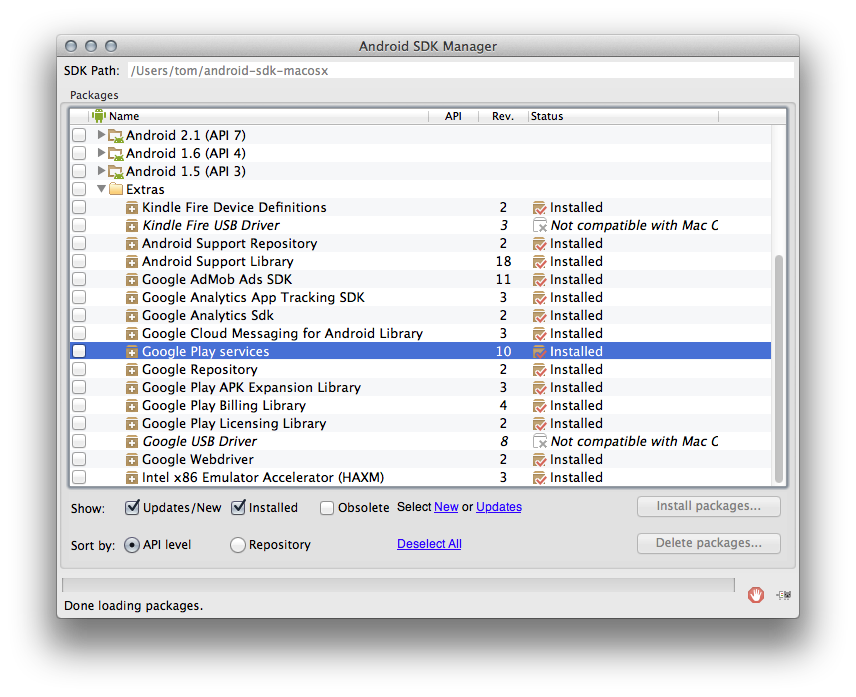
**Install Google Play services**

**Download the Google Play services**

Open the Android SDK Manager and install Extras ▸ Google Play services

If already Google Play services install ignore it.

****

**Create a library project for the Google Play Service library**

Add Google Play Services – Maps Component or Manage NuGet Packages to your Project.

**Google Maps API key**

Your application needs an API key to access the Google Maps servers. The type of key you need is an API key with restriction for **Android apps**. The key is free. You can use it with any of your applications that call the Google Maps Android API, and it supports an unlimited number of users.

You get this key via the <https://code.google.com/apis/console/> - Google APIs Console. You have to provide your application signature key (**SHA-1**) and the application package name. This is based on the key with which you sign your Android application during deployment. During development with Visual Studio or Xamarin Studio.

**Creating the SHA-1 for your signature key**

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

C:\Users\[USERNAME]\AppData\Local\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

C:\Program Files (x86)\Java\jdk[VERSION]\bin\keytool.exe

Run keytool using the following command (using the file paths shown above)

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

## Debug.keystore Example

Run this below command in Command Prompt.

C:\Program Files (x86)\Java\jdk1.7.0\_71\bin>keytool -list -v -keystore "C:\Users\[USERNAME]\AppData\Local\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, 2016

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

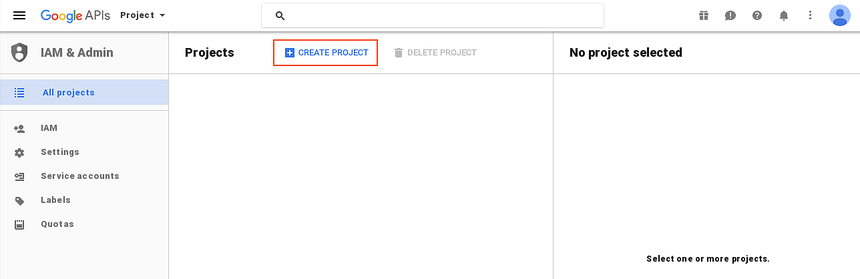
Signature algorithm name: SHA1withRSA

Version: 3

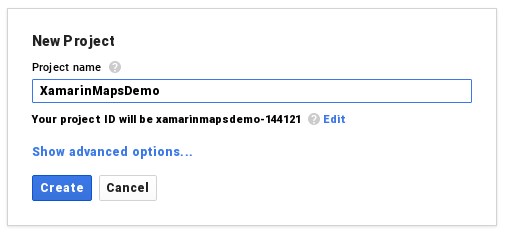
You will use the SHA-1 fingerprint (listed after **SHA1**) later in this guide.

**Register with the Google APIs Console**

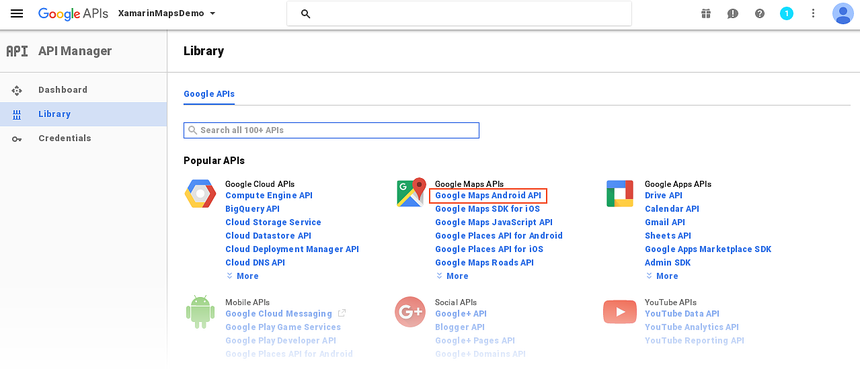
After you have retrieved the SHA-1 fingerprint of the signing keystore, you have to register in the Google APIs Console that you want to use Google Maps for Android. You can reach this console via the following link: <https://code.google.com/apis/console/> and click **CREATE PROJECT**:

****

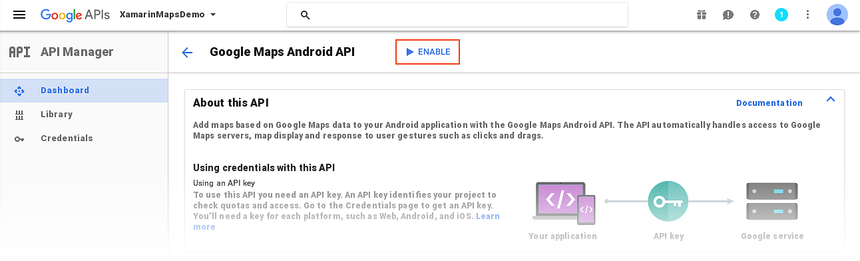
In the **New Project** dialog that appears, enter the project name. The dialog will manufacture a unique project ID that is based on your project name, as shown in this example:

****

Click the **Create** button. After a minute or so, the project is created and you are taken to the **API Manager** page. In the **Library** section, click **Google Maps Android API**:



At the top of the Google Maps Android API page, click ENABLE to turn on the service for this project

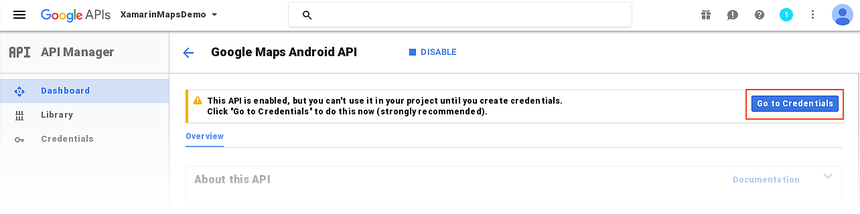
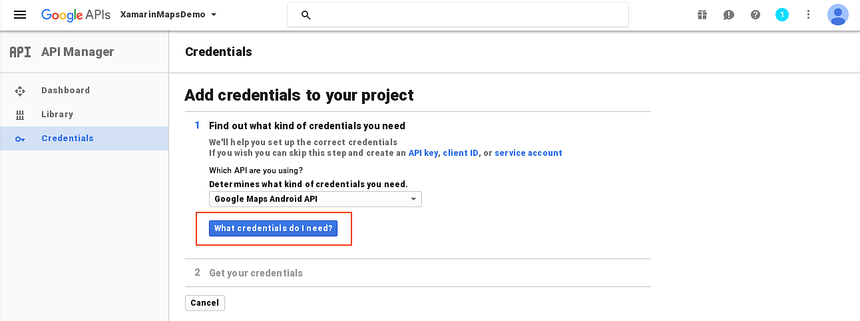


At this point the API project has been created and the Google Maps Android API has been added to it. However, you cannot use this API in your project until you create credentials for it. Next we will look at how to create an API key and white-list a Xamarin. Android application so that it is authorized to use this key.

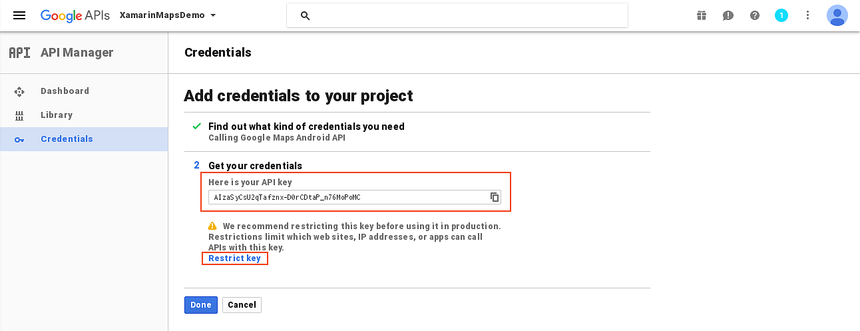
**Create API key for your Application**

After the **Google Developer Console** API project has been created, it is necessary to create an Android API key. Xamarin. Android applications must have an API key before they are granted access to Android Map API.

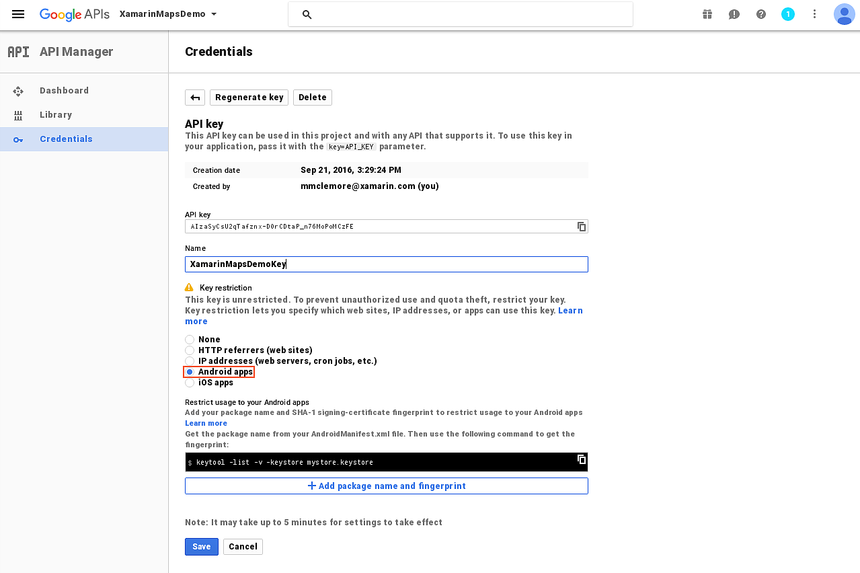
In the **Google Maps Android API** page that is displayed (after clicking **ENABLE** in the previous step), click the **Go to Credentials** button

In the **Credentials** page, click thebutton (**What** **credentials do I need?)**

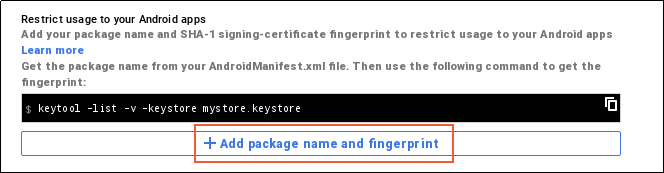
After this button is clicked, the API key is generated. Next it is necessary to restrict this key so that only your app can call APIs with this key. Click **Restrict key**

****

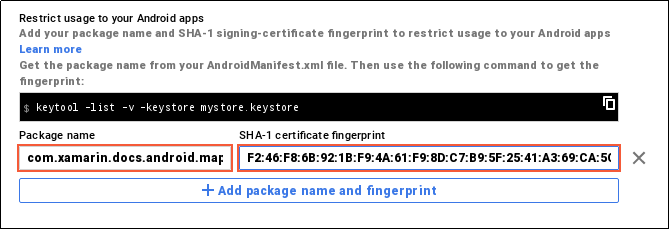
Change the **Name** field from **API Key 1** to a name that will help you remember what the key is used for (**XamarinMapsDemoKey** is used in this example). Next, click the **Android apps** radio button:



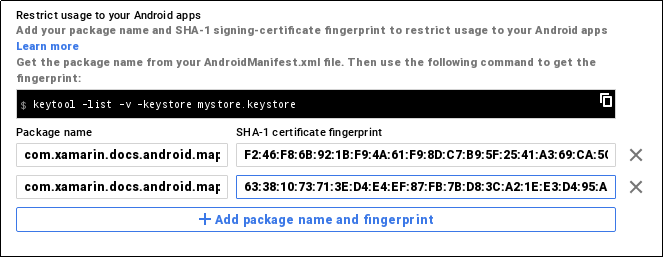
To add the SHA-1 fingerprint, click **+ Add package name and fingerprint**



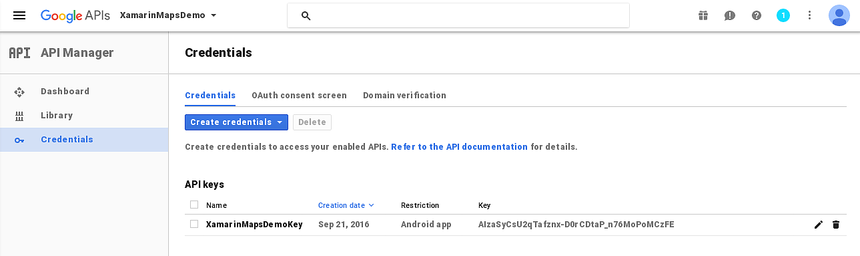
Enter your app's package name and enter the SHA-1 certificate fingerprint (obtained via keytool as explained earlier in this guide). In the following example, the package name for XamarinMapsDemo is entered, followed by the SHA-1 certificate fingerprint obtained from **debug.keystore**



Note that, in order for your APK to access Google Maps, you must include SHA-1 fingerprints and package names for every keystore (debug and release) that you use to sign your APK. For example, if you use one computer for debug and another computer for generating the release APK, you should include the SHA-1 certificate fingerprint from the debug keystore of the first computer and the SHA-1 certificate fingerprint from the release keystore of the second computer. Click **+ Add package name and fingerprint** to add another fingerprint and package name as shown in this example

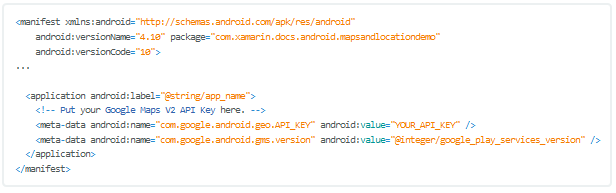


Click the **Save** button to save your changes. Next, you are returned to the list of your API keys. If you have other API keys that you have created earlier, they will also be listed here. In this example, only one API key (created in the previous steps) is listed.



**Adding the Key to Your Project**

Finally, add this API key to the **AndroidManifest.XML** file of your Xamarin. Android app. In the following example, YOUR\_API\_KEY is to be replaced with the API key generated in the previous steps

****

**Example:**

**API KEY**: AIzaSyCIlvD5uOwIKxM8Eve1FktlNjmzOeii\_aw