

Run Voice Chat within Your Unity Application

This document shows how you can configure and run your Unity Application with Agora IO's Voice SDK. The SDK supports iOS, Android, MacOS and Windows platform on Unity. Note that the functionalities of the Voice SDK is a subset of the Video SDK.

Prerequisites

- Unity Editor (2017 LTS or above)
- A [developer account](#) with Agora.io

Getting Started

Although you may start the integration with your existing project, in this short tutorial we will just use the demo project included in the SDK.

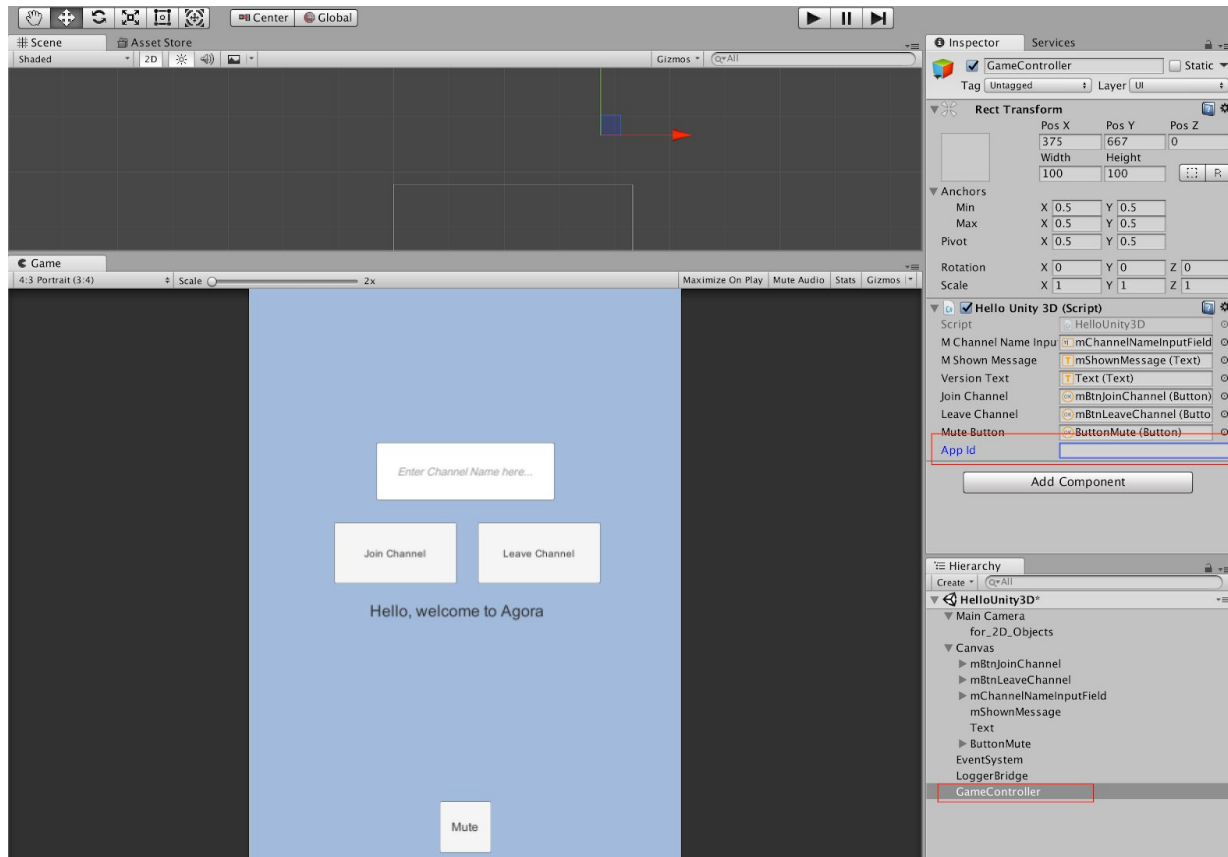
Open a new Unity project, and navigate to Unity Asset Store and search for "Agora Voice SDK". Download and import the assets.

Add Your AppID

Before you can build and run the project, you will need to add your AppID to the configuration. Go to your [developer account's project console](#), create a new AppId or copy the AppID from an existing project. Perform the following steps:

1. Open the HelloUnity3D scene.
2. Select the GameController from the Unity Editor's Hierarchy panel.
3. The GameController game object has a property App ID, this is where you will add your Agora App ID.

See the following screen capture:



At this step, you should be able to test the demo App within the Unity Editor. Just input your selected channel name to the input field and click Join. However, since this is SDK is built for interactive communication, you will need a separate device to run the same application in order to test voice chat.

Player Settings for Building the Sample Application

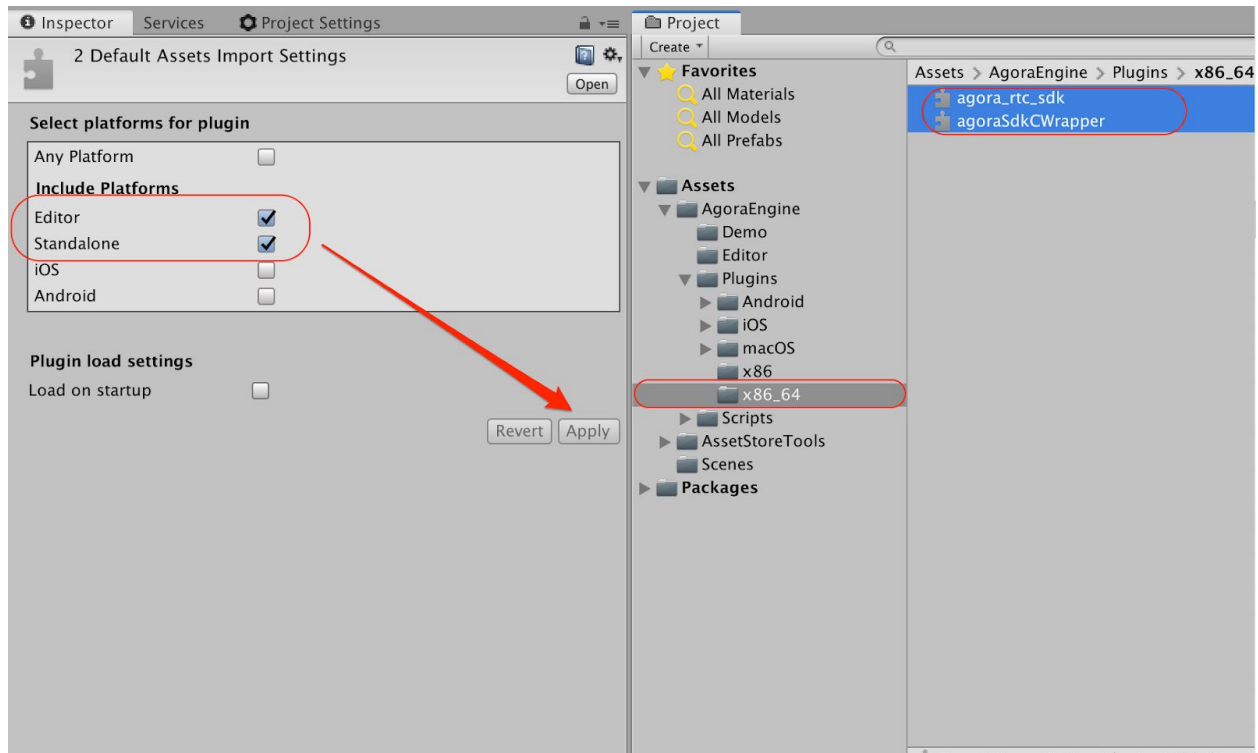
We will discuss the configuration for the four supported platforms in this section.

Common Setting

Open the **Build Settings** and drag `SceneHome` and `SceneHelloVideo` scenes from the assets list into the “Scenes in Build” list.

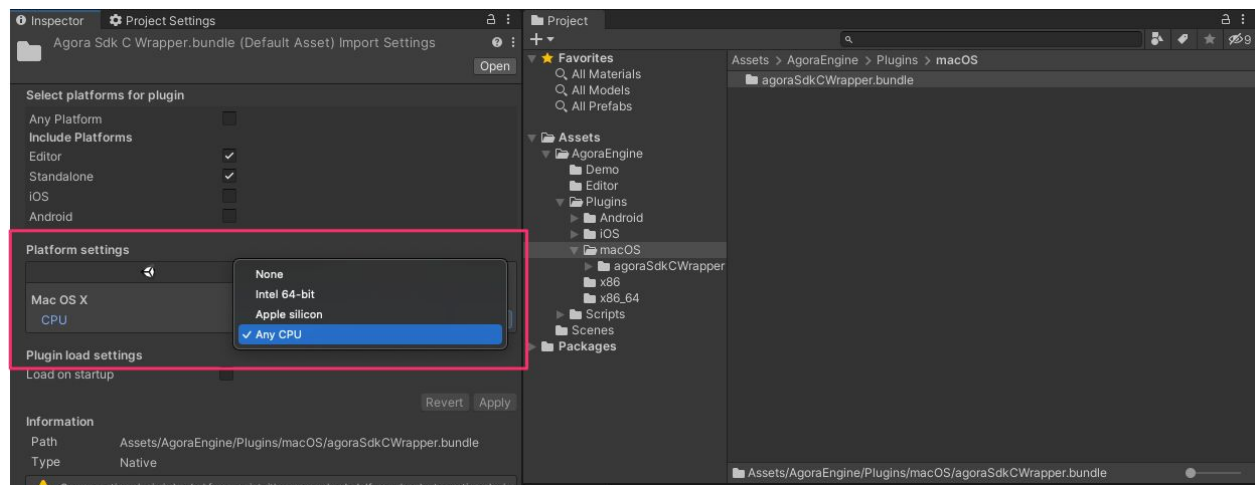
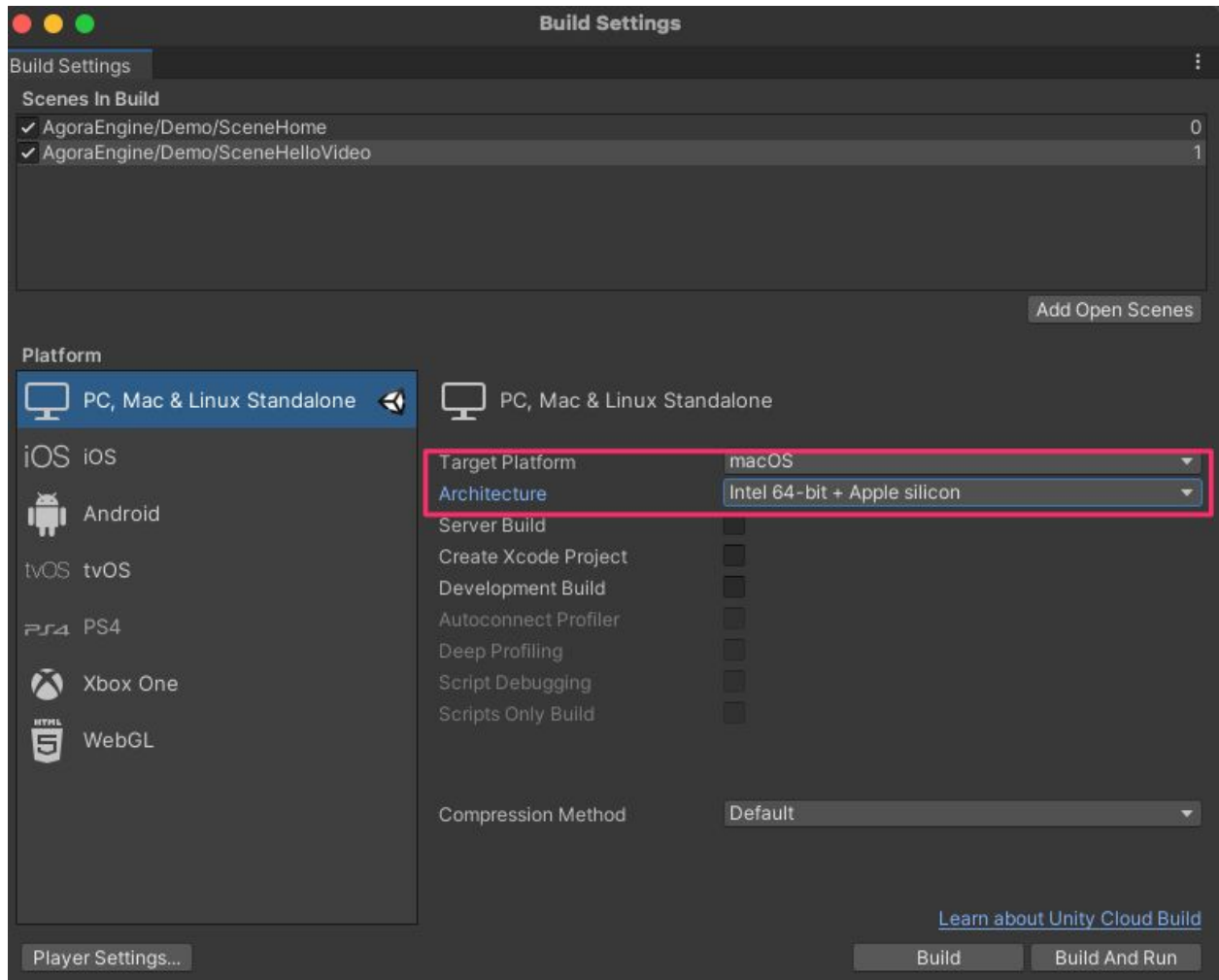
Setting Plugin Identities

For **Windows** builds, go into Assets->Plugins->x86_64 folder, select “Editor” and “Standalone” and then click Apply.



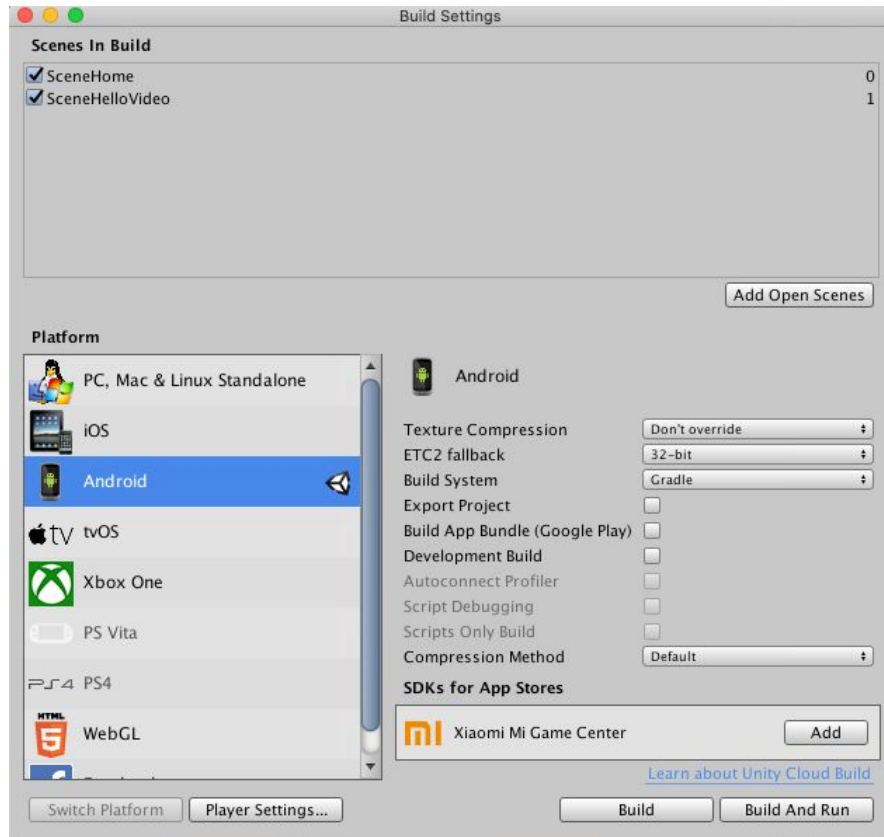
To ensure that the build plugin libraries don't collide, disable the identities of the files in the x86 folder.

For MacOS builds on Unity 2020.2 and up, make sure build target CPU type is selected to match library identities:



Android Build

Select **Android** from the platform list and click Switch Platform.

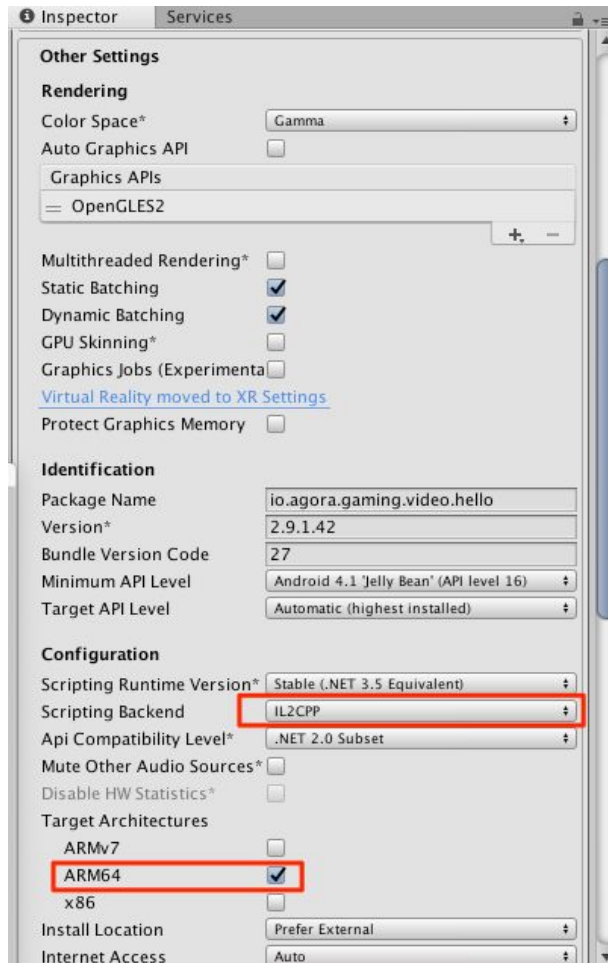


Android Build

Once Unity finishes the setup process, open the Player Settings and set a unique package name, e.g., io.agora.gaming.video.hello.

In order to comply with Google's [64 bit App requirement](#), make sure the following setting is selected:

1. Change the Scripting backend to IL2CPP.
2. Select ARM64 for the Target Architecture.



Android Build Settings

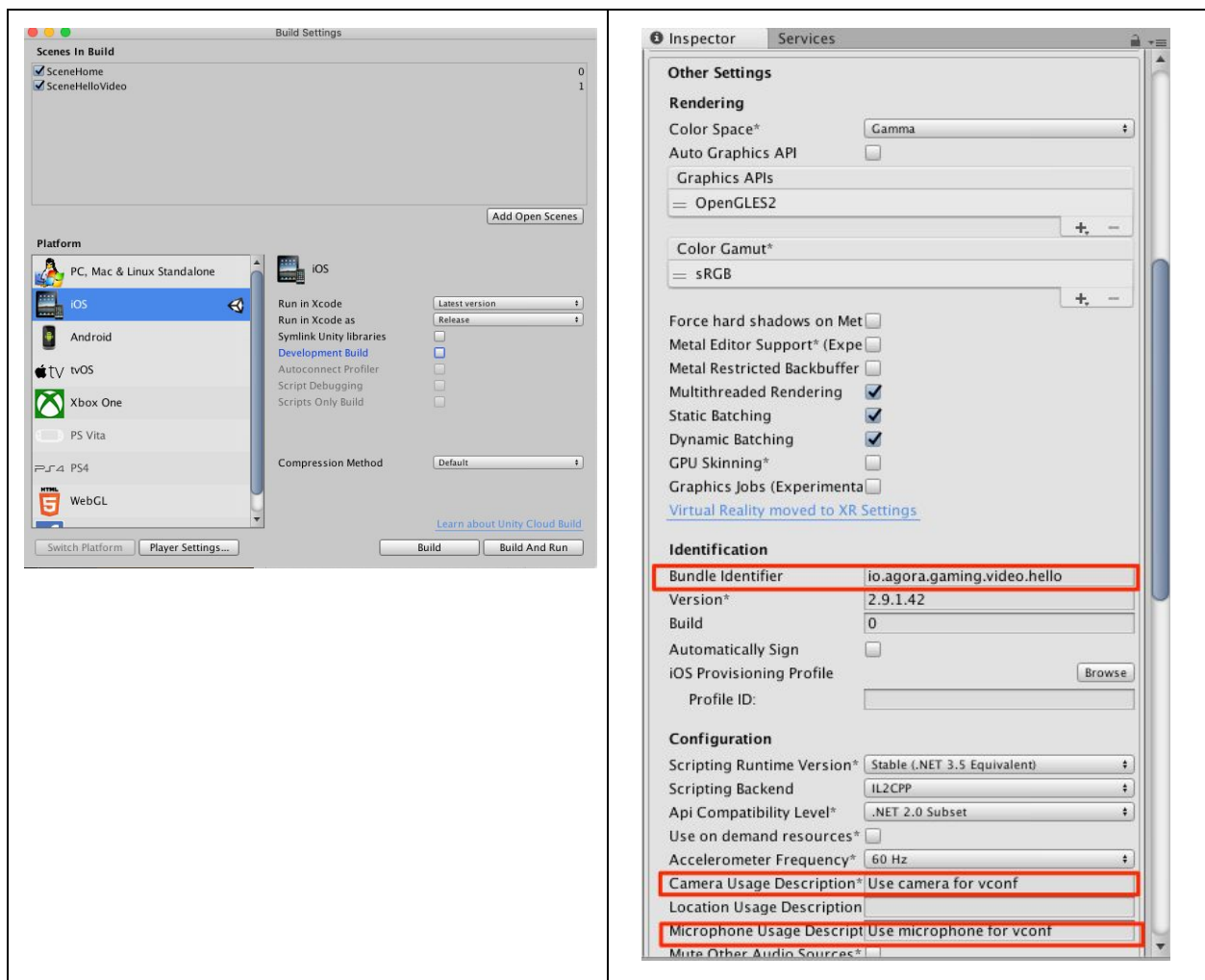
Leave other settings as is. For AR/VR enabled Applications, please refer to the separate README file.

iOS Build

Select **iOS** from the platform list and click Switch Platform.

The default build setting should work for most cases. The only custom settings are:

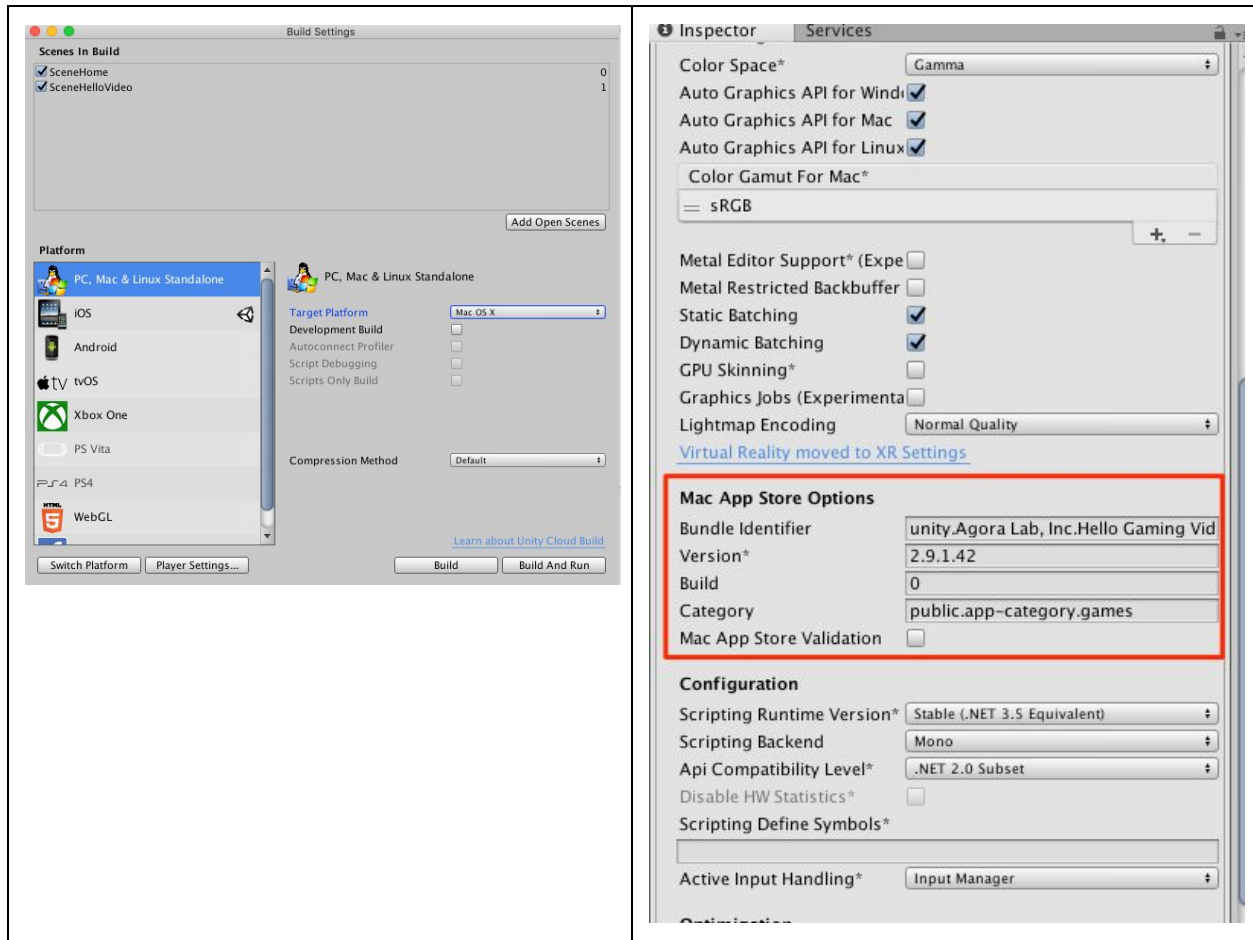
1. Change the Bundle Identifier to your own Bundle identifier so XCode can properly codesign the application.
2. Ensure the microphone permission has a description to allow the user to know why the microphone is being accessed by the application
3. Ensure the camera permission has a description to allow the user to know why the camera is being accessed by the application



iOS Build Settings

MacOS Build

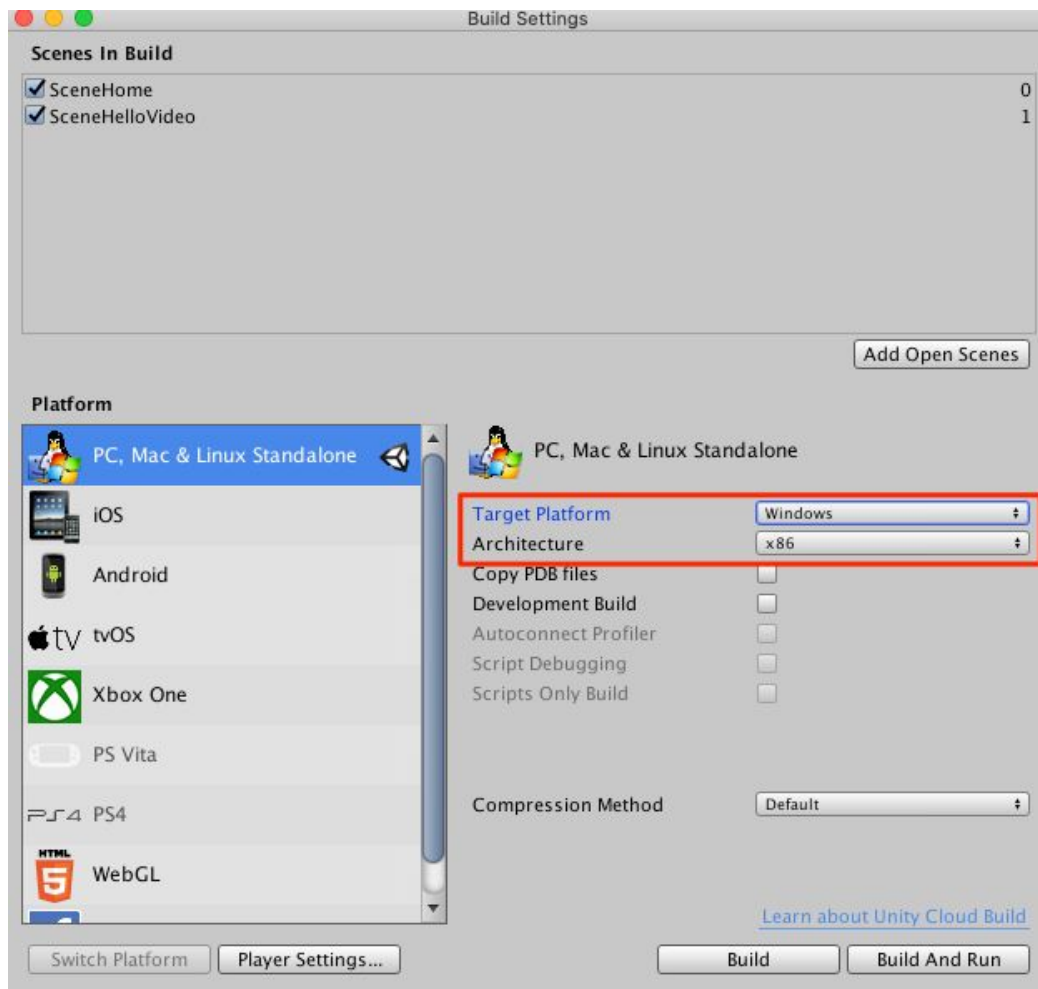
Select **“PC, Mac & Linux Standalone”** from the platform list and click Switch Platform. Then Select **“Mac OS X”** for the Target Platform. Fill in the appropriate Mac App Store Options are needed.



MacOS Build Settings

Windows Build

Select “**PC, Mac & Linux Standalone**” from the platform list and click Switch Platform. Then Select “**Windows**” for the Target Platform. Also choose x86 for 32 bit or x86_64 for 64 bit architecture according to your Application target.



Windows Build

Remember to set the appropriate Plugin library identities as described in the earlier section of this README file.

Conclusion

After the configuration has been set up according to the above rules, you should be able to build to the target platform. You can then talk to the other device using the Agora Voice SDK.

Other Resources

- The complete API documentation is available in the [Document Center](#).
- For technical support, submit a ticket using the [Agora Dashboard](#).
- Developer relations team: devrel@agora.io
- Release note:
https://docs.agora.io/en/Interactive%20Broadcast/release_unity_video?platform=Unity