

# Obtaining a Google Maps API Key

---

In order to use the Google Maps functionality in Android, you need to register for a Maps API key with Google. Until you do this, you will just see a blank grid instead of a map in your applications. You must obtain a Google Maps Android API v2 key - keys from the older Google Maps Android API key v1 will not work.

Obtaining a Maps API v2 key involves the following steps:

1. Retrieve the SHA1 fingerprint of the keystore that is used to sign the application.
2. Create a project in the Google APIs console.
3. Obtaining the API key.

## Obtaining your Signing Key Fingerprint

---

In order to request a Maps API key from Google, you need to know the SHA1 fingerprint of the keystore that is used to sign the application. Typically, this means you will have to determine the SHA1 fingerprint for the debug keystore, and then the SHA1 fingerprint for the keystore that is used to sign your application for release.

By default the keystore that is used to sign debug versions of a Xamarin.Android application can be found at the following location:

- **Windows** - `C:\Users\[USERNAME]\AppData\Local\Xamarin\Mono for Android\debug.keystore`
- **OSX** - `/Users/[USERNAME]/.local/share/Xamarin/Mono for Android/debug.keystore`

Information about a keystore is obtained by running the `keytool` command from the JDK. This tool is typically found in the Java bin directory:

- **OSX** - `/System/Library/Java/JavaVirtualMachines/[VERSION].jdk/Contents/Home/bin/keytool`
- **Windows** - `C:\Program Files (x86)\Java\jdk[VERSION]\bin\keytool.exe`

Run `keytool` using the following command:

```
keytool -list -v -keystore [STORE FILENAME] -alias [KEY NAME] -storepass [STORE PASSWORD] -keypass [KEY PASSWORD]
```

For the debug key, this command will look like:

```
keytool -list -v -keystore /Users/[USERNAME]/.local/share/Xamarin/Mono\ for\
Android/debug.keystore -alias androiddebugkey -storepass android -keypass android
```

You should see something like the following output in your console window:

```
Alias name: androiddebugkey
Creation date: Jan 01, 2013
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
```

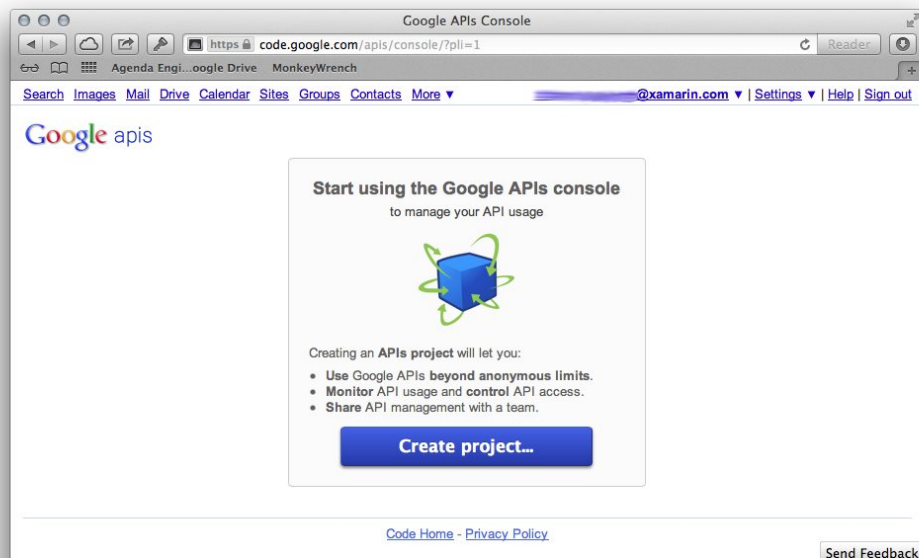
Owner: CN=Android Debug, O=Android, C=US  
Issuer: CN=Android Debug, O=Android, C=US  
Serial number: 4aa9b300  
Valid from: Mon Jan 01 08:04:04 UTC 2013 until: Mon Jan 01 18:04:04 PST 2033  
Certificate fingerprints:  
MD5: AE:9F:95:D0:A6:86:89:BC:A8:70:BA:34:FF:6A:AC:F9  
SHA1: BB:0D:AC:74:D3:21:E1:43:07:71:9B:62:90:AF:A1:66:6E:44:5D:75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

You will need the SHA1 fingerprint later on.

## Creating an API project

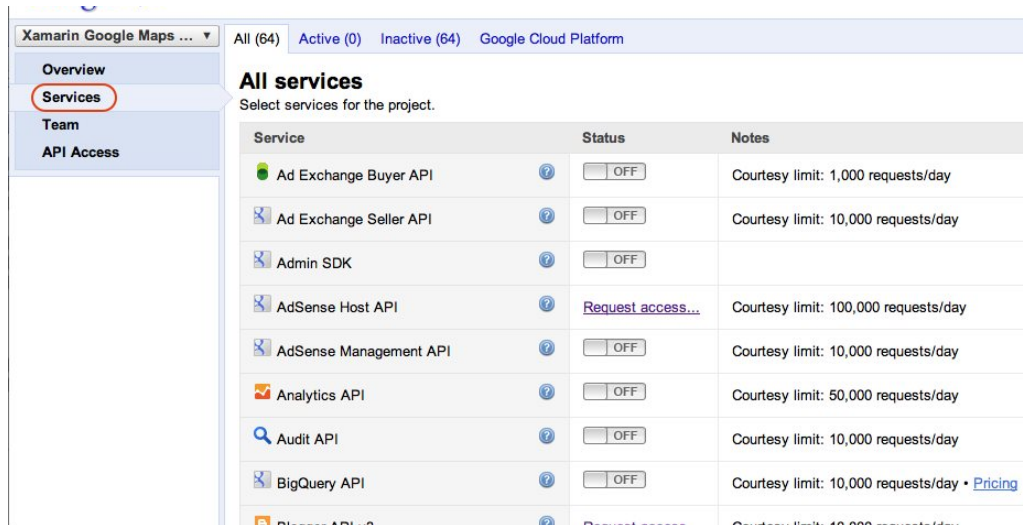
Once you have the SHA1 fingerprint of the signing keystores, it is necessary to create a new project in the Google APIs console, or to add the Google Maps Android API v2 service to an existing project.

1. In a browser, navigate to the [Google APIs Console](https://code.google.com/apis/console/).
2. The first time you use the Google API console, you will be prompted to create a new project, as shown in the following screenshot:



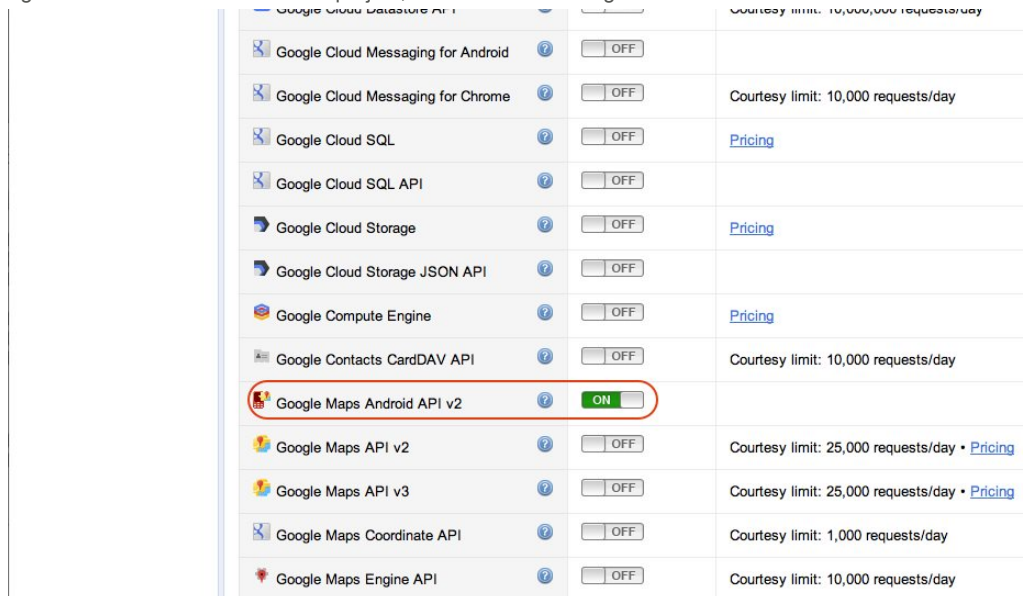
3. .
4. Click the Create Project button to create the project.
5. Click on the Services tab in the left navigation bar. This will display a list of all services that are available to a project. You can see an example of this in the screenshot below:

Google apis



6.

7. Scroll down the list of services until the Google Maps Android API v2 service is visible. Toggle the switch indicator at the right to turn the service on for this project, as shown in the image below:



8.

9. Once you turn the service on, the terms of service will be displayed to you. Click on the checkbox and click Accept.

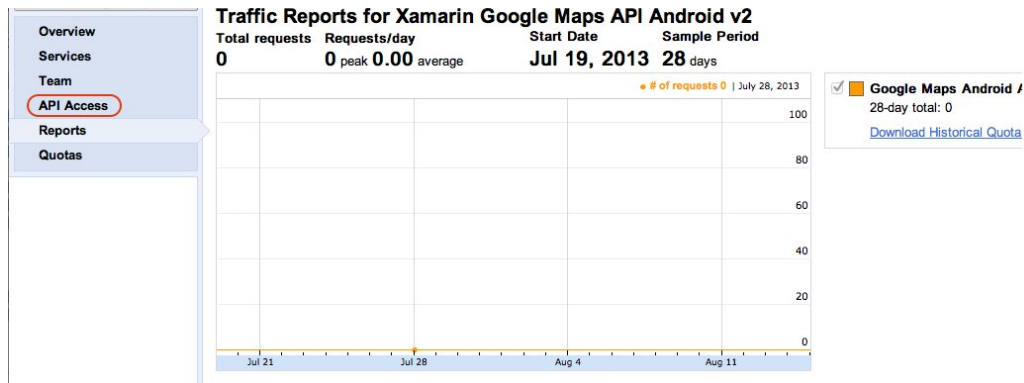
At this point the API project has been created and the Google Maps Android API v2 has been added to it. Next we will look at how to create an API key and white list a Xamarin.Android application so that it is authorized to use this key.

## Obtaining the API Key

Once the Console API project has been created, it is necessary to create an Android API key. Xamarin.Android applications require the API key before they will be granted access to Android Map API v2.

1. Navigate to the Console API, and click on the API access:





2.

3. In the next page, click on the Create New Android Key button.

**Google apis**

Xamarin Google Maps ...

**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 20 client IDs. [Learn more](#)

**Create an OAuth 2.0 client ID...**

**Simple API Access**

Use API keys to identify your project when you do not need to access user data. [Learn more](#)

Create new Server key... Create new Browser key... **Create new Android key...** Create new iOS key...

**Notification Endpoints**

Use notification endpoints to identify domains that may receive webhook notifications from your API. [Learn more](#)

Allowed Domains: No domains allowed [Edit...](#)

4.

5. Next, it will be necessary to white-list an application with this API key. In the dialog that pops up, enter the SHA1 fingerprint, followed by a semi-colon, followed by the package name of your application. The following line is an example:

6. BB:0D:AC:74:D3:21:E1:43:67:71:9B:62:91:AF:A1:66:6E:44:5D:75;com.example.android.mapexample

7. This can be seen in the following screenshot:

**Google apis**

Xamarin Google Maps ...

**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.

**Configure Android Key for Xamarin Google Maps API Android v2**

This key can be deployed in your Android applications.

API requests are sent directly to Google from your clients' Android devices. Google verifies that each request originates from an Android application that matches one of the certificate SHA1 fingerprints and package names listed below. You can discover the SHA1 fingerprint of your developer certificate using the following command:

```
keytool -list -v -keystore mystore.keystore
```

[Learn more](#)

**Accept requests from an Android application with one of the certificate fingerprints and package names listed below:**

```
55:63:73:70:D8:AB:FD:FB:DA:46:7B:70:BE:E8:FB:1A:02:22:0F;com.xamarin.demo.maps
```

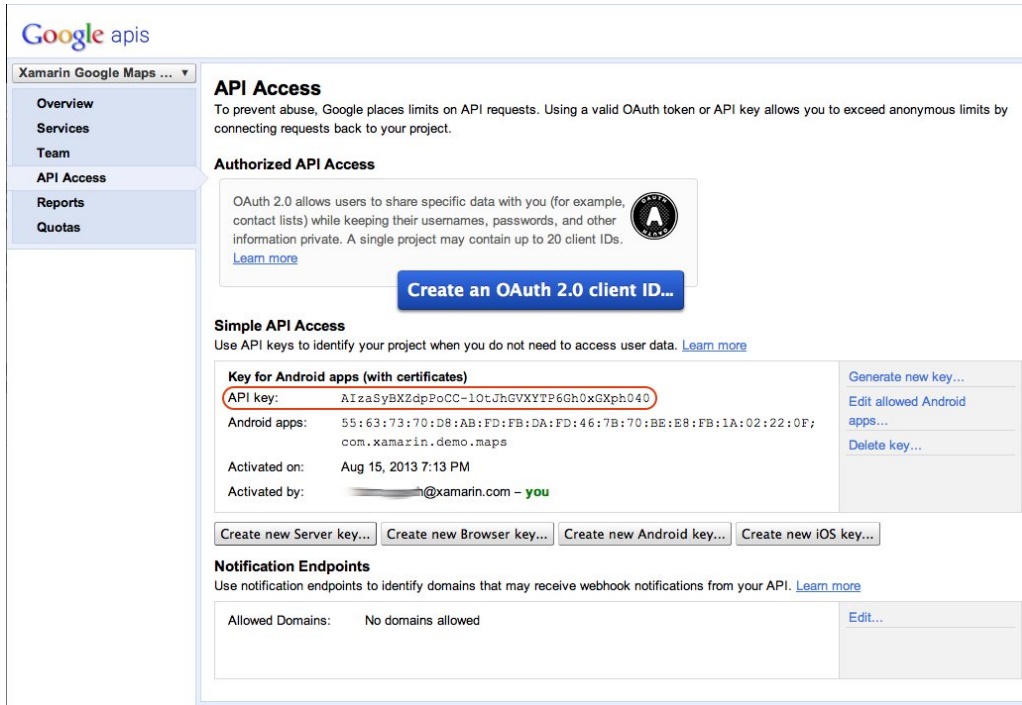
One SHA1 certificate fingerprint and package name (separated by a semicolon) per line. Example:

```
45:B5:E4:6F:36:AD:0A:98:94:B4:02:66:2B:12:17:F2:56:26:A0:E0;com.example
```

8.



9. Click on the Create button, which will return you to the API Access screen, which will display the API key and the Android apps that are authorized to use the API key. An example of this can be seen below:



10.

This is the API key that will be added to the AndroidManifest.XML file of a Xamarin.Android application.