For full documentation, visit the AVPro Movie Capture Developer Portal

Table of Contents

- Articles
  - About
    - Introduction
    - Features
    - Requirements
    - Download
    - Asset Files
  - Usage
    - Installation
    - Demos
    - Quick Start
    - Screen Capture
    - Camera Capture
    - Webcam/Texture Recording
    - 360 Rendering
    - Transparent Capture
    - Screenshot
    - Named Pipe
    - Audio Capture
    - Photo Library
    - Completing a Capture
    - Scripting
    - Codecs
  - Capture Components
    - Capture From Screen
    - Capture From Texture
    - Capture From Camera
    - Capture From Camera 360
    - Capture From Camera 360 ODS
  - Audio Capture Components
    - Capture From AudioListener
    - Capture From AudioRenderer
    - Capture From AudioMixer

- Capture From Wwise
- Utility Components
  - Camera Selector
  - AudioSource To WAV
  - Ambisonic Source
  - Ambisonic WAV Writer
  - Capture GUI
  - Mouse Cursor
  - Motion Blur
  - Timeline Controller
- Platform Notes
  - Windows
  - macOS
  - iOS
  - Android

Introduction

AVPro Movie Capture is a powerful cross-platform video capture/rendering plugin for Unity created by RenderHeads

We created this plugin primarily for use with our internal projects. At RenderHeads we create interactive installations for events and educational games for museums. Capturing videos of user interactions or rendering out content from Unity for clients are important aspects of our work so we created this tool to streamline our workflow and improve the quality of our work.

Since releasing it to the public we have made a lot of modifications based on user feedback, and we aim to make this the most reliable and easy to use video capture system for Unity.

Features

Features of AVPro Movie Capture include:

- **General**
  - Very easy to use
  - Optimised for high performance
- **Video Capture**
  - Realtime capture and offline rendering
  - Capture from screen, camera, webcam or texture
  - 180 and 360 degree VR capture (mono and stereo)
  - Omni-directional stereo (ODS) support for VR renders
  - Linear and Gamma colour spaces supported
  - Motion blur rendering option
  - Transparent alpha channel support
  - 8K and above video support (on supported hardware)
- **Audio Capture**
  - Capture audio as part of video capture
  - Realtime audio capture support from Unity or microphone
  - Offline Unity audio capture support
  - Offline Wwise audio capture support
  - Realtime system-wide audio capture on Windows
  - 1st, 2nd, 3rd order ambisonic audio capture to WAV file
- **Highly optimised**
  - GPU video encoding support
  - Optimised native Direct3D, OpenGL and Metal support
  - Focus on minimal garbage generation
- **Other**
  - Injection of stereo and 360 spherical data into videos
  - D3D12 support
- **Compatibility**
  - Unity 2017.x - 2022.x and above supported
  - Cross-platform with support for iOS, macOS, Windows and Android
  - Works in editor and standalone builds
- **Well supported**
  - Free watermarked trial version available (download here)
  - Good documentation
  - Public issue tracker

Requirements

System Requirements

Unity

- 2017.x, 2018.x, 2019.x, 2020.x, 2021.x, 2022.x
- 5.6.x (Windows and macOS only)

Platforms

- macOS
  - macOS High Sierra (10.13) and above
  - 64bit only (x86_64, arm64)
  - Metal and OpenGL Core graphics APIs
  - Unity 5.6.x only supports OpenGL Core
- iOS
  - iOS 11.0 and above
  - arm64 only
  - Metal graphics API
- Android
  - Android 8.0 (Oreo, API level 26) and above (ARM7, ARM64, x86 and x86_64)
  - OpenGLES3
- Windows
  - Windows XP SP3, 7, 8, 8.1, 10 and above (x86 and x86_64)
  - Direct3D 11 and 12 graphics APIs

Platforms not Supported

- Windows UWP
- WebGL
- tvOS
- Linux desktop
- Tizen
- Lumin (Magic Leap)
- Samsung TV
- Game Consoles (XBox, PS4 etc)

Downloads, Editions & Upgrades

Free Trial Version

1. Fully featured watermarked trial versions can be downloaded here: https://github.com/RenderHeads/UnityPlugin-AVProMovieCapture/releases
2. Once the .unitypackage file has been downloaded, follow the installation instructions.

Purchase

All editions of AVPro Movie Capture can be purchased via the Unity Asset Store:

- Basic Edition (all platforms with limits)
- Ultra Edition (all platforms)
- Desktop Edition (Windows and macOS only)
- Mobile Edition (Android and iOS only)

Editions

We've made several editions of AVPro Movie Capture available so you can pick the one that's best for your project:

	TRIAL EDITION	BASIC EDITION	DESKTOP EDITION	MOBILE EDITION	ULTRA EDITION
Platforms					
Windows standalone	Watermarked	**Yes**	**Yes**	Watermarked	**Yes**
macOS standalone	Watermarked	**Yes**	**Yes**	Watermarked	**Yes**
Android	Watermarked	**Yes**	Watermarked	**Yes**	**Yes**
iOS	Watermarked	**Yes**	Watermarked	**Yes**	**Yes**
Features					
Screen capture	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
Camera capture	**Yes**	.	**Yes**	**Yes**	**Yes**
Texture capture	**Yes**	.	**Yes**	**Yes**	**Yes**
Offline rendering	**Yes**	.	**Yes**	**Yes**	**Yes**
Other codecs besides H264/AAC/MP4	**Yes**	.	**Yes**	**Yes**	**Yes**
Custom bit-rate / frame-rate	**Yes**	.	**Yes**	**Yes**	**Yes**
Transparency	**Yes**	.	**Yes**	**Yes**	**Yes**

	TRIAL EDITION	BASIC EDITION	DESKTOP EDITION	MOBILE EDITION	ULTRA EDITION
External audio sources	**Yes**	.	**Yes**	**Yes**	**Yes**
Other					
Support Priority	Normal	Normal	Normal	Normal	Priority
Multi-site license	N/A	No	No	No	No
Price	Free	$50	$250	$250	$350
Link	Download	Store	Store	Store	Store

Upgrade Paths

In many cases a developer may own one edition and as their needs increase may need the features of other editions. To cater for this we have set up some upgrade paths on the Unity Asset Store to make this cost effective.

Owners of AVPro Movie Capture 4.x

NOTE

Despite AVPro Movie Capture 4.x being deprecated, existing owners of this version can still access it for download via the Unity Asset Store. Make sure that you're logged in with the same account that you used to originally purchase it and it should be visible in your list to download the last published version.

There is **discounted** pricing automatically applied (for a limited time) for owners of the retired product **AVPro Movie Capture 4.x**:

Upgrade Pricing

	BASIC EDITION	DESKTOP EDITION	MOBILE EDITION	ULTRA EDITION
Full Price:	**$50**	**$250**	**$250**	**$350**
Upgrading from another edition:				
Basic Edition	.	$200	$200	$300
Desktop Edition	.	.	.	$100
Mobile Edition	.	.	.	$100
Upgrading from AVPro Movie Capture 4.x:				
Windows Edition	.	$100	.	$250
macOS Edition	.	$100	.	$250
iOS Edition	.	.	$150	$250

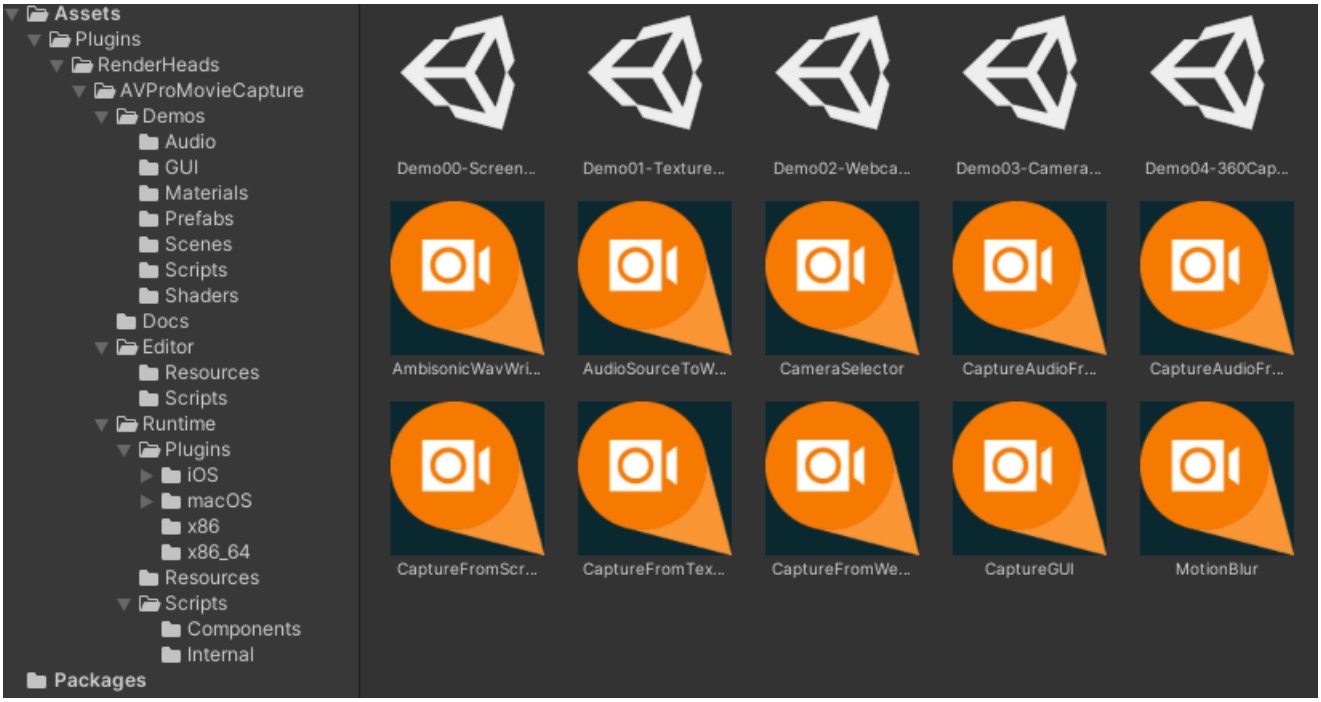
	BASIC EDITION	DESKTOP EDITION	MOBILE EDITION	ULTRA EDITION
Full Edition	.	.	.	$150

Future Updates

We plan to continue supporting this product until the next major version (6.0.0). This will include bug fixes, new features and improvements.

The product will be actively supported for around 2 years and then replaced with the next major version around July 2025. We feel that being open about our strategy for upgrades is beneficial to developers planning their projects. Owners of the current 5.x version will get a discounted price when upgrading to 6.x

Asset Files



The AVPro Movie Capture asset package contains files organised into 4 major folders:

1. Runtime Folder
2. Demos Folder
3. Editor Folder
4. Docs Folder

Runtime Folder

This folder contains the primary elements of AVPro Movie Capture. It builds to the `RenderHeads.AVProMovieCapture.Runtime` assembly.

- **Plugins/**

  Contains all native plugin files

- **Scripts/**

  - **Components/**

    Monobehaviour Components

  - **Internal/**

    Internal code that is not usually modified by users

- **Resources**

  Resources (mostly Shaders) used internally and always included in builds

Demos Folder

This folder contains several examples scenes showing how to use some of the components included with AVPro Movie Capture. This folder builds to its own assembly and is optional.

Editor Folder

This folder contains editor-only scripts which are not usually modified by users. It builds to the `RenderHeads.AVProMovieCapture.Editor` assembly.

Docs Folder

This folder contains a PDF version of the documentation

Installation

Installation Steps

If you are installing from scratch:

1. Either download the latest trial version: https://github.com/RenderHeads/UnityPlugin-AVProMovieCapture/releases
   or
   Purchase the latest version from the Unity Asset Store
2. Open your project in Unity
3. For the trial version, import the `unitypackage` file into your Unity project by double-clicking the file. For the Asset Store simply import the file directly from the Asset Store UI.
4. If prompted to upgrade some scripts click Yes

Upgrade Steps

If you are upgrading to a new version:

1. Close your Unity project
2. Open your project again and proceed to the next step immediately without running the scene or clicking on any AVPro Movie Capture components, as this can cause the plugin files to become locked
3. If updating from the Unity Asset Store, simple select the new version to import. If updating from a `unitypackage` file (eg the trial version), double-clicking the file to import it
4. If prompted to upgrade some scripts click Yes

Watermarked Trial Version

If you are using a trial version of the plugin then you will see a watermark displayed over the video. The watermark is in the form of a “RenderHeads” logo that animates around the screen, or a thin horizontal bar that moves around the screen.

Installation & Watermark Troubleshooting

It’s often a good idea to check that the correct version is reported after a plugin upgrade. You can check which version you have installed by adding t component to your scene and clicking on the “About / Help” button in the Inspector for that component. The version number is displayed in this box.

The full version of AVPro Movie Capture has no watermarks for any platforms. If you use one of the platform specific editions (eg AVPro Movie Capture (iOS), or AVPro Movie Capture (Windows) then you will not see the watermark on the platform you purchased for, but you will see the watermark on the other platforms. For example if you purchased AVPro Movie Capture (iOS) then you will still see the watermark in the Unity editor as this is running on Windows/macOS, but the videos played back when you deploy to your iOS device will be watermark-free.

Installing Multiple Single-Platform Packages

If you are not using the AVPro Movie Capture full package and instead have opted to purchase multiple individual single-platform packages then the installation must be done carefully, especially when upgrading to a new version.

If you have installed the iOS package then it will also contain plugins for all of the other platforms but with the watermark enabled. This means that if you then try to install another AVPro MovieCapture package it may not override the plugins correctly. Here is how to resolve this using the iOS and Windows package as examples:

1. Open a fresh Unity instance (this is important as otherwise Unity may have locked the plugin files which prevents them from being upgraded)

2. Import the iOS package
3. Import the Windows package, but make sure that you have the iOS native plugin file unticked (so that it is not overwritten) A similar process can be applied for other package combinations.

List of native plugin files that need to be selectively chosen when replacing specific platforms:

- macOS
    - Plugins/AVProMovieCapture.bundle
- iOS
    - Plugins/iOS/AVProMovieCapture.framework
- Windows
    - Plugins/x86/AVProMovieCapture.dll
    - Plugins/x86_64/AVProMovieCapture.dll

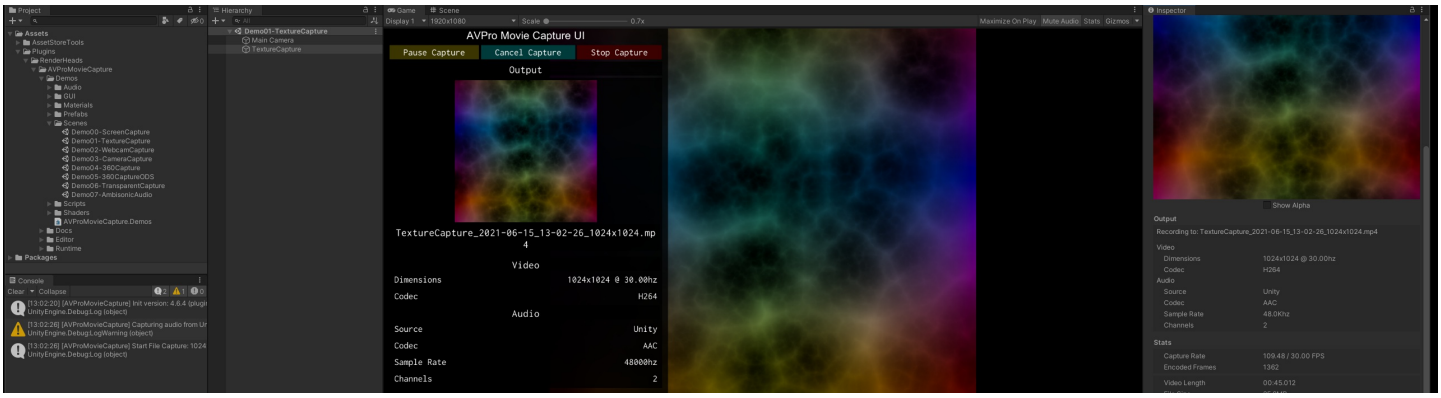
Demos

The package includes several demo scenes which demonstrate how to use the major components.

Demo00 - Screen Capture

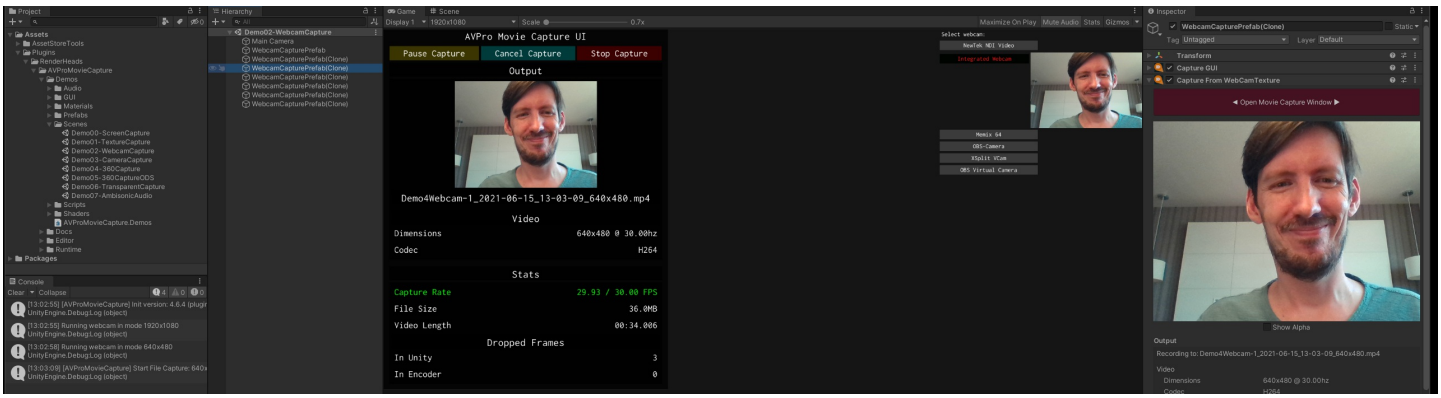
This scene is set up to capture the final output on the screen to a video file. It uses the `CaptureFromScreen` component. The capture runs in real-time capture mode.

Demo01 - Texture Capture



This scene is set up to capture a dynamic `RenderTexture` texture to a video file. It uses the `CaptureFromTexture` component. This capture runs in offline render mode so that every frame is captured and the render completes faster than real-time.

Demo02 - Webcam Capture

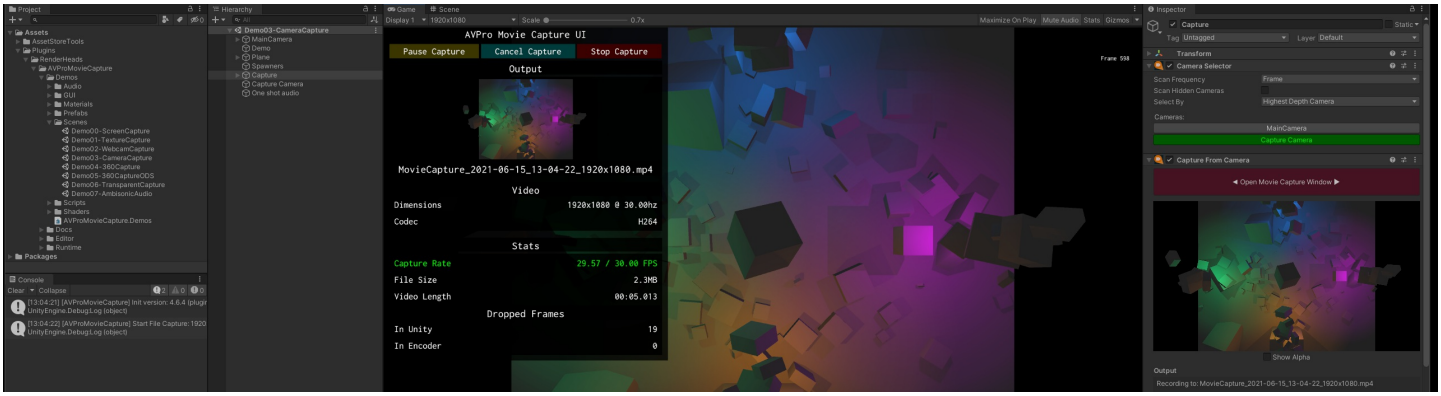


This scene is set up to capture a Unity `WebCamTexture` texture to a video file. It uses the `CaptureFromTexture` component. It uses manual triggering via script to inform at component each time the texture is updated so that no frames are missed.

NOTE

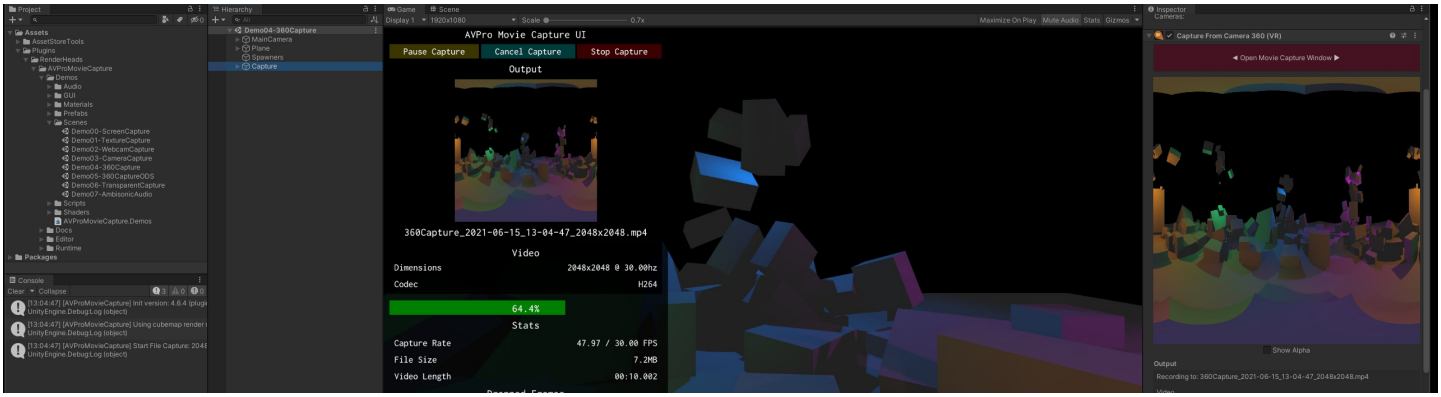
To use the `WebCamTexture` capture component/demo, the string `AVPRO_MOVIECAPTURE_WEBCAMTEXTURE_SUPPORT` must be added to `Scriping Define Symbols` in `Player Settings > Other Settings > Script Compilation`. This is because on some platforms (iOS, macOS, Android at least) Unity's build system automatically forces webcam permissions for apps with source code containing the string "WebCamTexture", which can be build issues if this feature is not required.

Demo03 - Camera Capture



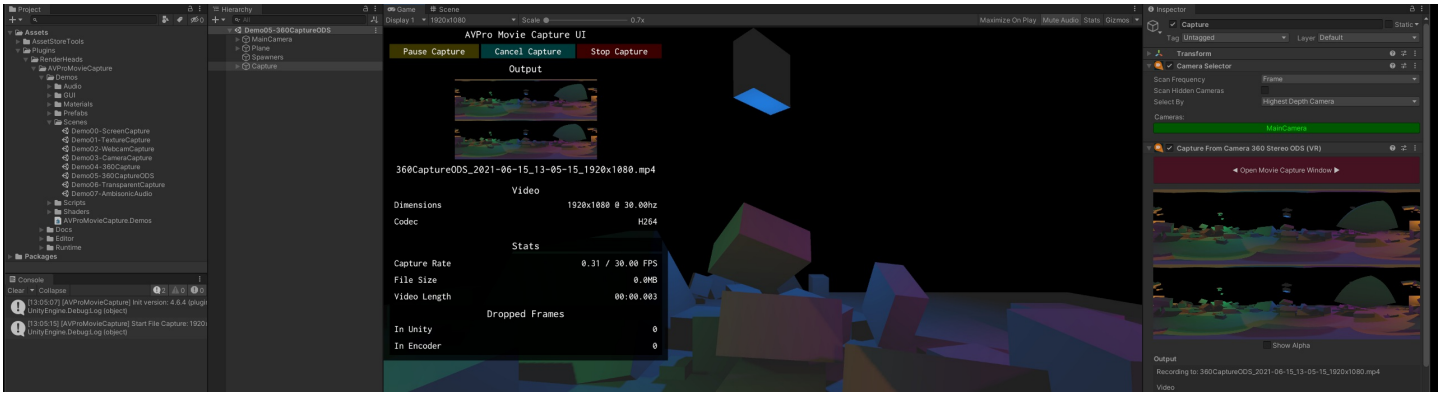
This scene is set up to capture a Unity `Camera` to a video file. It uses the `CaptureFromCamera` component. The capture runs in real-time capture mode. The `CameraSelector` component can be used to switch `Camera` at runtime.

Demo04 - 360 Capture



This scene is set up to capture a Unity `Camera` to a video file in 360 equirectangular format. It uses the `CaptureFromCamera360` component. This capture runs in offline render mode so that every frame is captured and the render completes faster than real-time.

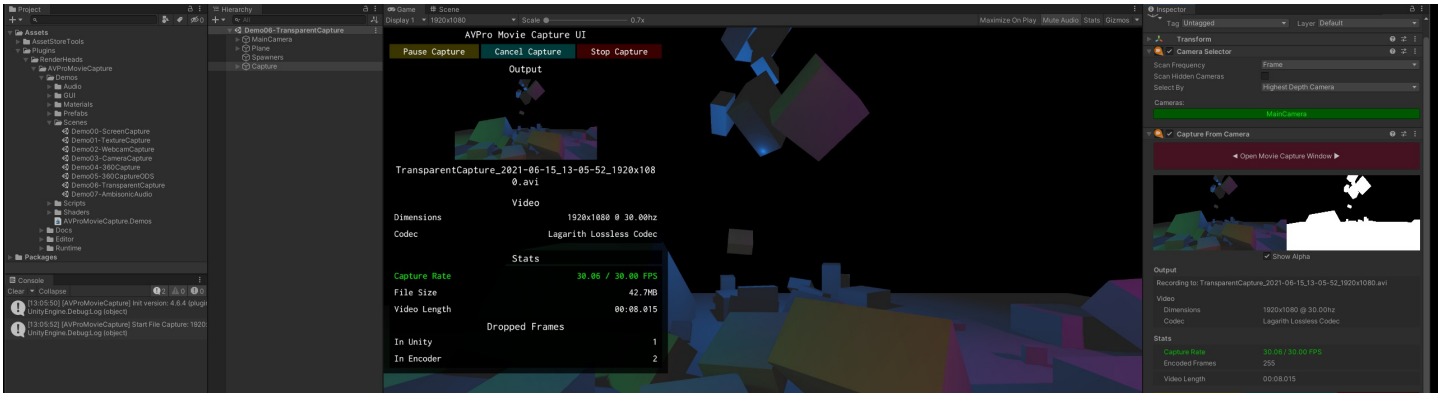
Demo05 - 360 ODS Capture



This scene is set up to capture a Unity `Camera` in accurate stereo to a video file in 360 equirectangular format. It uses the

`CaptureFromCamera360ODS` component. This capture runs in offline render mode so that every frame is captured, and is very slow due to the nature of ODS rendering.

Demo06 - Transparent Capture



This scene is set up to capture a Unity `Camera` to a video file, preserving the transparent/alpha channel data. It uses the `CaptureFromCamera` component. The capture runs in real-time capture mode.

Demo07 - Ambisonic Audio

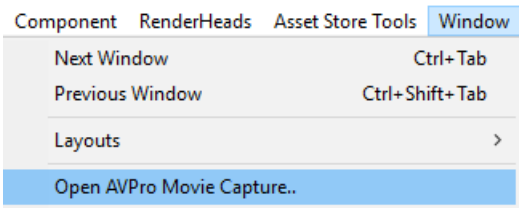
This scene is set up to capture moving `AudioSource` objects into an Ambisonic WAV file for further processing. It also captures a real-time 360 video using the `CaptureFromCamera360` component which can be used to validate that the audio is correct. This video is only low resolution so that it doesn't impact the audio capture.

Quick Start

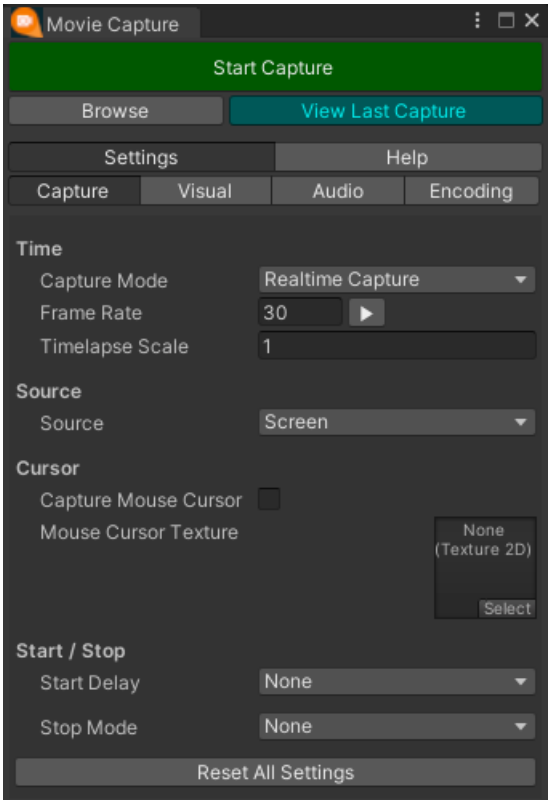
In-Editor Capture

The in-editor capture window is the easiest and fastest way to start making captures.

The capture window allow you to quickly and easily capture videos directly from inside the Unity editor without modifying your scene. Simply open the `AVPro Movie Capture editor window` from the Window menu bar:



You can now this UI panel to your editor layout to allow one-click video captures, or open and close it as needed.



This panel allows you to configure your recording options and codecs.

The first options to set is `"Capture Mode"` which controls whether the capture is real-time, or an offline render.

Next `"Source"` type should be set. This controls where the capture comes from - whether you're capturing the whole screen in general, or from a specific camera, or from a specific camera in 360 degrees.

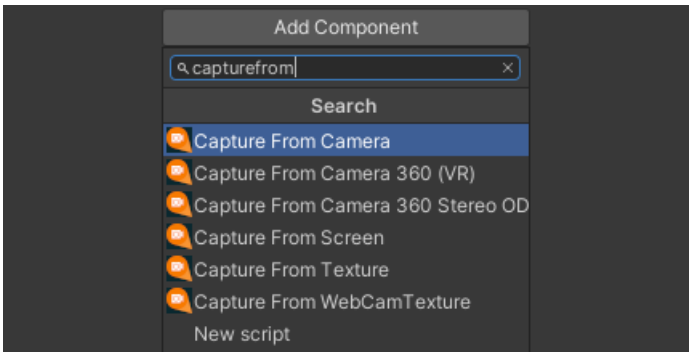
All settings are remembered between sessions so once it has been configured once you only need to press the `“Start Capture”` button. Once a capture starts the statistics of the capture, capture progress and controls are presented.

In-Game Capture

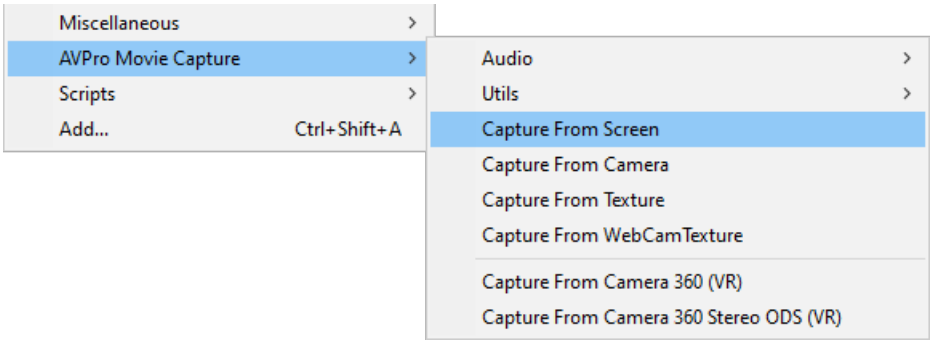
You can add one of the components (`CaptureFromScreen`, `CaptureFromCamera` etc) to your scene and trigger them to record

directly from your game. This works in the editor and standalone builds.

Components can be added via the `"Add Component"` button on the `GameObject`



or via the "Component" menu:



See the component documentation for more details on how to configure these beyond the default settings.

Real-time Capturing

For real-time capturing is it important to have the correct capture settings otherwise the capture frame-rate will not be sustainable, leading to video captures that aren't smooth.

Most of the codecs use hardware (GPU) encoding and should be able to handle 4K at 30 FPS without a problem. Beyond that the a powerful system is required and more care must be taken with the capture settings used.

Offline Rendering

For visuals that don’t require real-time input (cut-scenes, procedural / sequenced animations etc) you have the option to record in offline/non-realtime mode. This mode allows you to capture every frame of animation even if the playback runs very slowly and allows you to capture to any desired frame rate. For example you may have a mega complex/detailed sequence that renders very slowly in Unity. You can use offline recording to render this to a 60fps video. Offline rendering isn’t suitable for anything that requires real-time input. Since this type of rendering can take a while, you can use the `Auto Stop` option to set how many frames / seconds you want it to render for

```
// Create a 10 second offline screen render
CaptureFromScreen capture = go.AddComponent<CaptureFromScreen>();
capture.IsRealTime = false;
capture.FrameRate = 60f;
capture.StopMode = StopMode.FramesEncoded;
capture.StopAfterFramesElapsed = capture.FrameRate * 10f;
capture.StartCapture();
```

Demo Scenes

Read about the included demo scenes as these are also a good quick start reference for typical use-cases.

Capturing the Screen

Screen capture will capture the final rendering to screen, including all UI elements (even IMGUI) and post-processes effects.

> **NOTE**
>
> Currently capture is only possible from one of the displays. When using multiple displays the last one will be captured.

The final screen output can be captured in-editor using the Movie Capture editor window.

The `CaptureFromScreen` component can be used for generate capture as part of your application. Simple add this component to a `GameObject` and set the capture settings needed in the Inspector Window, or via Scripting.

Here is an example of capturing the screen via scripting:

```
using UnityEngine;
using RenderHeads.Media.AVProMovieCapture;

public class ScreenCaptureExample : MonoBehaviour
{
    void Start()
    {
        GameObject go = new GameObject();
        CaptureFromScreen capture = go.AddComponent<CaptureFromScreen>();
        capture.IsRealTime = false;
        capture.FrameRate = 60f;
        capture.StopMode = StopMode.FramesEncoded;
        capture.StopAfterFramesElapsed = capture.FrameRate * 10f;
        capture.StartCapture();
    }

    void Update()
    {
        if (KeyCode.GetKeyDown(KeyCode.S))
        {
            _capture.StartCapture();
        }
        if (KeyCode.GetKeyDown(KeyCode.S))
        {
            _capture.StopCapture();
        }
    }
}
```

Camera Capture

Camera capture will capture only what the specific camera (and it's chained children) are rendering. This can be useful when you only need to render specific elements to your video.

Post-process filters may be captured but it depends on the mechanism used.

> **NOTE**
>
> Camera capture is more expensive than Screen Capture as the Camera will be rendered again specifically for the capture. If the camera scene is complicated this will double the draw call count.

Webcam / Texture Recording

AVPro Movie Capture includes a demo scene that shows how to easily record a Unity webcam to video file. It uses the CaptureFromTexture component as Unity's `WebCamTexture` is just a normal texture. If you want to capture the audio as well, make sure to select the microphone, or leave the audio device index as -1 to use the default microphone.

On iOS, make sure to fill in the Camera Usage Description field in the Other Settings page of the `Player Settings` (NSCameraUsageDescription key in the built project's Info.plist), your app will crash otherwise.

NOTE

To use the `WebCamTexture` capture component/demo, the string `AVPRO_MOVIECAPTURE_WEBCAMTEXTURE_SUPPORT` must be added to `Scriping Define Symbols` in `Player Settings > Other Settings > Script Compilation`. This is because on some platforms (iOS, macOS, Android at least) Unity's build system automatically forces webcam permissions for apps with source code containing the string "WebCamTexture", which can be build issues if this feature is not required.

360 Rendering

360 degree (or 180) stereo or mono captures can be made using the CaptureFromCamera360 or CaptureFromCamera360ODS components. The ODS capture component captures more accurate stereo, however it is extremely slow.

Generally 360 rendering should be done as an off-line render instead of a real-time capture as it is very intensive - rendering is done using cubemaps, so the scene is rendered 6 times.

> **NOTE**
>
> It is not possible to capture screen-space post-process filters when rendering in 360.

When using `CaptureFromCamera360` the `XR 360 Stereo Capture` option in `Player Settings` must be enabled.

> **NOTE**
>
> Using SRP it is not possible to use the `CaptureFromCamera360` component to capture stereo, as SRP doesn't set support XR 360 Stereo Capture

Transparent Capture

AVPro Movie Capture supports encoding the alpha channel as long as a suitable codec is used. See the codecs page for information about which codecs support transparency.

To encode transparency:

1. Use a `CaptureFromCamera` component and make sure “Support Alpha” is ticked. Or if you use the Editor Window select “Camera” as the source type and make sure “Support Alpha” is ticked on the Visual tab.
2. Set the background colour of your capture camera and set the alpha value to zero.
3. Set the capture camera clear flags to “Solid Color”
4. If you’re using the `CaptureFromCamera360ODS` component then make sure to set the Camera Clear Color alpha to zero in the capture component settings.

NOTE

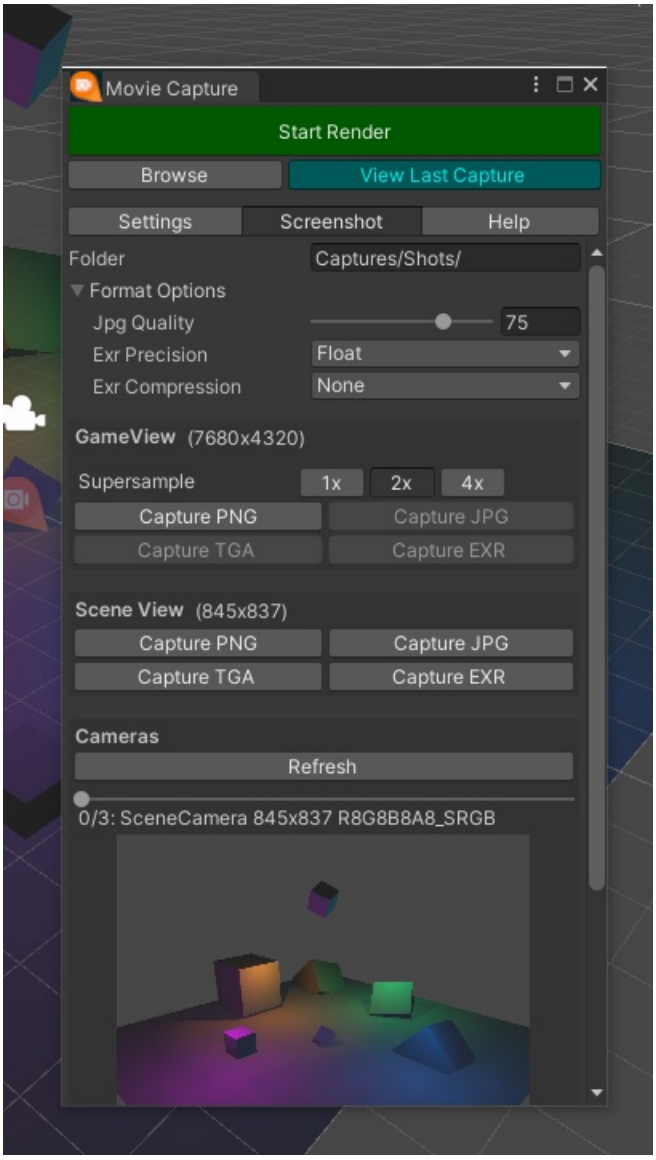
Unity by default is not a compositing system, so the alpha channels it generates are generally only for alpha blending usage during rendering and are not designed to be correct for exporting. You may need to create or tweak shaders to ensure that the correct alpha values are being rendered

NOTE

Be aware that many post-process filters do not preserve the alpha channel

Screenshot

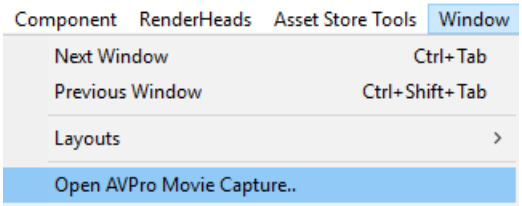
The Capture Window has a new panel for quickly and easily capturing screnshots.



Screenshots can be taken from the Game View, Scene View, any Camera or RenderTexture.

Pressing the "View Last Capture" or "Browse" buttons will take you to the last screenshot taken.

Access the AVPro Movie Capture Window from the menu bar:



Capture to Named Pipe

This is an experimental feature that allows the captured video to be written to a named pipe. This allows other software to listen for the incoming video frames.

Using FFMPEG you can connect to the named pipe for custom encoding:

```
ffmpeg.exe -y -f rawvideo -codec rawvideo -s 640x480 -r 30 -pix_fmt rgb32 -i \\.\pipe\test_pipe -an -c:v libx264 -pix_fmt yuv420p output.mp4
```

You must start the capture and THEN run ffmpeg, otherwise it will not find the named pipe.

Currently it only video outputs in rgb32 format and you need to specify the resolution and frame rate. Audio is not supported.

Another example using FFMPEG showing how to stream a video via RTP:

```
ffmpeg -r 30 -vcodec rawvideo -f rawvideo -pix_fmt yuv420p -s 1280x720 -i \\.\pipe\test_pipe -an -f rtp rtp://127.0.0.1:9090
```

NOTE

When you start the capture to the named pipe, Unity will freeze until that pipe is read from (eg by starting FFMPEG).

Audio Capture

The audio capture source can be:

- From a microphone/recording device
- From Unity
- From Unity Audio Mixer
- System-wide
- Manually via scripting
- From Wwise

Microphone

To record from a microphone the index of the microphone needs to be set (0 for the default device). Certain platforms (macOS, iOS, Android) require that permission has been granted in order to capture from the microphone (or other device). See here for more information.

Unity

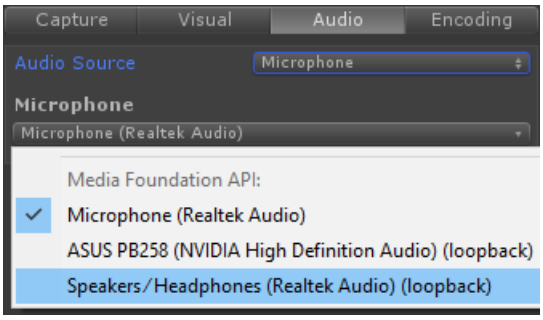
The plugin supports capturing audio directly from Unity. See the CaptureAudioFromAudioListener and CaptureAudioFromAudioRenderer components.

Unity

The plugin supports capturing audio directly from Unity Audio Mixer. See the CaptureAudioFromAudioMixer component.

System-wide

On Windows 10 you can capture all of the audio that is being played to a particular sound device, regardless of which application produces the audio. This is useful as it also allows audio to be captured via Unity components that don't play audio through Unity's audio pipeline (eg FMOD, Wwise, AVPro Video etc). This is done through via WASAPI loopback device support which is not supported by all audio devices.



In the AVPro Movie Capture Window the loopback device will appear in the list of microphones with `(loopback)` appended to the name. This means that you'll be able to record all audio sent to that device from any application. If there are no loopback devices listed then your audio driver doesn't support this feature.

To select a loopback device via the `CaptureFrom` components, you need to set the `Audio Source` to `Microphone` and then set the correct device index. The device index can be assigned via code by looking for devices with the `(loopback)` string at the end of the name. Note that there can be multiple of these devices.

Wwise

See CaptureAudioFromWwise component.

Ambisonic Audio Capture

The plugin supports encoding 3D positional audio in a Unity scene to the ambisonic format which is suitable for encoding with 360 videos. See the Ambisonic Demo.

Photo Library

You can capture videos directly to the Photo Library on macOS and iOS.

Requirements

macOS

macOS 10.15 and later is required.

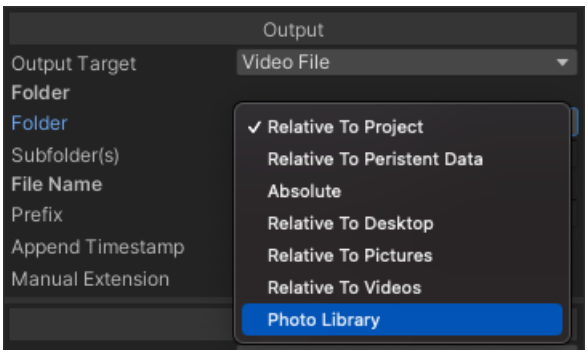
> **NOTE**
>
> Photo Library support is not available in the Unity editor on macOS, because the required usage description keys are not included in the Unity Editor applications Info.plist file.

iOS

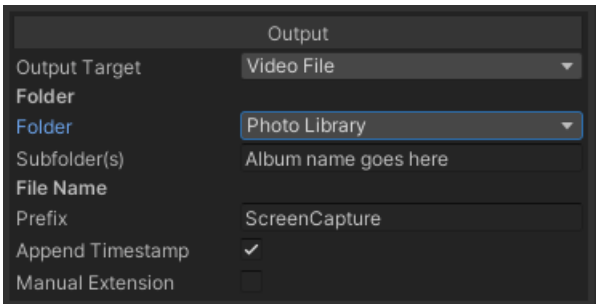
iOS 11.0 and later is required.

Usage

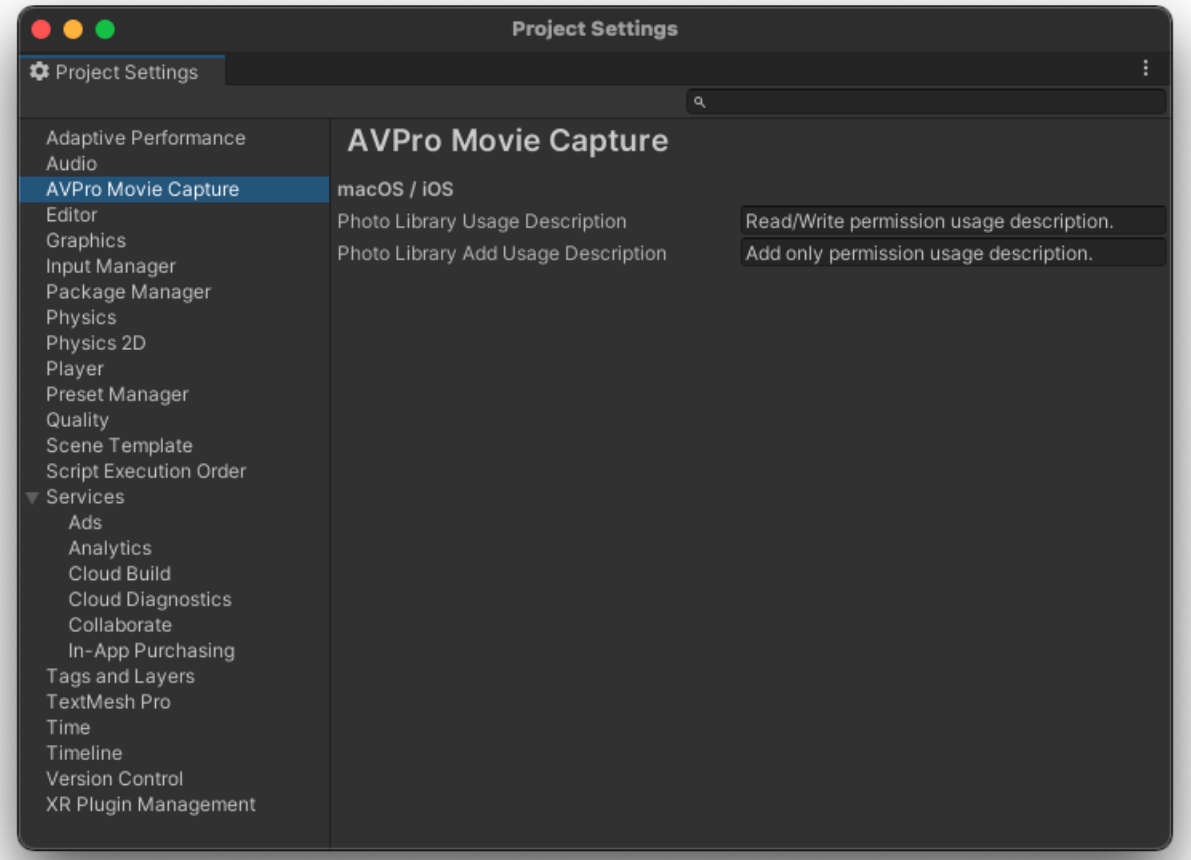
Select `Photo Library` from the `Folder` dropdown menu in the `Output` section of the capture component inspector as shown.



If you want your captures to be added to an album, put the desired album name in the `Subfolder(s)` text field. Adding captures to an an album will required full read/write access to the photo library.



You will need to request permission from the user in order to be able to access the photo library. To do this fill in the relevant usage description in the AVPro Movie Capture section of the Project Settings inspector as shown below. These fields should be added to your generated application or Xcode project as part of the build process.



You can then request authorisation using the following code snippet where `capture` is the capture component instance:

```
IEnumerator Start()
{
#if (UNITY_STANDALONE_OSX || UNITY_IOS) && !UNITY_EDITOR
    CaptureBase.PhotoLibraryAccessLevel photoLibraryAccessLevel = CaptureBase.PhotoLibraryAccessLevel.AddOnly;

    // If we're trying to write to the photo library, make sure we have permission
    if (capture.OutputFolder == CaptureBase.OutputPath.PhotoLibrary)
    {
        // Album creation (album name is taken from the output folder path) requires read write access.
        if (capture.OutputFolderPath != null && capture.OutputFolderPath.Length > 0)
            photoLibraryAccessLevel = CaptureBase.PhotoLibraryAccessLevel.ReadWrite;

        switch (CaptureBase.HasUserAuthorisationToAccessPhotos(photoLibraryAccessLevel))
        {
            case CaptureBase.PhotoLibraryAuthorisationStatus.Authorised:
                // All good, nothing to do
                break;

            case CaptureBase.PhotoLibraryAuthorisationStatus.Unavailable:
                Debug.LogWarning("The photo library is unavailable, will use RelativeToPeristentData instead");
                capture.OutputFolder = CaptureBase.OutputPath.RelativeToPeristentData;
                break;

            case CaptureBase.PhotoLibraryAuthorisationStatus.Denied:
                // User has denied access, change output path
                Debug.LogWarning("User has denied access to the photo library, will use RelativeToPeristentData instead");
                capture.OutputFolder = CaptureBase.OutputPath.RelativeToPeristentData;
                break;

            case CaptureBase.PhotoLibraryAuthorisationStatus.NotDetermined:
                // Need to ask permission
                yield return CaptureBase.RequestUserAuthorisationToAccessPhotos(photoLibraryAccessLevel);
                // Nested switch, everbodies favourite
                switch (CaptureBase.HasUserAuthorisationToAccessPhotos(photoLibraryAccessLevel))
                {
                    case CaptureBase.PhotoLibraryAuthorisationStatus.Authorised:
                        // All good, nothing to do
                        break;

                    case CaptureBase.PhotoLibraryAuthorisationStatus.Denied:
                        // User has denied access, change output path
                        Debug.LogWarning("User has denied access to the photo library, will use RelativeToPeristentData instead");
                        capture.OutputFolder = CaptureBase.OutputPath.RelativeToPeristentData;
                        break;

                    case CaptureBase.PhotoLibraryAuthorisationStatus.NotDetermined:
                        // We were unable to request access for some reason, check the logs for any error information
                        Debug.LogWarning("Authorisation to access the photo library is still undetermined, will use RelativeToPeristentData
instead");
                        capture.OutputFolder = CaptureBase.OutputPath.RelativeToPeristentData;
                        break;
                }
                break;
        }
    }
#endif
    yield return null;
}
```

Waiting for a Capture to Complete

When a capture is stopped, there will still be remaining frames to process and some file post-processing operations to perform. The file is not usable until these operations have completed. These operations happen in the background so they do not block the main thread. They usually only take a second or two to complete, but it can take longer if the file is very large and post-process operations are enabled (eg fast-start, stereo injection).

The `FileWritingHandler` class can be used to know when a captured video file is ready for use.

There are two events that can be used:

- `BeginFinalFileWritingAction`
- `CompletedFileWritingAction`

If you only want to know when the file is ready then the `OnCompleteFinalFileWriting` can be subscribed to:

```
using UnityEngine;
using RenderHeads.Media.AVProMovieCapture;

public class FileCompleteExample : MonoBehaviour
{
    [SerializeField] CaptureBase _capture = null;

    void Start()
    {
        _capture.CompletedFileWritingAction += OnCompleteFinalFileWriting;
    }

    void OnDestroy()
    {
        _capture.CompletedFileWritingAction -= OnCompleteFinalFileWriting;
    }

    void OnCompleteFinalFileWriting(FileWritingHandler handler)
    {
        Debug.Log("The file is at: " + handler.Path);
    }
}
```

If you want to track the progress from the beginning of file writing then use `BeginFinalFileWritingAction`, which can then be polled for the ready status, or could be used in conjunction with the `CompletedFileWritingAction`:

```
using UnityEngine;
using RenderHeads.Media.AVProMovieCapture;

public class FileWriteHandlerExample : MonoBehaviour
{
    [SerializeField] CaptureBase _capture = null;
    private FileWritingHandler _fileWritingHandler;

    void Start()
    {
        _capture.BeginFinalFileWritingAction += OnBeginFinalFileWriting;
    }

    void OnDestroy()
    {
        _capture.BeginFinalFileWritingAction -= OnBeginFinalFileWriting;
        if (_fileWritingHandlers != null)
        {
            _fileWritingHandlers.Dispose();
        }
    }

    void OnBeginFinalFileWriting(FileWritingHandler handler)
    {
        _fileWritingHandlers = handler;
    }

    void Update()
    {
        if (_fileWritingHandlers != null)
        {
            // Poll for the file writing to complete
            if (_fileWritingHandlers.IsFileReady())
            {
                Debug.Log("File is ready at: " + _fileWritingHandlers.Path);
                _fileWritingHandlers = null;
            }
        }
    }
}
```

Retreiving captured file path

If you use the `FileWriteHandler` then you can retrieve the capture file path by using the `Path` property as shown in the two examples above.

Otherwise, the `CaptureFrom` component has a `LastFilePath` that will return the full path to the last file written:

```
using UnityEngine;
using RenderHeads.Media.AVProMovieCapture;

public class GetLastPathExample : MonoBehaviour
{
    [SerializeField] CaptureBase _capture = null;

    void Update()
    {
        Debug.Log("The last captured file is at: " + _capture.LastFilePath);
    }
}
```

Scripting

Before are some useful scripting examples:

Codec Scripting

```
// Iterate over found video codecs
foreach (Codec codec in CodecManager.VideoCodecs)
{
    Debug.Log(codec.Name + " - " codec.Index);
}

// Searching for a specific codec
Codec h264Codec = CodecManager.FindCodec(CodecType.Video, "H264");
Codec aacCodec = CodecManager.FindCodec(CodecType.Audio, "AAC");
if (h264Codec != null && aacCodec != null)
{
    Debug.Log("Codecs found");
}

// Assign a specific codec for capture
capture.NativeForceVideoCodecIndex = h264Codec.Index;
capture.NativeForceAudioCodecIndex = aacCodec.Index;

// Set a list of acceptable codecs to use, instead of specifying one directly
capture.NativeForceVideoCodecIndex = -1;
capture.VideoCodecPriorityWindows = new string[] { "H264", "HEVC" };
```

Audio Input Device Scripting

```
// Iterate over found audio input devices
foreach (Device device in DeviceManager.AudioInputDevices)
{
    Debug.Log(device.Name + " - " codec.Index);
}

// Search for a specific device
if (CodecManager.AudioInputDevices.Count > 0)
{
    Device firstDevice = CodecManager.AudioInputDevices.Devices[0];
    if (firstDevice != null)
    {
        Debug.Log("Device found");
    }
}

// Assign a device for audio capture
capture.AudioCaptureSource = AudioCaptureSource.Microphone;
capture.ForceAudioInputDeviceIndex = firstDevice.Index;
```

Capture From Screen

```
using UnityEngine;
using RenderHeads.Media.AVProMovieCapture;

public class ScreenCaptureExample : MonoBehaviour
{
    void Start()
    {
        GameObject go = new GameObject();
        CaptureFromScreen capture = go.AddComponent<CaptureFromScreen>();
        capture.IsRealTime = false;
        capture.FrameRate = 60f;
        capture.StopMode = StopMode.FramesEncoded;
        capture.StopAfterFramesElapsed = capture.FrameRate * 10f;
        capture.NativeForceVideoCodecIndex = -1;
        capture.VideoCodecPriorityWindows = new string[] { "H264", "HEVC" };
        capture.StartCapture();
    }

    void Update()
    {
        if (KeyCode.GetKeyDown(KeyCode.S))
        {
            _capture.StartCapture();
        }
        if (KeyCode.GetKeyDown(KeyCode.S))
        {
            _capture.StopCapture();
        }
    }
}
```

Codecs

AVPro Movie Capture doesn't include any built-in codecs and instead uses codecs that are already included / registered with the operating system. Most operating systems have several built-in codecs and optionally allow 3rd-party codecs to be installed.

macOS / iOS Codecs

AVPro Movie Capture running on macOS and iOS support the following codecs:

CODEC	CONTAINERS	IOS SUPPORT	TRANSPARENCY	NOTES
Video Codecs				
H.264	mp4/m4v/mov	□	.	
HEVC (H.265)	mp4/m4v/mov	□	□	Please ensure your mac has support for hardware HEVC encoding. Results using the software encoder will be... varied. HEVC with alpha is also supported for capturing transparency, and requires the QuickTime file container (.mov) to work
JPEG	m4v/mov	□	.	
ProRes 422	mov	.	.	
ProRes 4444	mov	.	□	
Uncompressed	mov	.	.	If you like HUGE media files then this is the codec for you. It's unlikely to be useful. Use the png image sequence capture mode if you want uncompressed video frames. Uncompressed frames are raw RGB
Audio Codecs				
AAC	mp4/m4v/mov	□	.	
FLAC	mov	□	.	The free lossless audio codec
Apple Lossless	m4v/mov	□	.	
Linear PCM	mov	□	.	16-bit linear PCM interleaved samples at the source sample rate (usually 48000Hz)
Uncompressed	mov	□	.	The raw audio from the audio source, likely to be linear pcm encoded. For instance with Unity this will be 32bit floating point linear pcm at the current audio sample rate (usually 48000Hz)
Image Codecs				
PNG	png	□	□	
JPEG	jpeg	□	.	

CODEC	CONTAINERS	IOS SUPPORT	TRANSPARENCY	NOTES
TIFF	tiff	□	□	
HEIF	heif	□	□	Requires macOS 10.13.4 or later

Containers

MP4 Container

The MPEG-4 container format. This is the default for the H264 and HEVC video codecs and the AAC audio codec.

M4V Container

Apple's proprietary video container based on mp4. Supports the H264, HEVC, Apple ProRes 422 and Apple ProRes 4444 video codecs, and AAC and Apple Lossless audio codecs.

QuickTime Container

The Apple Quicktime movie format. Recognised file extensions are: mov and qt. This is the default container for the MJPEG, AppleProRes 422, AppleProRes 4444, uncompressed video codecs and for the FLAC, LinearPCM, uncompressed audio codecs.

Android Codecs

AVPro Movie Capture running on Android support the following codecs:

CODEC	CONTAINERS	NOTES
Video Codecs		
H.264	mp4	
HEVC (H.265)	mp4	
Audio Codecs		
AAC	mp4/m4v/mov	
Image Codecs		
PNG	png	
JPEG	jpeg	

Containers

MP4 Container

The MPEG-4 container format. This is the default for the H264 and HEVC video codecs and the AAC audio codec.

Windows Desktop Codecs

AVPro Movie Capture supports both the built-in Windows codecs, and any 3rd-party codecs installed on the system. Codecs support varies based on the API:

VIDEO API	CONTAINERS	LEGACY	HARDWARE ENCODING	3RD-PARTY CODECS	MIN OS SUPPORT
Media Foundation	MP4	.	□	.	Windows 7
DirectShow	AVI/MP4	□	.	□	Windows XP
VFW (Video for Windows)	AVI	□	.	□	Windows XP

Media Foundation Codecs

Using Media Foundation (MF) codecs is the most modern option and the most typical use case. Only a fixed subset of codecs are supported:

VIDEO API	HARDWARE ENCODING	MIN OS SUPPORT	SPECS
Video Codecs			
H.264	□	Windows 7	Codec Spec
HEVC (H.265)	□	Windows 10	Codec Spec
Audio Codecs			
AAC	.	Windows 7	Codec Spec
FLAC	.	Windows 10	.
Image Codecs			
Uncompressed PNG	.	Windows XP	.

H.264 Codec

This is the most common and compatible codec by far and should be used in most cases. You can create easy to share videos with this codec.

HEVC (H.265) Codec

This is a very common and compatible codec with higher compression than H.264 but requires more modern hardware and operating system support.

AAC / FLAC Audio Codec

AAC is the most common audio codec and is widely compatible and should be used in most cases. FLAC is useful if you require lossless audio encoding but isn't as widely compatible.

DirectShow & VFW Legacy Codecs

These legacy codecs can be used for cases where the Media Foundation codecs are not suitable for the use case. Windows does include a few built-in legacy codecs but most of these are not useful. 3rd-party codecs can be installed, here are a few we recommend:

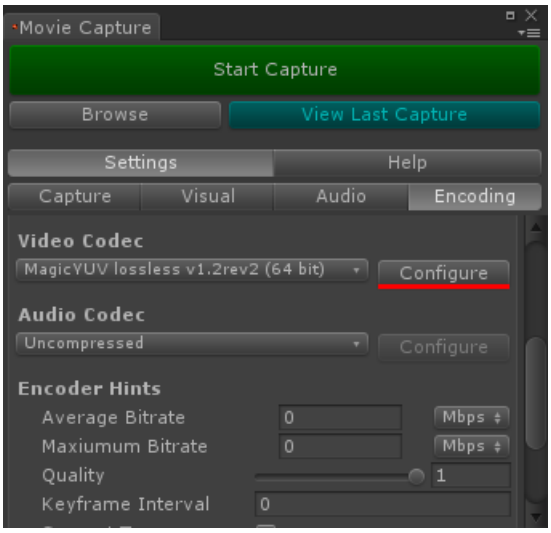
CODEC	REALTIME	LOSSLESS	FILE SIZE	ALPHA CHANNEL	LINK
Lagarith	□	□	Large	□*	Link
MagicYUV	□	□	Large	□*	Link
x264	□*	□*	Small-Medium	.	Link

*after configuring the codec

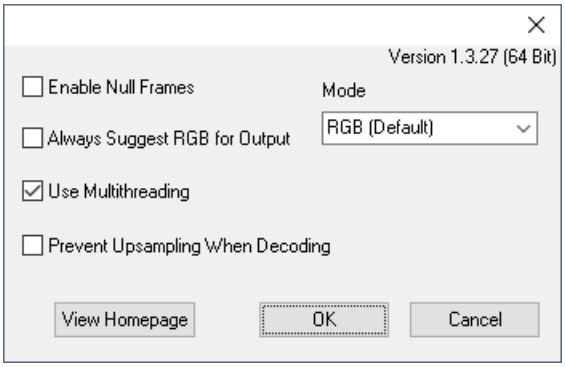
Lagarith Codec

Lagarith is a great general purpose codec. It uses the AVI file container. It's fast enough for real-time (due to great multi-threading) and lossless. Naturally the files it generates are very large and not suitable for sharing. We use Lagarith as an intermediate codec for real-time capturing and then re-encode to something else like H.264 MP4 offline.

Always check your codec settings. You can do this directly in the plugin via the Configure button:



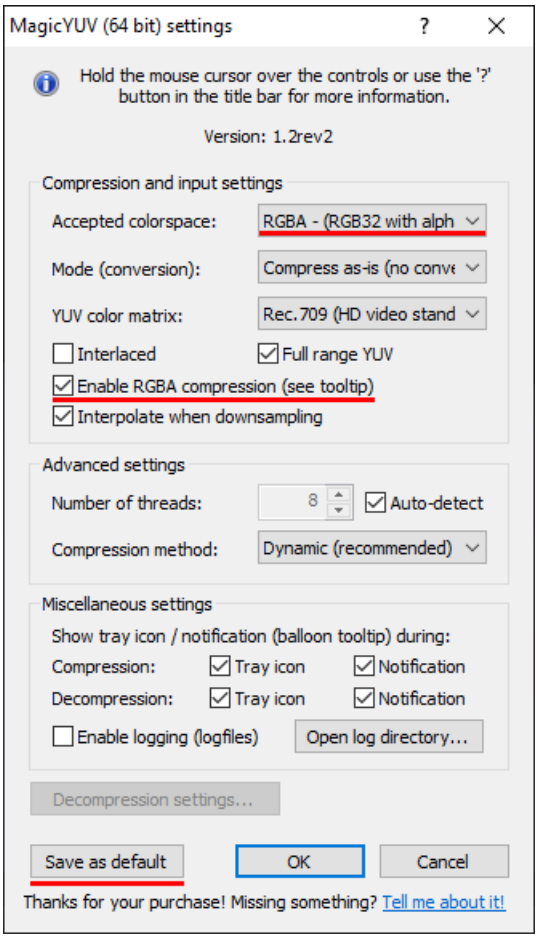
Recommended settings:



1. Enable "Use Multithreading"
2. Set Mode to RGB or RGBA if you need to capture alpha channel

MagicYUV Codec

MagicYUV is another fast lossless codec for realtime capture. It uses the AVI file container. It can capture transparent videos but first needs to be configured to do so:



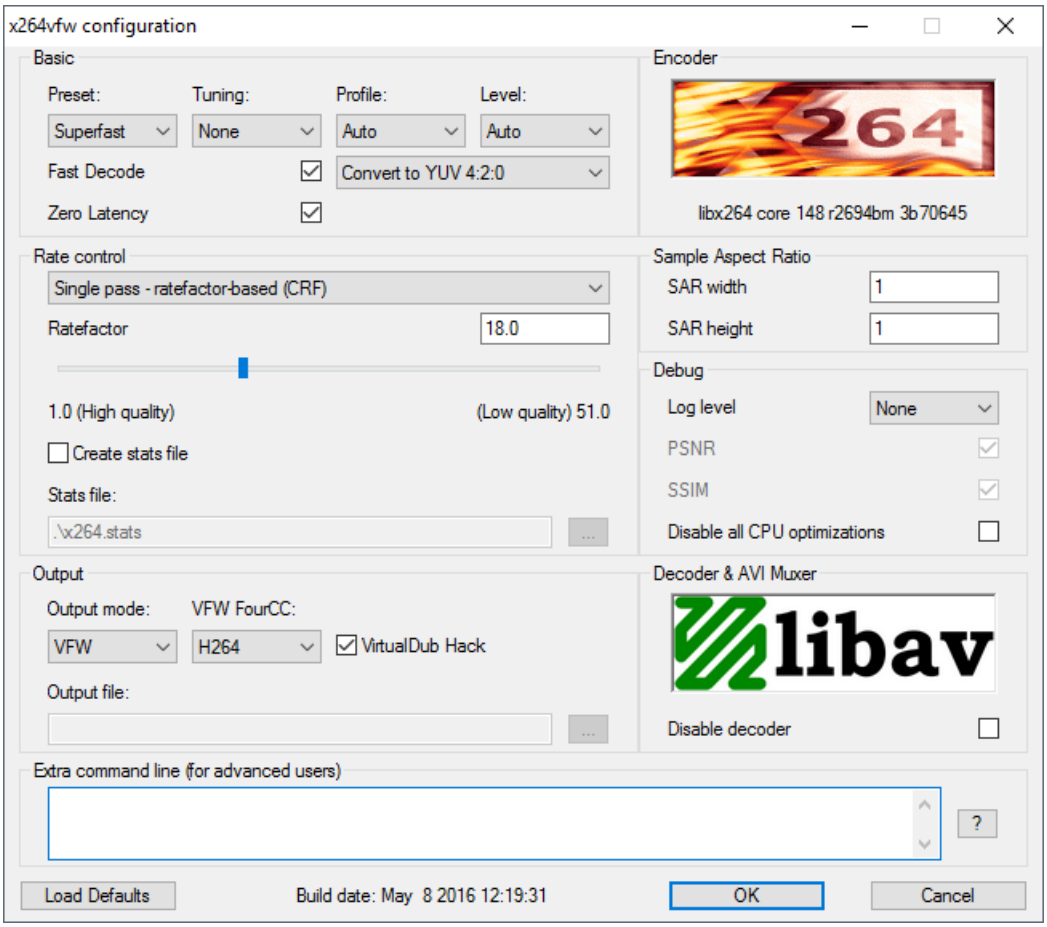
1. Open the codec configuration panel by going to the Start Menu and running "MagicYUV VFW codec configuration"
2. Set "Accepted Colorspace" to "RGBA (RGB32 with alpha)"
3. Tick "Enable RGBA compression"
4. Click "Save defaults"

x264 Codec

x264 is a highly tunable codec and suits almost any need. By default it's set up for off-line processing which produces tiny files but generally uses way too much CPU for real-time capture. We tweak x264 for real-time capture and use it to directly generate video files suitable for sharing. x264 can also be used with an MP4 muxer to generate MP4 files instead of AVI files (see FAQ).

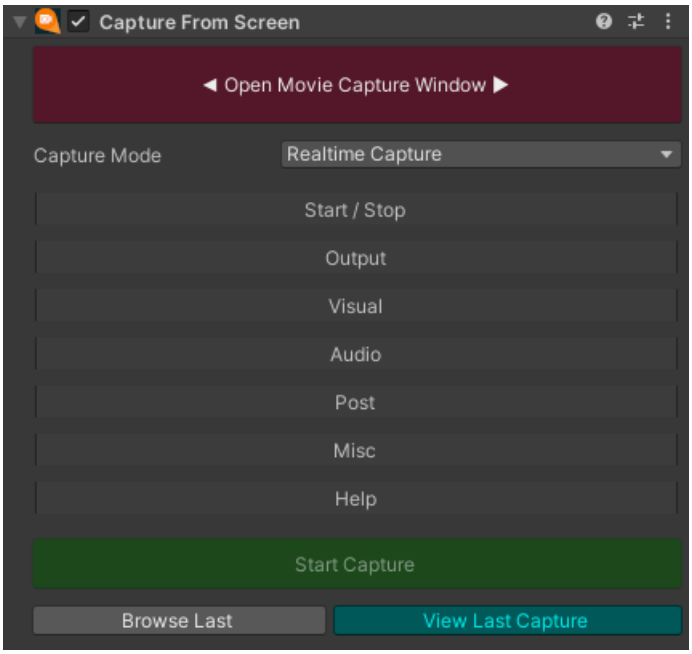
Always check your codec settings. You can do this directly in the plugin via the Configure button, or in 3rd party software like Virtualdub (video > compression menu). Recommended settings:

- Preset: Fast/Veryfast/Superfast
- Enable "Fast Decode"
- Set "Ratefactor" to your desired quality level
- Enable "VirtualDub Hack"
- Set Debug "Log level" to None
- Checking 'Zero Latency' can help with AV sync and also improve cross-platform playback (for some reason).

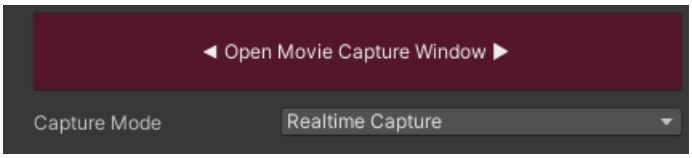


Note: You may have to run this configuration twice - once for the 32-bit x264 config, and again for the 64-bit x264 config. These settings are separate for 32bit and 64bit so if you're running 64-bit Unity it will use the 64-bit settings, but then if you build a 32-bit standalone app it will use the 32-bit settings - so it is often best to just build the same "bitness" so you're using in Unity editor to avoid such confusion.

Capture From Screen



This is the preferred component for capturing as it simply captures everything on the screen (backbuffer), including UI and IMGUI. The hardware mouse cursor is the only thing not captured, but this component has an option to capture the mouse cursor by drawing it using IMGUI.

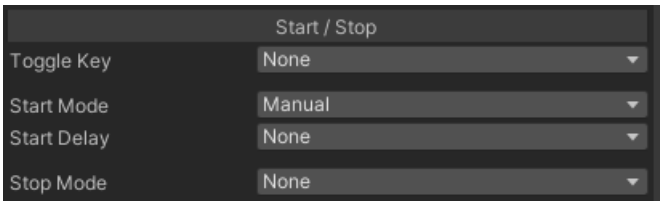


- The "Open Movie Capture Window" button opens the in-editor capture window which allows capturing without adding components to your scene

Properties

PROPERTY	FUNCTION
Capture Mode	Capture can either be realtime, or an offline render

Start/Stop

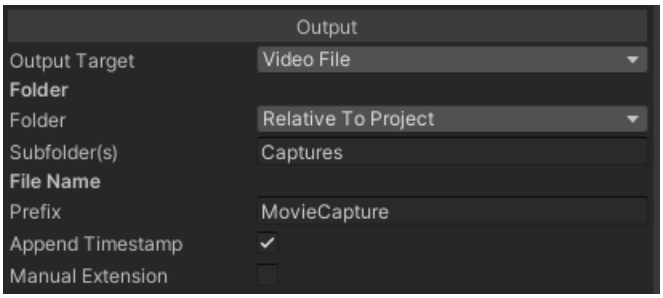


Properties

PROPERTY	FUNCTION
Toggle Key	Select a key to toggle the start and end of capturing
Start Mode	The capture can start either when the component starts, or wait until it is manually triggered by the user via scripting
Start Delay	An optional delay can be specified before the frame capturing actually starts

PROPERTY	FUNCTION
Stop Mode	Without a stop mode the capture will continue forever until it is stopped by the user or script. You can set a stop mode to make the capture stop when it reaches either a certain number of frames, or a duration in seconds

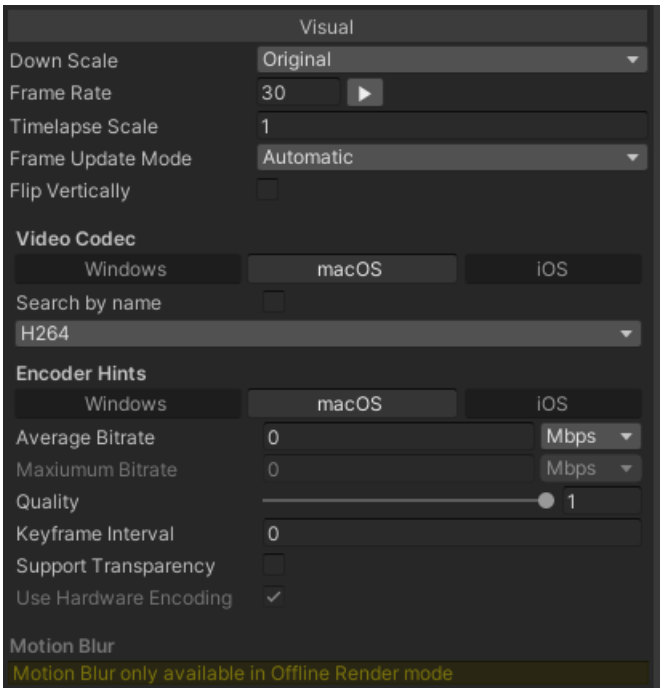
Output



Properties

PROPERTY	FUNCTION
Output Target	Select if you want to output to a video file, image sequence, or a named pipe
Folder	
Folder	Select the relative folder to output to
Subfolder(s)	Select a subfolder to output to
File Name	
Prefix	The start of the file name
Append Timestamp	Whether to append an auto-generated timestamp to the file name for videos
Manual Extension	Whether to manually specify an extension to the file name for videos
Image Sequence	
Format	The format of the image sequence (per-platform)
Start Frame	The number to start the frame count with for the file name
Zero Digits	The number to digits to use for the frame count in the file name
Named Pipe	
Pipe Path	The path of the pipe to write to (eg `\\.\pipe\test_pipe`)

Visual

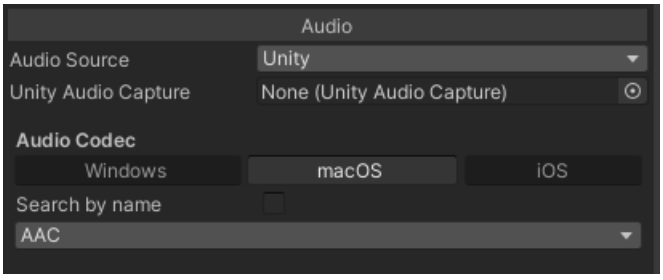


Properties

PROPERTY	FUNCTION
Down Scale	Reduce the resolution of the final output
Frame Rate	The frame rate per second for output file
Timelapse Scale	For real-time captures a scale can be set to allow for timelapse captures
Frame Update Mode	Allows the frame update to be controlled automatically, or manually via scripting. This can be useful when capturing an element that doesn't necessarily update each Unity frame, eg a webcam or custom texture rendering
Flip Vertically	Flip the output image vertically (debug only)
Video Codecs	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don’t have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Video Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index
Encoder Hints	
Average Bitrate	Ideal average bitrate of output video
Maximum Bitrate	Ideal maximum bitrate of output video

PROPERTY	FUNCTION
Quality	0..1 range to specify quality preference over other factors (eg encoding speed)
Keyframe Interval	The distance between keyframes
Support Transparency	Whether or not transparency should be supported if possible
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Enable Fragmented Writing	Enables fragmented writing of QuickTime (mov, mp4, etc.) files. Enabling this feature allows you to open and play a partially written asset should something unrecoverable occur during the writing process. Adjust `Movie Fragment Interval` to control the fragment size
Movie Fragment Interval	The interval in seconds at which movie fragments should be written, the default is 120 seconds. Making this value too small may result in captures failing
Motion Blur	
Use Motion Blur	Enables an experimental option which is available for offline rendering. It accumulates frames in between the capture frames to create motion blur. This is very useful for high quality rendering but it is very expensive as it is a brute force approach to motion blur
Samples	The number of sub-frames to render which get accumulated into a single output frame
Cameras	

Audio

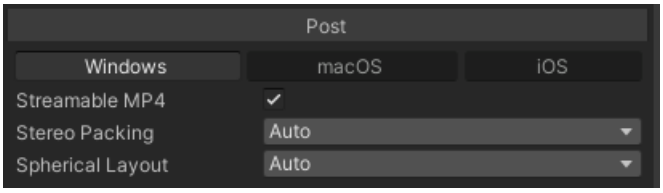


Properties

PROPERTY	FUNCTION
Audio Source	Specified where to capture audio from, this can be set to None, Unity, Microphone, Manual, or Unity Audio Mixer
Unity Audio Source	

PROPERTY	FUNCTION
Unity Audio Capture	The component to use for capturing audio from Unity. This can be CaptureAudioFromAudioListener, CaptureAudioFromAudioRenderer or CaptureAudioFromWwise
Microphone Audio Source	
Force Audio Device Index	The index of the microphone/recording device to use. Default is zero
Manual Audio Source	
Sample Rate	The sample rate for manual audio encoding
Channels	The number of audio channels for audio audio encoding
Audio Codec	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don't have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Audio Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index

Post

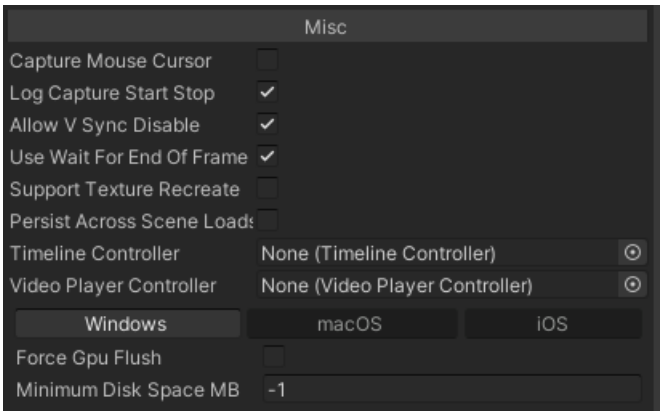


After the video file has been created, an optional post-process can be run to make addition changes to the file.

Properties

PROPERTY	FUNCTION
Streamable MP4	Makes the MP4/MOV file suitable for streaming by using 'fast start' encoding method
Stereo Packing	Injects data to specify the stereo format of the video
Spherical Layout	Injects data to specify the spherical format of the video

Misc

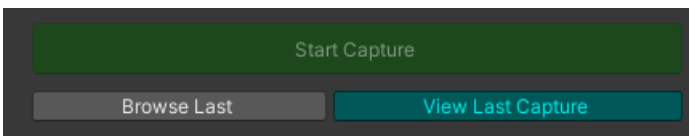


Properties

PROPERTY	FUNCTION
Capture Mouse Cursor	This option is only available when using the CaptureFromScreen component and allows the mouse cursor to be captured. You have to specify a texture to use for the cursor
Log Capture Start Stop	Log each time the capture is started or stopped. Disable this for less garbage generation
Allow V-Sync Disable	For off-line capturing, allow vsync to be disabled which allows captures to run as fast as possible
Use Wait For End Of Frame	Makes sure capture happens right at the end of the frame - this allows some features such as skinning to be resolved correctly
Support Texture Recreation	
Persist Across Scene Loads	The GameObject will not be destroyed when the scene is unloaded.
Timeline Controller	Optional TimelineController component when capturing Timelines using offline rendering
VideoPlayer Controller	Optional component when capturing VideoPlayers using offline rendering (not currently working)
Windows Only	
Force GPU Flush	Flushing the GPU during each capture results in less latency, but can slow down rendering performance for complex scenes.
Minimum Disk Space MB	Set to -1 to ignore, otherwise it will keep checking the disk space and stop the capture if the free disk space gets below this number of MB

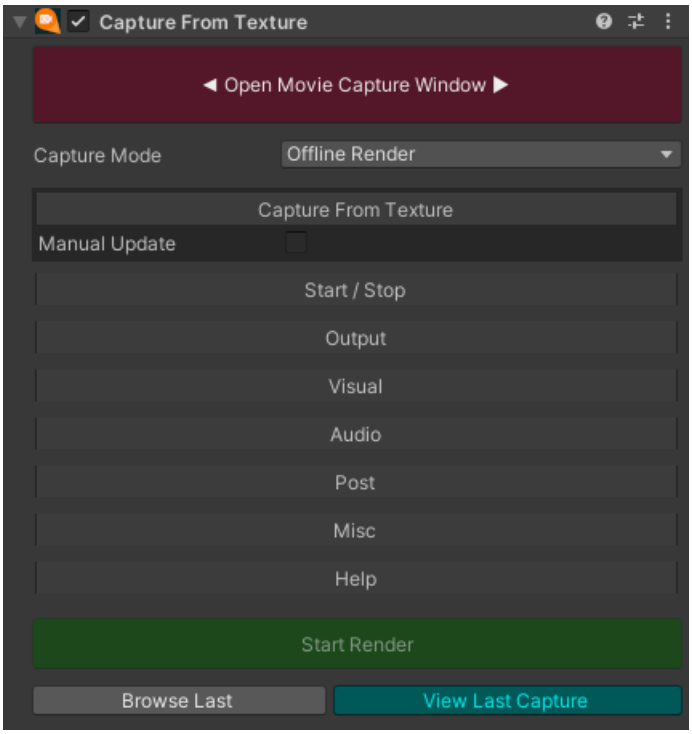
Help

Shows the plugin version number and has links to the website, issue tracker, documentation etc



- The "Start Capture" button will begin capturing. The scene needs to be running.
- The "Browse Last" button will show a Finder/Explorer window with the last capture selected.
- The "View Last Capture" button will play back the last captured video using the system media player.

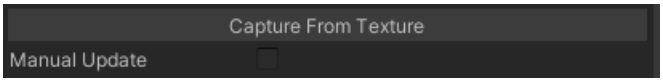
Capture From Texture



This component captures from any Unity texture (`Texture2D`, `RenderTexture`, `WebCamTexture` etc).

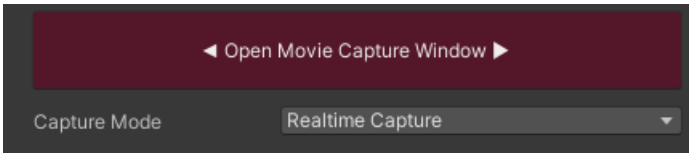
The texture to capture is assigned via scripting by calling the `SetSourceTexture()` method.

From Texture



Properties

PROPERTY	FUNCTION
Manual Update	Specified whether the component will try to capture the texture each Unity frame, or to wait for a scripted signal that the texture has updated. For example a webcam texture has it's own update rate, so for best results it should be signalled when this texture has updated via `UpdateSourceTexture()`.

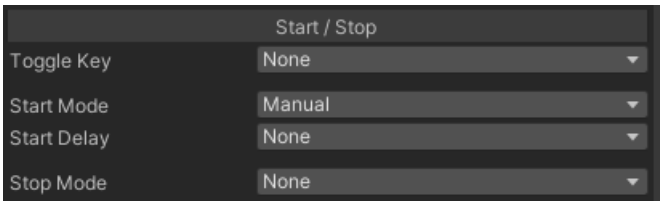


- The "Open Movie Capture Window" button opens the in-editor capture window which allows capturing without adding components to your scene

Properties

PROPERTY	FUNCTION
Capture Mode	Capture can either be realtime, or an offline render

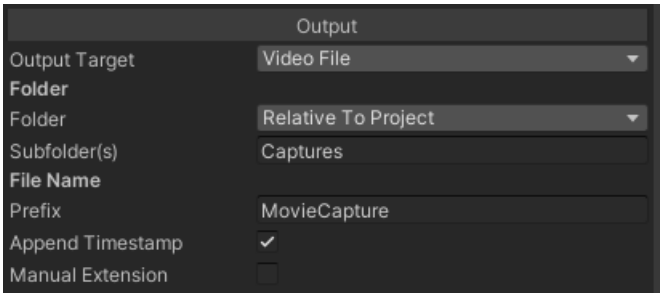
Start/Stop



Properties

PROPERTY	FUNCTION
Toggle Key	Select a key to toggle the start and end of capturing
Start Mode	The capture can start either when the component starts, or wait until it is manually triggered by the user via scripting
Start Delay	An optional delay can be specified before the frame capturing actually starts
Stop Mode	Without a stop mode the capture will continue forever until it is stopped by the user or script. You can set a stop mode to make the capture stop when it reaches either a certain number of frames, or a duration in seconds

Output

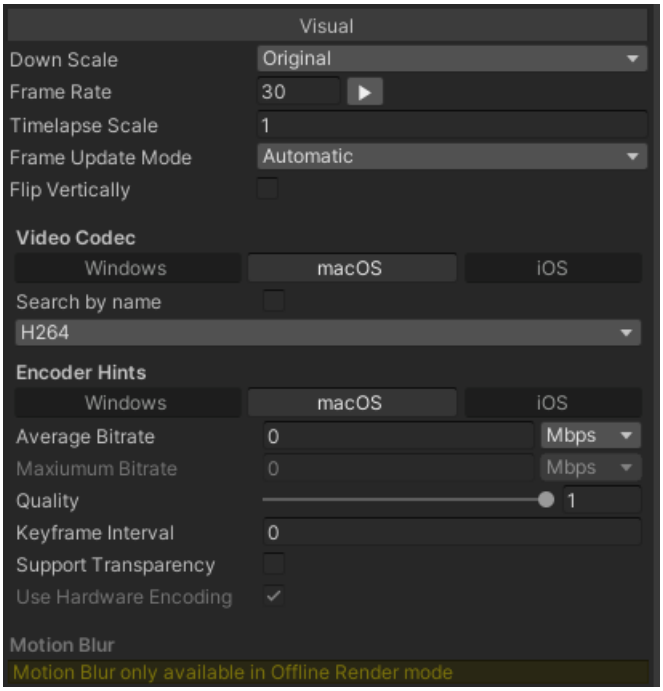


Properties

PROPERTY	FUNCTION
Output Target	Select if you want to output to a video file, image sequence, or a named pipe
Folder	
Folder	Select the relative folder to output to
Subfolder(s)	Select a subfolder to output to
File Name	
Prefix	The start of the file name
Append Timestamp	Whether to append an auto-generated timestamp to the file name for videos
Manual Extension	Whether to manually specify an extension to the file name for videos
Image Sequence	
Format	The format of the image sequence (per-platform)

PROPERTY	FUNCTION
Start Frame	The number to start the frame count with for the file name
Zero Digits	The number to digits to use for the frame count in the file name
Named Pipe	
Pipe Path	The path of the pipe to write to (eg `\\.\pipe\test_pipe`)

Visual



Properties

PROPERTY	FUNCTION
Down Scale	Reduce the resolution of the final output
Frame Rate	The frame rate per second for output file
Timelapse Scale	For real-time captures a scale can be set to allow for timelapse captures
Frame Update Mode	Allows the frame update to be controlled automatically, or manually via scripting. This can be useful when capturing an element that doesn't necessarily update each Unity frame, eg a webcam or custom texture rendering
Flip Vertically	Flip the output image vertically (debug only)
Video Codecs	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.

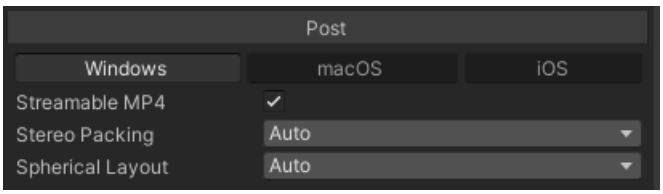
PROPERTY	FUNCTION
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don't have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Video Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index
Encoder Hints	
Average Bitrate	Ideal average bitrate of output video
Maximum Bitrate	Ideal maximum bitrate of output video
Quality	0..1 range to specify quality preference over other factors (eg encoding speed)
Keyframe Interval	The distance between keyframes
Support Transparency	Whether or not transparency should be supported if possible
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Enable Fragmented Writing	Enables fragmented writing of QuickTime (mov, mp4, etc.) files. Enabling this feature allows you to open and play a partially written asset should something unrecoverable occur during the writing process. Adjust `Movie Fragment Interval` to control the fragment size
Movie Fragment Interval	The interval in seconds at which movie fragments should be written, the default is 120 seconds. Making this value too small may result in captures failing
Motion Blur	
Use Motion Blur	Enables an experimental option which is available for offline rendering. It accumulates frames in between the capture frames to create motion blur. This is very useful for high quality rendering but it is very expensive as it is a brute force approach to motion blur
Samples	The number of sub-frames to render which get accumulated into a single output frame
Cameras	

Audio

Properties

PROPERTY	FUNCTION
Audio Source	Specified where to capture audio from, this can be set to None, Unity, Microphone, Manual, or Unity Audio Mixer
Unity Audio Source	
Unity Audio Capture	The component to use for capturing audio from Unity. This can be CaptureAudioFromAudioListener, CaptureAudioFromAudioRenderer or CaptureAudioFromWwise
Microphone Audio Source	
Force Audio Device Index	The index of the microphone/recording device to use. Default is zero
Manual Audio Source	
Sample Rate	The sample rate for manual audio encoding
Channels	The number of audio channels for audio audio encoding
Audio Codec	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don't have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Audio Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index

Post

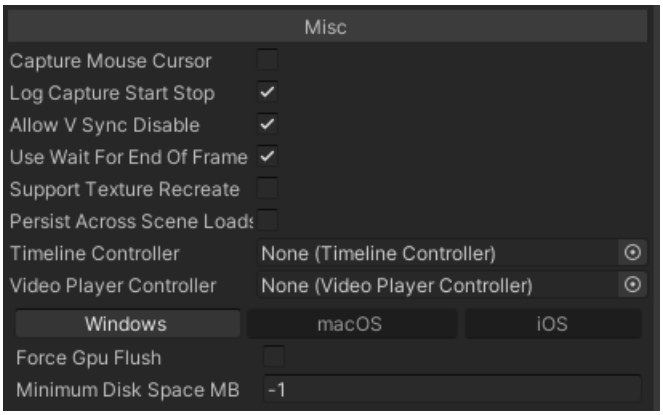


After the video file has been created, an optional post-process can be run to make addition changes to the file.

Properties

PROPERTY	FUNCTION
Streamable MP4	Makes the MP4/MOV file suitable for streaming by using 'fast start' encoding method
Stereo Packing	Injects data to specify the stereo format of the video
Spherical Layout	Injects data to specify the spherical format of the video

Misc

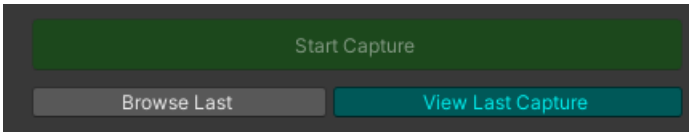


Properties

PROPERTY	FUNCTION
Capture Mouse Cursor	This option is only available when using the CaptureFromScreen component and allows the mouse cursor to be captured. You have to specify a texture to use for the cursor
Log Capture Start Stop	Log each time the capture is started or stopped. Disable this for less garbage generation
Allow V-Sync Disable	For off-line capturing, allow vsync to be disabled which allows captures to run as fast as possible
Use Wait For End Of Frame	Makes sure capture happens right at the end of the frame - this allows some features such as skinning to be resolved correctly
Support Texture Recreation	
Persist Across Scene Loads	The GameObject will not be destroyed when the scene is unloaded.
Timeline Controller	Optional TimelineController component when capturing Timelines using offline rendering
VideoPlayer Controller	Optional component when capturing VideoPlayers using offline rendering (not currently working)
Windows Only	
Force GPU Flush	Flushing the GPU during each capture results in less latency, but can slow down rendering performance for complex scenes.
Minimum Disk Space MB	Set to -1 to ignore, otherwise it will keep checking the disk space and stop the capture if the free disk space gets below this number of MB

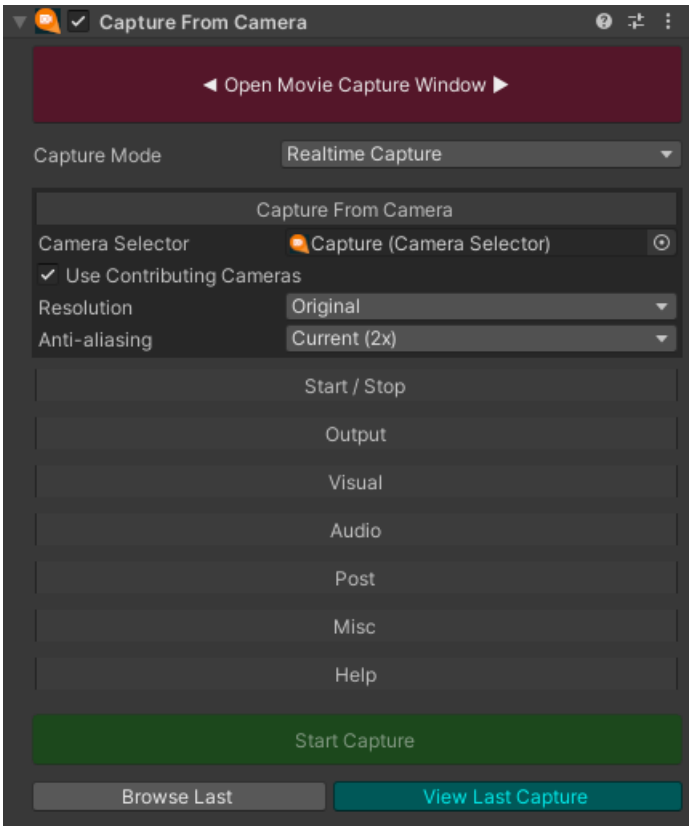
Help

Shows the plugin version number and has links to the website, issue tracker, documentation etc



- The "Start Capture" button will begin capturing. The scene needs to be running.
- The "Browse Last" button will show a Finder/Explorer window with the last capture selected.
- The "View Last Capture" button will play back the last captured video using the system media player.

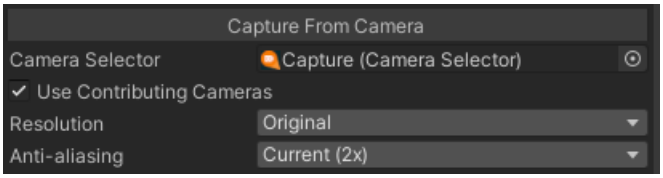
Capture From Camera



This component captures the rendering of a Unity camera (including post effects and uGUI). IMGUI is not captured (for this you have to use the CaptureFromScreen component). You can capture from a single camera, or from multiple cameras if you have several cameras in your scene that contribute to the final camera rendering.

This component allows you to render at a higher resolution than the app is running at, including 8K output (if the codec supports this resolution).

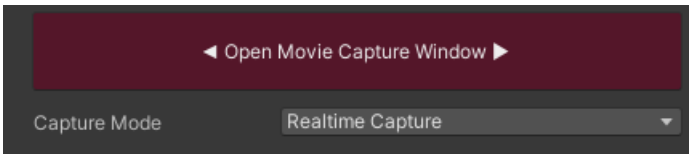
From Camera



Properties

PROPERTY	FUNCTION
Camera Selector	Selects the camera to render from. Using the CameraSelector component allows easy switching between cameras.
Camera	Selects the camera to render from if only one camera is required.
Use Contributing Cameras	Whether to work out which others cameras are also contributing to the rendering (camera chaining) and render those as well
Resolution	The resolution of the final output (before downscale)
Anti-aliasing	Anti-aliasing to use during rendering

PROPERTY	FUNCTION
Include Scene View Gizmos	When capturing from the Scene View camera, whether to render the Gizmos

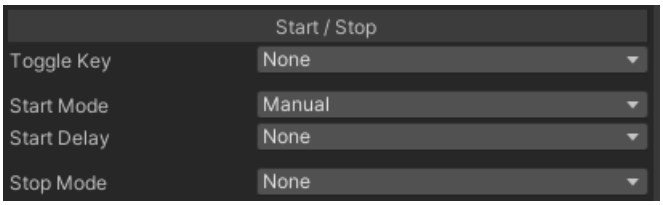


- The "Open Movie Capture Window" button opens the in-editor capture window which allows capturing without adding components to your scene

Properties

PROPERTY	FUNCTION
Capture Mode	Capture can either be realtime, or an offline render

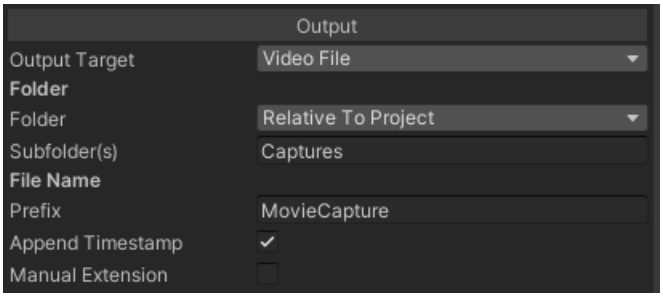
Start/Stop



Properties

PROPERTY	FUNCTION
Toggle Key	Select a key to toggle the start and end of capturing
Start Mode	The capture can start either when the component starts, or wait until it is manually triggered by the user via scripting
Start Delay	An optional delay can be specified before the frame capturing actually starts
Stop Mode	Without a stop mode the capture will continue forever until it is stopped by the user or script. You can set a stop mode to make the capture stop when it reaches either a certain number of frames, or a duration in seconds

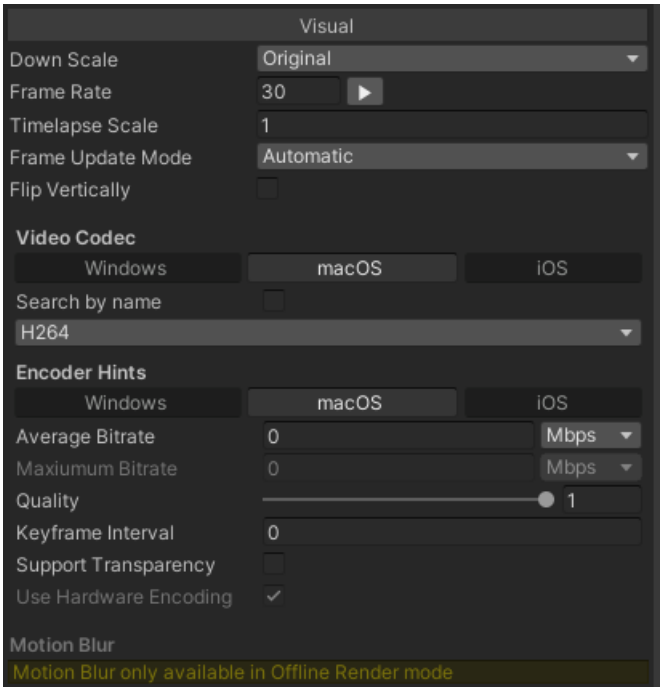
Output



Properties

PROPERTY	FUNCTION
Output Target	Select if you want to output to a video file, image sequence, or a named pipe
Folder	
Folder	Select the relative folder to output to
Subfolder(s)	Select a subfolder to output to
File Name	
Prefix	The start of the file name
Append Timestamp	Whether to append an auto-generated timestamp to the file name for videos
Manual Extension	Whether to manually specify an extension to the file name for videos
Image Sequence	
Format	The format of the image sequence (per-platform)
Start Frame	The number to start the frame count with for the file name
Zero Digits	The number to digits to use for the frame count in the file name
Named Pipe	
Pipe Path	The path of the pipe to write to (eg `\\.\pipe\test_pipe`)

Visual



Properties

PROPERTY	FUNCTION
Down Scale	Reduce the resolution of the final output
Frame Rate	The frame rate per second for output file
Timelapse Scale	For real-time captures a scale can be set to allow for timelapse captures
Frame Update Mode	Allows the frame update to be controlled automatically, or manually via scripting. This can be useful when capturing an element that doesn't necessarily update each Unity frame, eg a webcam or custom texture rendering
Flip Vertically	Flip the output image vertically (debug only)
Video Codecs	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don't have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Video Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index
Encoder Hints	
Average Bitrate	Ideal average bitrate of output video
Maximum Bitrate	Ideal maximum bitrate of output video
Quality	0..1 range to specify quality preference over other factors (eg encoding speed)
Keyframe Interval	The distance between keyframes
Support Transparency	Whether or not transparency should be supported if possible
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Enable Fragmented Writing	Enables fragmented writing of QuickTime (mov, mp4, etc.) files. Enabling this feature allows you to open and play a partially written asset should something unrecoverable occur during the writing process. Adjust `Movie Fragment Interval` to control the fragment size

PROPERTY	FUNCTION
Movie Fragment Interval	The interval in seconds at which movie fragments should be written, the default is 120 seconds. Making this value too small may result in captures failing
Motion Blur	
Use Motion Blur	Enables an experimental option which is available for offline rendering. It accumulates frames in between the capture frames to create motion blur. This is very useful for high quality rendering but it is very expensive as it is a brute force approach to motion blur
Samples	The number of sub-frames to render which get accumulated into a single output frame
Cameras	

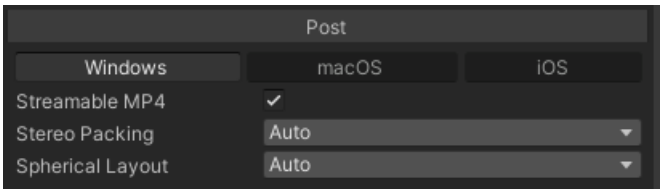
Audio

Properties

PROPERTY	FUNCTION
Audio Source	Specified where to capture audio from, this can be set to None, Unity, Microphone, Manual, or Unity Audio Mixer
Unity Audio Source	
Unity Audio Capture	The component to use for capturing audio from Unity. This can be CaptureAudioFromAudioListener, CaptureAudioFromAudioRenderer or CaptureAudioFromWwise
Microphone Audio Source	
Force Audio Device Index	The index of the microphone/recording device to use. Default is zero
Manual Audio Source	
Sample Rate	The sample rate for manual audio encoding
Channels	The number of audio channels for audio audio encoding
Audio Codec	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.

PROPERTY	FUNCTION
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don't have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Audio Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index

Post

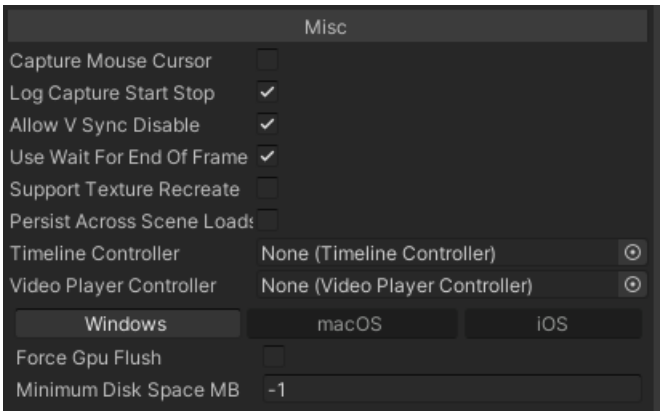


After the video file has been created, an optional post-process can be run to make addition changes to the file.

Properties

PROPERTY	FUNCTION
Streamable MP4	Makes the MP4/MOV file suitable for streaming by using 'fast start' encoding method
Stereo Packing	Injects data to specify the stereo format of the video
Spherical Layout	Injects data to specify the spherical format of the video

Misc



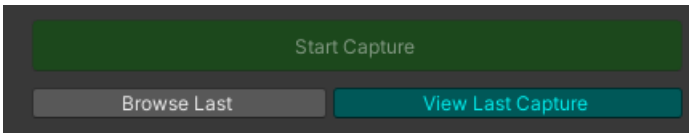
Properties

PROPERTY	FUNCTION
Capture Mouse Cursor	This option is only available when using the CaptureFromScreen component and allows the mouse cursor to be captured. You have to specify a texture to use for the cursor
Log Capture Start Stop	Log each time the capture is started or stopped. Disable this for less garbage generation
Allow V-Sync Disable	For off-line capturing, allow vsync to be disabled which allows captures to run as fast as possible
Use Wait For End Of Frame	Makes sure capture happens right at the end of the frame - this allows some features such as skinning to be resolved correctly

PROPERTY	FUNCTION
Support Texture Recreation	
Persist Across Scene Loads	The GameObject will not be destroyed when the scene is unloaded.
Timeline Controller	Optional TimelineController component when capturing Timelines using offline rendering
VideoPlayer Controller	Optional component when capturing VideoPlayers using offline rendering (not currently working)
Windows Only	
Force GPU Flush	Flushing the GPU during each capture results in less latency, but can slow down rendering performance for complex scenes.
Minimum Disk Space MB	Set to -1 to ignore, otherwise it will keep checking the disk space and stop the capture if the free disk space gets below this number of MB

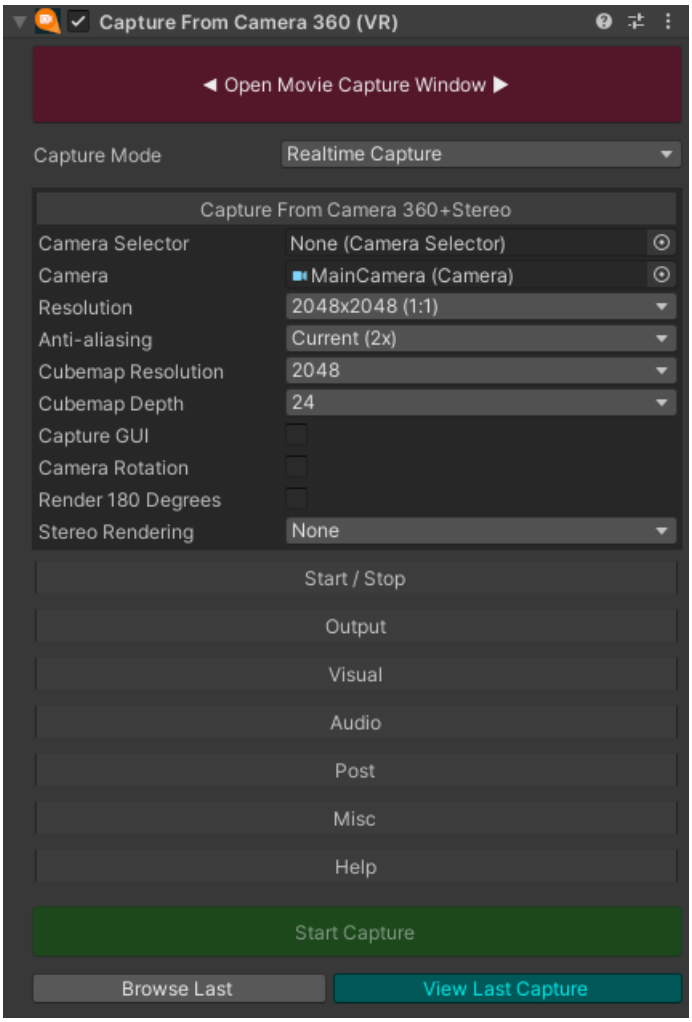
Help

Shows the plugin version number and has links to the website, issue tracker, documentation etc



- The "Start Capture" button will begin capturing. The scene needs to be running.
- The "Browse Last" button will show a Finder/Explorer window with the last capture selected.
- The "View Last Capture" button will play back the last captured video using the system media player.

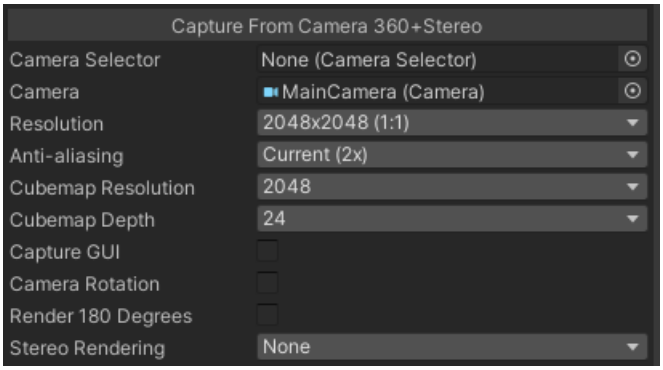
Capture From Camera 360



This component allows you to capture full 360 degree views of your scene in equi-rectangular format suitable for viewing in VR.

This component works by rendering to 6 faces of a cubemap (so 6 renders of the scene per frame) and then resolving the cubemap into the final equi-rectangular image.

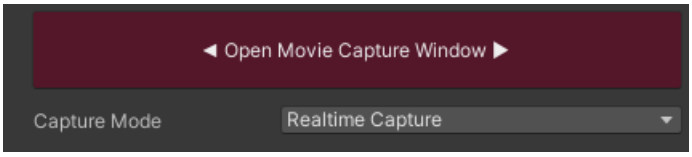
From Camera 360



Properties

PROPERTY	FUNCTION
Camera Selector	Selects the camera to render from. Using the CameraSelector component allows easy switching between cameras.
Camera	Selects the camera to render from if only one camera is required.

PROPERTY	FUNCTION
Resolution	The resolution of the final output (before downscale)
Anti-aliasing	Anti-aliasing to use during rendering
Cubemap Resolution	The resolution of the cubemap faces used during rendering
Cubemap Depth	The bit-depth of the cubemap depth buffer used during rendering
Capture GUI	Whether or not this cubemap renderer needs to capture GUI. Capturing UI requires a slightly slower rendering path
Camera Rotation	Whether or not this capture needs to support camera rotation. Doing so requires a slightly slower rendering path, but it is by default enabled
Render 180 Degrees	Selecting this option will output only 180 degree rendering
Stereo Rendering	Optionally you can select to create a stereo output in either top-bottom or left-right packing formats. The left eye is always rendered first (top / left). Note that this method of stereo rendering is just a fast approximation. For higher quality stereo the ODS component should be used, but it is much much slower
Interpupillary Distance (IPD)	The distance between the pupils. This is set to 0.064 which is the standard for an american male adult. You may wish to alter this if creating content for a wide range of users and quality is important. This value is in meters and assumes a scale of 1 unit = 1 meter

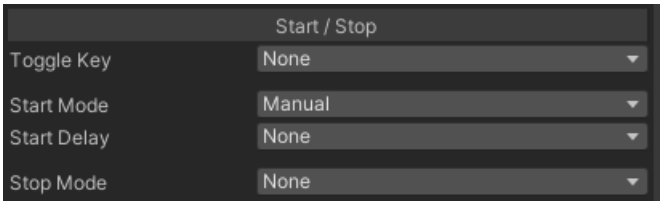


- The "Open Movie Capture Window" button opens the in-editor capture window which allows capturing without adding components to your scene

Properties

PROPERTY	FUNCTION
Capture Mode	Capture can either be realtime, or an offline render

Start/Stop

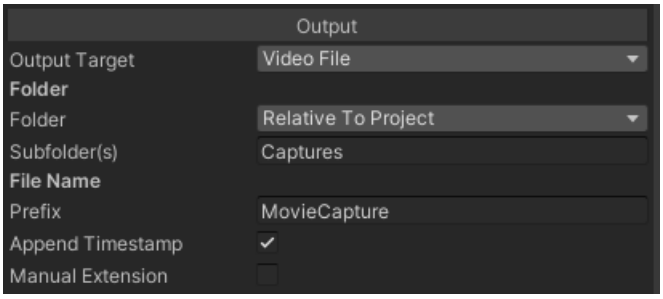


Properties

PROPERTY	FUNCTION
Toggle Key	Select a key to toggle the start and end of capturing
Start Mode	The capture can start either when the component starts, or wait until it is manually triggered by the user via scripting

PROPERTY	FUNCTION
Start Delay	An optional delay can be specified before the frame capturing actually starts
Stop Mode	Without a stop mode the capture will continue forever until it is stopped by the user or script. You can set a stop mode to make the capture stop when it reaches either a certain number of frames, or a duration in seconds

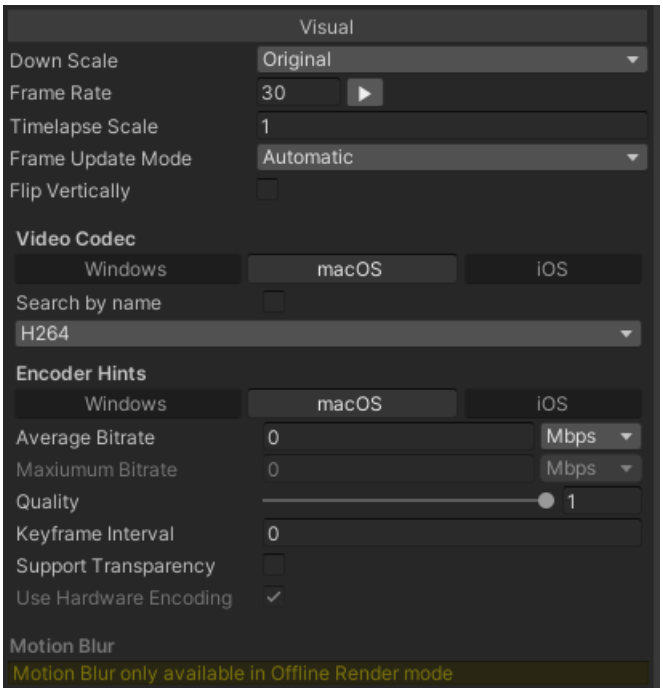
Output



Properties

PROPERTY	FUNCTION
Output Target	Select if you want to output to a video file, image sequence, or a named pipe
Folder	
Folder	Select the relative folder to output to
Subfolder(s)	Select a subfolder to output to
File Name	
Prefix	The start of the file name
Append Timestamp	Whether to append an auto-generated timestamp to the file name for videos
Manual Extension	Whether to manually specify an extension to the file name for videos
Image Sequence	
Format	The format of the image sequence (per-platform)
Start Frame	The number to start the frame count with for the file name
Zero Digits	The number to digits to use for the frame count in the file name
Named Pipe	
Pipe Path	The path of the pipe to write to (eg `\\.\pipe\test_pipe`)

Visual

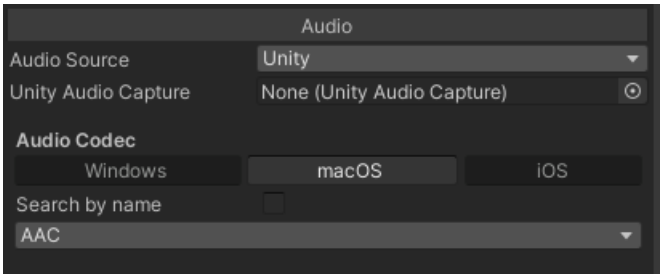


Properties

PROPERTY	FUNCTION
Down Scale	Reduce the resolution of the final output
Frame Rate	The frame rate per second for output file
Timelapse Scale	For real-time captures a scale can be set to allow for timelapse captures
Frame Update Mode	Allows the frame update to be controlled automatically, or manually via scripting. This can be useful when capturing an element that doesn't necessarily update each Unity frame, eg a webcam or custom texture rendering
Flip Vertically	Flip the output image vertically (debug only)
Video Codecs	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don’t have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Video Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index
Encoder Hints	
Average Bitrate	Ideal average bitrate of output video
Maximum Bitrate	Ideal maximum bitrate of output video

PROPERTY	FUNCTION
Quality	0..1 range to specify quality preference over other factors (eg encoding speed)
Keyframe Interval	The distance between keyframes
Support Transparency	Whether or not transparency should be supported if possible
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Enable Fragmented Writing	Enables fragmented writing of QuickTime (mov, mp4, etc.) files. Enabling this feature allows you to open and play a partially written asset should something unrecoverable occur during the writing process. Adjust `Movie Fragment Interval` to control the fragment size
Movie Fragment Interval	The interval in seconds at which movie fragments should be written, the default is 120 seconds. Making this value too small may result in captures failing
Motion Blur	
Use Motion Blur	Enables an experimental option which is available for offline rendering. It accumulates frames in between the capture frames to create motion blur. This is very useful for high quality rendering but it is very expensive as it is a brute force approach to motion blur
Samples	The number of sub-frames to render which get accumulated into a single output frame
Cameras	

Audio

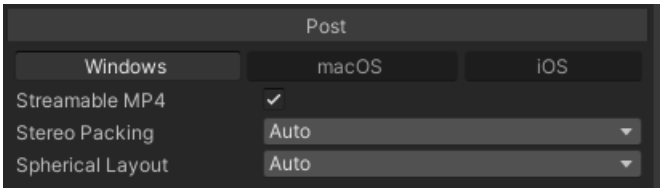


Properties

PROPERTY	FUNCTION
Audio Source	Specified where to capture audio from, this can be set to None, Unity, Microphone, Manual, or Unity Audio Mixer
Unity Audio Source	

PROPERTY	FUNCTION
Unity Audio Capture	The component to use for capturing audio from Unity. This can be CaptureAudioFromAudioListener, CaptureAudioFromAudioRenderer or CaptureAudioFromWwise
Microphone Audio Source	
Force Audio Device Index	The index of the microphone/recording device to use. Default is zero
Manual Audio Source	
Sample Rate	The sample rate for manual audio encoding
Channels	The number of audio channels for audio audio encoding
Audio Codec	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don't have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Audio Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index

Post

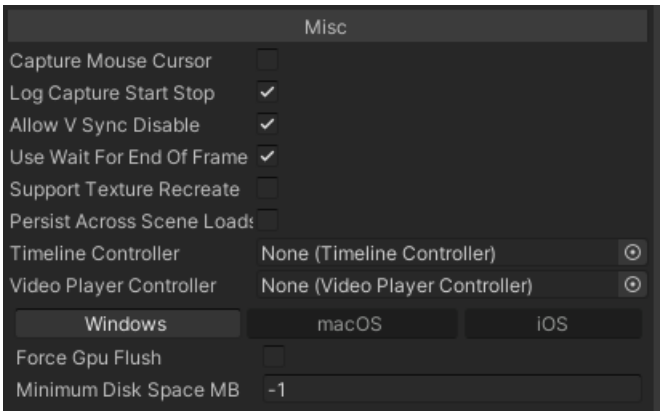


After the video file has been created, an optional post-process can be run to make addition changes to the file.

Properties

PROPERTY	FUNCTION
Streamable MP4	Makes the MP4/MOV file suitable for streaming by using 'fast start' encoding method
Stereo Packing	Injects data to specify the stereo format of the video
Spherical Layout	Injects data to specify the spherical format of the video

Misc

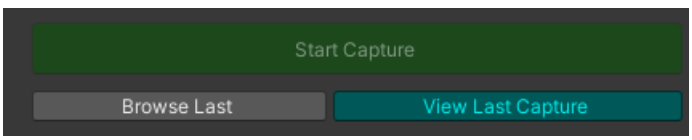


Properties

PROPERTY	FUNCTION
Capture Mouse Cursor	This option is only available when using the CaptureFromScreen component and allows the mouse cursor to be captured. You have to specify a texture to use for the cursor
Log Capture Start Stop	Log each time the capture is started or stopped. Disable this for less garbage generation
Allow V-Sync Disable	For off-line capturing, allow vsync to be disabled which allows captures to run as fast as possible
Use Wait For End Of Frame	Makes sure capture happens right at the end of the frame - this allows some features such as skinning to be resolved correctly
Support Texture Recreation	
Persist Across Scene Loads	The GameObject will not be destroyed when the scene is unloaded.
Timeline Controller	Optional TimelineController component when capturing Timelines using offline rendering
VideoPlayer Controller	Optional component when capturing VideoPlayers using offline rendering (not currently working)
Windows Only	
Force GPU Flush	Flushing the GPU during each capture results in less latency, but can slow down rendering performance for complex scenes.
Minimum Disk Space MB	Set to -1 to ignore, otherwise it will keep checking the disk space and stop the capture if the free disk space gets below this number of MB

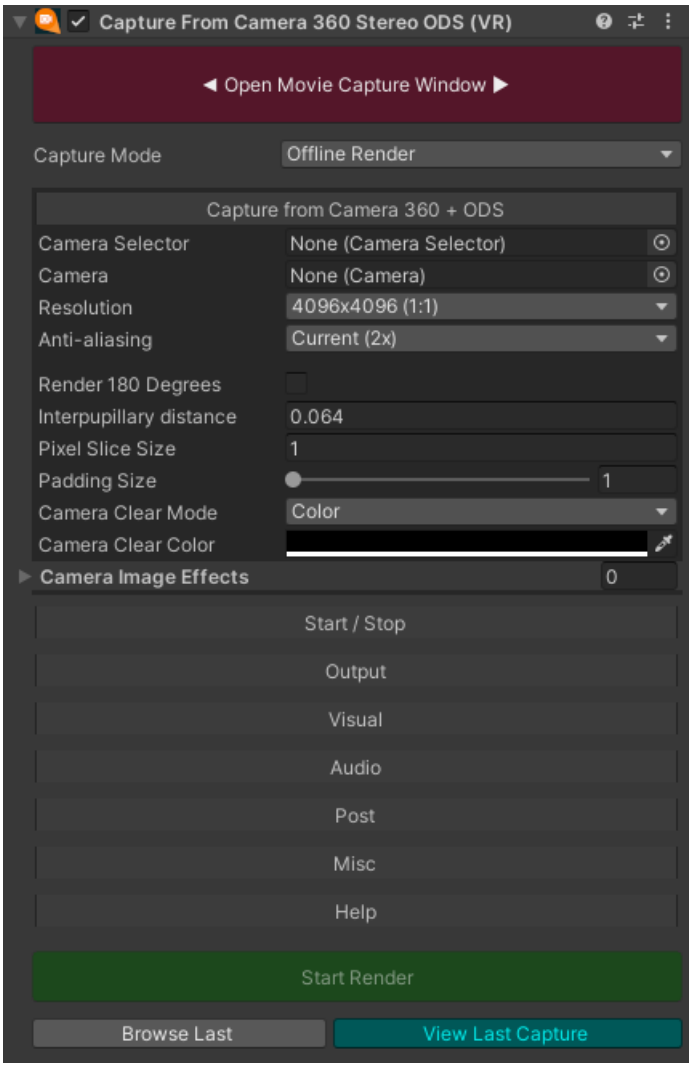
Help

Shows the plugin version number and has links to the website, issue tracker, documentation etc



- The "Start Capture" button will begin capturing. The scene needs to be running.
- The "Browse Last" button will show a Finder/Explorer window with the last capture selected.
- The "View Last Capture" button will play back the last captured video using the system media player.

Capture From Camera 360 ODS



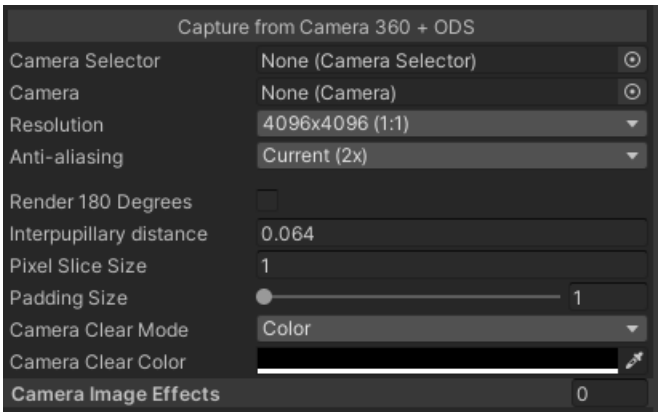
This component allows accurate rendering of stereo 360 scenes to equi-rectangular format via a technique called Omni-Directional Stereo (ODS) and based on this paper: https://developers.google.com/vr/jump/rendering-ods-content.pdf

This component takes a very long time to render each frame and is only suitable for offline rendering. This is due to the complex rendering method, but the stereo results are much more accurate than using the stereo option in the CaptureFromCamera360 component. This component always outputs a stereo image and shouldn't be used for monoscopic rendering.

Note that Unity may become unresponsive when using this component. This is because each frame takes a long time to render so the Unity editor will only update at the end of each frame. Hopefully we will improve this in the future to keep the UI responsive. It is best to set the "Auto Stop" mode when using this component so that it will stop at the end of a sequence without user intervention.

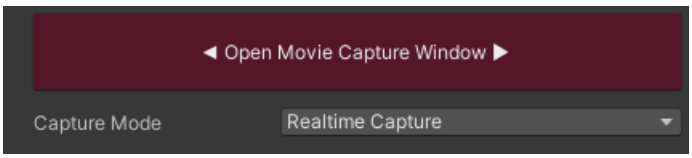
Also, please note that this capture component can use a ton of memory if the Unity profiler is enabled due to how many rendering calls it does per frame. Even a simple scene can take over 32GB of memory due to the profiler. You must disable the profiler completely when using this component. This is done by opening the profiler, disabling the record option, closing the profiler, remove the tab if it is docked, then restarting Unity

From Camera 360 ODS



Properties

PROPERTY	FUNCTION
Camera Selector	Selects the camera to render from. Using the CameraSelector component allows easy switching between cameras.
Camera	Selects the camera to render from if only one camera is required.
Resolution	The resolution of the final output (before downscale)
Anti-aliasing	Anti-aliasing to use during rendering
Render 180 Degrees	Selecting this option will output only 180 degree rendering
Interpupillary Distance (IPD)	The distance between the pupils. This is set to 0.064 which is the standard for an american male adult. You may wish to alter this if creating content for a wide range of users and quality is important. This value is in meters and assumes a scale of 1 unit = 1 meter
Pixel Slice Size	The size in pixels of each vertical slice. Scene renders per frame = (Resolution.width / Pixel Slice). Increasing this number will make rendering faster, but will decrease quality. Usually best to leave this set to 1 pixel
Padding Size	The amount of padding in pixels to add to each vertical slice. If you're using post-process effects (bloom etc) then you may need to increase this above 1 so that these effects work.
Camera Clear Mode	How the camera clears
Camera Clear Color	The colour to clear the camera with
Camera Image Effects	Here you can add any legacy Image Effects that are applied to your camera. These effects will then get applied as the rendering is done.

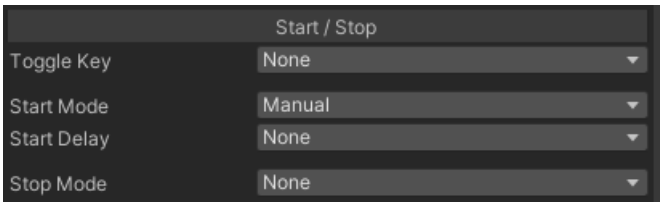


- The "Open Movie Capture Window" button opens the in-editor capture window which allows capturing without adding components to your scene

Properties

PROPERTY	FUNCTION
Capture Mode	Capture can either be realtime, or an offline render

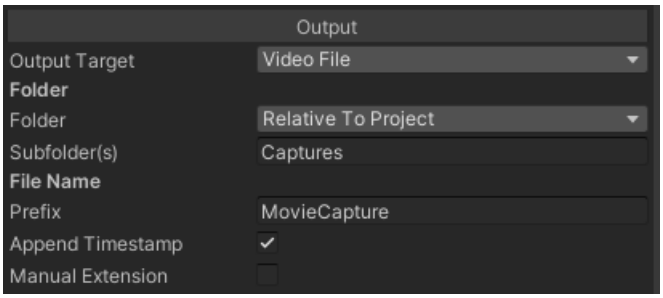
Start/Stop



Properties

PROPERTY	FUNCTION
Toggle Key	Select a key to toggle the start and end of capturing
Start Mode	The capture can start either when the component starts, or wait until it is manually triggered by the user via scripting
Start Delay	An optional delay can be specified before the frame capturing actually starts
Stop Mode	Without a stop mode the capture will continue forever until it is stopped by the user or script. You can set a stop mode to make the capture stop when it reaches either a certain number of frames, or a duration in seconds

Output

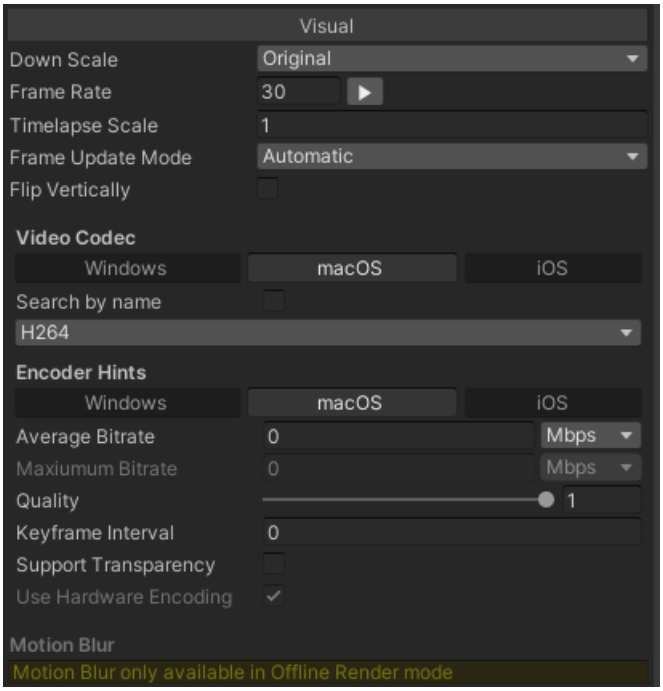


Properties

PROPERTY	FUNCTION
Output Target	Select if you want to output to a video file, image sequence, or a named pipe
Folder	
Folder	Select the relative folder to output to
Subfolder(s)	Select a subfolder to output to
File Name	
Prefix	The start of the file name
Append Timestamp	Whether to append an auto-generated timestamp to the file name for videos

PROPERTY	FUNCTION
Manual Extension	Whether to manually specify an extension to the file name for videos
Image Sequence	
Format	The format of the image sequence (per-platform)
Start Frame	The number to start the frame count with for the file name
Zero Digits	The number to digits to use for the frame count in the file name
Named Pipe	
Pipe Path	The path of the pipe to write to (eg `\\.\pipe\test_pipe`)

Visual



Properties

PROPERTY	FUNCTION
Down Scale	Reduce the resolution of the final output
Frame Rate	The frame rate per second for output file
Timelapse Scale	For real-time captures a scale can be set to allow for timelapse captures
Frame Update Mode	Allows the frame update to be controlled automatically, or manually via scripting. This can be useful when capturing an element that doesn't necessarily update each Unity frame, eg a webcam or custom texture rendering
Flip Vertically	Flip the output image vertically (debug only)
Video Codecs	

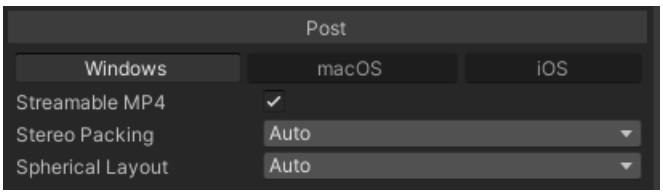
PROPERTY	FUNCTION
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don't have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Video Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index
Encoder Hints	
Average Bitrate	Ideal average bitrate of output video
Maximum Bitrate	Ideal maximum bitrate of output video
Quality	0..1 range to specify quality preference over other factors (eg encoding speed)
Keyframe Interval	The distance between keyframes
Support Transparency	Whether or not transparency should be supported if possible
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Use Hardware Encoding	Whether to prefer to use hardware or software encoding
Enable Fragmented Writing	Enables fragmented writing of QuickTime (mov, mp4, etc.) files. Enabling this feature allows you to open and play a partially written asset should something unrecoverable occur during the writing process. Adjust `Movie Fragment Interval` to control the fragment size
Movie Fragment Interval	The interval in seconds at which movie fragments should be written, the default is 120 seconds. Making this value too small may result in captures failing
Motion Blur	
Use Motion Blur	Enables an experimental option which is available for offline rendering. It accumulates frames in between the capture frames to create motion blur. This is very useful for high quality rendering but it is very expensive as it is a brute force approach to motion blur
Samples	The number of sub-frames to render which get accumulated into a single output frame
Cameras	

Audio

Properties

PROPERTY	FUNCTION
Audio Source	Specified where to capture audio from, this can be set to None, Unity, Microphone, Manual, or Unity Audio Mixer
Unity Audio Source	
Unity Audio Capture	The component to use for capturing audio from Unity. This can be CaptureAudioFromAudioListener, CaptureAudioFromAudioRenderer or CaptureAudioFromWwise
Microphone Audio Source	
Force Audio Device Index	The index of the microphone/recording device to use. Default is zero
Manual Audio Source	
Sample Rate	The sample rate for manual audio encoding
Channels	The number of audio channels for audio audio encoding
Audio Codec	
Search By Name	Whether or not to search for a codec based on a prioritised list of codec names, or whether to force codec use. On macOS and iOS it is possible to specify the codec directly as they are all generally supported. On Windows it is not always known which codecs are available, so it can be useful to specify a search list.
Codec Search Order	Allows you to list the names of codecs to search for. The first codec in the list that it finds installed on the system will be the one used for encoding. This is useful if you are deploying an app to a system you don't have codec control over so that it can look for a number of codecs before finally falling back to uncompressed. If you want uncompressed simply add a blank entry to the list
Force Audio Codec Index	Only for Windows, allows you to override the search priority list and select a codec directly by index

Post

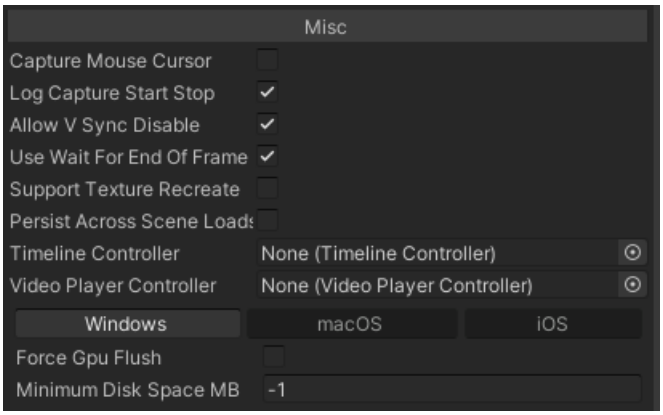


After the video file has been created, an optional post-process can be run to make addition changes to the file.

Properties

PROPERTY	FUNCTION
Streamable MP4	Makes the MP4/MOV file suitable for streaming by using 'fast start' encoding method
Stereo Packing	Injects data to specify the stereo format of the video
Spherical Layout	Injects data to specify the spherical format of the video

Misc

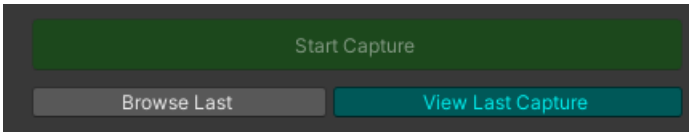


Properties

PROPERTY	FUNCTION
Capture Mouse Cursor	This option is only available when using the CaptureFromScreen component and allows the mouse cursor to be captured. You have to specify a texture to use for the cursor
Log Capture Start Stop	Log each time the capture is started or stopped. Disable this for less garbage generation
Allow V-Sync Disable	For off-line capturing, allow vsync to be disabled which allows captures to run as fast as possible
Use Wait For End Of Frame	Makes sure capture happens right at the end of the frame - this allows some features such as skinning to be resolved correctly
Support Texture Recreation	
Persist Across Scene Loads	The GameObject will not be destroyed when the scene is unloaded.
Timeline Controller	Optional TimelineController component when capturing Timelines using offline rendering
VideoPlayer Controller	Optional component when capturing VideoPlayers using offline rendering (not currently working)
Windows Only	
Force GPU Flush	Flushing the GPU during each capture results in less latency, but can slow down rendering performance for complex scenes.
Minimum Disk Space MB	Set to -1 to ignore, otherwise it will keep checking the disk space and stop the capture if the free disk space gets below this number of MB

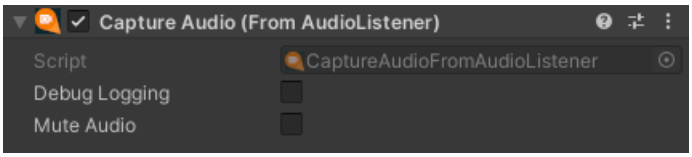
Help

Shows the plugin version number and has links to the website, issue tracker, documentation etc



- The "Start Capture" button will begin capturing. The scene needs to be running.
- The "Browse Last" button will show a Finder/Explorer window with the last capture selected.
- The "View Last Capture" button will play back the last captured video using the system media player.

Capture Audio From AudioListener



This component can be added to capture the audio from Unity during a real-time capture.

This must be added to a GameObject that contains the `AudioListener` you want to record (usually Main Camera).

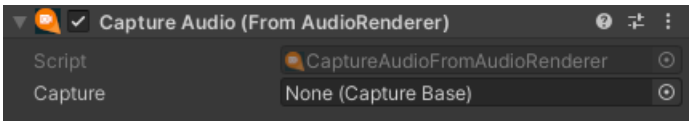
One thing to be noted about this component is that it is implemented as a filter, therefore enabling bypass effects on an `AudioSource` would exclude that `AudioSource` from being captured.

This component must then be connected to the CaptureFrom* component via the "Unity Audio Capture" field.

Properties

PROPERTY	FUNCTION
Debug Logging	Used for debugging only
Mute Audio	Mute the capturing

Capture Audio From AudioRenderer



This component can be added to capture the audio from Unity during an offline render.

This component must then be connected to the CaptureFrom* component via the “Unity Audio Capture” field.

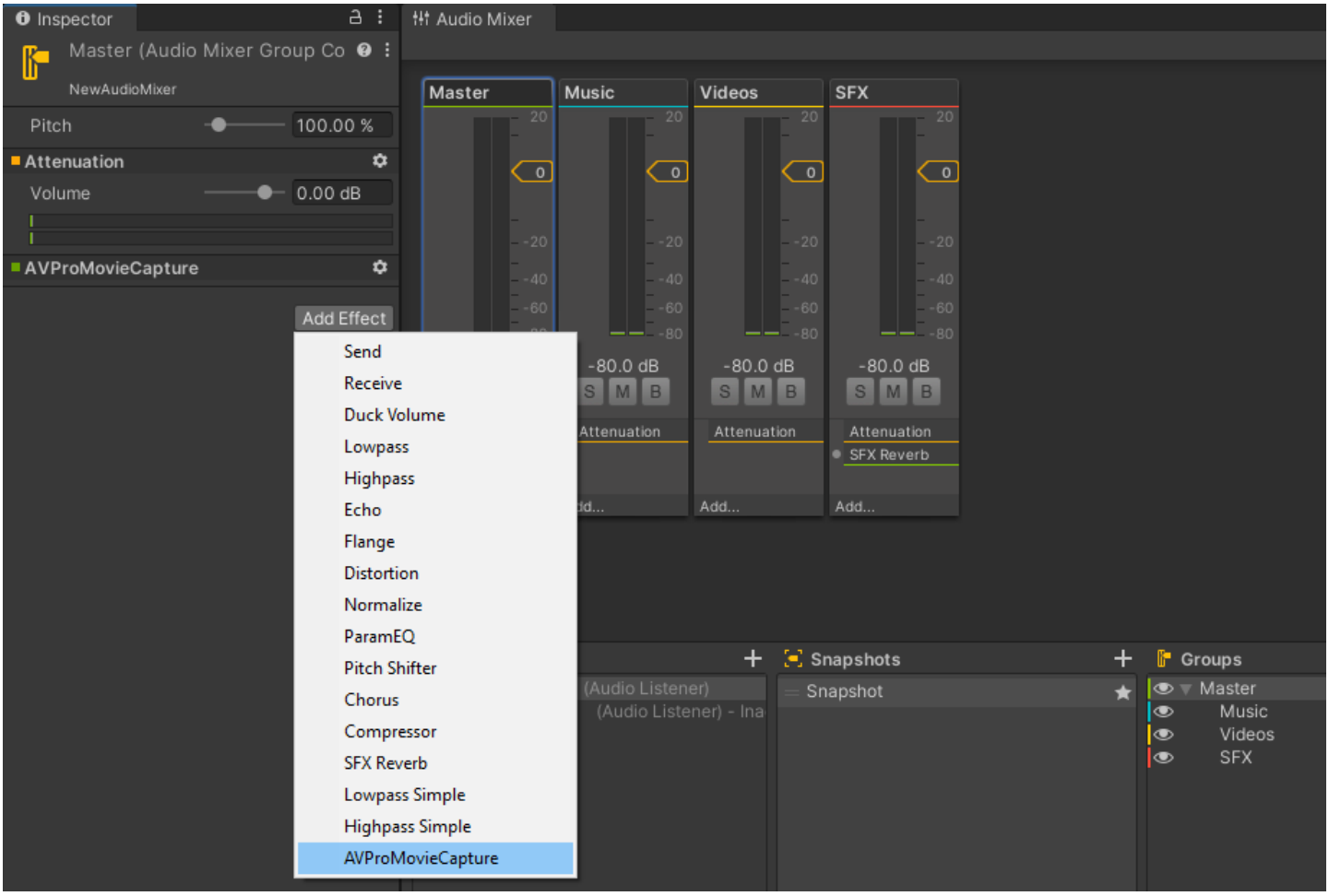
During capturing this component will set Unity into recording mode, so no audio will be played by Unity.

This component requires Unity 2017.3 or newer.

Properties

PROPERTY	FUNCTION
Capture	The CaptureFrom component where the audio will be written to

Capture Audio From AudioMixer



This is not actually a component, but an audio plugin that allows direct capture from Unity's `Audio Mixer`.

The advantage of this component over the other methods of capturing Unity audio is that it is more performant, so to achieve a high frame-rate real-time captures this is the best method.

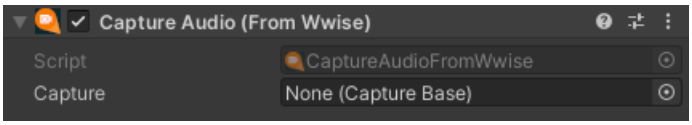
Requirements: 1) This feature is currently for Windows only 2) Ensure that any audio playback (eg from an `AudioSource`) is set to use the correct `Audio Mixer`, otherwise the audio will not be captured

Usage: 1) Ensure that the audio plugin is set to auto-load on startup. To do this, go to the `AVPMovieCapture/Runtime/Plugins/Windows/x86_64` folder, select the file `AudioPluginAVProMovieCapture` and enable the option `Load on startup`. 2) Select the `Audio Mixer` you want to record 3) Add the "AVProMovieCapture" audio effect 4) In your CaptureFrom component Audio section, select "Unity Audio Mixer" as the audio source

> **⚠ WARNING**
>
> Ensure that you only have a single "AVProMovieCapture" audio effect active at once. If you have multiple active then audio capture will not be correct and it may impact capture performance.

Limitations: 1) This feature only works for real-time captures and will not work with offline rendering. For rendering consider using the CaptureAudioFromAudioRenderer component instead 2) This feature only works with a single Capture instance, so it doesn't work when using multiple capture instances in parallel.

Capture Audio From Wwise



This component can be added to capture the audio from Wwise during an offline render.

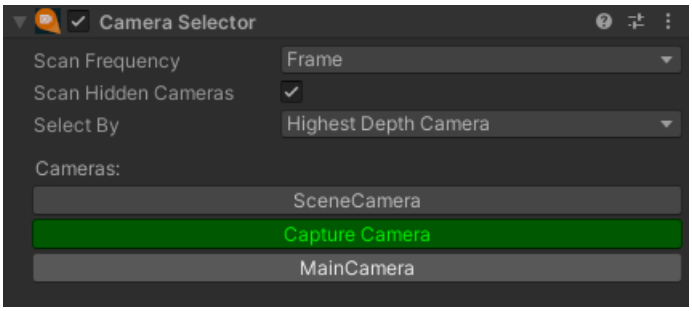
To use this component some setup is required:

1. The Wwise package (2021.1.4 or newer) must be installed in your project.
2. The string `AVPRO_MOVIECAPTURE_WWISE_SUPPORT` must be added to `Scriping Define Symbols` in `Player Settings > Other Settings > Script Compilation`.
3. In the assembly definition (.asmdef) file `AVProMovieCapture.Runtime` set the WWise assembly definition file as a dependency.
4. Assign this component in your CaptureFrom* component via the "Unity Audio Capture" field.

Properties

PROPERTY	FUNCTION
Capture	The CaptureFrom component where the audio will be written to

Camera Selector



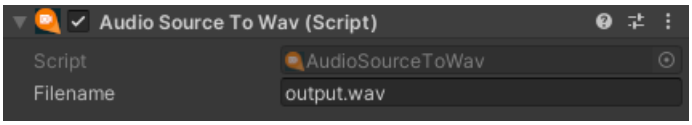
This utility component is used by `CaptureFromCamera`, `CameraFromCamera360` and `CapturefromCamera360ODS` components to give better control over the camera used during capturing.

This allows various options to be specified which will determine which camera is used in the scene. This can help for capture situations where the desired capture camera changes over time, or across scene loads

Properties

PROPERTY	FUNCTION
Scan Frequency	How often to scan for the best camera - every frame, every scene load, or manually triggered via scripting
Scan Hidden Cameras	Look for hidden cameras (such as the Scene View camera)
Select By	The criteria to select the camera by. Cameras can be found by name, tag, depth or manually specified
Cameras	In Play mode the list of cameras are shown to allow easy camera switching

AudioSource To WAV



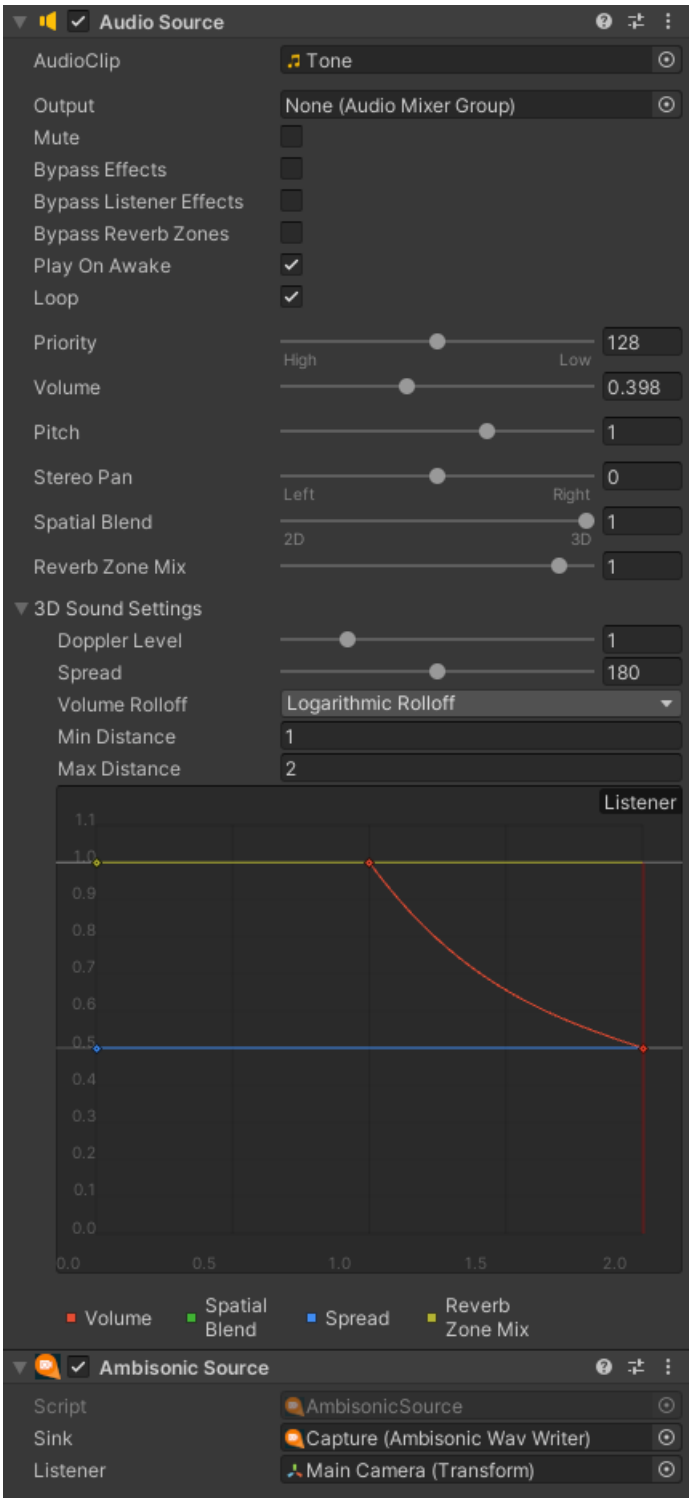
This utility component is used to record an `AudioSource` component directly to a WAV file.

The WAV file is saved into the `Application.persistentDataPath` folder.

Properties

PROPERTY	FUNCTION
File Name	the name of the file to generate (eg "audio.wav")

Ambisonic Source



This component captures audio generated by an `AudioSource` and so usually is added to a `GameObject` containing an `AudioSource` component.

The `AudioSource` needs to be positional (3D) so set the Spatial Blend to 1. The `AudioSource` should ideally be a mono source, and so the 3D Spread option should be set to 180 so that the audio loudness is equal in all channels.

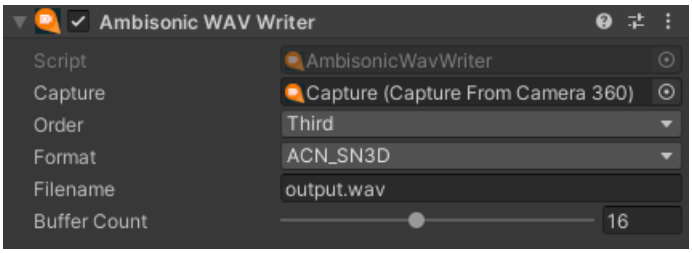
Distance and Rolloff can be used so the audio gets quieter with distance.

Properties

PROPERTY	FUNCTION
Sink	The audio is encoded as ambisonic audio and sent to the sink for processing. The only sink currently is the AmbisonicWavWriter component.

PROPERTY	FUNCTION
Listener	An optional Listener `Transform` can be specified. This is useful if the listener (usually the Main Camera) isn't fixed at 0,0,0 and allows the positions of the audio to be calculated relative to the listener.

Ambisonic WAV Writer



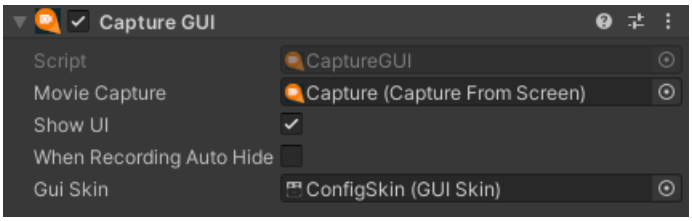
This component will generate a WAV file containing the Ambisonics audio that was captured by the AmbisonicSource components.

The WAV Writer will start collecting audio when the main CaptureFrom component begins capturing, so that the audio will be in sync with the video captured.

Properties

PROPERTY	FUNCTION
Capture	The CaptureFrom component to use for starting and stopping this audio capture
Order	The ambisonic order to encode with. Higher orders require more audio channels but have improved spatialisation. The WAV file will have 4 channels for 1st order, 9 channels for 2nd order, and 16 channels for 3rd order Ambisonics.
Format	Whether to use FUMA or SN3D channel ordering and normalisation standards
Filename	The filename to output (eg audio.wav). The file will be stored relative to Unity's `Application.persistentDataPath`
Buffer Count	Number of buffers to allocate for performance

Capture GUI



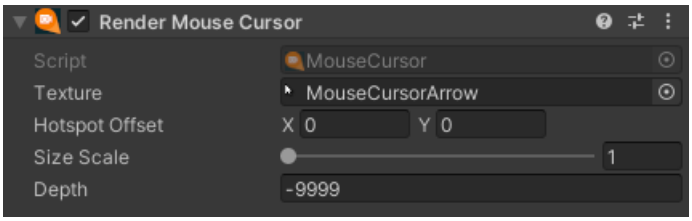
This component renders a GUI to the game view (using `IMGUI`) to control the `CaptureFrom` component. It allows the codecs to be selected, capture resolution and frame rate to be chosen and the capture itself to be controlled via Start / Pause / Resume / Stop buttons.

This component is included as it can be useful but it is intended as more of an example than something to include in your builds.

Properties

PROPERTY	FUNCTION
Movie Capture	The CaptureFrom component this GUI controls
Show UI	Whether to currently draw this GUI or not
When Recording Auto Hide	Whether to hide the GUI during recording
Gui Skin	Optional IMGUI skin to use

Mouse Cursor

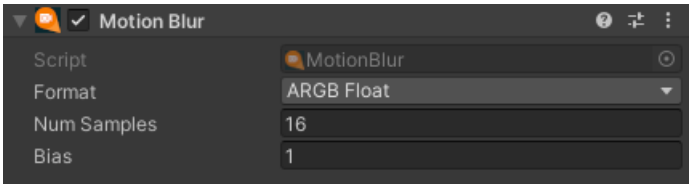


Uses `IMGUI` to render a custom mouse cursor texture to the screen. This is useful when using the only a hardware mouse cursor is used and the CaptureFromScreen component needs to capture a mouse cursor.

Properties

PROPERTY	FUNCTION
Texture	The mouse cursor texture to render
Hotspot Offset	The pixel coordinates for the offset of the cursor tip
Size Scale	Scale
Depth	The IMGUI depth/layer to render at

Motion Blur



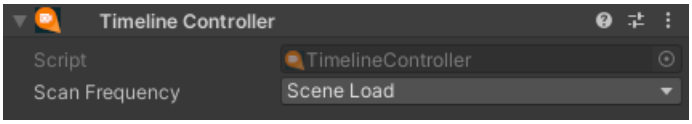
This is an internal component used when Motion Blur is enabled.

It handles accumulation of sub-frames into a texture.

Properties

PROPERTY	FUNCTION
Format	The texture format to use for accumulation
Num Samples	The number of sub-frames to accumulate
Bias	

Timeline Controller



This component is used for scenes that use the Playable Directors / Timeline feature of Unity when doing non-realtime captures. It allows the plugin to take control over the timestep of the Timeline so that it stays in sync with the capture.

If you are using the Movie Capture Window then this component is automatically used for offline captures. If you are using one of the `CaptureFrom` components, then simply add this component to the same `GameObject` and set the `TimelineController` field in the CaptureFrom component under the Misc section, or via the API using the `TimeController` property

Properties

PROPERTY	FUNCTION
Scan Frequency	The frequency at which to scan for new Timeline's

Windows Desktop Platform

Plugin Specs

- Compatibility
  - Unity 5.6 - 2017.x - 2021.x are supported
  - Supported CPU architectures are 32-bit and 64-bit x86
  - Windows XP (SP3) - 10 are supported, however some versions of Windows allow for more features than others
- Rendering
  - For rendering we support Direct3D 11, Direct3D 12 (requires minimum Unity 2019.3)
  - Multi-threaded rendering is supported
- Internals
  - Under the hood we're using the Media Foundation and DirectShow API's. Media Foundation supported on Windows 8 and above, and DirectShow is supported from Windows XP.
  - The only 3rd-party libraries used in the Windows Desktop binaries are:
    - GDCL Mpeg-4 https://github.com/roman380/gdcl.co.uk-mpeg4
    - GLEW http://glew.sourceforge.net/

Supported Codecs

See the Codecs section for more information.

Troubleshooting

Windows N / KN editions

- There are some editions of Windows (N and KN) that ship with greatly reduced built-in media playback capabilities.
- It seems like these editions don't include MFPlat.DLL, but do include some basic DirectShow components. This means the Media Foundation playback path will not work.
- These editions of Windows require either a 3rd party codec installed (such as the LAV Filters for DirectShow), or the Microsoft Media Feature Pack:
  - Media Feature Pack for Windows 7 SP1 https://www.microsoft.com/en-gb/download/details.aspx?id=16546
  - Media Feature Pack for Windows 8.1 https://www.microsoft.com/en-gb/download/details.aspx?id=40744
  - Media Feature Pack for Windows 10 https://www.microsoft.com/en-gb/download/details.aspx?id=48231

macOS Platform

Plugin Specs

- Compatibility
  - Unity 2017.x - 2021.x are supported
  - macOS 10.13 and later are supported
  - Only 64-bit (x86_64, arm64) builds are supported
- Rendering
  - Metal and OpenGL Core rendering APIs are supported
- Internals
  - Under the hood we’re using Apple's AVFoundation API

Supported Codecs

See the Codecs section for more information.

Troubleshooting

Notarising

- We notarise the plugin bundle so you shouldn't have to code-sign it prior to building your package for submission to the App Store.

iOS Platform

Plugin Specs

- Compatibility
  - Unity 2018.x - 2021.x are supported
  - iOS 11.0 and later are supported
- Rendering
  - Only the Metal rendering API is supported
- Internals
  - Under the hood we’re using Apple's AVFoundation API

Supported Codecs

See the Codecs section for more information.

Android Platform

Plugin Specs

- Compatibility
  - Unity 2018.x - 2022.x are supported
  - Android Oreo 8.0 (API 26) and later are supported
  - Android 9.0 (API 28) and later recommended
  - Supported CPU architectures are arm-v7a, arm64-v8a, x86 and x86-64
- Rendering
  - OpenGL ES 3.0 and above is supported
  - Vulkan graphics API is not yet supported
  - Multi-threaded rendering is supported
- Internals
  - Under the hood we're using Android's MediaCodec API
  - The only 3rd-party libraries used are:
    - stb_image (Trial version only)
    - stb_image_write (Used for png and jpg writing on APIs 28 and 29)
    - rapidxml

Supported Codecs

See the Codecs section for more information.

Troubleshooting

Collecting logcat output

We often need to see the device logs to work out why something isn't working. For this the device should be connected via USB.

If you're using Android Studio then you can click on the `Logcat` tab and choose `No Filters` in the bar on the top right. You should see logs being produced and can copy-paste all of them into a text file.

Another useful tool on Windows is called mLogcat and is a GUI tool for monitoring and capturing logs from Android.

Alternatively the command-line tool adb can be used from the Android SDK. Just run the following from a console (Windows) or terminal (macOS) window:

```
adb logcat
```

Portrait captures are landscape with 90° rotation

Some Android devices have hardware encoders that are less flexible and provide better support for a landscape orientation.

For instance, the Xiaomi Redmi Note 10 is able to capture HEVC at a maximum resolution of 3840x2160. When making a screen capture in landscape this is fine and is able to fully capture the screen resolution of 2400x1080. However in portrait the vertical resolution of 2400 pixels is greater than the maximum supported vertical resolution of the encoder at 2160 pixels. Therefore, in order to provide screen captures at full resolution we capture in landscape and add a 90° rotation to the videos metadata.

You can disable this behaviour by enabling "No Capture Rotation" in the Android platform specific options of the Capture component inspector.

Fails to create recorder when using the microphone as the audio source

This is most likely due to permission not being granted for using the microphone.

Because we do not use Unity's Microphone class, Unity is not aware that it needs to add the recording permission to the manifest. Details of how to do this are available here. You need to add the following line before the `application` section of the manifest:
`<uses-permission android:name="android.permission.RECORD_AUDIO" android:required="false" />`

You will also need to request permission at run time. See the Requst Audio Permission section for more details.