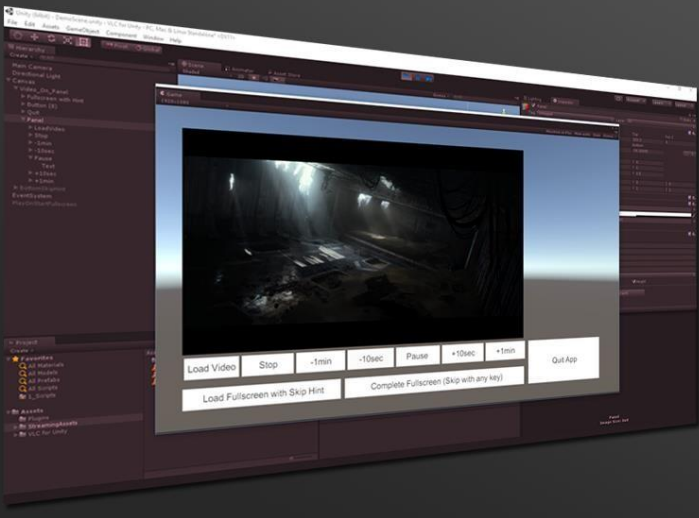



VLC FOR UNITY INTEGRATION README

Thanks for purchasing this asset. Please read this file for an easy introduction on how everything works.

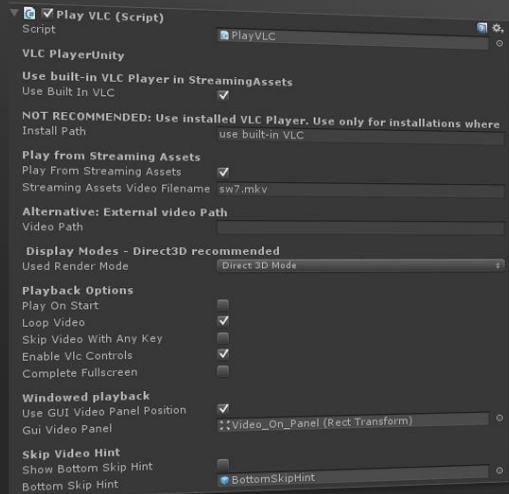
Introduction and Supported Media


VLC for Unity makes it easy to play any kind of video file in your Windows builds with the power of VLC. **You can use any video codec, file type, bitrate or resolution** that VLC supports. Since the video will be played on your GPU, even 4K videos should be handled easily by most PCs.



**VLC Media Player
for Unity**

- Play any format and codec
- 4K with Direct3D on GPU
- Fullscreen and Windowed
- Render on UI Elements
- Position and Test in Editor
- Control functions
- Handles aspect ratios...
- and screen sizes



**VLC Media Player
for Unity**

Integrate VLC and Videos into Build or use installed Copy

Play any video from streaming assets, or use a path to any location accessible

Optional: Choose Display Mode

Choose playback options

Optional: Render on UI Elements

Use a Skip-hint: E.g. "Press space to skip"

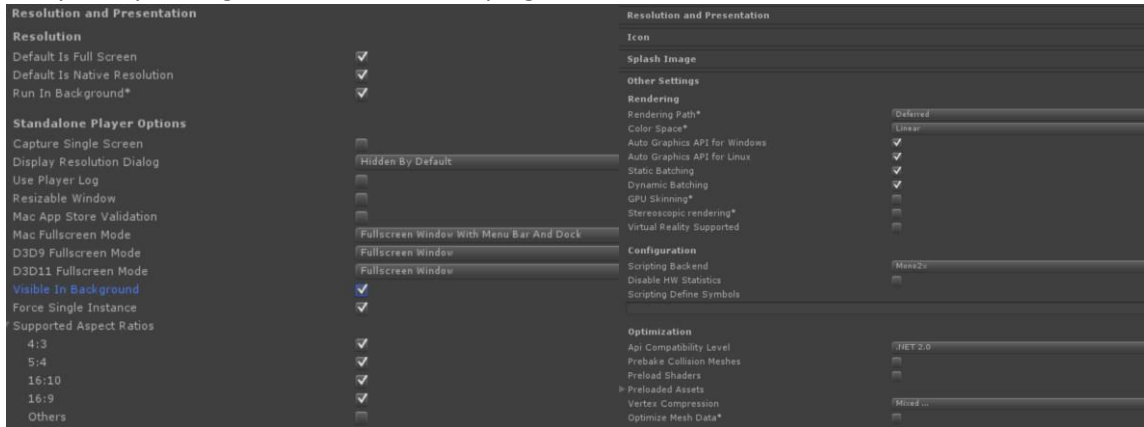
Short setup

Player Settings Requirements

Operating Systems supported (builds and editor): Windows 7, 8, 10. Other operating systems may be supported later on as updates.

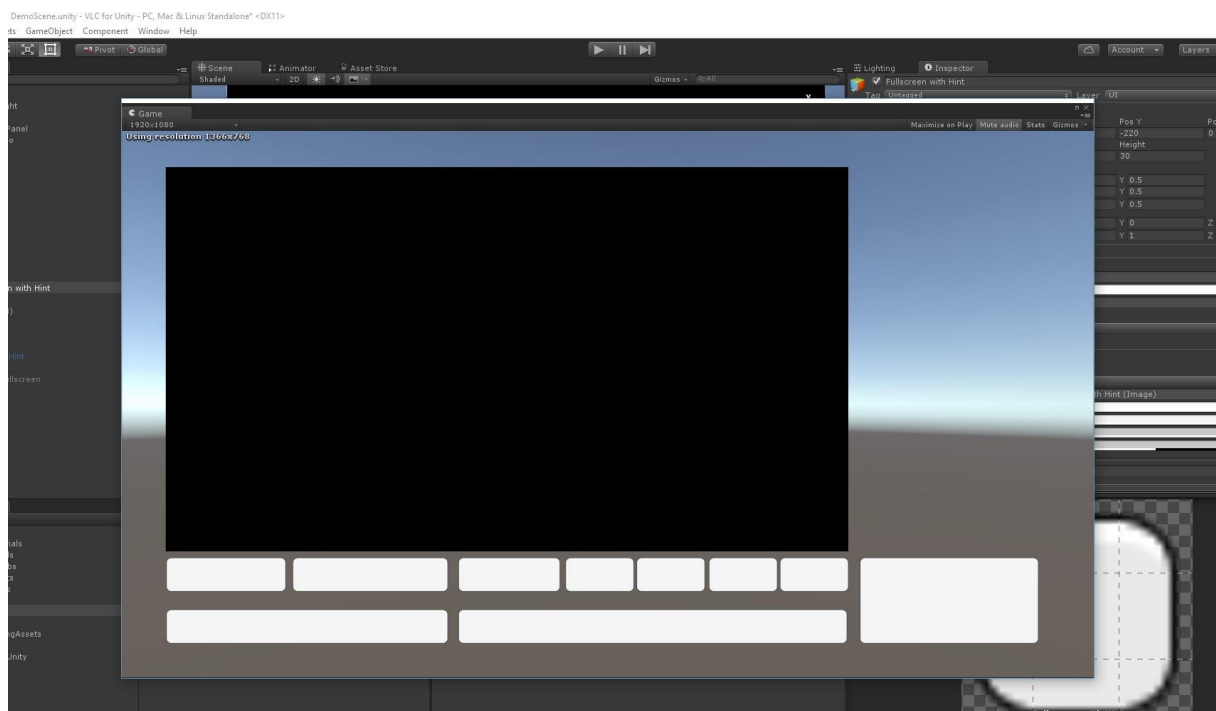
Project Settings: These are the recommended project settings for the plugin.

Particularly important is *Running in Background*, *Visible in Background* and using the *.NET 2.0* API compatibility setting – otherwise needed plugins can't be loaded or built.



Requirements for proper playback in Editor

When testing your videos *windowed on GUI elements* in the Editor, please make sure of the following things. **This does not apply to builds, in your builds everything will look properly ALL THE TIME.**



- Drag the game view from the editor.
- Make sure you set the resolution to **Free Aspect OR..**
- ...if using a **specific resolution**, resize the window so **no grey overflow area is around your game view** in the game window.

If you don't do these steps, the windowed video *might be a bit off in the editor view*. I'll try to fix this issue for the editor, but again: **everything will always be placed exactly right in builds.**

Play a video with VLC for Unity

To simply play a video:

- Assign the PlayVLC.cs Script to any GameObject.
- Make sure "Use Built in VLC" is checked
- Put any video in the StreamingAssets folder if you want to include the video in your build. Write the filename into the "Streaming Assets Video Filename"-Field and check "Play from Streaming Assets" Box
- OR: Play the video from any location and paste the path to the video in the "Video Path"-Field while making sure "Play from Streaming Assets" is unchecked
- Check the "Play On Start" and "Skip Video With Any Key"-Box ☐ Start to play video

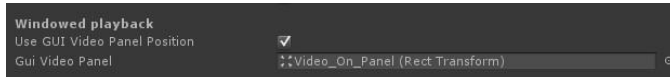
For a quick overview, open the **Demo Scene** that comes with this plugin.

Playing a video on GUI Elements

ATTENTION: Make sure the *pivot is in the upper left corner* or use the prefab that comes with the package. This will be fixed later.

Follow these steps to easily place a video on UI-Panels in Unity:

- Create a UI Panel and put it to the desired location / size
- Assign the PlayVLC.cs script to the Panel or any other GameObject in your scene
- Make sure the **pivot** of the Panel is in the **upper left corner**, or use the Panel prefab that comes with the plugin
- Check these settings below and assign the **RectTransform** of the Panel to the Script



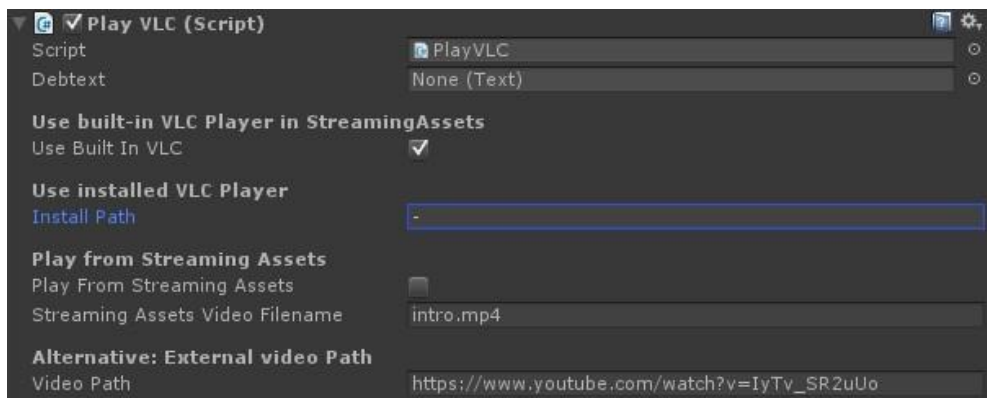
- Optional: Check "Enable VLC Controls" box and use UI Buttons to control the video

Using command line arguments

By using command line options, you can use hundreds of arguments to customize basically anything for your video playback. This is an extremely powerful tool to get exactly the behaviour you want.

All command line arguments can be found here: https://wiki.videolan.org/VLC_command-line_help/

Playing a YouTube Video



Just use a copied YouTube URL and put it into the Video Path field.

Streaming a video to Texture2D

NEW: You can now stream videos to Texture 2D as a free add-on to this plugin.
The add-on is still in BETA, so no warranties! This solution is probably going to be replaced by something better in the future, but it does the trick already.

To stream a video to texture2D, download this free add-on here from my website and put it into your project (so you always have the latest version, asset store updates take quite a long time):

<http://www.christianholzer.com/VLCfU/VLCTextureStream.rar>

- The add-on is in BETA: it is free of charge and separate from the VLC for Unity Plugin! That means: No warranties that everything works, use it entirely at your own risk.
- For GStreamer Libraries and Codecs please check out and be aware of the following page on licensed codecs before using: <https://gstreamer.freedesktop.org/documentation/licensing.html>.
- See the demo scenes on how to stream a single movie or YouTube video to a 3d object and how to stop video playback. Not all features, formats and controls work at the moment!

REQUIREMENTS:

- 64bit Editor
- For playback in editor (in builds it will always work), start Unity in DirectX 9 mode (cmd: ...\\Unity.exe -force-d3d9) or set graphics API to DX9 in player settings

- For builds use x86_64
- If your video doesn't work, try downloading this and place it in your project directory - that should solve it: http://www.christianholzer.com/VLCfU/GStreamerPlugins_x86_64.rar

USAGE AND OPTIONS:

Check the demo scenes on how it works. Put PlayVLC.cs and VLCtoGStreamTextureStream.cs on a GameObject with a renderer or on a UI element. Use the VLCtoGStreamTextureStream.TextureStreamPlay(), TextureStreamStop() to start/stop video playback. You can also check PlayOnStart on PlayVLC.cs.

Experiment with the following options to get the best result. Options for the VLCtoGStreamTextureStream Script are as follows:

- **NoTranscoding:** Whether to disable transcoding for the VLC stream. Requires UseVLCStreamOrYoutube to be true to have an effect. Experiment with this if a video is not working, for YouTube streams or h264 mp4s you can leave this to true.
- **UseVLCStreamOrYoutube:** Use VLC for the streaming to Texture2D, enables you to play YouTube videos on a texture for example. If this is unchecked, you might need the additional GStreamer plugins from here http://www.christianholzer.com/VLCfU/GStreamerPlugins_x86_64.rar and YouTube streams won't work, but it might be faster.
- The video path settings come from the PlayVLC.cs script as in the normal mode of the plugin.

Used open source content:

GST Unity Bridge -- <https://github.com/ua-i2cat/gst-unity-bridge/>

GStreamer Libs -- <https://gstreamer.freedesktop.org/>

Video in Background: Playing a video behind your Scene/Unity Window

UPDATE: You can now also use the stream to Texture2D feature and play your video on a plane behind your scene.

You can play a video behind your actual Unity window, so your UI and 3D Elements will remain visible and intractable, while a video plays in the background (pretty much like in the Fallout4 Main Menu).

ATTENTION: This works only in Builds at the moment!

To do this, check *VideoInBackground* and assign the *VideoInBackgroundCameraPrefab* to the PlayVLC.cs script instance.

Take a look at the “**DemoScene_VideoInBG**” and the “All Settings Overview”-Part of this readme below.

Playing a video with buttons or from script, use video controls

To play a video from script, you can call the *Play()* function of a PlayVLC-Script Instance.

To stop a video, call the stop *StopVideo()* function. The video stops automatically after playing, if looping is set to false. Otherwise you need to call the stop function or allow skipping by users.

If you enable video controls with the “*Enable Vlc Controls*” checkbox, you can additionally call the following functions from scripts or UI elements:

Pause(): pauses the video and resumes it when called again

SeekForwardShort(): jumps 10 seconds ahead

SeekForwardMedium(): jumps 1 minute ahead

SeekForwardLong(): jumps 5 mins ahead

SeekBackwardShort(): jumps 10 seconds back

SeekBackwardMedium(): jumps 1 minute back

SeekBackwardLong(): jumps 5 mins back

ToggleMute(): Turns audio on and off.

VolumeDown(): reduces audio volume by 5%

VolumeUp(): increases audio volume by 5%

NEW COMMAND REQUEST CONTROLS

Here are the new control requests that you can use **to control the video**. They allow for more advanced options, such as setting the time or jumping a specific time forward for example. You can now **retrieve video information and statistics** also. The new control functions can control more than one VLC instance, so you can now **manipulate multiple videos** at the same time.

To use them, you must **enable them in the PlayVLC.cs instance** you are using. In addition, if you want **to control multiple videos at the same time**, you have to **assign each instance a unique port**, so that every VLC instance gets the control requests by its script.

New Info-Functions are:

CR_PrintVideoInfo(): Prints all video parameters to console

CR_GetCurrentTime(): returns current time in seconds

CR_GetVideoDuration(): returns total video duration in s

CR_GetCurrentVolumeLevel(): returns current volume level

CR_GetCurrentPlaybackRate(): returns playback rate (float)

CR_GetCurrentVideoSeekPosition(): get the exact position of the current time in the timeline (useful for a progress bar)

New control functions are:

CR_TogglePlayPause(): Use this instead of standard Pause(), works for multiple instances of VLC

CR_JumpToSecond(int second): Jump to a specific second

CR_NextVideoInList() + PreviousVideoInList(): Navigate through playlist

CR_SetPlayBackspeedMultiplier(float multi): set playback speed, values <1 make videos slower

CR_SetVolumeLevelAbsolute(int val) + SetVolumeAdditive(int val): Control volume, or add values to current volume

CR_ToggleLoopWhilePlaying(): toggle looping

CR_ToggleRandomPlayback: toggle random, only useful for playlist

FYI: VLC/Unity needs permission from firewall to do this.

QuitVLCHelper.cs

When you want to kill all videos, you can use this helper script. Comes in handy if you want to:

- Change a scene while playing a video or multiple videos and you don't stop them yourself
- Quit the app while playing a video or multiple videos and you don't stop them yourself

Just call *QuitAllVideos()* and *QuitApplication()* for this.

All Settings Overview

bool UseBuiltInVLC: Use a special VLC version that will be included in your builds. This is **strongly recommended**, since you can't know if or which VLC your users will have installed.

string InstallPath: Put the install path to vlc.exe here (Most often C:\Program Files\VideoLAN\VLC\vlc.exe). This VLC player will be used if *UseBuiltInVLC* is **false**. Use only for installations where you can install a proper version that you can test yourself (many VLC builds have different issues that could interfere with the Unity Integration).

UPDATE: since 1.03, everything should work OK with all VLC versions installed.

bool PlayFromStreamingAssets: Play videos from StreamingAssets folder. These videos will be copied to your build automatically. Recommended option.

string StreamingAssetsVideoFilename: Name of the file to play – “name.ext”

string[] StreamingAssetsVideoPlaylistItems: Add multiple videos as a playlist – “name.ext”

string[] VideoPaths: If *PlayFromStreamingAssets* is false, use these paths to the video file(s). It can be located anywhere, but must of course be accessible for the user. Use more array items to make a playlist.

bool UseSubtitles Use subtitles?

string StreamingAssetsSubtitle: Name of the subtitle file to use, must be in streaming assets

UsedRenderMode: You can choose from 3 Options

- **Direct3D**: Best option, highly recommended. This is the most performant solution.
- **Fullscreen Overlay on Primary Display**: Works only on primary display. **[In development]**
- **QT Interface**: When you need to use full VLC options with interface. **[In development]**

bool PlayOnStart: Play on startup? For intros.

bool LoopVideo: Loop video. Make sure the user is able to skip the video. If false, video stops automatically.

bool SkipVideoWithAnyKey: Skip videos with any key on your keyboard or button on gamepad. You should use this for fullscreen videos, where you don't have any other possibility to skip. Note: videos skip automatically after playback, except when *LoopVideo* is true.

bool EnableVlcControls: Allows you to call control functions for the last active video. See demo scene and the explanations on the functions on the previous pages.

bool CompleteFullscreen: If this is set to **false**, fullscreen videos will be rendered atop of Unity's window. This means: if you're running in windowed mode, the video will be "fullscreen" atop the game window, but not fullscreen over the whole screen. If this is set to **true**, it will be forced to render across the complete screen. Make sure to activate *SkipVideoWithAnyKey* when using this, especially if *LoopVideo* is true!

bool UseGUIVideoPanelPosition: Render on top of this UI element. Makes it easy to position windowed videos.

bool AudioOnly: Use only audio? Useful for music, for example from YouTube.

bool NoAudio: Disable audio.

GuiVideoPanel: The UI RectTransform that the video will be rendered on. **IMPORTANT**: The pivot must be in the **upper left corner**. Use the prefab that comes with this script if you want. Apply the poster image that you want to use on the Image component of the RectTransform if you wish.

bool ShowBottomSkipHint: Show a hint that tells the user how to skip the cutscene. If true, the BottomSkipHint will be shown.

BottomSkipHint: UI Element that will be shown under the video (you can use the prefab in the package, or use a custom one). See demo scene.

bool FlickerFix: Better performance at video start. Recommended.

bool VideoInBackground: Play video behind your scene. See the readme above.

VideoInBackgroundCameraPrefab: Place the prefab that comes with this package here, or use your own or modified prefab. This camera keys the background from your scene and makes the Unity window transparent there – which shows the video that is rendered beneath. See the demo scene.

bool UseVlc210OrHigher: Since VLC 2.0.8, there are a few bugs that have not yet been fixed by the VLC team. If you want to use a more recent version, like the built-in 2.2.1 in VLC for Unity 1.03, check this box to ensure proper functionality.

bool PinVideo: Pin the video to the UI its rendered upon, or to the Unity window. If you move either, the window will resize and use the best aspect to fill the new window or UI size automatically.

bool EnableControlRequests: Use the new controls (see readme).

int VLCControlPort: Unique port to control each VLC instance. Must be set, different VLC instances must have different ports!

New versions, planned features, Support and Contact

1.06

- Better & more controls and video info functions
- Use all VLC command line args to customize behaviour
- Better playlist support

1.04

- Audio Control: Set Audio Level to desired amount (steps of 5) from 0 to 200 %
- SRT Subtitles
- Stream to Texture2D (BETA!)

1.03

- Pin Video to UI / Unity Window: You can now move/resize the video while playing, not just at video start. Best aspect ratio inside the available space is handled automatically.
- Youtube streaming: just put in the URL
- Audio Controls
- Replaced demo video with Youtube URLs in the scripts to save filesize of the plugin.

1.02

- Better performance at video start ☐ Play video in background.

1.01

- Minor fixes.

Here is what I am working on and what I am planning for the next updates:

- Video Effects
- Add complete folders full of media
- Automatic detection of installed VLC media player and its version

I hope I can get everything working in the near future!

If you have feedback, need help or have ideas what I could improve please write me – I'd like to hear from you! I'll answer as soon as possible.

You can reach me by mail: chunityassets@gmail.com

PS: Check out my other stuff on the Asset Store:

<https://www.assetstore.unity3d.com/en/#!/search/page=1/sortby=popularity/query=publisher:5166>

Thanks again for purchasing,

Christian