Chapter 17: Working with Maps

Section 17.1: Adding a map in Xamarin.Forms (Xamarin Studio)

You can simply use the native map APIs on each platform with Xamarin Forms. All you need is to download the *Xamarin.Forms.Maps* package from nuget and install it to each project (including the PCL project).

Maps Initialization

First of all you have to add this code to your platform-specific projects. For doing this you have to add the `Xamarin.FormsMaps.Init` method call, like in the examples below.

iOS project

File AppDelegate.cs

```
[Register("AppDelegate")]
public partial class AppDelegate : Xamarin.Forms.Platform.iOS.FormsApplicationDelegate
{
    public override bool FinishedLaunching(UIApplication app, NSDictionary options)
    {
        Xamarin.Forms.Forms.Init();
        Xamarin.FormsMaps.Init();

        LoadApplication(new App());

        return base.FinishedLaunching(app, options);
    }
}
```

Android project

File MainActivity.cs

```
[Activity(Label = "MapExample.Droid", Icon = "@drawable/icon", Theme = "@style/MyTheme",
MainLauncher = true, ConfigurationChanges = ConfigChanges.ScreenSize | ConfigChanges.Orientation)]
public class MainActivity : Xamarin.Forms.Platform.Android.FormsAppCompatActivity
{
    protected override void OnCreate(Bundle bundle)
    {
        TabLayoutResource = Resource.Layout.Tabbar;
        ToolbarResource = Resource.Layout.Toolbar;

        base.OnCreate(bundle);

        Xamarin.Forms.Forms.Init(this, bundle);
        Xamarin.FormsMaps.Init(this, bundle);

        LoadApplication(new App());
    }
}
```

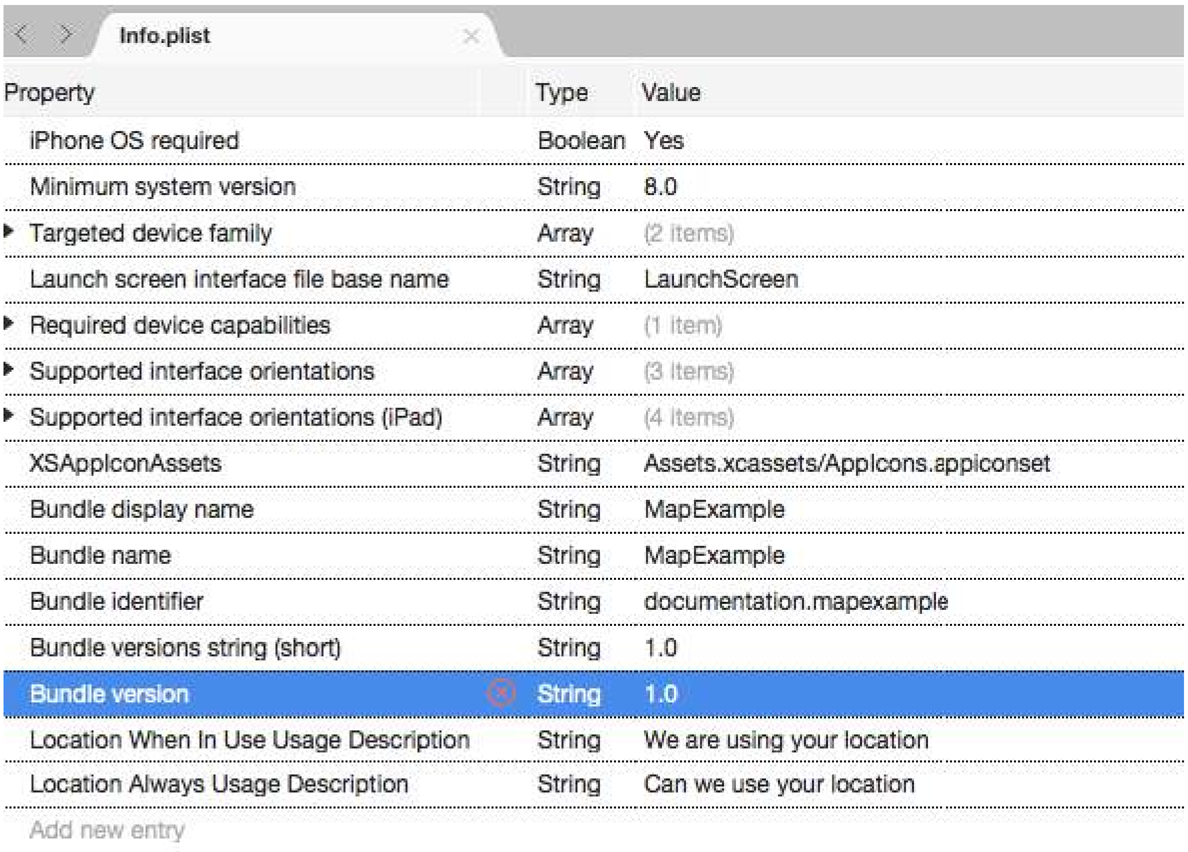
Platform Configuration

Additional configuration steps are required on some platforms before the map will display.

iOS project

In iOS project you just have to add 2 entries to your *Info.plist* file:

- `NSLocationWhenInUseUsageDescription` *string* with value `We are using your location`
- `NSLocationAlwaysUsageDescription` *string* with value `Can we use your location`



Property	Type	Value
iPhone OS required	Boolean	Yes
Minimum system version	String	8.0
▸ Targeted device family	Array	(2 items)
Launch screen interface file base name	String	LaunchScreen
▸ Required device capabilities	Array	(1 item)
▸ Supported interface orientations	Array	(3 items)
▸ Supported interface orientations (iPad)	Array	(4 items)
XSAppIconAssets	String	Assets.xcassets/AppIcons.appiconset
Bundle display name	String	MapExample
Bundle name	String	MapExample
Bundle identifier	String	documentation.mapexample
Bundle versions string (short)	String	1.0
Bundle version	String	1.0
Location When In Use Usage Description	String	We are using your location
Location Always Usage Description	String	Can we use your location
Add new entry		

Android project

To use Google Maps you have to generate an API key and add it to your project. Follow the instruction below to get this key:

1. (Optional) Find where your keytool tool location (default is `/System/Library/Frameworks/JavaVM.framework/Versions/Current/Commands`)
2. (Optional) Open terminal and go to your keytool location:

   ```
   cd /System/Library/Frameworks/JavaVM.framework/Versions/Current/Commands
   ```

3. Run the following keytool command:

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

   Where [USERNAME] is, obviously, your current user folder. You should get something similar to this in the output:

   ```
   Alias name: androiddebugkey
   Creation date: Jun 30, 2016
   Entry type: PrivateKeyEntry
   ```

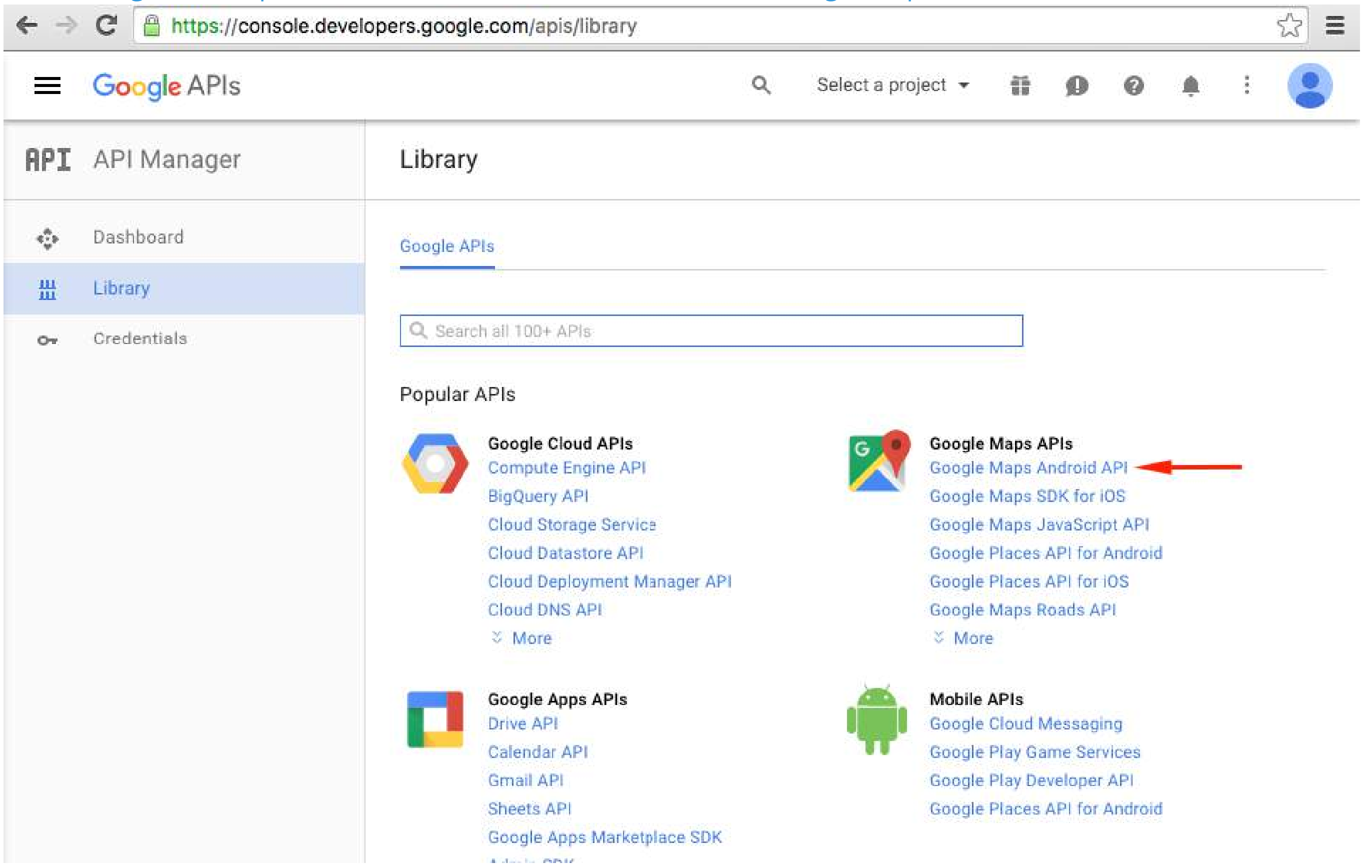
```
Certificate chain length: 1
Certificate[1]:
Owner: CN=Android Debug, O=Android, C=US
Issuer: CN=Android Debug, O=Android, C=US
Serial number: 4b5ac934
Valid from: Thu Jun 30 10:22:00 EEST 2016 until: Sat Jun 23 10:22:00 EEST 2046
Certificate fingerprints:
MD5:  4E:49:A7:14:99:D6:AB:9F:AA:C7:07:E2:6A:1A:1D:CA
SHA1: 57:A1:E5:23:CE:49:2F:17:8D:8A:EA:87:65:44:C1:DD:1C:DA:51:95
SHA256:
70:E1:F3:5B:95:69:36:4A:82:A9:62:F3:67:B6:73:A4:DD:92:95:51:44:E3:4C:3D:9E:ED:99:03:09:9F:90:
3F
Signature algorithm name: SHA256withRSA
Version: 3
```

4. All we need in this output is the SHA1 certificate fingerprint. In our case it equals to this:

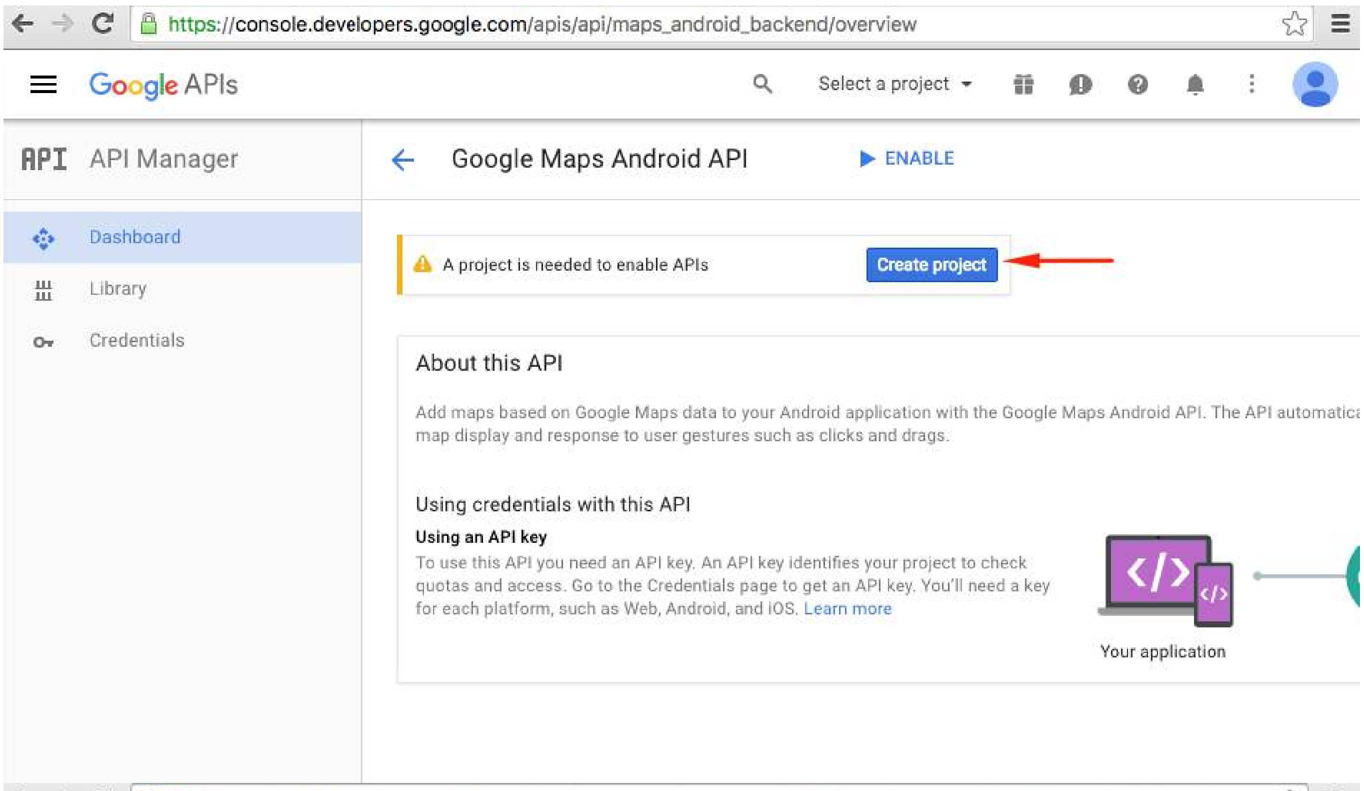
```
57:A1:E5:23:CE:49:2F:17:8D:8A:EA:87:65:44:C1:DD:1C:DA:51:95
```

Copy or save somewhere this key. We will need it later on.

5. Go to Google Developers Console, in our case we have to add Google Maps Android API, so choose it:



6. Google will ask you to create a project to enable APIs, follow this tip and create the project:


https://console.developers.google.com/apis/api/maps_android_backend/overview

Google APIs — Select a project

API Manager

- Dashboard
- Library
- Credentials

Google Maps Android API — ENABLE

A project is needed to enable APIs — Create project

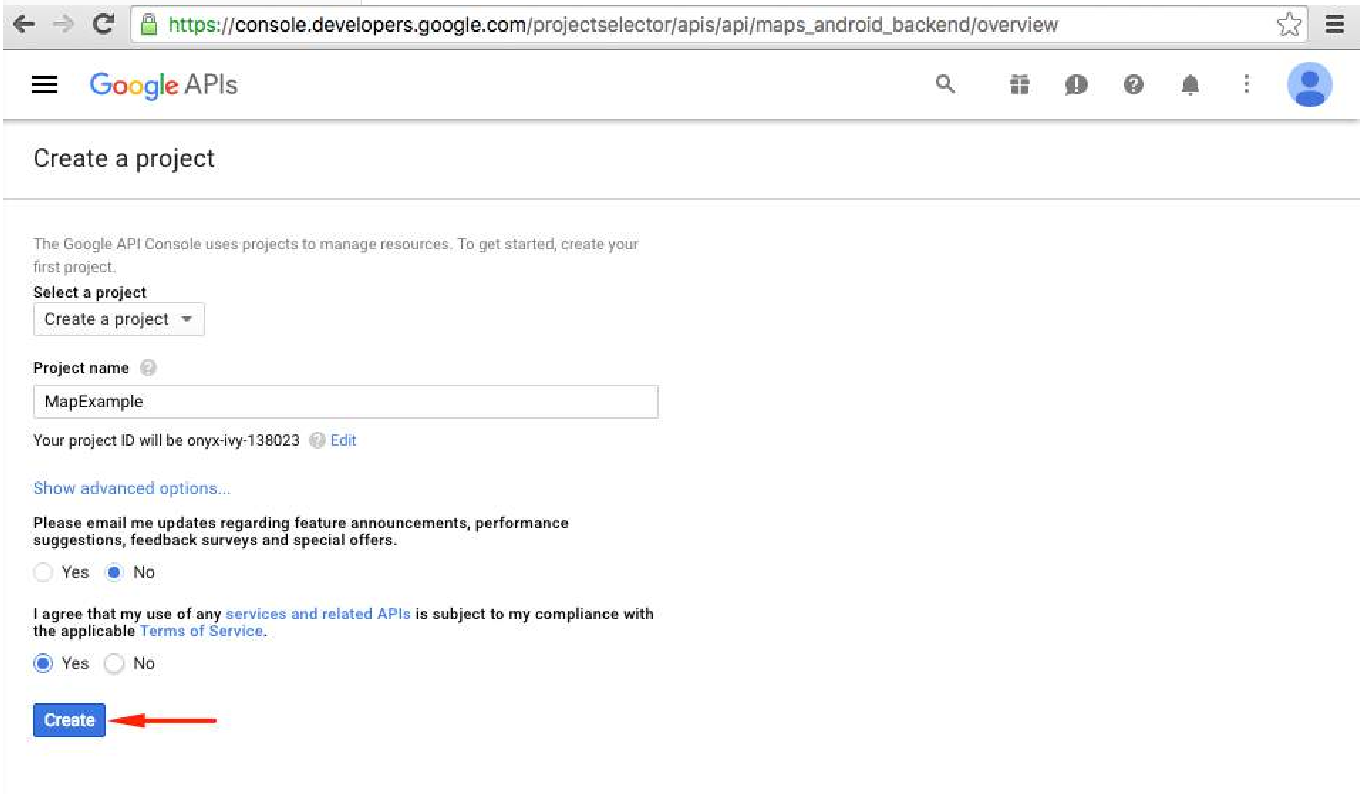
About this API

Add maps based on Google Maps data to your Android application with the Google Maps Android API. The API automatica
map display and response to user gestures such as clicks and drags.

Using credentials with this API

Using an API key

To use this API you need an API key. An API key identifies your project to check quotas and access. Go to the Credentials page to get an API key. You'll need a key for each platform, such as Web, Android, and iOS. Learn more


https://console.developers.google.com/projectselector/apis/api/maps_android_backend/overview

Google APIs

Create a project

The Google API Console uses projects to manage resources. To get started, create your first project.

Select a project: Create a project

Project name: MapExample

Your project ID will be onyx-ivy-138023 Edit

Show advanced options...

Please email me updates regarding feature announcements, performance suggestions, feedback surveys and special offers.

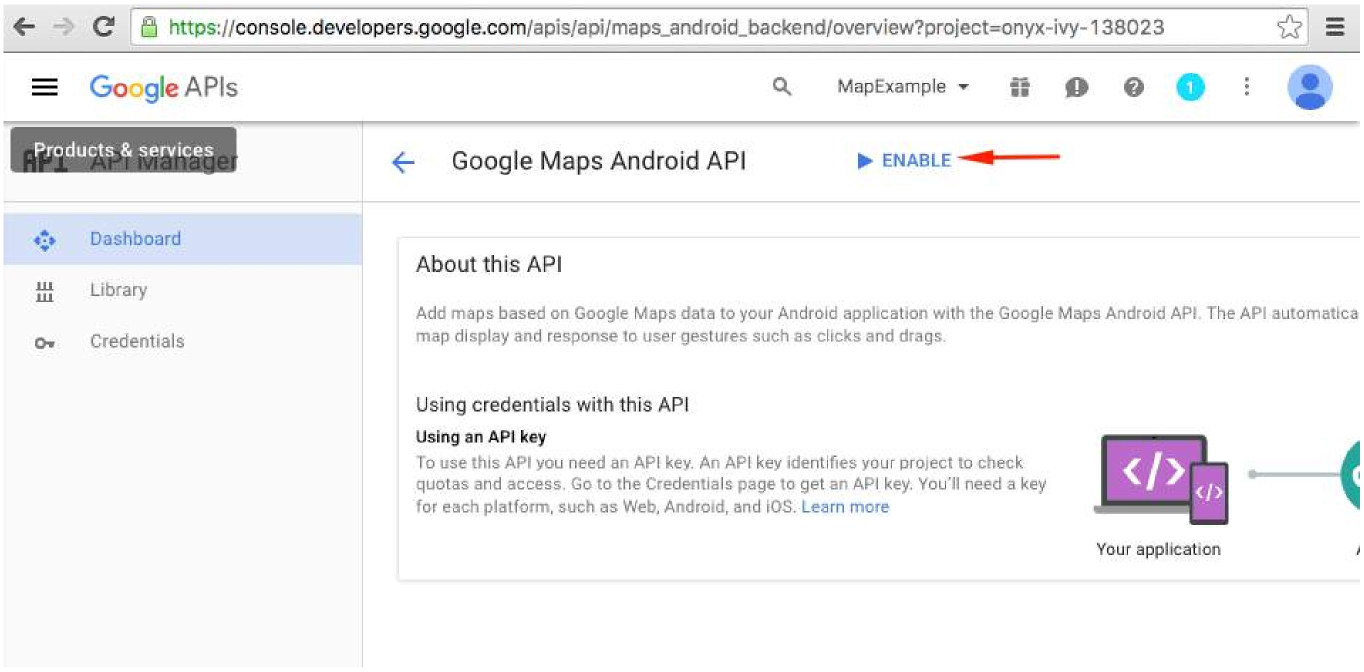
Yes [✓ No]

I agree that my use of any services and related APIs is subject to my compliance with the applicable Terms of Service.

[✓ Yes] No

Create

7. Enable Google Maps API for your project:


https://console.developers.google.com/apis/api/maps_android_backend/overview?project=onyx-ivy-138023

About this API

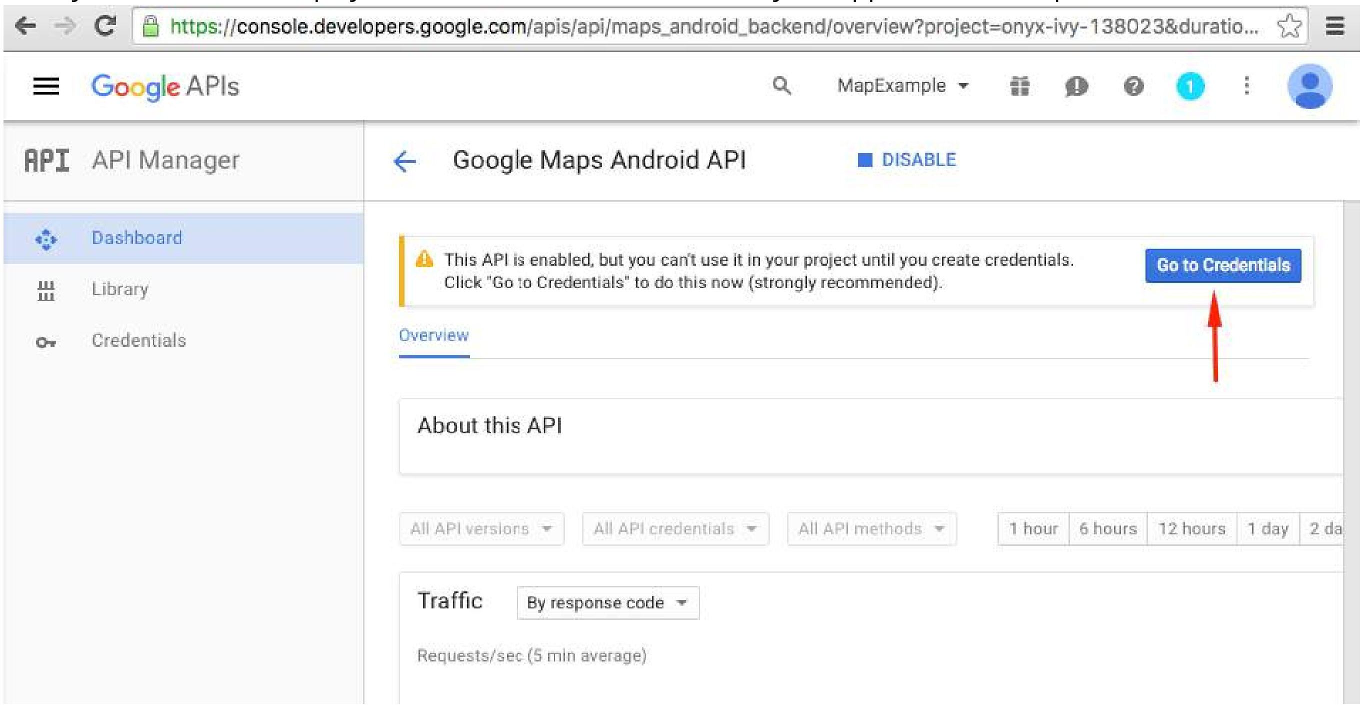
Add maps based on Google Maps data to your Android application with the Google Maps Android API. The API automatica
map display and response to user gestures such as clicks and drags.

Using credentials with this API

Using an API key

To use this API you need an API key. An API key identifies your project to check quotas and access. Go to the Credentials page to get an API key. You'll need a key for each platform, such as Web, Android, and iOS. Learn more

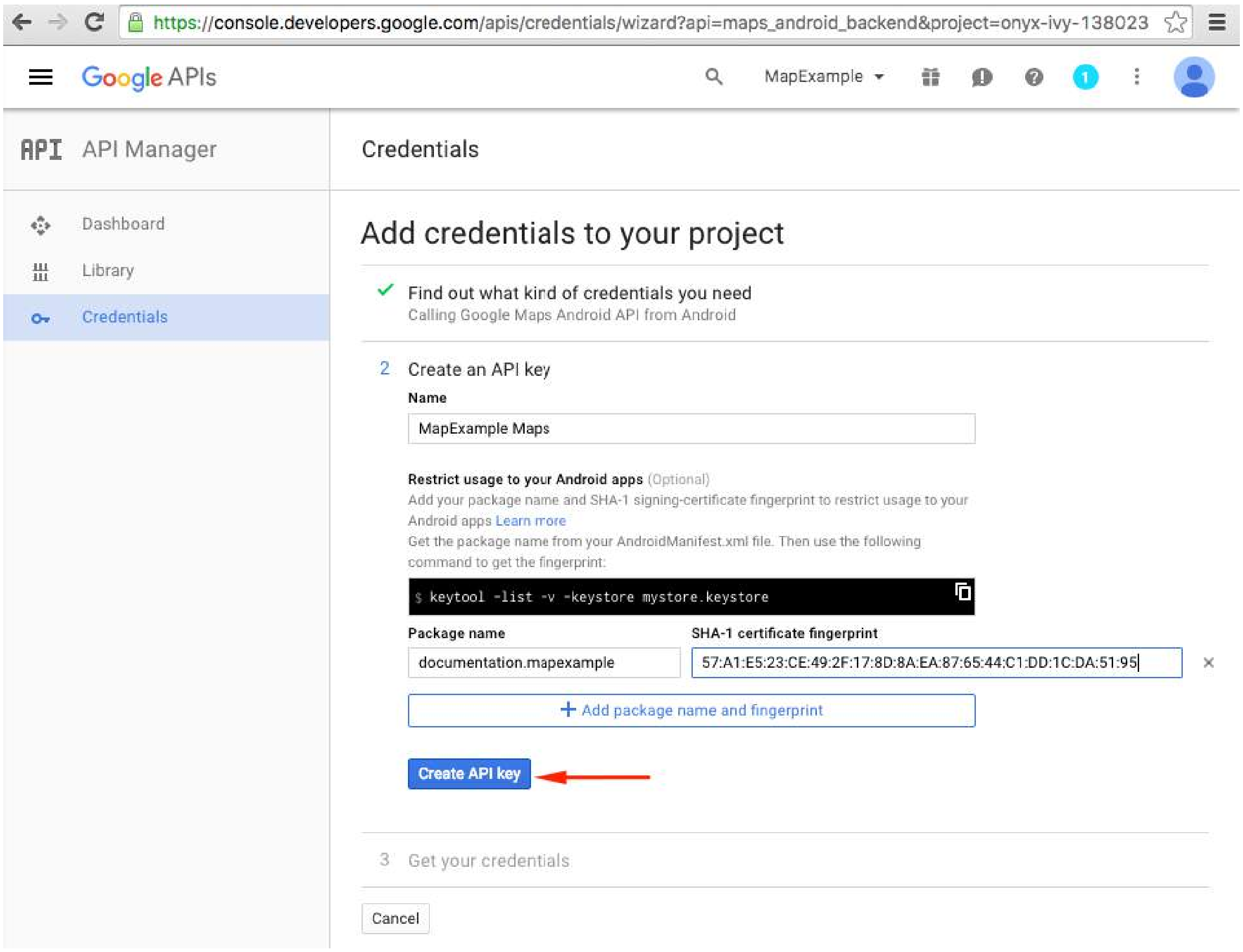
After you have enabled api, you have to create credentials for your app. Follow this tip:


https://console.developers.google.com/apis/api/maps_android_backend/overview?project=onyx-ivy-138023&duratio...

This API is enabled, but you can't use it in your project until you create credentials.
Click "Go to Credentials" to do this now (strongly recommended).

About this API

8. On the next page choose the Android platform, tap on "What credentials do I need?" button, create a name for your API key, tap on "Add package name and fingerprint", enter your package name and your SHA1 fingerprint from the step 4 and finally create an API key:


https://console.developers.google.com/apis/credentials/wizard?api=maps_android_backend&project=onyx-ivy-138023

Credentials

Add credentials to your project

✓ Find out what kind of credentials you need
Calling Google Maps Android API from Android

2 Create an API key

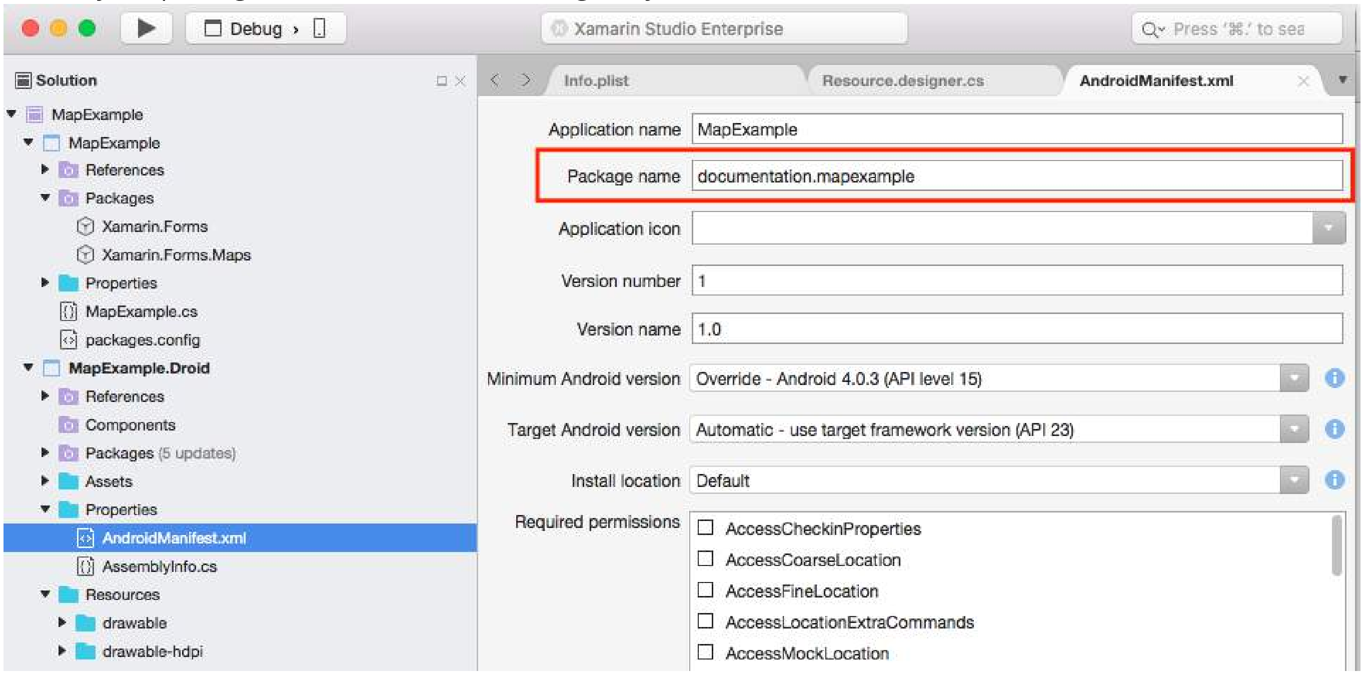
Name: MapExample Maps

Restrict usage to your Android apps (Optional)
Add your package name and SHA-1 signing-certificate fingerprint to restrict usage to your Android apps Learn more
Get the package name from your AndroidManifest.xml file. Then use the following command to get the fingerprint:

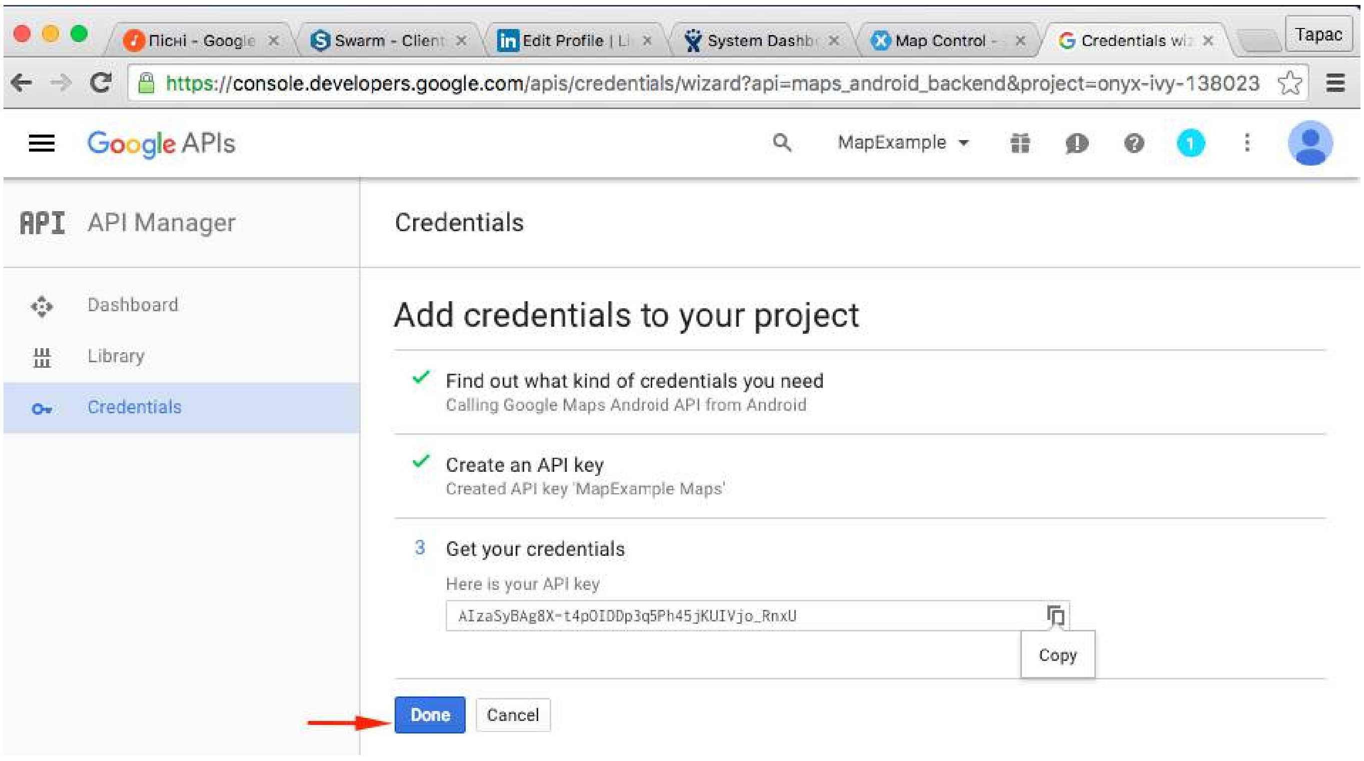
```
$ keytool -list -v -keystore mystore.keystore
```

Package name: documentation.mapexample
SHA-1 certificate fingerprint: 57:A1:E5:23:CE:49:2F:17:8D:8A:EA:87:65:44:C1:DD:1C:DA:51:95

To find your package name in Xamarin Studio go to your .Droid solution -> AndroidManifest.xml:


Application name: MapExample
Package name: documentation.mapexample
Application icon:
Version number: 1
Version name: 1.0
Minimum Android version: Override - Android 4.0.3 (API level 15)
Target Android version: Automatic - use target framework version (API 23)
Install location: Default
Required permissions:
- ☐ AccessCheckinProperties
- ☐ AccessCoarseLocation
- ☐ AccessFineLocation
- ☐ AccessLocationExtraCommands
- ☐ AccessMockLocation

9. After creation copy the new API key (don't forget to press the "Done" button after) and paste it to your `AndroidManifest.xml` file:


Credentials

Add credentials to your project

✓ Find out what kind of credentials you need
Calling Google Maps Android API from Android

✓ Create an API key
Created API key 'MapExample Maps'

3 Get your credentials
Here is your API key
AIzaSyBAg8X-t4pOIDDp3q5Ph45jKUIVjo_RnxU

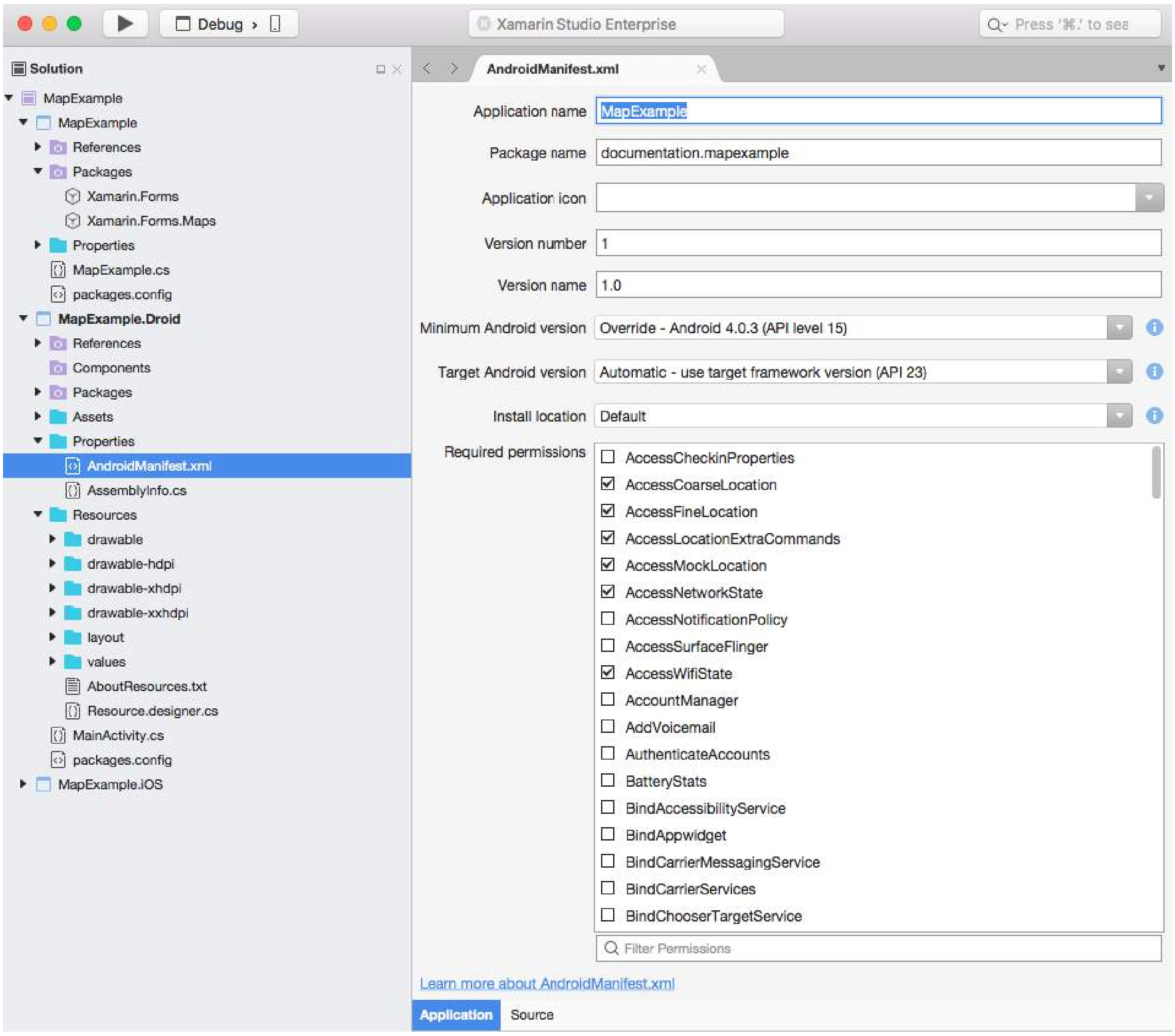
https://console.developers.google.com/apis/credentials/wizard?api=maps_android_backend&project=onyx-ivy-138023
Google APIs
MapExample
API Manager
Dashboard
Library
Credentials
Copy
Done
Cancel

File AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:versionCode="1"
   android:versionName="1.0"
   package="documentation.mapexample">
    <uses-sdk
       android:minSdkVersion="15" />
    <application
       android:label="MapExample">
        <meta-data
           android:name="com.google.android.geo.API_KEY"
           android:value="AIzaSyBAg8X-t4pOIDDp3q5Ph45jKUIVjo_RnxU" />
        <meta-data
           android:name="com.google.android.gms.version"
           android:value="@integer/google_play_services_version" />
    </application>
</manifest>
```

You'll also need to enable some permissions in your manifest to enable some additional features:

- Access Coarse Location
- Access Fine Location
- Access Location Extra Commands
- Access Mock Location
- Access Network State
- Access Wifi State
- Internet



Application name	MapExample
Package name	documentation.mapexample
Application icon	
Version number	1
Version name	1.0
Minimum Android version	Override - Android 4.0.3 (API level 15)
Target Android version	Automatic - use target framework version (API 23)
Install location	Default

Required permissions

- ☐ AccessCheckinProperties
- ☑ AccessCoarseLocation
- ☑ AccessFineLocation
- ☑ AccessLocationExtraCommands
- ☑ AccessMockLocation
- ☑ AccessNetworkState
- ☐ AccessNotificationPolicy
- ☐ AccessSurfaceFlinger
- ☑ AccessWifiState
- ☐ AccountManager
- ☐ AddVoicemail
- ☐ AuthenticateAccounts
- ☐ BatteryStats
- ☐ BindAccessibilityService
- ☐ BindAppwidget
- ☐ BindCarrierMessagingService
- ☐ BindCarrierServices
- ☐ BindChooserTargetService

Although, the last two permissions are required to download Maps data. Read about Android permissions to learn more. That's all the steps for Android configuration.

> *Note*: if you want to run your app on android simulator, you have to install Google Play Services on it. Follow this tutorial to install Play Services on Xamarin Android Player. If you can't find google play services update after the play store installation, you can update it directly from your app, where you have dependency on maps services

Adding a map

Adding map view to your crossplatform project is quite simple. Here is an example of how you can do it (I'm using PCL project without XAML).

PCL project

File MapExample.cs

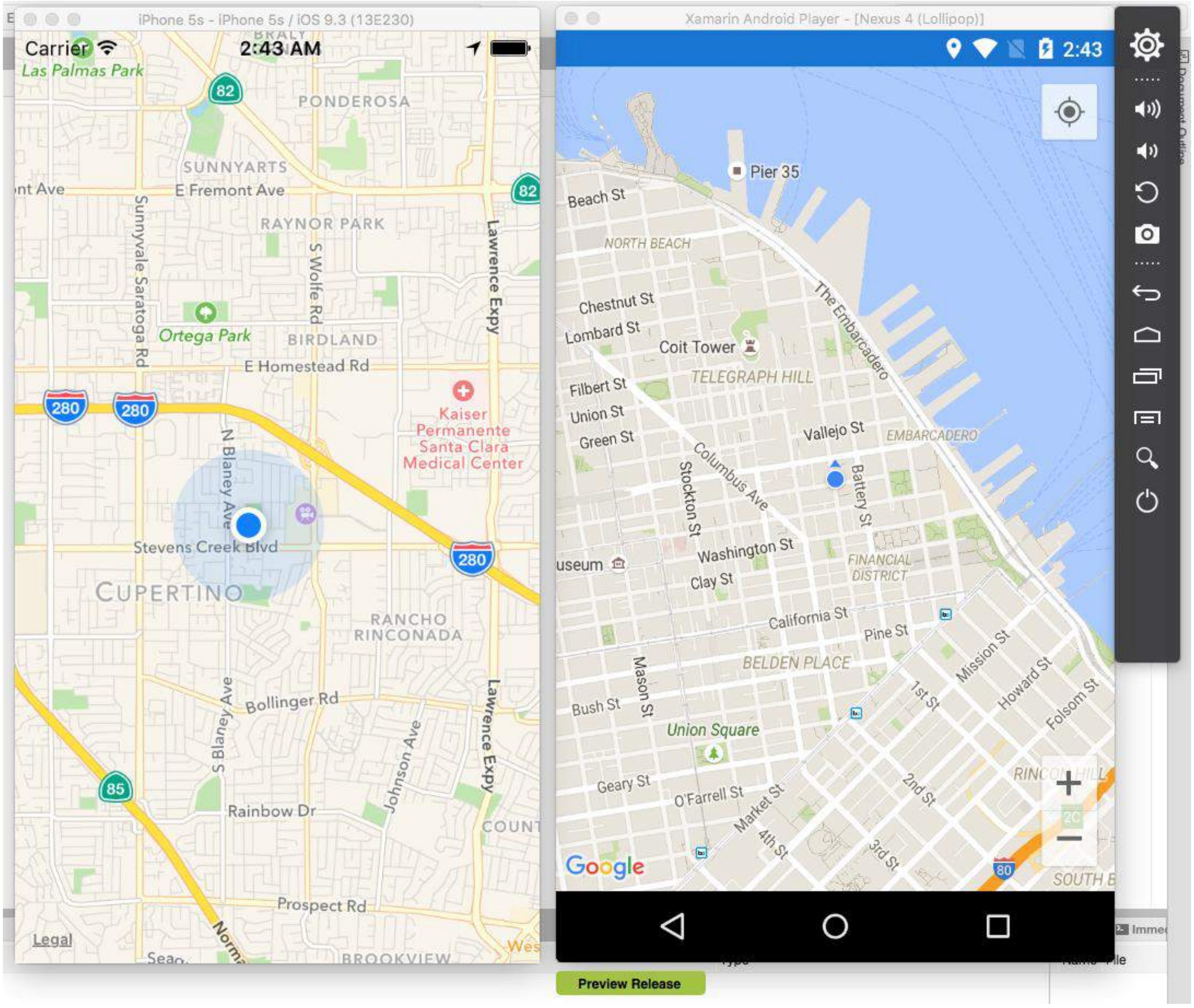
```
public class App : Application
```

```
{
    public App()
    {
        var map = new Map();
        map.IsShowingUser = true;

        var rootPage = new ContentPage();
        rootPage.Content = map;

        MainPage = rootPage;
    }
}
```

That's all. Now if you'll run your app on iOS or Android, it will show you the map view:


iPhone 5s - iPhone 5s / iOS 9.3 (13E230)
Carrier
2:43 AM
Xamarin Android Player - [Nexus 4 (Lollipop)]
2:43
Google
Legal
Preview Release