VR Video Player

Introduction:

VR Video Player assets will help you to setup virtual reality video experience in your Unity app.

Supported Platform:
Windows
MacOS
Android
iOS

Supported VR SDK:
Google Cardboard
Google Daydream
Oculus GearVR
Oculus Rift
HTC Vive

Supported video player:
Unity Movie Texture
Unity Video Player
Android Media Player
HTC Media Decoder
Easy Movie Texture

Supported VR video format:
2D/3D
180/180-3D
360/360-3D
3D format include Side-by-Side and Up-Down.

Roadmap:

Support all video container format.
Support remote online video streaming url.
Support API follow https://developer.android.com/reference/android/media/MediaPlayer.html.

Player prefab introduction:

Player Prefab	VR Video format
Evereal/VRVideoPlayer/Prefab/FlatVideoPlayer/FlatVideoPlayer	Flat 2D video
Evereal/VRVideoPlayer/Prefab/FlatVideoPlayer/FlatVideoPlayer_LR	Stereo side by side video
Evereal/VRVideoPlayer/Prefab/FlatVideoPlayer/FlatVideoPlayer_UD	Stereo up down video
Evereal/VRVideoPlayer/Prefab/360VideoPlayer/360VideoPlayer	360 video
Evereal/VRVideoPlayer/Prefab/360VideoPlayer/360VideoPlayer_LR	Stereo side by side 360 video
Evereal/VRVideoPlayer/Prefab/360VideoPlayer/360VideoPlayer_UD	Stereo up down 360 video
Evereal/VRVideoPlayer/Prefab/180VideoPlayer/180VideoPlayer	180 video
Evereal/VRVideoPlayer/Prefab/180VideoPlayer/180VideoPlayer_LR	Stereo side by side 180 video
Evereal/VRVideoPlayer/Prefab/180VideoPlayer/180VideoPlayer_UD	Stereo up down 180 video

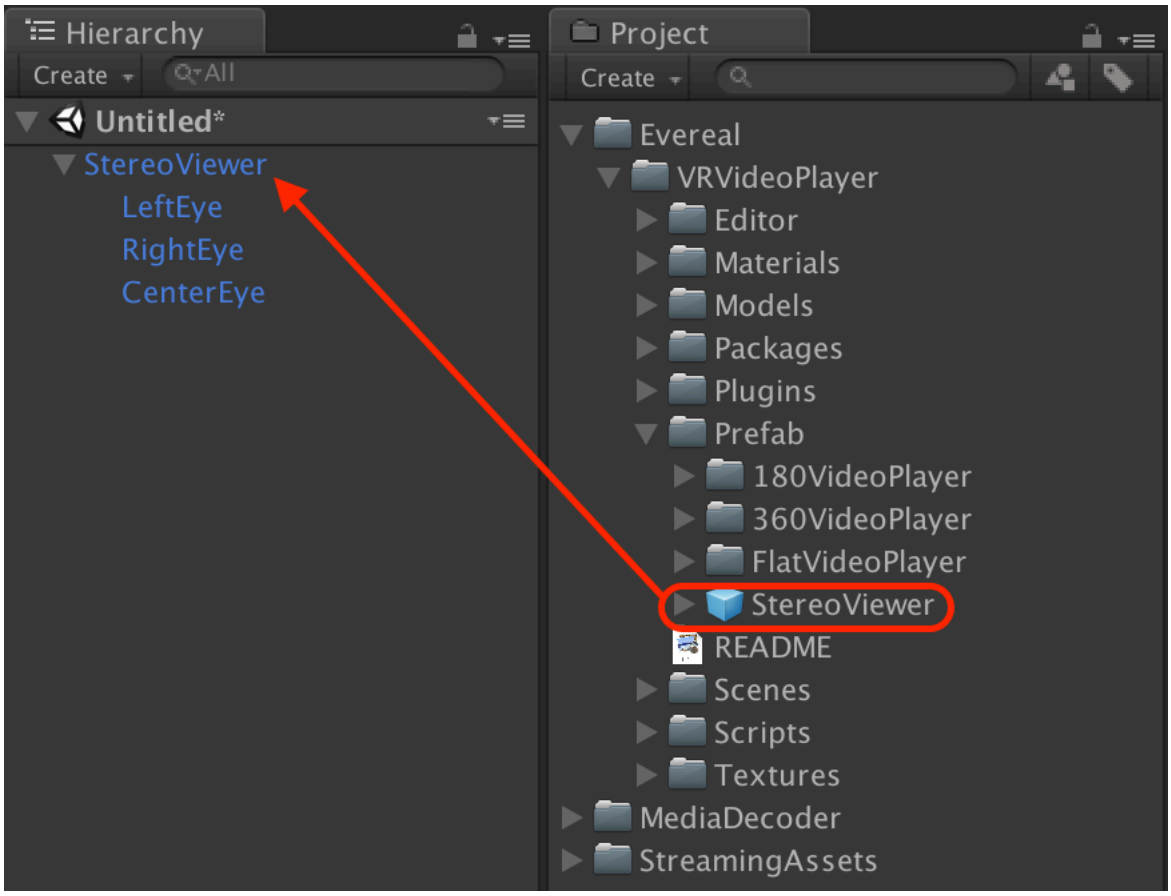
Note: Usually the stereo 360 video should be up down format, and stereo 180 video should be side by side format.

Setup Guide:

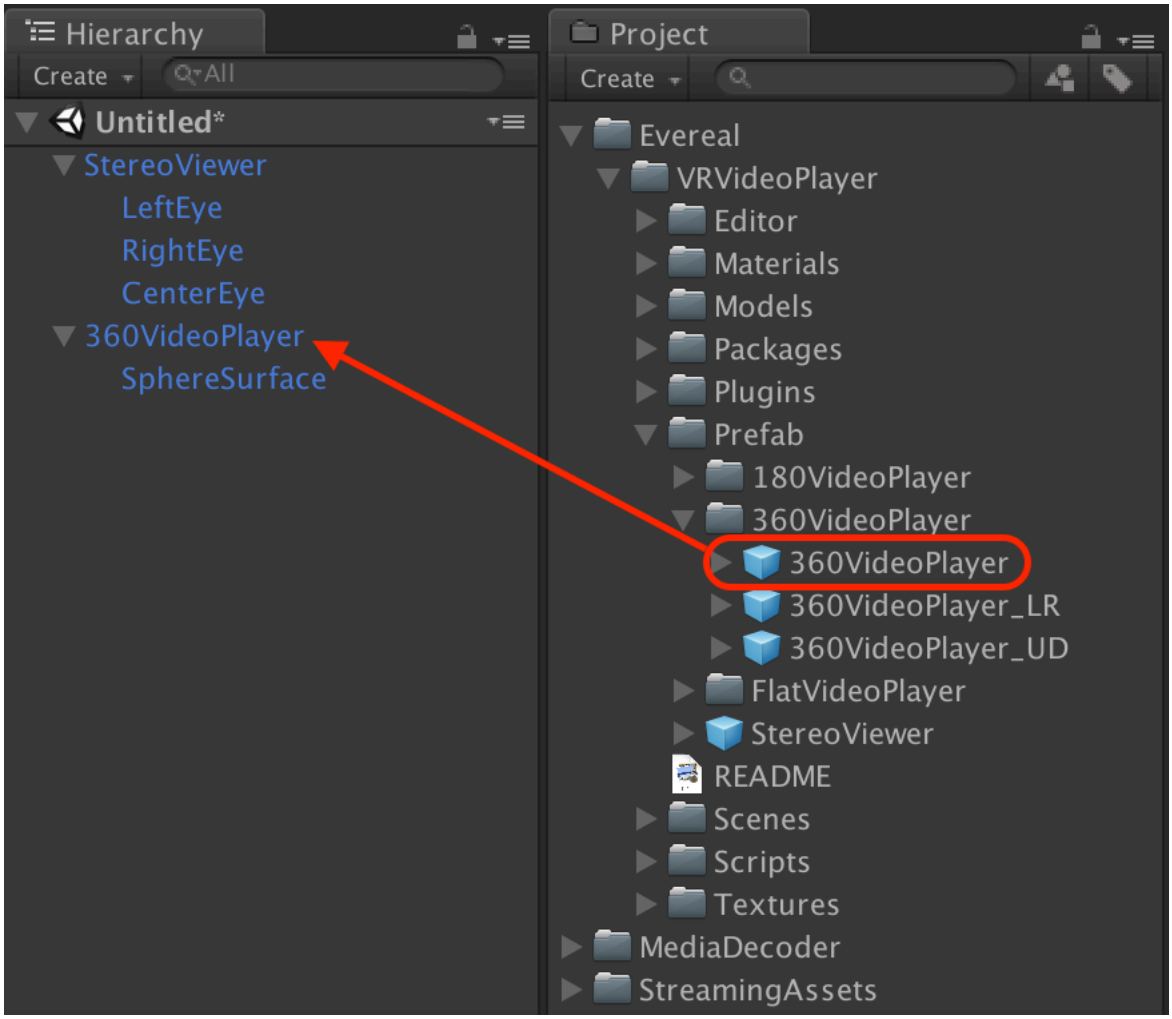
Setup process will be extremely easy, you can enable the all VR formats video play functionality only in 15 seconds!

Setup for Oculus GearVR:

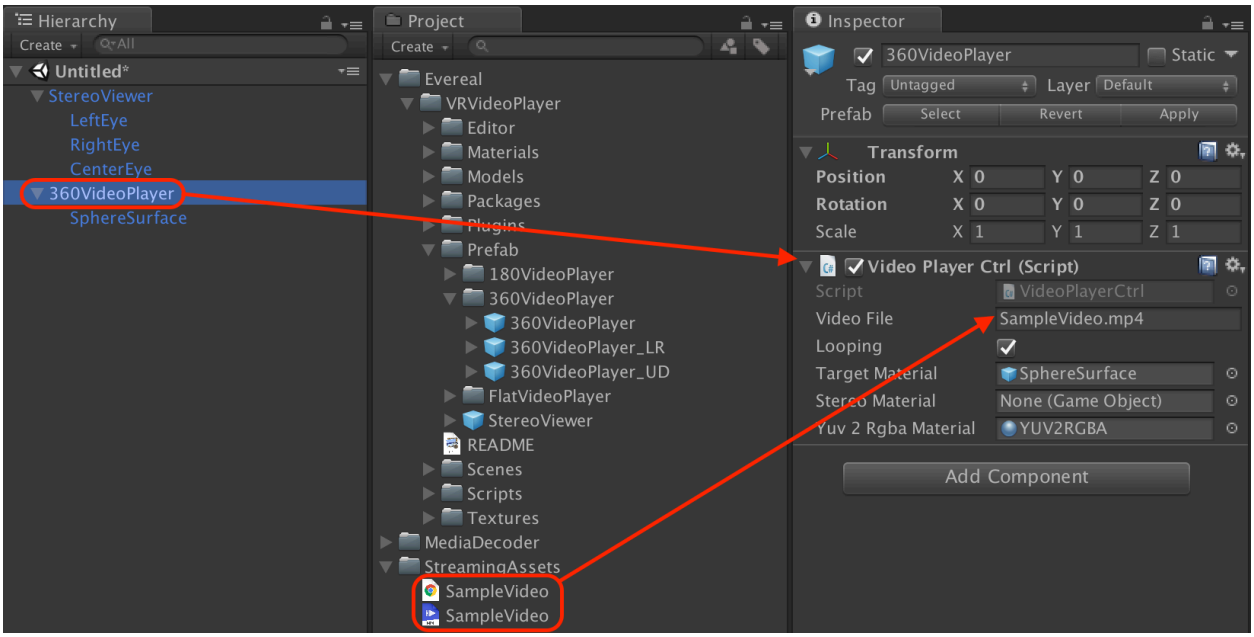
Drag the *Evereal/VRVideoPlayer/Prefab/StereoViewer* prefab into your scene and disable you default main camera:



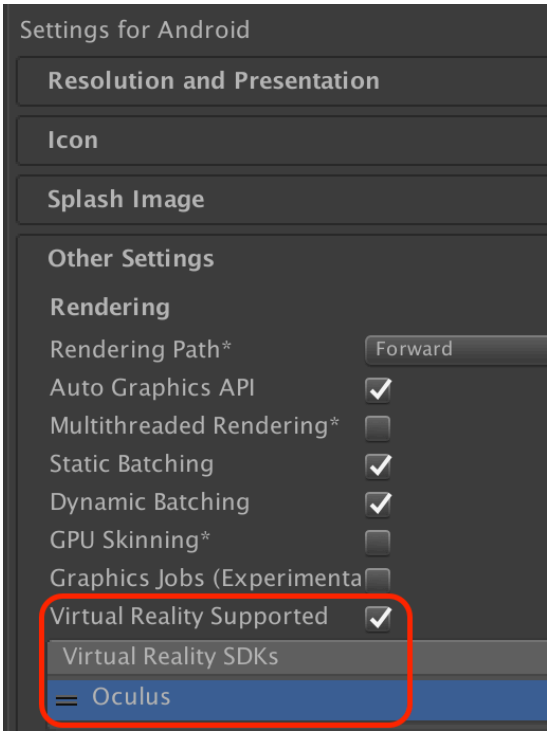
Depends on your requirement, drag the corresponding player (you can chose 180, 360 or flat player) prefab into your scene:



Fill in the correct video file name you want play with in *VideoPlayerCtrl* script, the video file should be place in *StreamingAssets* folder:



Enable Virtual Reality Supported for Oculus SDK:



Note: Please refer to https://unity3d.com/cn/learn/tutorials/topics/virtual-reality/deploying-your-vr-project for more configure information.

Then you should be able to build the player app on Android, and use it with Gear VR glass:

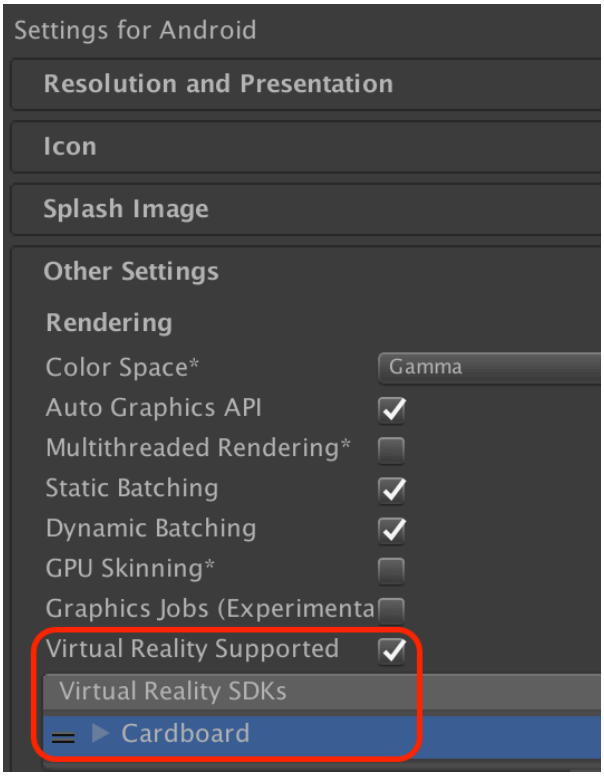
Setup for GoogleVR/Cardboard:

You need download GoogleVR SDK package from:
https://developers.google.com/vr/unity/download

Note: This plugin has been tested with Google SDK v0.6, v1.2 and v1.5, any version between and newer should work.

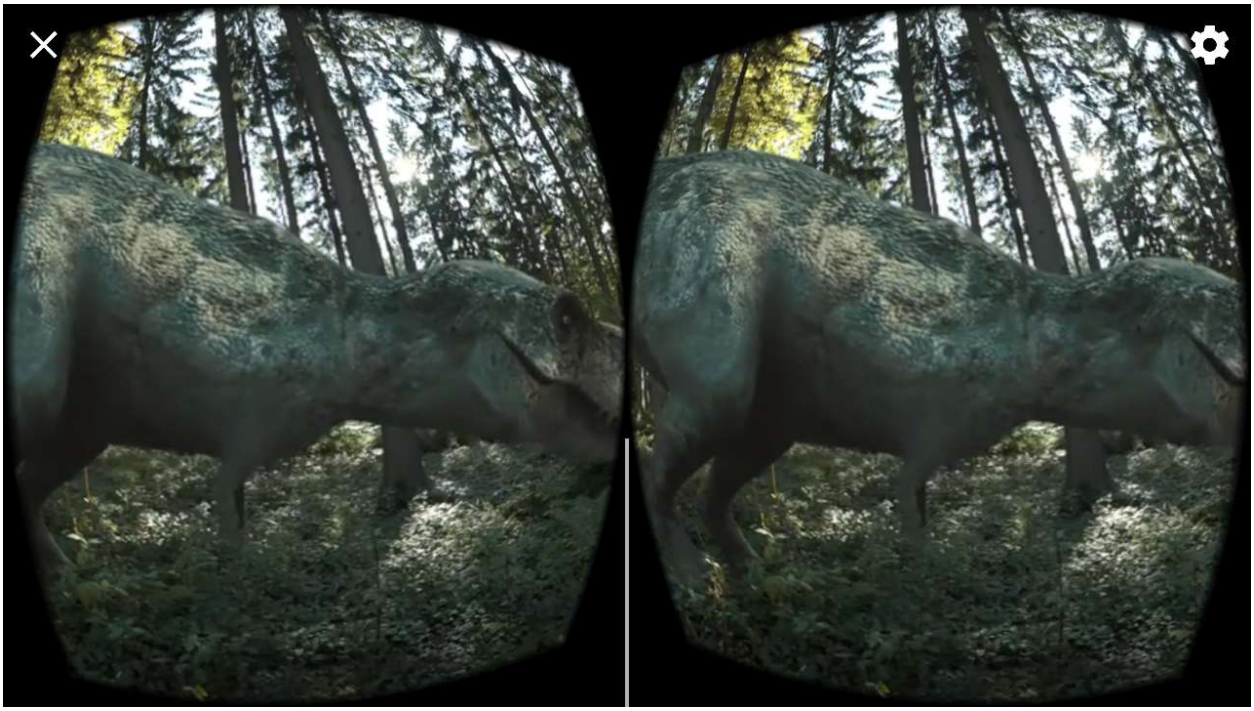
Unity 5.6 with GoogleVR SDK 1.5 or newer:

Its same with steps of Oculus GearVR except of enable Virtual Reality Supported for Google Cardboard SDK:



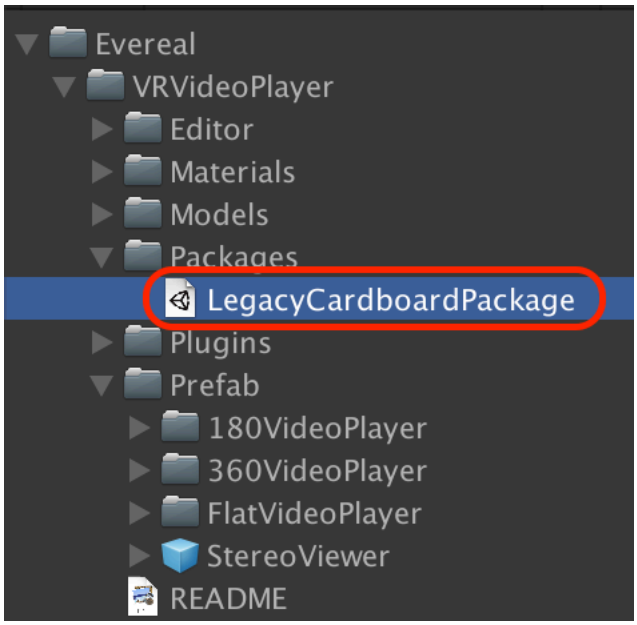
Note: Please refer to https://developers.google.com/vr/unity/get-started for more configure information.

Build it on Android and use it with Cardboard:

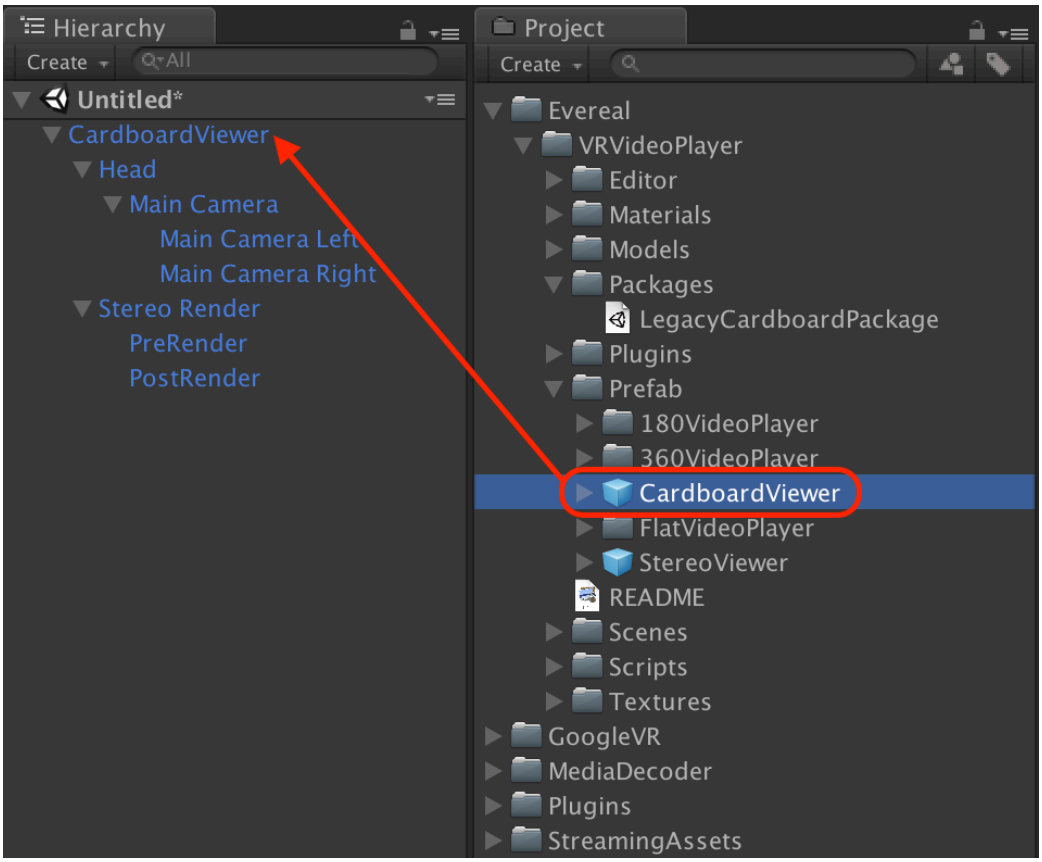


Unity with Legacy GoogleVR SDK (before v1.5):

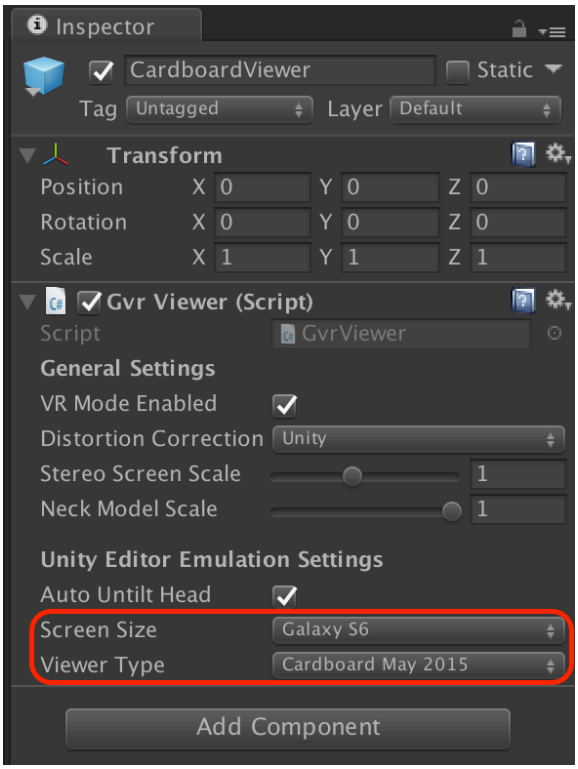
Open *Evereal/VRVideoPlayer/Packages/LegacyCardboardPackage*:



Drag the *Evereal/VRVideoPlayer/Prefabs/CardboardViewer* prefab into your scene and disable you default main camera:



Setup correct device type and screen size:



The rest of steps are same with steps of Oculus GearVR except that enable Virtual Reality Supported is not required, build it on Android and use it with Google Cardboard.

Note: The GoogleVR SDK also support iOS platform, you have two option to make it work, use EasyMovieTexture or use Unity 5.6.2f1 or later.

Setup for Oculus Rift / HTC Vive:

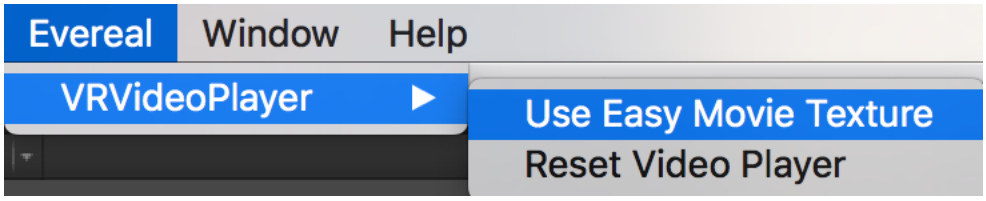
The video player backend for Oculus Rift or HTC Vive devices on Windows use MediaDecoder plugin, and if your Unity version is 5.6.0f3 or later, you can chose Unity VideoPlayer as alternative.

For macOS platform, this plugin use Unity VideoPlayer on Unity 5.6.0f3 or later and use Unity MovieTexture before Unity 5.6.0f3, the MovieTexture is very limitation and only support ogv video format, however, there are many tools can help you get ogv video format, such as: https://convertio.co/mp4-ogv/, you can convert then import to your game to make it work.

The setup process should be same as Oculus GearVR. Please contact me if you have any issue of integration.

Setup for EasyMovieTexture:

If you have purchased EasyMovieTexture plugin and want to keep using it as video decoder, import EasyMovieTexture and then click the menu item of *Use Easy Movie Texture*, then its done!



Note: This plugin has been tested with EasyMovieTexture v3.59.

Core API Reference:

Use VideoPlayerCtrl Script:

Public API Functions:

```
/// <summary>
/// Play/Resume the video.
/// </summary>
public void Play() {
/// <summary>
/// Pause the video.
/// </summary>
public void Pause() {
/// <summary>
/// Stop the video.
/// </summary>
public void Stop() {
```

Video Attributes:

```
/// <summary>
/// Get the state of the current video player.
/// </summary>
public StateType CurrentState {
/// <summary>
/// Get the width of the video.
/// </summary>
public int VideoWidth {
/// <summary>
/// Get the height of the video.
/// </summary>
public int VideoHeight {
/// <summary>
/// Get the duration of the video.
/// </summary>
public int VideoDuration {
```

Video States Flow:

```
/// <summary>
/// VideoPlayerCtrl state. Take reference from:
/// https://developer.android.com/reference/android/media/MediaPlayer.ht
///
///                       Idle
///                        |
///          Initialize()  |
///                        | LoadVideo()
///                        |
///                        v
///                   Initialized
///                        |
///             Prepare()  |
///                        |
///                        v
///                    Prepared
///                        |
///                Play()  |
///                        |
///                        v          Pause()
///             ------> Started ------------------> Paused
///             |          |  ^                        |
///             |   Stop()  |  |        Play()         |
///     Play()  |          |  --------------------------
///             |          v
///             -------- Stopped
///
///                        .
///    WaitVideoComplete()  |
///                        v
///                       End
///
/// </summary>
public enum StateType
{
    Idle = 0,
    Initialized = 1,
    Prepared = 2,
    Started = 3,
    Paused = 4,
    Stopped = 5,
    End = 6,
    Error = 7
}
```

Callback on Video Event:

You can register *OnVideoReady* and *OnVideoEnd* event:

```
public VideoReadyEvent OnVideoReady;
public VideoEndEvent OnVideoEnd;
```

Here is sample usage script:

```
public class SampleEventCtrl : MonoBehaviour
{
    public VideoPlayerCtrl videoPlayerCtrl;

    private const string TAG = "[VRVideoPlayer]";

    private void OnEnable()
    {
        videoPlayerCtrl.OnVideoReady += HandleVideoReadyEvent;
        videoPlayerCtrl.OnVideoEnd += HandleVideoEndEvent;
    }

    private void OnDisable()
    {
        videoPlayerCtrl.OnVideoReady -= HandleVideoReadyEvent;
        videoPlayerCtrl.OnVideoEnd -= HandleVideoEndEvent;
    }

    private void HandleVideoReadyEvent()
    {
        // Add you custom function here.
        Debug.Log(TAG + " Video is ready to play.");
    }

    private void HandleVideoEndEvent()
    {
        // Add you custom function here.
        Debug.Log(TAG + " Video playback is end.");
    }
}
```

FAQ:

1) Here is an error on YUV2RGBA shader file, how to fix?

If get an error in shader file with o.vertex = UnityObjectToClipPos(v.vertex), replace this line with o.vertex = mul(UNITY_MATRIX_MVP, v.vertex).

2) When using HTC Media Decoder, its failed to play video when build to standalone.

It is because the build failed to find required dll, you need place avcodec-57.dll, avformat-57.dll, avutil-55.dll, libeay32.dll, swresample-2.dll in the same folder with the built exe file.

Feedback:

For any issue or feedback, please leave a comment on Unity Asset Store or contact me at:

dev@evereal.com