequest mLocationRequest = new LocationRequest();
    mLocationRequest.setInterval(10000);
    mLocationRequest.setFastestInterval(5000);
    mLocationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
}
```

- setInterval() - This method sets the rate in milliseconds at which your app prefers to receive location updates.
- setFastestInterval() - This method sets the fastest rate in milliseconds at which your app can handle location updates.
- setPriority() - This method sets the priority of the request, which gives the Google Play services location services a strong hint about which location sources to use.

Stop updates

```
@Override
protected void onPause() {
    super.onPause();
    stopLocationUpdates();
}

protected void stopLocationUpdates() {
    LocationServices.FusedLocationApi.removeLocationUpdates(
            mGoogleApiClient, this);
}
```



THIS LESSON TEACHES YOU HOW TO

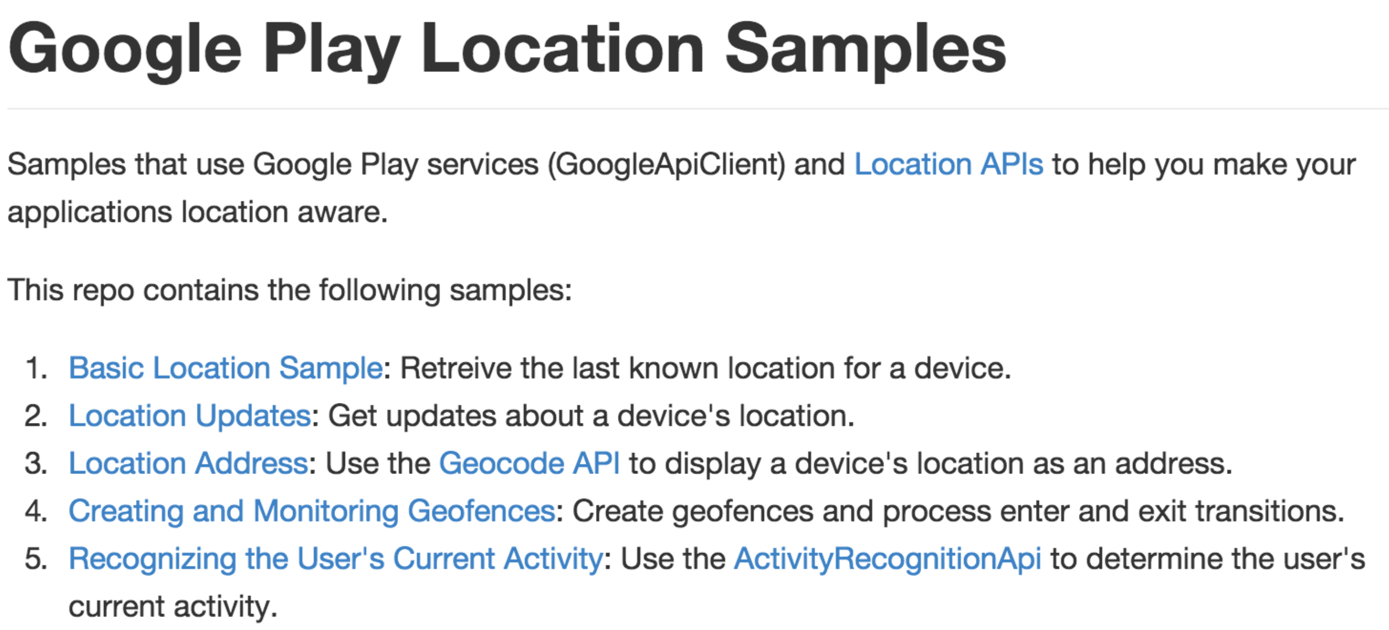
1. Connect to Location Services
2. Set Up a Location Request
3. Request Location Updates
4. Define the Location Update Callback
5. Stop Location Updates
6. Save the State of the Activity

YOU SHOULD ALSO READ

- Setting up Google Play Services
- Getting the Last Known Location

TRY IT OUT

- LocationUpdates



Google Play Location Samples

Samples that use Google Play services (GoogleApiClient) and Location APIs to help you make your applications location aware.

This repo contains the following samples:

1. Basic Location Sample: Retreive the last known location for a device.
2. Location Updates: Get updates about a device's location.
3. Location Address: Use the Geocode API to display a device's location as an address.
4. Creating and Monitoring Geofences: Create geofences and process enter and exit transitions.
5. Recognizing the User's Current Activity: Use the ActivityRecognitionApi to determine the user's current activity.

https://github.com/googlesamples/android-play-location

Javadoc

```
/**
 * Runs when a GoogleApiClient object successfully connects.
 */
@Override
public void onConnected(Bundle connectionHint) {
    Log.i(TAG, "Connected to GoogleApiClient");

    // If the initial location was never previously requested, we use
    // FusedLocationApi.getLastLocation() to get it. If it was previously requested, we store
    // its value in the Bundle and check for it in onCreate(). We
    // do not request it again unless the user specifically requests location updates by pressing
    // the Start Updates button.
    //
    // Because we cache the value of the initial location in the Bundle, it means that if the
    // user launches the activity,
    // moves to a new location, and then changes the
    // is displayed as the activity is re-created.
    if (mCurrentLocation == null) {
        mCurrentLocation = LocationServices.FusedLoc
        mLastUpdateTime = DateFormat.getTimeInstance
        updateUI();
    }

    // If the user presses the Start Updates button
    // mRequestingLocationUpdates to true (see start
    // the value of mRequestingLocationUpdates and i
    if (mRequestingLocationUpdates) {
        startLocationUpdates();
    }
}
```

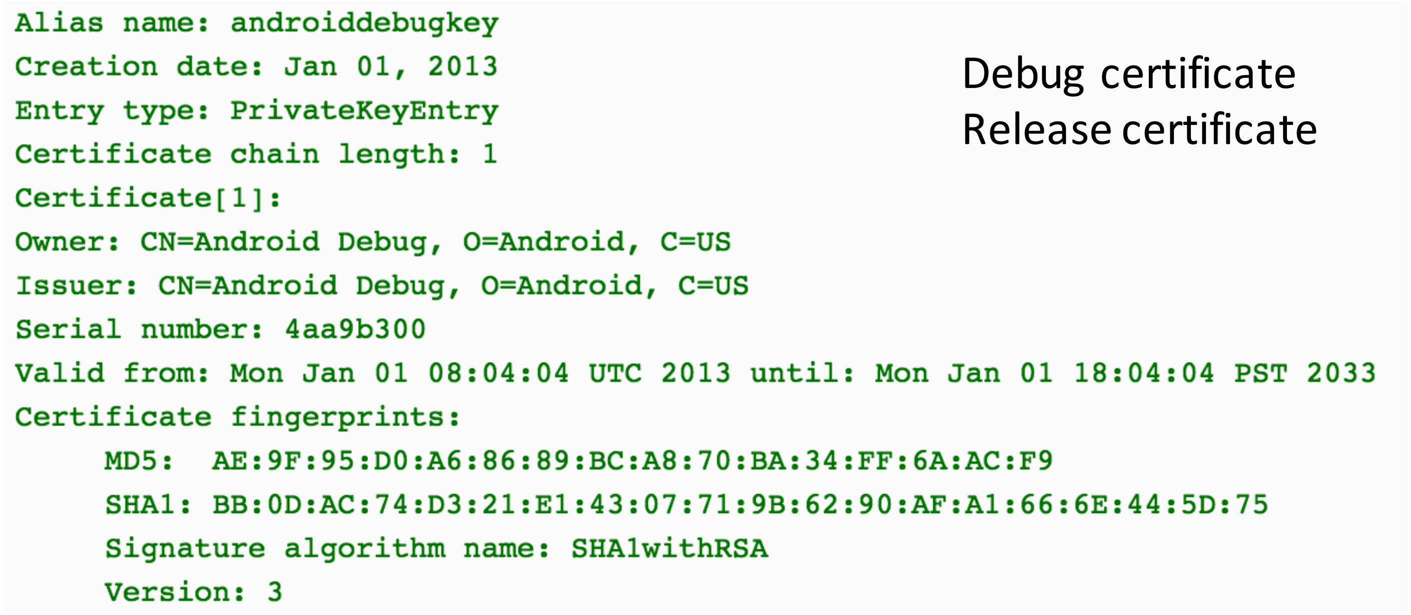
```
/**
 * Short one line description.                              (1)
 * <p>
 * Longer description. If there were any, it would be       [2]
 * here.
 * <p>
 * And even more explanations to follow in consecutive
 * paragraphs separated by HTML paragraph breaks.
 *
 * @param  variable Description text text text.            (3)
 * @return Description text text text.
 */
public int methodName (...) {
    // method body with a return statement
}
```

MAPS

Configure

1. Retrieve information about your application's certificate.
2. Register a project in the Google APIs Console and add the Maps API as a service for the project.
3. Request one or more keys.
4. Add your key to your application and begin development.

SHA-1 fingerprint

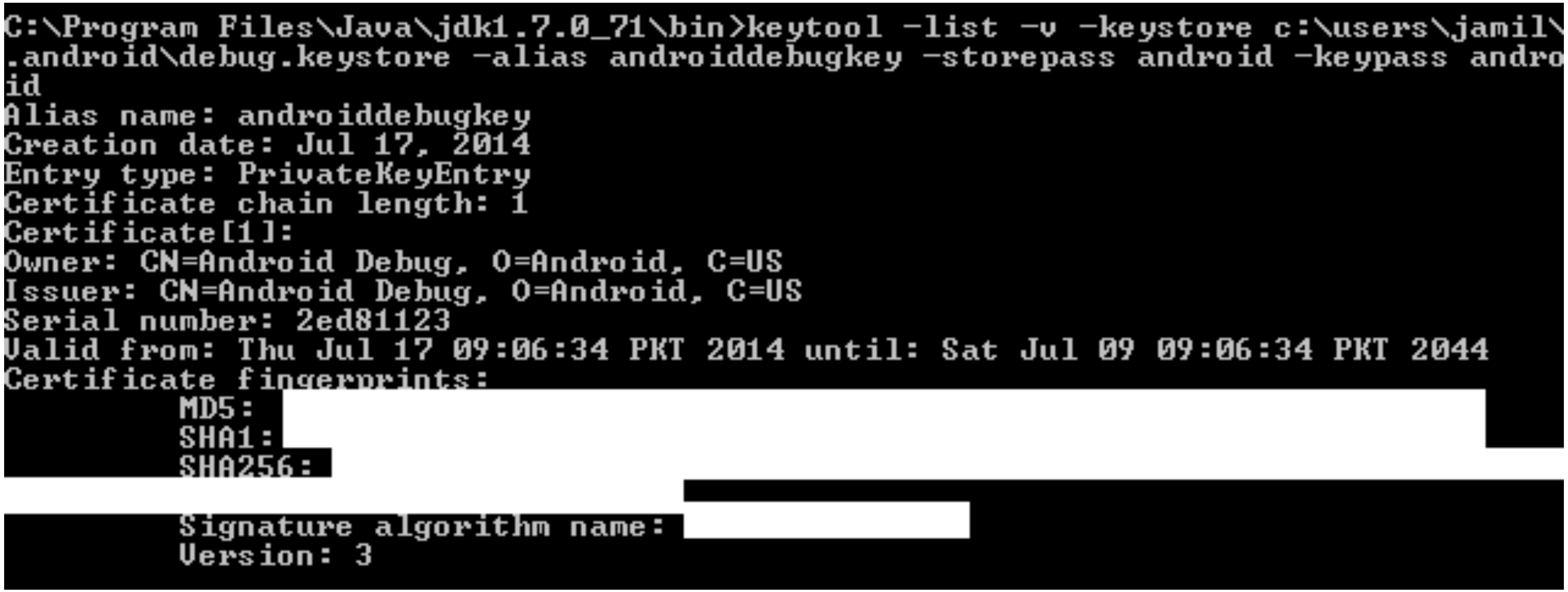


```
Alias name: androiddebugkey
Creation date: Jan 01, 2013
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=Android Debug, O=Android, C=US
Issuer: CN=Android Debug, O=Android, C=US
Serial number: 4aa9b300
Valid from: Mon Jan 01 08:04:04 UTC 2013 until: Mon Jan 01 18:04:04 PST 2033
Certificate fingerprints:
     MD5:  AE:9F:95:D0:A6:86:89:BC:A8:70:BA:34:FF:6A:AC:F9
     SHA1: BB:0D:AC:74:D3:21:E1:43:07:71:9B:62:90:AF:A1:66:6E:44:5D:75
     Signature algorithm name: SHA1withRSA
     Version: 3
```

Debug certificate
Release certificate

https://developers.google.com/maps/documentation/android/start?hl=fr#install_and_configure_the_google_play_services_sdk

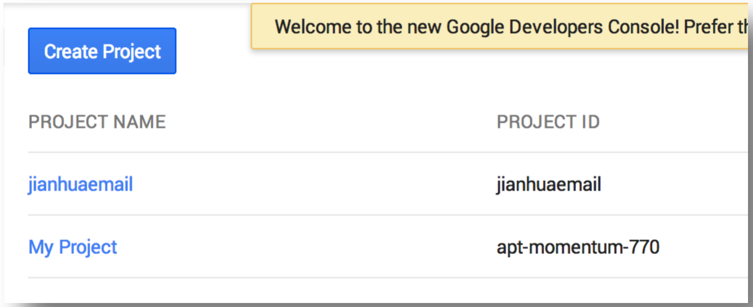
SHA-1 fingerprint



```
C:\Program Files\Java\jdk1.7.0_71\bin>keytool -list -v -keystore c:\users\jamil\
.android\debug.keystore -alias androiddebugkey -storepass android -keypass andro
id
Alias name: androiddebugkey
Creation date: Jul 17, 2014
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=Android Debug, O=Android, C=US
Issuer: CN=Android Debug, O=Android, C=US
Serial number: 2ed81123
Valid from: Thu Jul 17 09:06:34 PKT 2014 until: Sat Jul 09 09:06:34 PKT 2044
Certificate fingerprints:
         MD5: [redacted]
         SHA1:
         SHA256:

         Signature algorithm name: [redacted]
         Version: 3
```

http://stackoverflow.com/questions/27609442/how-to-get-the-sha1-fingerprint-certificate-in-android-studio-for-debug-mode

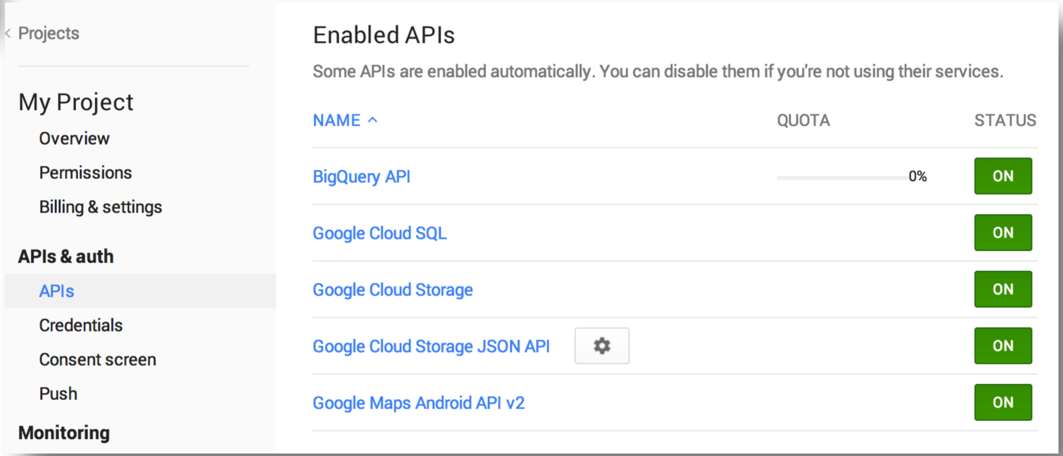
Google Developers Console



Create Project

Welcome to the new Google Developers Console! Prefer th

PROJECT NAME	PROJECT ID
jianhuaemail	jianhuaemail
My Project	apt-momentum-770

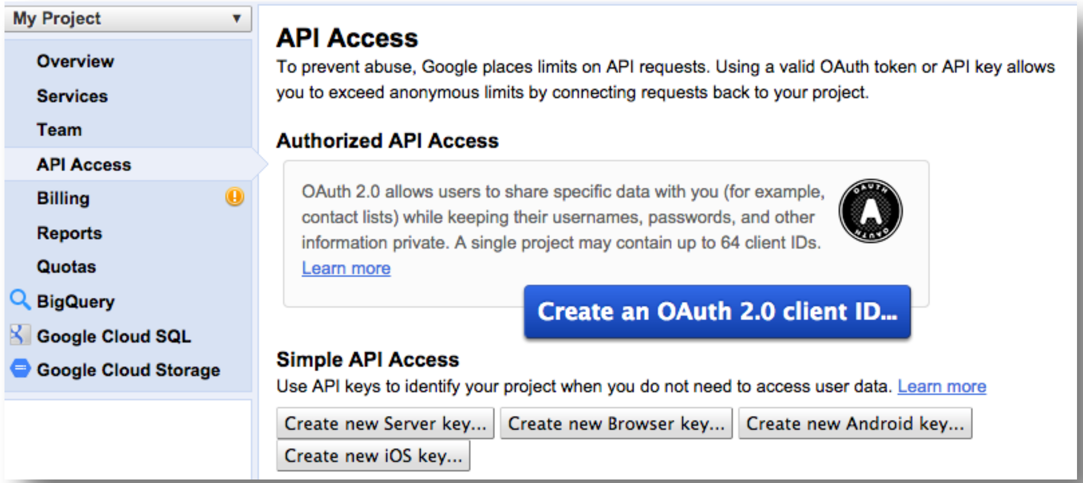


Enabled APIs

Some APIs are enabled automatically. You can disable them if you're not using their services.

NAME ^	QUOTA	STATUS
BigQuery API	0%	ON
Google Cloud SQL		ON
Google Cloud Storage		ON
Google Cloud Storage JSON API		ON
Google Maps Android API v2		ON

The API key



API Access

To prevent abuse, Google places limits on API requests. Using a valid OAuth token or API key allows you to exceed anonymous limits by connecting requests back to your project.

Authorized API Access

OAuth 2.0 allows users to share specific data with you (for example, contact lists) while keeping their usernames, passwords, and other information private. A single project may contain up to 64 client IDs. Learn more

Simple API Access

Use API keys to identify your project when you do not need to access user data. Learn more

```xml
<meta-data
    android:name="com.google.android.maps.v2.API_KEY"
    android:value="API_KEY"/>
```

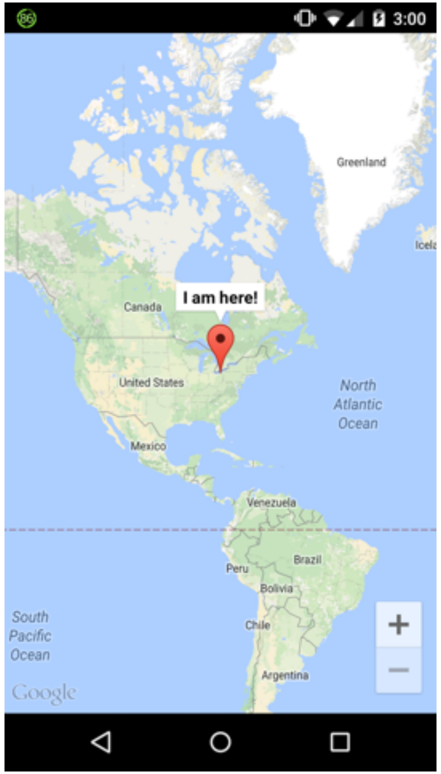
Basic map

```xml
<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
          android:id="@+id/map"
          android:layout_width="match_parent"
          android:layout_height="match_parent"
          android:name="com.google.android.gms.maps.MapFragment"/>
```

```java
public class MainActivity extends Activity {

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

Markers

```
private void handleNewLocation(Location location) {
    Log.d(TAG, location.toString());

    double currentLatitude = location.getLatitude();
    double currentLongitude = location.getLongitude();
    LatLng latLng = new LatLng(currentLatitude, currentLongitude);

    MarkerOptions options = new MarkerOptions()
        .position(latLng)
        .title("I am here!");
    mMap.addMarker(options);
    mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
}
```



http://blog.teamtreehouse.com/beginners-guide-location-android