Run Video Chat within Your Unity Application

This document shows how you can configure and run your Unity Application with Agora IO's Video SDK. The SDK supports iOS, Android, MacOS and Windows platform on Unity.

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.

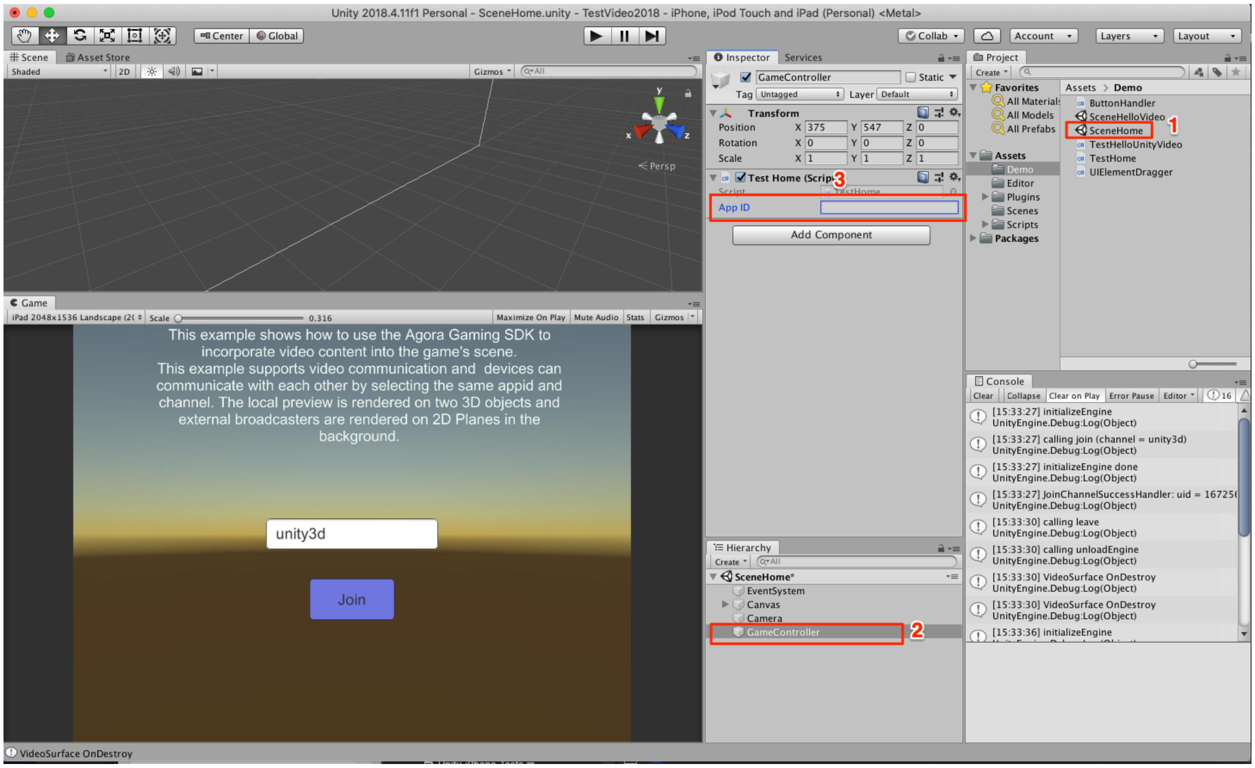
Open a new Unity project, and navigate to Unity Asset Store and search for "Agora Video SDK". Download and import the assets. Please scroll to the last section about upgrade if you are replacing the SDK on top of the old one.

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 Assets/Demo/SceneHome 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:



Unity 2018.4.11f1 Personal - SceneHome.unity - TestVideo2018 - iPhone, iPod Touch and iPad (Personal) <Metal>

This example shows how to use the Agora Gaming SDK to incorporate video content into the game's scene. This example supports video communication and devices can communicate with each other by selecting the same appid and channel. The local preview is rendered on two 3D objects and external broadcasters are rendered on 2D Planes in the background.

unity3d

GameController ☐ Static

Tag Untagged Layer Default

Transform

Position X 375 Y 547 Z 0

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

☑ Test Home (Script)

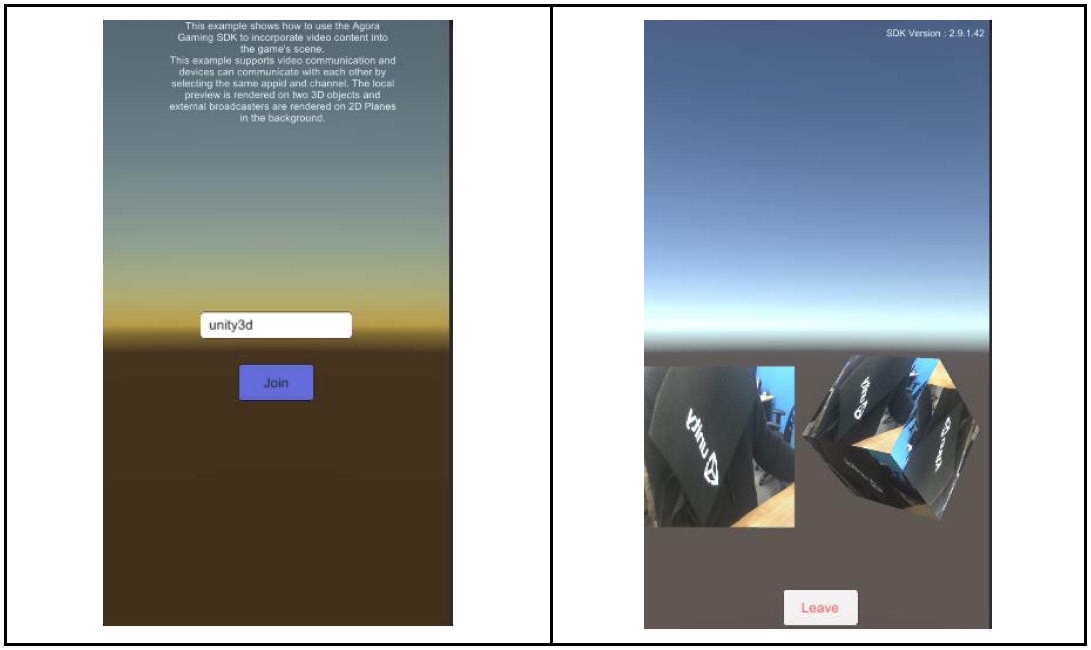
Script TestHome

App ID

```
[15:33:27] initializeEngine
UnityEngine.Debug:Log(Object)
[15:33:27] calling join (channel = unity3d)
UnityEngine.Debug:Log(Object)
[15:33:27] initializeEngine done
UnityEngine.Debug:Log(Object)
[15:33:27] JoinChannelSuccessHandler: uid = 167256
UnityEngine.Debug:Log(Object)
[15:33:30] calling leave
UnityEngine.Debug:Log(Object)
[15:33:30] calling unloadEngine
UnityEngine.Debug:Log(Object)
[15:33:30] VideoSurface OnDestroy
UnityEngine.Debug:Log(Object)
[15:33:30] VideoSurface OnDestroy
UnityEngine.Debug:Log(Object)
[15:33:36] initializeEngine
```

VideoSurface OnDestroy

At this step, you should be able to test the demo App within the Unity Editor. Input your desired channel name to the input field and click Join. You will be taken to the next scene and see a Quad and a Cube showing the local camera stream.



This example shows how to use the Agora Gaming SDK to incorporate video content into the game's scene. This example supports video communication and devices can communicate with each other by selecting the same appid and channel. The local preview is rendered on two 3D objects and external broadcasters are rendered on 2D Planes in the background.

unity3d

Player Settings for Building the Sample Application

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

Common Setting

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

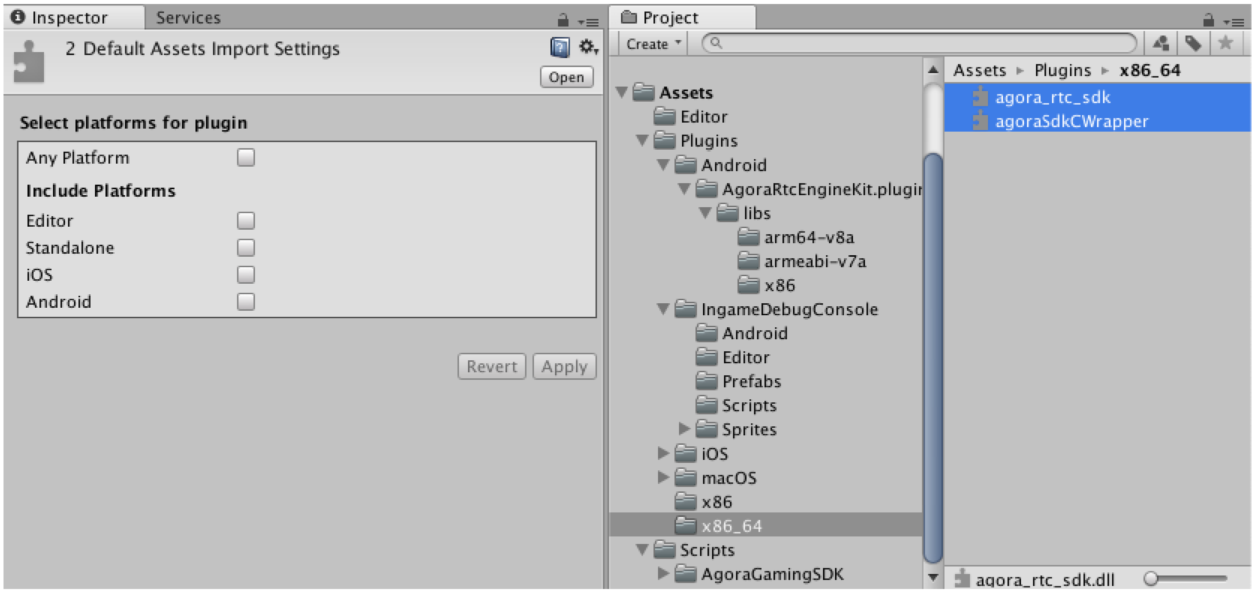
Setting Plugin Identities

* This only applies to **Windows** platform. Normally the library identities has been setup with the bundled SDK plugins. In case of manual override needed, follow the following steps:

For 64 bit **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.

Do the opposite for 32 bit Windows.



2 Default Assets Import Settings

Select platforms for plugin

Any Platform □

Include Platforms

Editor □

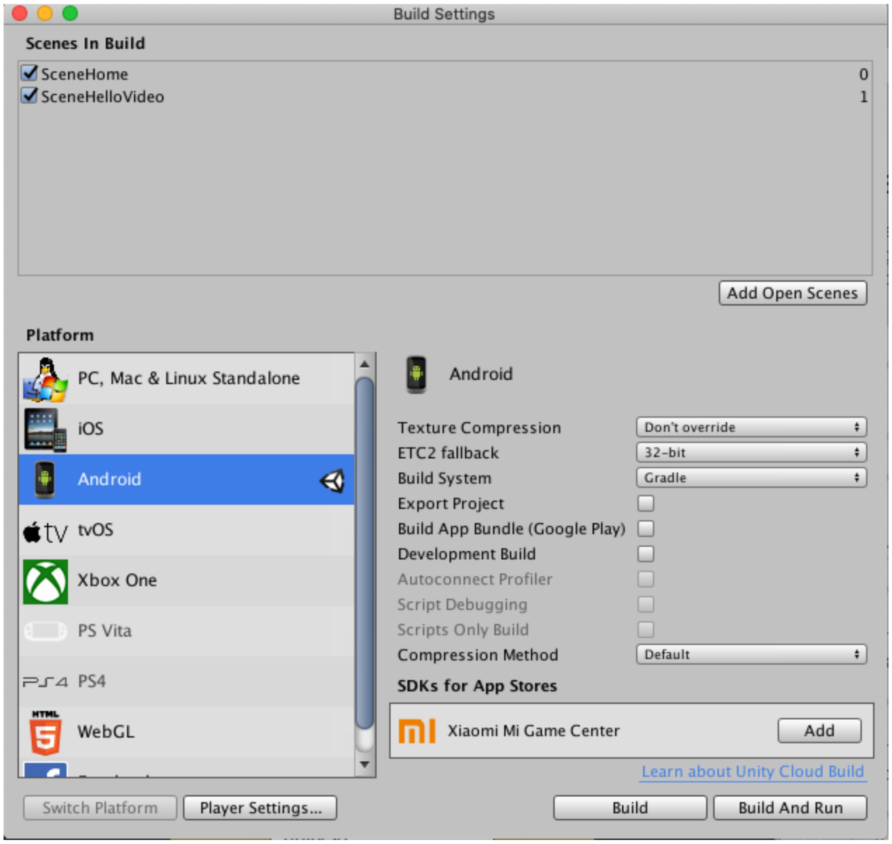
Standalone □

iOS □

Android □

Android Build

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



☑ SceneHome 0
☑ SceneHelloVideo 1

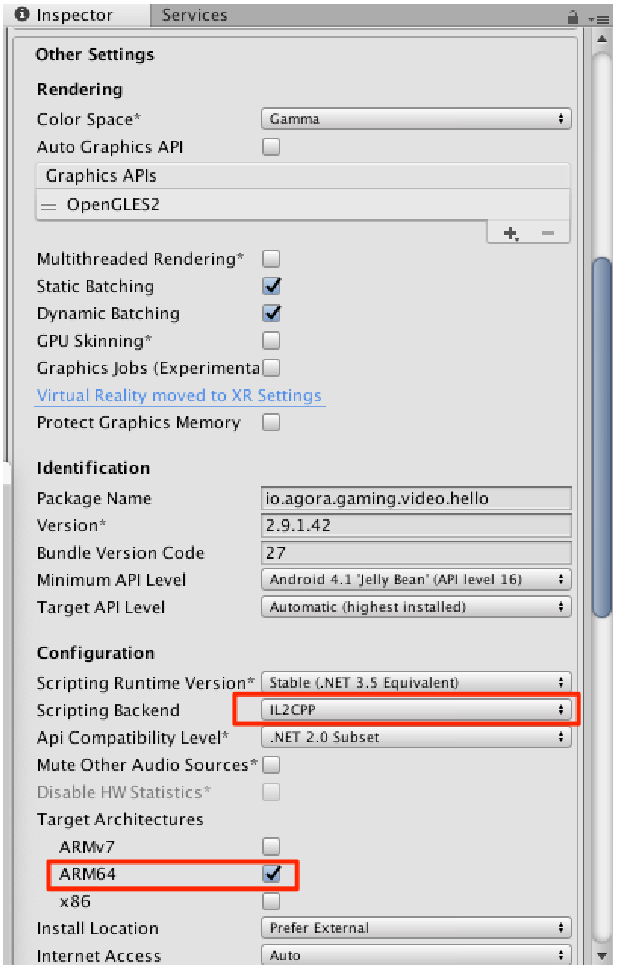
Texture Compression: Don't override
ETC2 fallback: 32-bit
Build System: Gradle
Export Project □
Build App Bundle (Google Play) □
Development Build □
Autoconnect Profiler □
Script Debugging □
Scripts Only Build □
Compression Method: Default

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.



Other Settings

Rendering

Color Space* Gamma

Auto Graphics API □

Graphics APIs

OpenGLES2

Multithreaded Rendering* □

Static Batching ☑

Dynamic Batching ☑

GPU Skinning* □

Graphics Jobs (Experimenta □

Virtual Reality moved to XR Settings

Protect Graphics Memory □

Identification

Package Name io.agora.gaming.video.hello

Version* 2.9.1.42

Bundle Version Code 27

Minimum API Level Android 4.1 'Jelly Bean' (API level 16)

Target API Level Automatic (highest installed)

Configuration

Scripting Runtime Version* Stable (.NET 3.5 Equivalent)

Scripting Backend IL2CPP

Api Compatibility Level* .NET 2.0 Subset

Mute Other Audio Sources* □

Disable HW Statistics* □

Target Architectures

ARMv7 □

ARM64 ☑

x86 □

Install Location Prefer External

Internet Access Auto

Android Build Settings

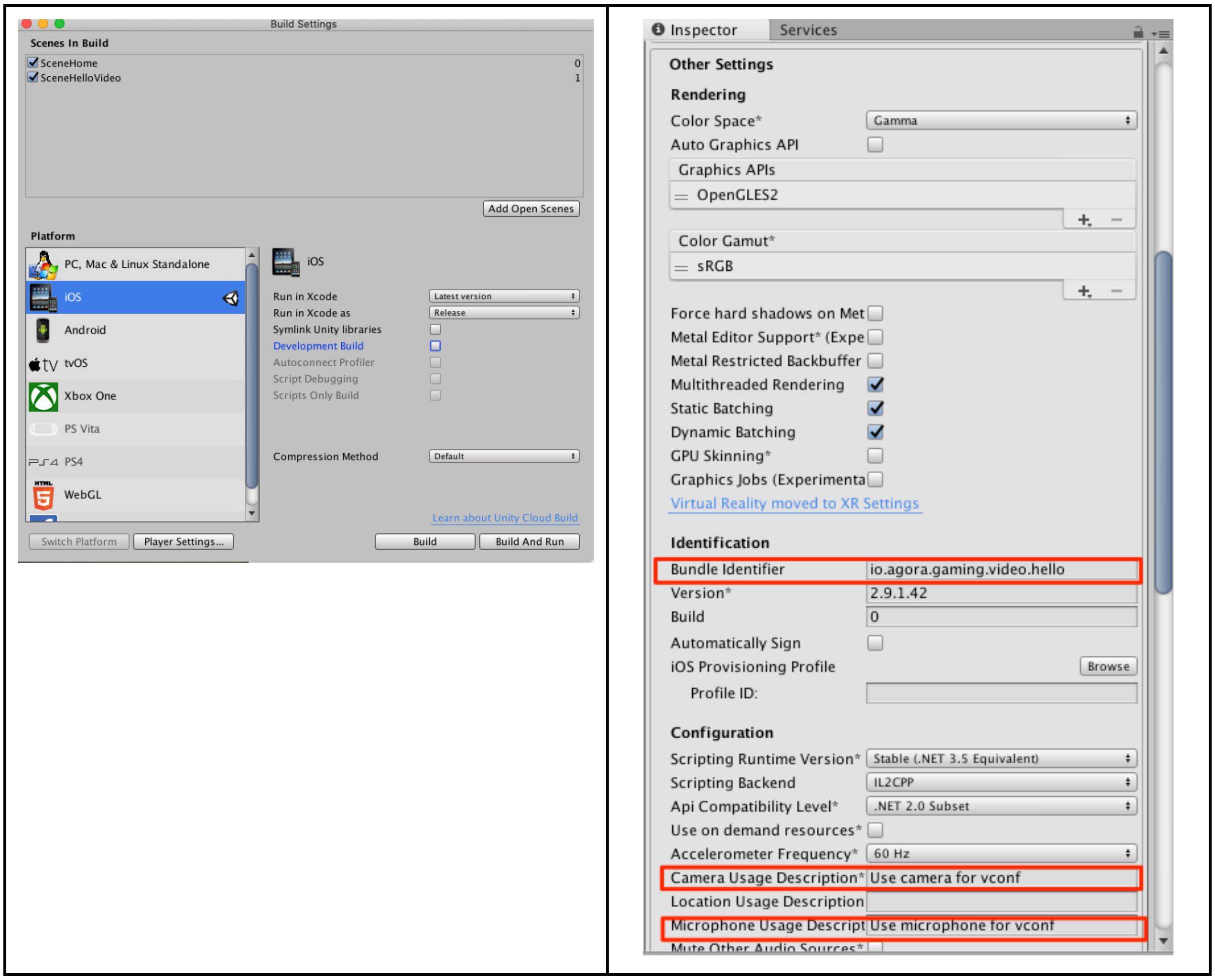
Leave other settings as is. For AR/VR enabled Applications, please refer to the separate README file.

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



Build Settings

Scenes In Build

- ☑ SceneHome 0
- ☑ SceneHelloVideo 1

Platform: iOS

- Run in Xcode: Latest version
- Run in Xcode as: Release
- Symlink Unity libraries: □
- Development Build: □
- Autoconnect Profiler: □
- Script Debugging: □
- Scripts Only Build: □
- Compression Method: Default

Inspector — Other Settings

Rendering

- Color Space*: Gamma
- Auto Graphics API: □
- Graphics APIs: OpenGLES2
- Color Gamut*: sRGB
- Force hard shadows on Met: □
- Metal Editor Support* (Expe: □
- Metal Restricted Backbuffer: □
- Multithreaded Rendering: ☑
- Static Batching: ☑
- Dynamic Batching: ☑
- GPU Skinning*: □
- Graphics Jobs (Experimenta: □
- Virtual Reality moved to XR Settings

Identification

- Bundle Identifier: io.agora.gaming.video.hello
- Version*: 2.9.1.42
- Build: 0
- Automatically Sign: □
- iOS Provisioning Profile
- Profile ID:

Configuration

- Scripting Runtime Version*: Stable (.NET 3.5 Equivalent)
- Scripting Backend: IL2CPP
- Api Compatibility Level*: .NET 2.0 Subset
- Use on demand resources*: □
- Accelerometer Frequency*: 60 Hz
- Camera Usage Description*: Use camera for vconf
- Location Usage Description:
- Microphone Usage Descript: Use microphone for vconf
- Mute Other Audio Sources*: □

iOS Build Settings

Embedding Frameworks

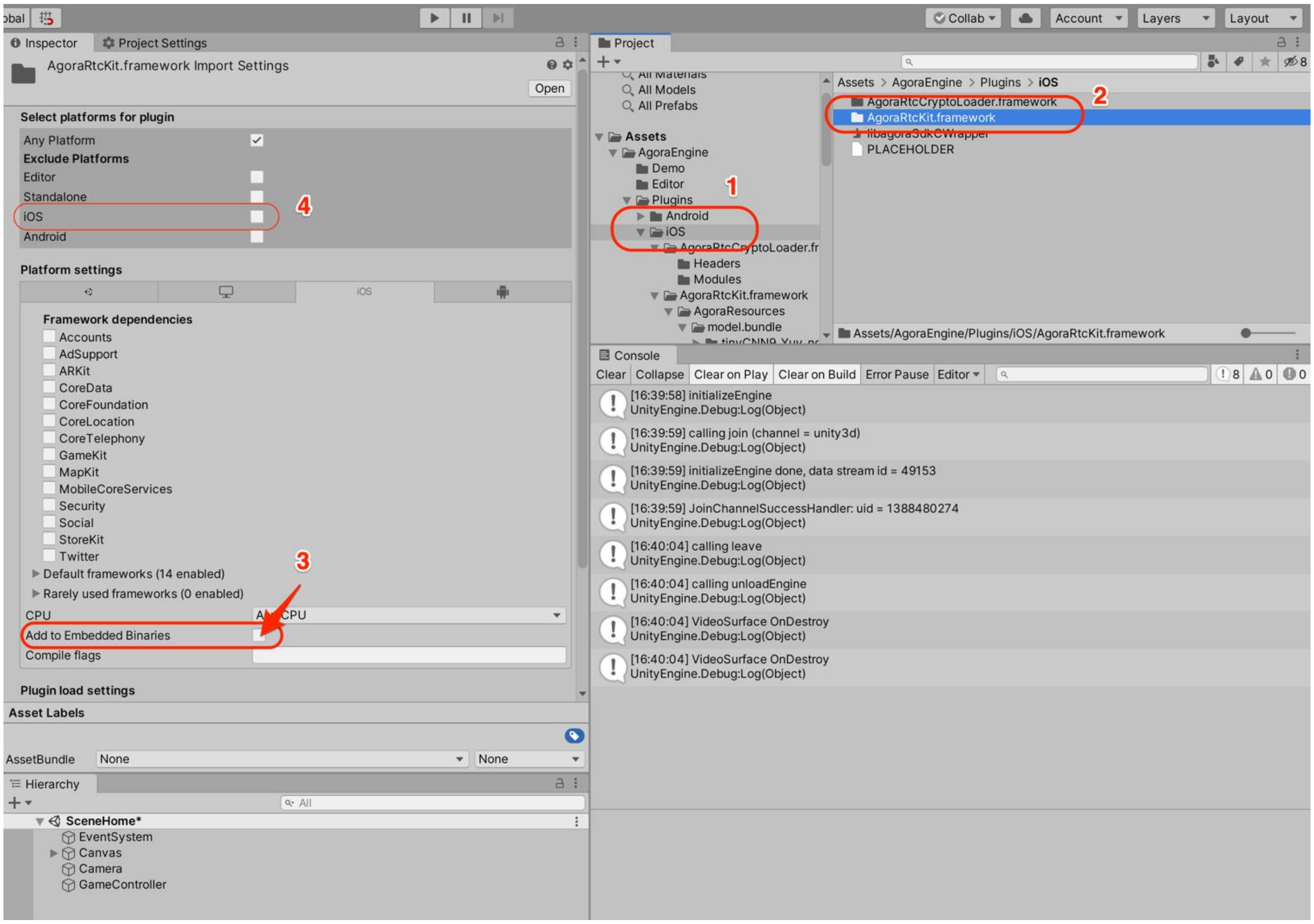
Starting with SDK version 3.0.1, the iOS plugin frameworks are dynamically loaded and need to be embedded into the libraries. The Post Processing build script BL_BuildPostProcess.cs should have taken care of this. If your iOS build runs perfectly, then no need to read on for the following help text.

Just in case of anything missed on building your own projects, please be sure that the following Frameworks and library is put into Embedded section for XCode:

Make sure you do for the three items in the iOS folder:

- AgoraRtcCryptoLoader.framework
- AgoraRtcKit.framework
- libagoraSdkCWrapper

Make sure you choose **iOS** as the identity for these files, especially libagoraSdkCWrapper.



AgoraRtcKit.framework Import Settings

Select platforms for plugin

Any Platform ☑

Exclude Platforms

Editor □

Standalone □

iOS □

Android □

Platform settings

Framework dependencies

□ Accounts
□ AdSupport
□ ARKit
□ CoreData
□ CoreFoundation
□ CoreLocation
□ CoreTelephony
□ GameKit
□ MapKit
□ MobileCoreServices
□ Security
□ Social
□ StoreKit
□ Twitter

▶ Default frameworks (14 enabled)

▶ Rarely used frameworks (0 enabled)

CPU: AnyCPU

Add to Embedded Binaries □

Compile flags

Plugin load settings

Asset Labels

AssetBundle: None / None

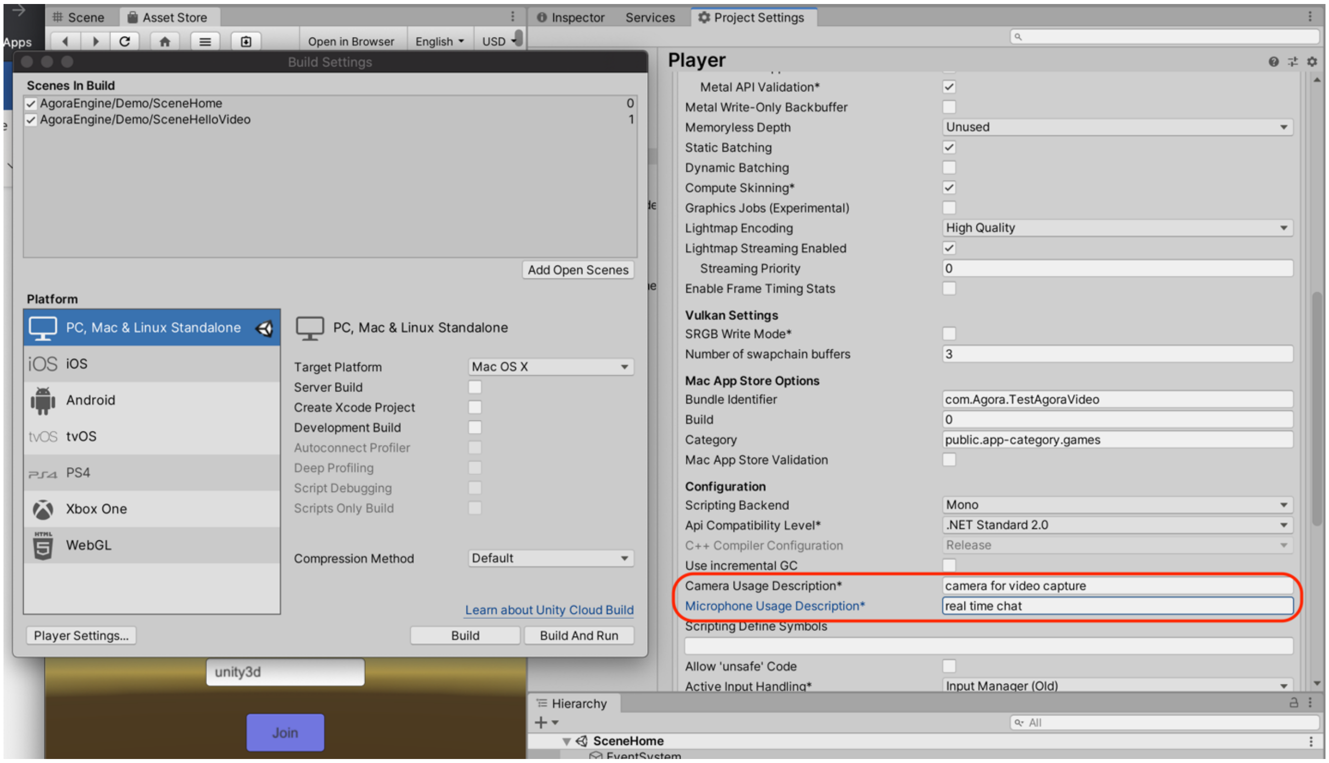
Assets/AgoraEngine/Plugins/iOS/AgoraRtcKit.framework

```
[16:39:58] initializeEngine
UnityEngine.Debug:Log(Object)
[16:39:59] calling join (channel = unity3d)
UnityEngine.Debug:Log(Object)
[16:39:59] initializeEngine done, data stream id = 49153
UnityEngine.Debug:Log(Object)
[16:39:59] JoinChannelSuccessHandler: uid = 1388480274
UnityEngine.Debug:Log(Object)
[16:40:04] calling leave
UnityEngine.Debug:Log(Object)
[16:40:04] calling unloadEngine
UnityEngine.Debug:Log(Object)
[16:40:04] VideoSurface OnDestroy
UnityEngine.Debug:Log(Object)
[16:40:04] VideoSurface OnDestroy
UnityEngine.Debug:Log(Object)
```

Unity Editor Embedded Binary setting (Unity 2018,2019 and up)

MacOS Build

Select **“PC, Mac & Linux Standalone”** from the platform list and click Switch Platform. Then Select “**Mac OS X**” for the Target Platform. Besides regular App Store required information, it is **very important** to fill in the **Camera and Microphone** usage description. Without this your app may crash on MacOS Catalina for privacy restriction reasons.



Build Settings

Scenes In Build

☑ AgoraEngine/Demo/SceneHome 0
☑ AgoraEngine/Demo/SceneHelloVideo 1

Add Open Scenes

Platform

- PC, Mac & Linux Standalone
- iOS
- Android
- tvOS
- PS4
- Xbox One
- WebGL

PC, Mac & Linux Standalone

Target Platform	Mac OS X
Server Build	□
Create Xcode Project	□
Development Build	□
Autoconnect Profiler	□
Deep Profiling	□
Script Debugging	□
Scripts Only Build	□
Compression Method	Default

Learn about Unity Cloud Build

Player Settings... | Build | Build And Run

Player

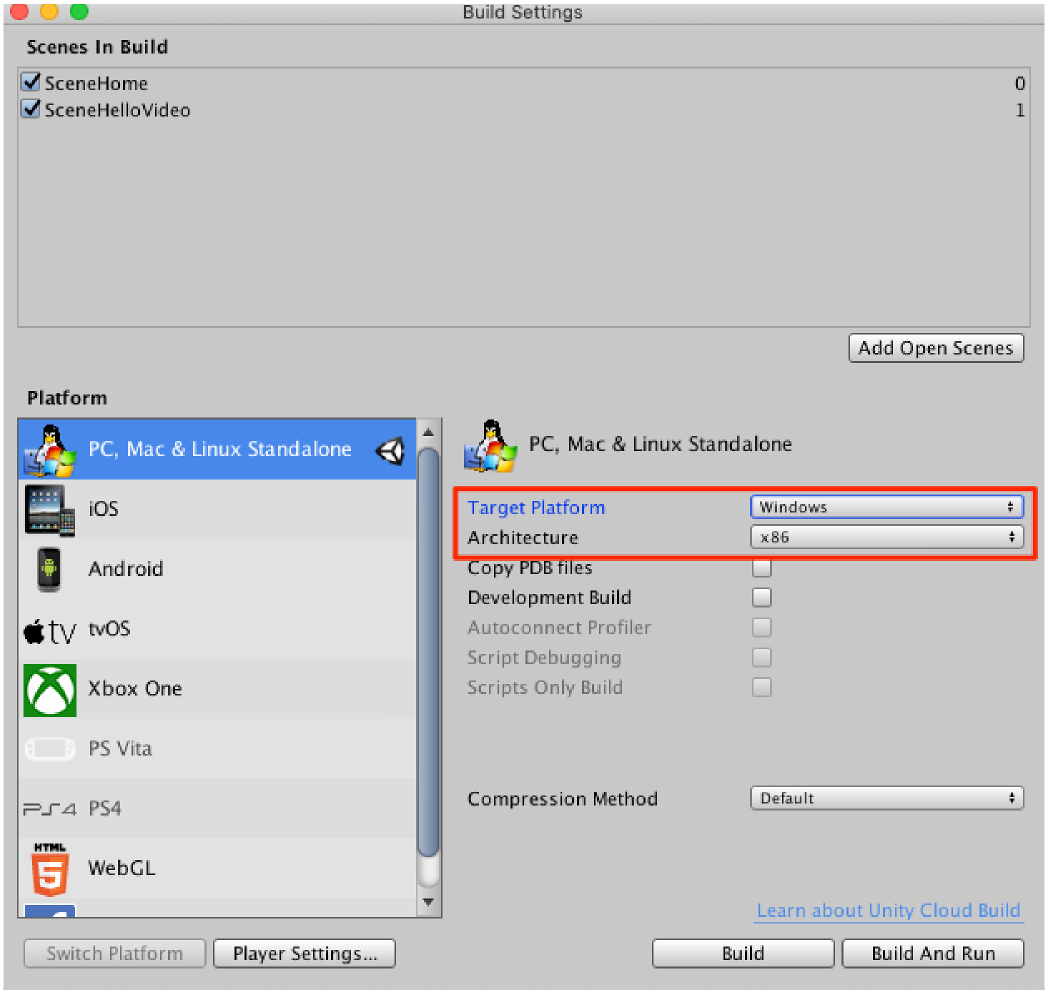
Metal API Validation*	☑
Metal Write-Only Backbuffer	□
Memoryless Depth	Unused
Static Batching	☑
Dynamic Batching	□
Compute Skinning*	☑
Graphics Jobs (Experimental)	□
Lightmap Encoding	High Quality
Lightmap Streaming Enabled	☑
Streaming Priority	0
Enable Frame Timing Stats	□
Vulkan Settings	
SRGB Write Mode*	□
Number of swapchain buffers	3
Mac App Store Options	
Bundle Identifier	com.Agora.TestAgoraVideo
Build	0
Category	public.app-category.games
Mac App Store Validation	□
Configuration	
Scripting Backend	Mono
Api Compatibility Level*	.NET Standard 2.0
C++ Compiler Configuration	Release
Use incremental GC	□
Camera Usage Description*	camera for video capture
Microphone Usage Description*	real time chat
Scripting Define Symbols	
Allow 'unsafe' Code	□
Active Input Handling*	Input Manager (Old)

SceneHome
EventSystem

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.



Build Settings

Scenes In Build

☑ SceneHome 0
☑ SceneHelloVideo 1

PC, Mac & Linux Standalone

Target Platform: Windows
Architecture: x86
Copy PDB files □
Development Build □
Autoconnect Profiler □
Script Debugging □
Scripts Only Build □

Compression Method: Default

Windows Build

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

Upgrading SDK

From versions prior to 2.9.1:

- Remove files according to this SDK directory tree: https://bit.ly/3eO16Jy
- Clean up Unity Asset cache (explained later)

From version 2.9.x to 3.0.x:

- Remove the entire file tree under AgoraEngine
- Importing the new SDK

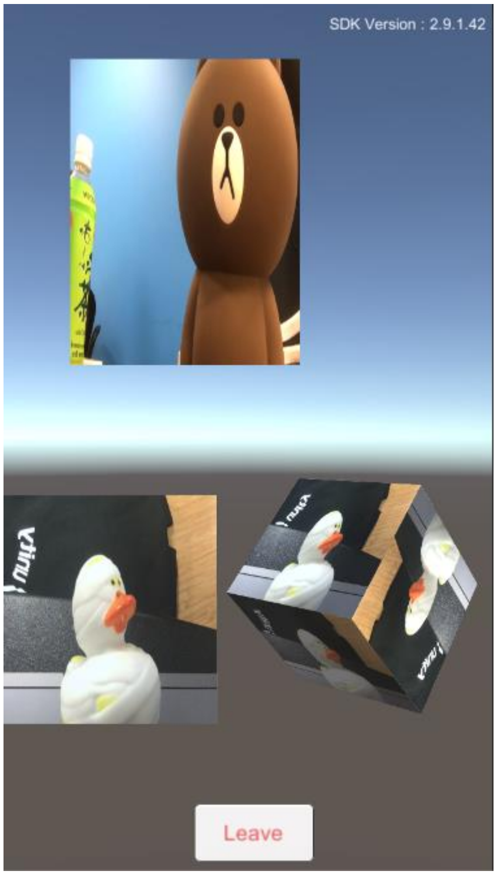
How to **Clean up old Unity Asset Cache**

- Refer to this documentation link to find the location of the cache folder: https://docs.unity3d.com/Manual/AssetPackages.html
- Note that the Asset Store fold may have a version attached to it. For example, this is the current location of the cache to the Agora Video SDK on MacOS:

```
~/Library/Unity/Asset Store-5.x/Agoraio/ScriptingVideo
```

Conclusion

After the configuration has been set up according to the above rules, you should be able to build to the target platform. The following is the sample demo running on device. The remote player will show up in the RawImage generated by code in the upper screen area. User may drag the image around to change its position when more players join the video chat channel.



Other Resources

- The complete API documentation is available in the [Document Center](.).
- For technical support, submit a ticket using the Agora Dashboard or
- Join our slack community: https://bit.ly/39j4I5h
- Help each other at https://agoraiodev.slack.com/messages/unity-help-me
- Developer relations team: devrel@agora.io
- Release note: https://docs.agora.io/en/Interactive%20Broadcast/release_unity_video?platform=Unity