



NextNav Pinnacle Unity Plugin Development Guide

Version: 2.1

Date Released: 04/30/2022

NextNav Confidential and Proprietary

All data and information contained in or disclosed by this document is confidential and proprietary information, and includes, without limitation, trade secrets of NextNav LLC and all rights therein are expressly reserved. By accepting this material, the recipient agrees that this material and the information contained therein is to be held in confidence and in trust and will not be disseminated, distributed, copied, reproduced in whole or in part, nor its contents revealed in any manner to others without the express written permission of NextNav LLC.

COPYRIGHT

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher. Material contained herein is reserved for members of NextNav and designated contractors, and therefore is not to be quoted, cited or shown outside the company without written permission.

Table of Contents

Introduction	3
Important note for developers	3
What versions of Unity should I use?	3
How can I obtain a developer evaluation key?	3
Getting Started	5
Creating the Managers	6
Methods.....	6
Callback methods.....	8
Setting the Developer Key and Host Url	9
Host URL:.....	9
Setting the Google Maps Key (Optional)	10
Android.....	10
iOS	10
Understanding the Plugin	11
Optional: Building for iOS.....	14
Optional: Building for Android	16
Optional: Testing the Demo Scene.....	17
Optional: Creating the Google Maps API Key	18
Errors & Warnings	20

Introduction

The NextNav Pinnacle Service generates industry-leading accuracy for vertical positioning data (also called z-axis data) using barometric pressure and 2D location information from a device, then optimizing the positioning data with NextNav's Pinnacle Altitude Service Network. NextNav has deployed the Pinnacle commercial service nationwide in the US, enabling FCC-compliant vertical accuracy in over 4,400 cities and towns addressing over 90% of buildings more than 3 floors high. The NextNav Pinnacle Service coverage area can be viewed at <https://www.nextnav.com/>.

Important note for developers

Due to the reliance of the plugin on the device's HW capabilities, it's not possible to test the Plugin directly inside the Unity Editor. In order to properly build and test the SDK, the device must have a Barometric Sensor and acquire a unique developer evaluation key.

What versions of Unity should I use?

This plugin is compatible with Unity version: 2019.4.X, 2020.3.X, 2021.1.X 2022.1.X

How can I obtain a developer evaluation key?

The Pinnacle Unity SDK plug-in is available with a built-in Key to quickly get started to evaluate the Pinnacle service for trial purposes. The built-in key for trial purposes will be available for a limited time period, therefore if further testing and time is required, NextNav recommends that you request a Developer Eval-Key by visiting <https://partner.nextnav.com/>. The Developer Eval-Key comes with additional benefits as you progress in your development cycle.

The Eval-Key is free to use for 60 days and provides 50K API requests/month for you to trial NextNav Pinnacle service.

The Eval-Key is unique for the application bundle ID for your mobile application (App-bundle-ID) or to the application ID (App-ID) of your client application.

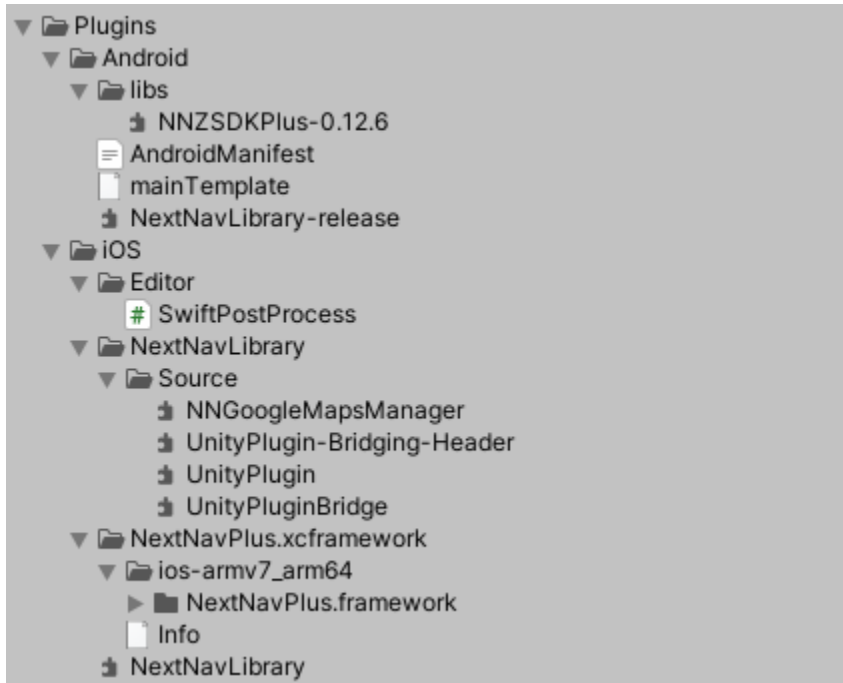
The built-in key can be easily updated with a unique Eval-Key for your application by passing it into { SetDeveloperKey(string key) } or via the demo scene referenced on page 8 - this will override the built-in key for extended evaluation.

When you're ready, NextNav can assist you with a commercial agreement and the associated commercial billing key (KEY) for your application.

This release is built based on Pinnacle SDK (iOS v1.2.2, Android v1.4.3.1).

Getting Started

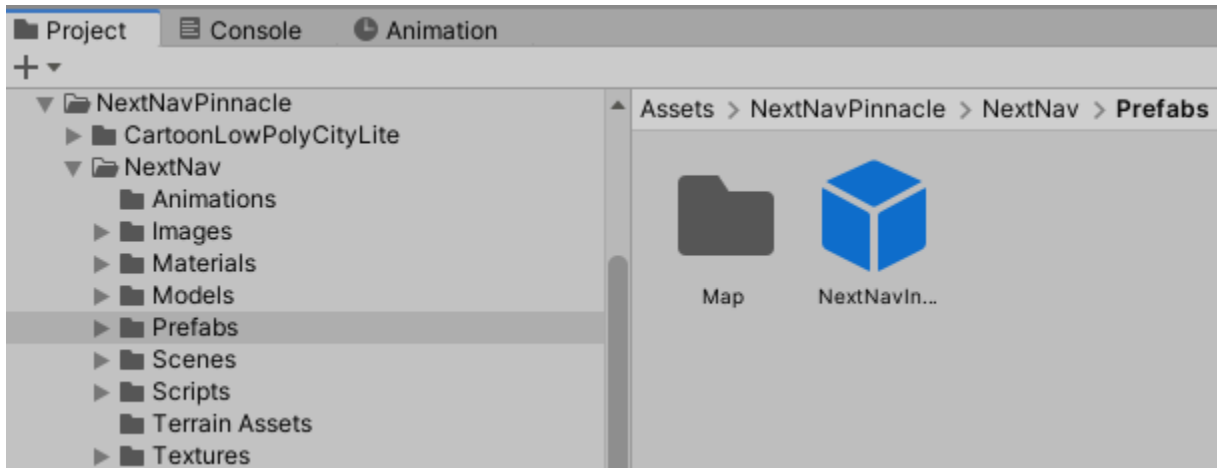
After importing the plugin, make sure that the project has this folder structure:



Plugins folder must be in Assets/Plugins to work

Creating the Managers

You can add the plugin to your Scene by adding the “NextNavInterface” prefab in your scene which is located here:



The prefab has the main script, “NextNavInterface” that automatically handles the current platform based on your context (Android or iOS) and receives the calls that are then passed to the specific native SDKs (NextNavAndroid script or NextNaviOS script).

Methods

Here’s the list of Methods available through the NextNavInterface:

Method	Description
200	Successful assistance data delivery, including HAE, HAT and uncertainty data
void InitSdk()	This will Initialize the plugin
string StartAltitudeCalculationOnce()	Get altitude data only once (Only Android)
string StartAltitudeCalculationEachSecond()	Get altitude data every second
string StartAltitudeCalculationEvery30Second()	Get altitude data every 30 seconds
string StartAltitudeCalculationEvery60Second()	Get altitude data every 60 seconds (Only Android)
void StopAltitudeCalculation()	Stop the current altitude calculation
void ShowCalibrationUI()	Launch the calibration screens
void Stop()	Turns off the plugin

void Callback (string stringData)	This method handles all the communication received from iOS or Android environment
void CallbackStatus (Constants.SdkStatus status, int statusCode)	Here you can handle the status and status code received
void CallbackData (string heightHat, string heightUncertaintyHat, string height, string heightUncertainty)	Here you can handler the differents heights received
void CallbackError (string error)	Here you can handler the error occurred
void RequestPermissions ()	Request permissions in android context (you can enable the auto request permissions boolean to avoid calling it manually and it will be triggered at the start of the app)
void SetDeveloperKey (string key)	You can set the developer key before calling the Init method
void SetHostURL (string hostURL)	You can set the Host Url before calling the Init method
void Start2DLocationInjection (string latitude, string longitude)	You can start an altitude calculation using latitude and longitude.

NOTE:

- When you use Start2DLocationInjection method, the plugin will no longer automatically fetch 2D location from the device for altitude calculation.
- You must ensure that the injected location value is updated as needed by calling the **StopAltitudeCalculation** function followed by a fresh **Start2DLocationInjection** function call with a new 2D location data.
- The location data may become stale if the device moves and the injected 2D Location is not refreshed.

Callback methods

You can receive three types of Callback methods from the SDK: “SDK_Status”, “Data” or “Error”

- SDK_Status: In this case you will receive a status code. This triggers the CallbackStatus() method in the NextNavInterface script, so you can add your code there.
- Data: Here you can access the heights values: “Height HAT” “Height HAT Uncertainty” and the “Timestamp” of each height received. This triggers the CallbackData() method in the NextNavInterface script, so you can add your code there.
- Error: Will give you the error description. This triggers the CallbackError() method in the NextNavInterface script, so you can add your code there.

```
1 referencia | trickbucket, Hace 25 días | 1 autor, 1 cambio
private void CallbackStatus(Constants.SdkStatus status, int statusCode)
{
    //You can replace this code with yours

    UIManager.Instance.UpdateUIBasedOnSdkStatus(status);
    UIManager.Instance.AddTextToDebugScroll("Sdk Status Code: " + statusCode);
}

1 referencia | FranciscoGregorio, Hace 1 hora | 1 autor, 1 cambio
private void CallackData(string heightHat, string heightUncertaintyHat, string timeStamp)
{
    //You can replace this code with yours

    if (UIManager.Instance == null)
        return;

    float _heightHat;
    float _heightUncertaintyHat;

    if (float.TryParse(heightHat, out _heightHat) && float.TryParse(heightUncertaintyHat, out _heightUncertaintyHat))
    {
        if (PlayerController.Instance != null)
            PlayerController.Instance.SetPlayerHeight(_heightHat);

        UIManager.Instance.UpdateHeightsHat(_heightHat, _heightUncertaintyHat);

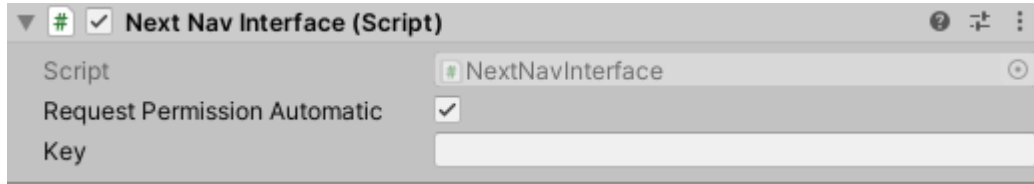
        string heightWithTimeText = _heightHat + " - TimeStamp: " + timeStamp;
        UIManager.Instance.AddTextToDebugScroll(heightWithTimeText);
    }
    else
    {
        UIManager.Instance.AddTextToDebugScroll("Error: Error trying To Parse the heights hats");
    }
}

1 referencia | trickbucket, Hace 25 días | 1 autor, 1 cambio
private void CallbackError(string error)
{
    //You can replace this code with yours

    if (UIManager.Instance != null)
        UIManager.Instance.AddTextToDebugScroll(error);
}
```

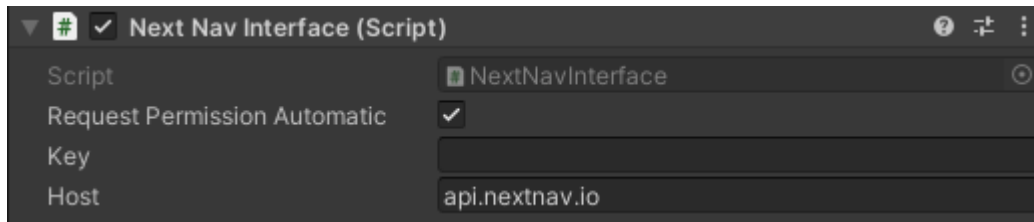

Setting the Developer Key and Host Url

Please refer to page 3 for information related to obtaining a developer key. Once you have acquired a developer key, just add it in the Key field in the NextNavInterface inspector (if you are in the demo scene you will need to add it in the NextNavInterfaceDemo script too).



You need to do this before calling the InitSdk method, otherwise the SDK will fail to initialize.

Host URL:



The parameter “baseUrl” is the NextNav service environment URL, developers should use “api.nextnav.io” during the evaluation phase. When the application is ready to be launched, the commercial services agreement for this application will provide the final “baseUrl” and “secretKey” for commercial purposes. Please email pinnacle.support@nextnav.com for any additional questions related to these parameters.

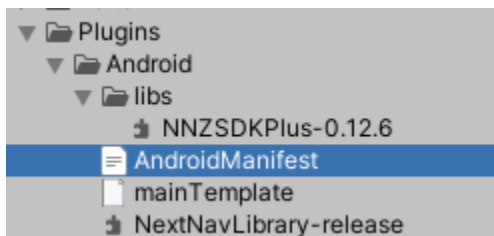
Setting the Google Maps Key (Optional)

In order for Calibration and Altitude to work, the SDK needs to be able to display a Map and be updated with Latitude and Longitude values.

Android

The Plugin is already configured to utilize Google Maps, but developers must provide their own key. Getting a Google Maps Key is totally free and more information can be found here: <https://developers.google.com/maps/documentation/geocoding/get-api-key>. After obtaining the Key, follow the steps below.

Start by changing the metadata in the android manifest file located here:



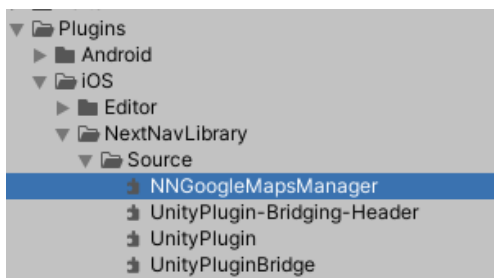
Add your google api key in the android:value"Api key here". Additionally, make sure that your Google Maps Api Key (Step 09 - Create Google Api Key for more info about this) has "Android SDK" enabled in the Google Developer Console. See Section 9 for Reference.

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

iOS

(if you use apple maps, this is not necessary)

Start by opening the NNGoogleMapsManager.swift file located in this path:



And add the Google Maps Api Key here and make sure that your Google Maps Api Key has "iOS SDK" enabled (Step 09 - Create Google Api Key for more info about this) in the Google Developer Console. See Section 9 for Reference.

```
class NNGoogleMapsManager: NSObject, GMSMapViewDelegate {  
    private static let GOOGLEMAPSAPIKEY = ""
```

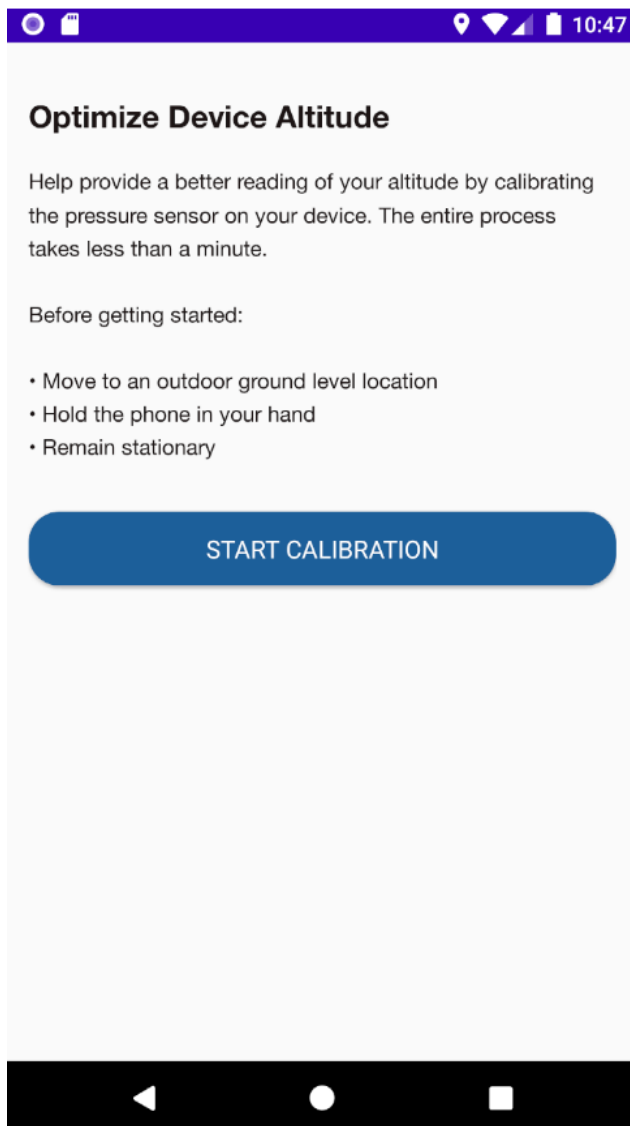
Understanding the Plugin

There is a very specific set of steps that need to be followed in order to properly Initialize, Calibrate and Receive Altitude Data.

IMPORTANT REMINDER: due to the reliance of the plugin on the device's HW capabilities, it's not possible to utilize the Plugin in the editor.

The steps below must be performed in sequence for the Plugin to work properly:

1. Call **InitSdk** and wait until you receive Status Code 800 (SDK Initialization Success).
2. Launch the calibration screens by calling **ShowCalibrationUI**. You should see the following screen:



3. Follow the instructions and wait until you receive Status Code 870 (Manual Baro Calibration Success). If calibration fails, you can troubleshoot using the table below.
4. Invoke any of the **StartAltitudeCalculation** methods (there are three alternatives that obtain Altitude at different frequencies: 1, 30 or 60 seconds, as well as one single time). For more information, refer to the section above with description of methods.

Any other Status Codes can be troubleshooted utilizing the following reference table:

Status Code	Description	Note
200	Successful assistance data delivery, including HAE, HAT and uncertainty data	
400	General server failure	
403	Invalid API Key	
600	2D location is out of NextNav Pinnacle service area	
610	Inside Pinnacle coverage/service area but reference pressure data is NOT available	
620	Inside Pinnacle coverage/service area but barometer calibration data is NOT available	<i>This status code is typically not observed, but provided for reference</i>
630	Combination of 610 and 620 Codes	<i>This status code is typically not observed, but provided for reference</i>
640	Barometer calibration data is stale	<i>This status code is typically not observed, but provided for reference</i>

SdkStatusNotification supports the following error codes:

Error Code	Description	Notes
800	SDK initialization success	
801	SDK initializing in progress	
802	Initializing pressure provider	<i>The test mobile phone may not include a barometric sensor</i>
803	Initializing location provider	
804	No location available	<i>Consider as initialization failed after 10 seconds</i>

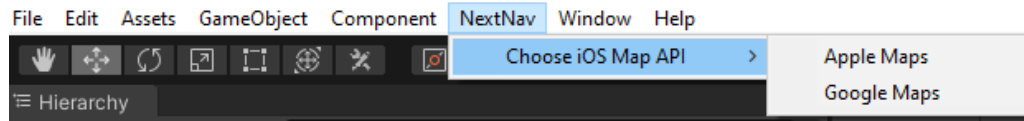
805	No pressure available	<i>Consider as initialization failed after 10 seconds</i>
806	SDK initialization failed	
808	SDK is already initialized	
810	SDK is not initialized	
811	SDK stop successfully	
815	Input host URL is invalid	
830	Altitude mode is activated	
832	Waiting for pressure for assistance fetch	
833	Waiting for location for assistance fetch	
834	Waiting for NNCS	
835	Altitude mode is deactivated	
870	Manual barometer calibration success	
871	Manual barometer calibration in progress	
872	Manual barometer calibration failed	<i>The end-user may be directed to manually calibrate in a different place/location and at a different time</i>
876	Device pressure reading is changing too fast and/or user doesn't seem to be still	<i>The Pinnacle SDK cannot run the manual-calibration operation due to rapid movement/change. The end user may be directed that a calibration is needed when the user's device is not moving.</i>
877	Not a good place (Ex: outside coverage area etc.) to collect data	<i>The end-user may be directed to manually calibrate in a different place/location and at a different time</i>
896	Activity permission is needed	
898	Location permission is needed	
899	Storage permission is needed	

IMPORTANT: on Android, if you receive Status Code 896, some permissions have not been granted successfully. This can happen depending on your project's configuration and also AndroidManifest's merge issues. If that's the case, simply press the P button located on the top right corner of the screen to force the Permissions dialog to appear.

Optional: Building for iOS

Building for iOS requires a few extra steps:

- A) You can change which maps api will the sdk use (Apple or Google maps) from the unity editor menu:



- B) You must use Metal Graphics API (You can change it in Build/PlayerSettings/OtherSettings)
- C) Create the xcode workspace.

The steps from “D” to “H” are only necessary if you want to use google maps api otherwise skip to step “I”

- D) Open the terminal on MacOS.
- E) If you don't have cocoapods installed. You need to write “sumo gem install cocoapods”.
- F) Use “cd” until you reach the project path. Example: “cd downloads/ProjectExample/”.
 - 1. If one of your folders has a space you need to add “ ” in the path.
 - 2. Example: cd ‘downloads/Project Example/’.
- G) Now you can use “pod init” to create a podfile file inside the project path.
- H) Open podfile file and change the lines for this:

```
source 'https://github.com/CocoaPods/Specs.git'

platform :ios, '12.0'

target 'UnityFramework' do

  pod 'GoogleMaps', '5.0.0'

end
```

This is needed in order to install Google Maps in your xcode project and Calibration to work.

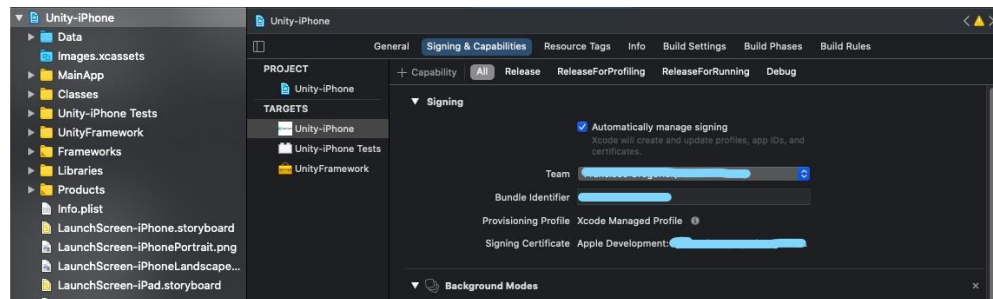
I) Now type “po


```
You have either:  
* out-of-date source repos which you can update with `pod repo update`.  
* mistyped the name or version.  
* not added the source repo that hosts the Podspec to your Podfile.
```

install” in the terminal. If you get the following error:

then you should update cocoa with this command: “pod repo update”.

J) Now you need to create the certificate in Signing & capabilities on Unity-iPhone target



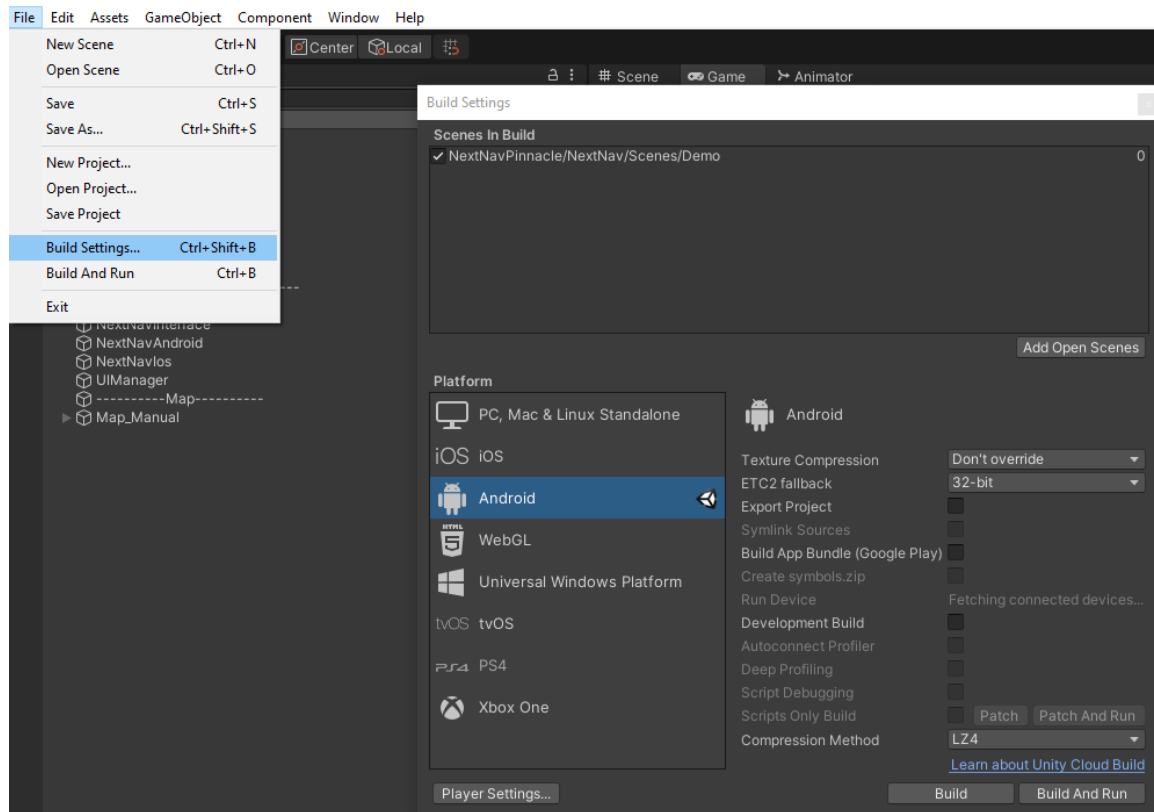
K) The only thing left is to add the marker image on “Images.xcassets”. You can use this  image as example. You need to drag the image to Images.xcassets and the image must have this name “markerPin” to be recognized as a marker on Google Maps.

L) You should be able to run the project on your device.

Important: If you want to launch your app to the AppStore or Testflight you will need to request Apple approval to use the SensorKit, otherwise you won't be able to create the provisioning profile with the entitlement. The app will crash when you try to open it without the profile entitlement.

Optional: Building for Android

In order to build for Android go to File => Build Settings and then press Build. No additional steps are required.



Which SDK versions should I use?

The min sdk supported is 22 and the target sdk supported is 31 or earlier.

The last thing about android that you should know is that we use an android manifest located in Assets\Plugins\Android.

Important Note:

We have 4 folders on this path: "Assets\NextNavPinnacle\NextNav\Unity Different Versions Support" which are to support different versions of unity, in case that you have any problems with gradles, android manifest or build errors related to this, you can try replacing the files, but you don't need to do anything, because this is handled automatically when you import the asset. Maybe the only thing you may need to do is add the platform android to the .aar library located on the final path "Assets\Plugins\Android".

Optional: Testing the Demo Scene

It contains a UI that follows the steps of the flow described in the **Understanding the Plugin** section well as multiple examples of how the altitude data can be displayed inside a Game environment:

- On an altitude UI,
- Directly on the World/Character,
- As text on a Log section.

Make sure to explore the provided scripts to better understand how the plugin works.

You can open the **Demo** scene located in NextNav/Scenes and see how it works.



Initialize or Stop the SDK



Move the character up (for testing purposes)



Move the character down (for testing purposes)



Open calibration screen

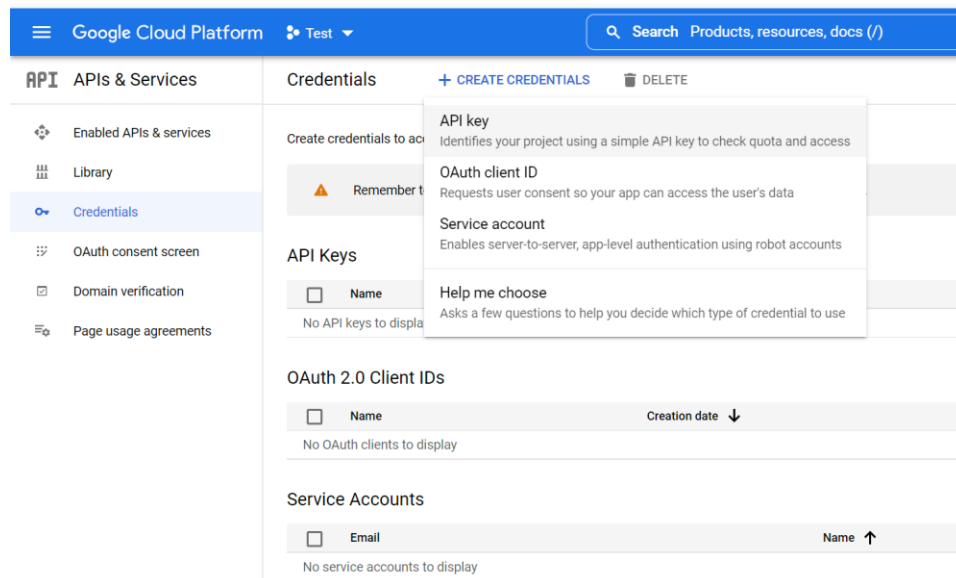


Start/Stop Altitude Calculation

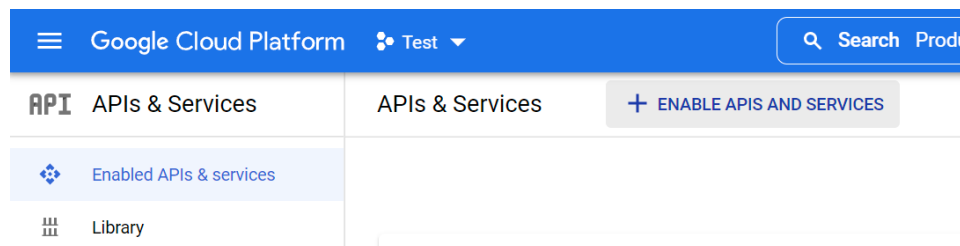
Optional: Creating the Google Maps API Key

Below is a summary of steps to get you started. Reference the [Link](#) for the official Google documentation.

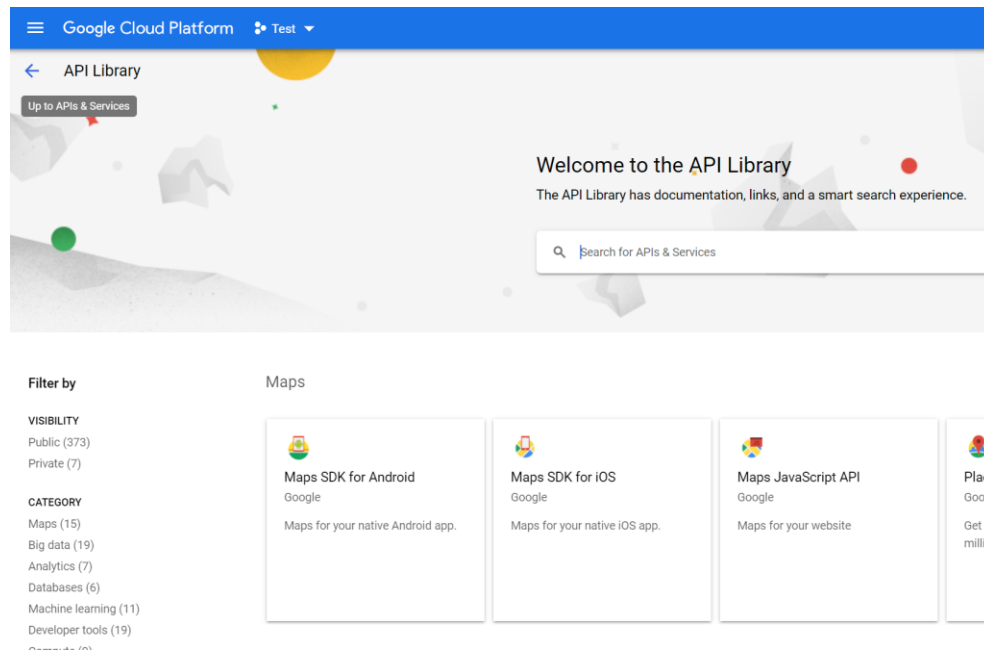
1. Open the Google Developer Console - [Link](#)
2. You will need an account so create or use one that you already have
3. Then create a project that will have the linked google api key
4. Go the left panel on the top and select Api & Services / Credentials, then create new credentials like this:



5. Then go again to Api & Services but now select Panel and click on “Enable Api & Services”



6. Now search for “Google Maps iOS” & “Google Maps Android” and enable both to be able to use google maps calibration screens in the plugin.



7. Done! Now you can use the Google Maps API in your project.

Errors & Warnings

1. Namespace error:

*“Assets\Plugins\iOS\Editor\SwiftPostProcess.cs(6,19): error CS0234un
The type or namespace name 'iOS' does not exist in the namespace
'UnityEditor’”*

You need to import/add the iOS module to the project and it will be fixed.

2. Missing 'package' key attribute on element package at [:arcore_client:]
AndroidManifest.xml:30:9-54

You need to follow the steps in this page:

https://developers.google.com/ar/develop/unity/android-11-build#unity_20193_20194_and_20201

Source: <https://forum.unity.com/threads/arcore-xr-plugin-2-12-requires-updated-gradle-on-unity-2019-4.1003444/>