UNIVERSAL MEDIA PLAYER

User Manual

Version 1.7

CONTENTS

USER SUPPORT

If you need support or have any questions or suggestions, please contact with me:

- unitydirectionkit@gmail.com (Main)
- unitydirectionkit@outlook.com (Additional)

INTRODUCTION

Universal Media Player (UMP) is cross platform media framework plugin for Unity that based on VLC and FFmpeg native libraries:

Platforms	CPUs	Library	Checked platforms/Graphic API
Windows	x86/x86_64	VLC	Windows 7+ (D3D9, D3D11, OpenGL)
OSX	x86_x64	VLC	10.10 Yosemite+ (OpenGL)
Linux	x86/x86_64/Universal	VLC	Ubuntu 12.04.5 LTS+ (OpenGL)
Android	armeabi-v7a/x86	VLC/Native	Android API level 14+ (Android 4.0+) (OpenGLES 2.0 or 3.0)
iOS	arm64, armv7	FFmpeg	iOS 6+ (OpenGLES 2.0 or Metal)
WebGL (experimental)	x86/x86_64	HTML5 Video	Firefox, Chrome, Opera

Possibilities of current version:

- Fast and flexible video playback (fast native texture updates: **Direct3D9**, **Direct3D11** and **OpenGL**);
- Fully compatible with Unity Editor (in-editor playback for Windows, OSX and Linux platforms);
- Supported Unity "Audio Source" component (works only on Windows, OSX and Linux platforms);
- Supported possibility to use external libVLC libraries (works only on Windows, OSX and Linux platforms);
- Supported possibility to easy switch subtitles (SPU) and audio tracks (works only on Windows, OSX and Linux platforms);
- Supported videos with transparent channel (works only on Windows, OSX and Linux platforms);
- Supported possibility to get pixels (snapshot) of playback video frame;
- Supported "**Youtube**" video playback;
- Supported main video file formats playback: 3GPP, AVI, FLV, SWF, M4V, Matroska, Ogg Video, QuickTime File Format, WebM, Windows Media Video and streaming media protocols: HTTPS, HTTP, HLS, RTSP, RTMP (if you have some additional stream or file format, please write me on my support email and I will check it for you, before you buy my asset:);
- Supported main video player events system: **Opening**, **Buffering**, **Prepared**, **Playing**, **Paused**, **Stopped**, **Ended**, **Error**;

- Supported full logging system from native library in Unity Editor for more debugging possibility with different depth: Warning, Debug, Error (works only on Windows, OSX platforms);
- Supported main video player features, like: play, pause, mute, playback rate, rewind, snapshot and other.

IMPORTANT

Please note that this package contains pre-compiled binaries for libvlc which are licensed under LGPL. Also this package contains not all of the libVLC modules, because some of them are licensed under GPL, so they has removed.

VLC native library source code used in this package is available: Winsows, OSX, Linux.

INSTALLATION GUIDELINE

Import the UMP package from the Asset Store. You should now have a folder named "UniversalMediaPlayer" with the following structure in your Unity project:

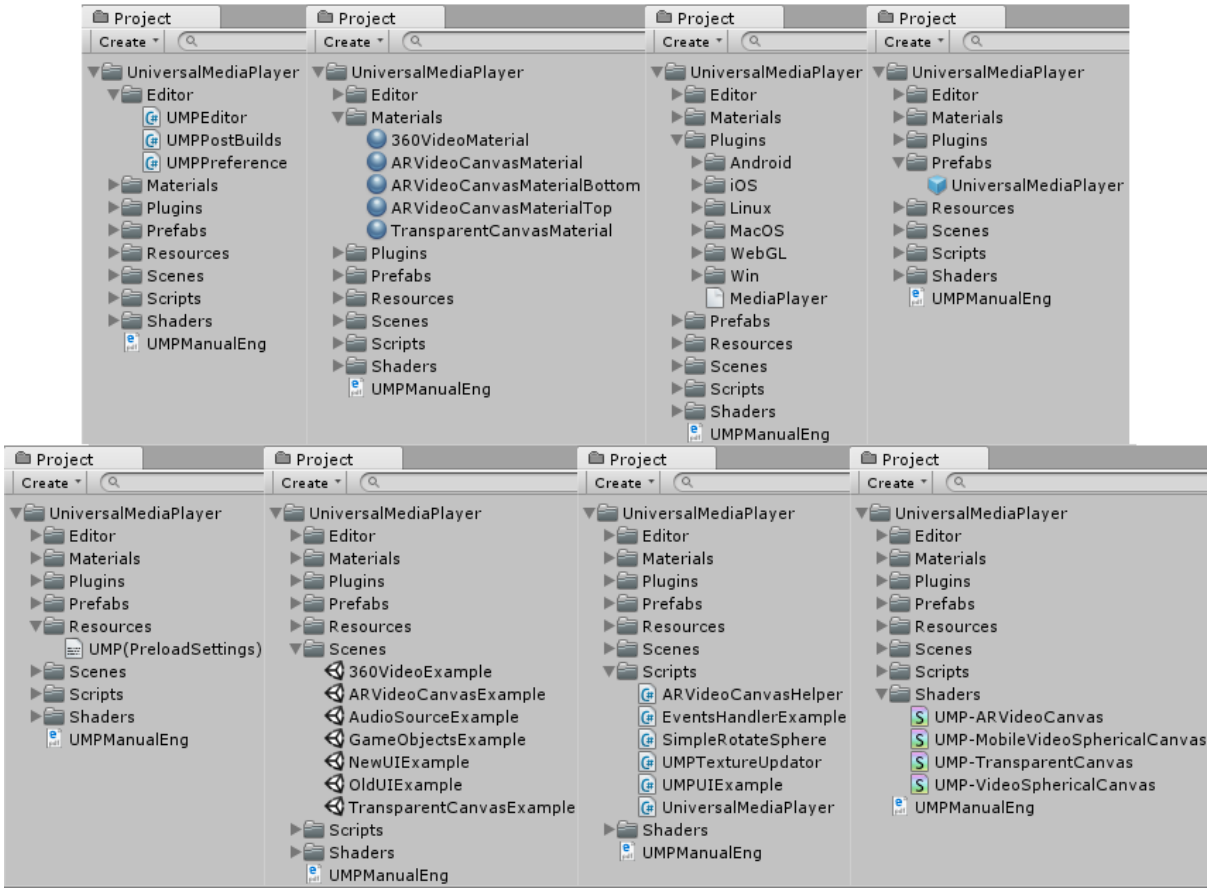

Figure 3.1: Structure of UMP asset

In short:

- **Editor**: Custom Editor, Preferences and Build scripts;
- **Materials**: Custom materials that used in example scenes that based on custom shaders;
- **Plugins**: All the native dlls and shared libraries;
- **Prefabs**: Special UMP prefab that used for easy setup your project with UMP asset;
- **Resources**: Special file where UMP asset stored all preloaded setting information;
- **Scenes**: Demo scenes that's show all components in a ready setup and how work with some additional features of UMP asset;
- **Scripts**: C# classes that show how to work with UMP;
- **Shaders**: Custom shaders that can be used for dynamic aspect handling and for video with transparent channel.

UMP Asset Parts

Full UMP asset consists of two independent parts:

- **UMP (Win, Mac, Linux, WebGL)** – for desktop platforms support;
- **UMP (Android, iOS)** – for mobile platforms support.

So, if you bought only UMP for mobile platforms it's doesn't work in standalone builds (PC, Mac & Linux platforms), but you will have possibility to playback video in Unity Editor – more information in External VLC Player Support subitem. The same with UMP for desktop platforms – you will have possibility to play videos in Unity Editor and standalone builds (PC, Mac & Linux platforms), but it's doesn't work on mobile (Android, iOS) platforms.

To correctly combine two independent parts into one full UMP asset you need:

1. Check if you have the last versions of UMP asset for desktop and mobile platforms;
2. At first, import **UMP (Android, iOS)** part into your project (press "All" and after it "Import" button in "Import Unity Package" window);
3. Import **UMP (Win, Mac, Linux, WebGL)** part into your project (press "All" and after it "Import" button in "Import Unity Package" window);

After all this steps you wil have possibility to use full UMP features in one full asset.

External VLC Player Support

UMP asset has possibility to support external libVLC libraries that will be installed with regular VLC player. It's also give you possibility to playback video in Unity Editor witout internal libVLC libraries that included in UMP asset by default (for example if you don't want to sotore this libraries in git repositories) and you can easily playback videos if you have only UMP (Android, iOS) part of full UMP asset.

To correctly use regular VLC player with UMP asset you need:

1. Download correct version of regular VLC player depending from Unity Editor bit version that you use:
    - Unity Editor (32bit)
    - Unity Editor (64bit)
2. Check if all setup correctly in special UMP menu in Unity Preferences (more information in UMP Preference Item chapter).

Updating Exist Version

When updating an existing UMP installation you should just delete the old "UniversalMediaPlayer" folder before importing the new version of UMP asset. If you had entered the play mode from Unity once in your editor session you have to close Unity and

restart it. Otherwise the native dlls will be cached from Unity and could not be updated properly.

Additional iOS Framework

If you want to load some specific video formats/streams on iOS platform or your video file/stream don't work properly with default UMP framework you can try to download the special "**UniversalMediaPlayer.framework**", that contains all possible supported codecs and replace to old one that located in " UniversalMediaPlayer\Plugins\iOS" folder. This updated framework not contains by default, because he has very big size and in main cases is not needed, but you can download it separately from this link: UniversalMediaPlayer.framework.

USAGE GUIDELINE

UMP Preference Item

It's a special possibility to setup UMP asset with some additional settings. You can find this menu in:

- **Windows**: Edit->Preferences...->UMP
- **OSX**: Unity->Preferences...->UMP
- **Linux**: Edit->Preferences...->UMP

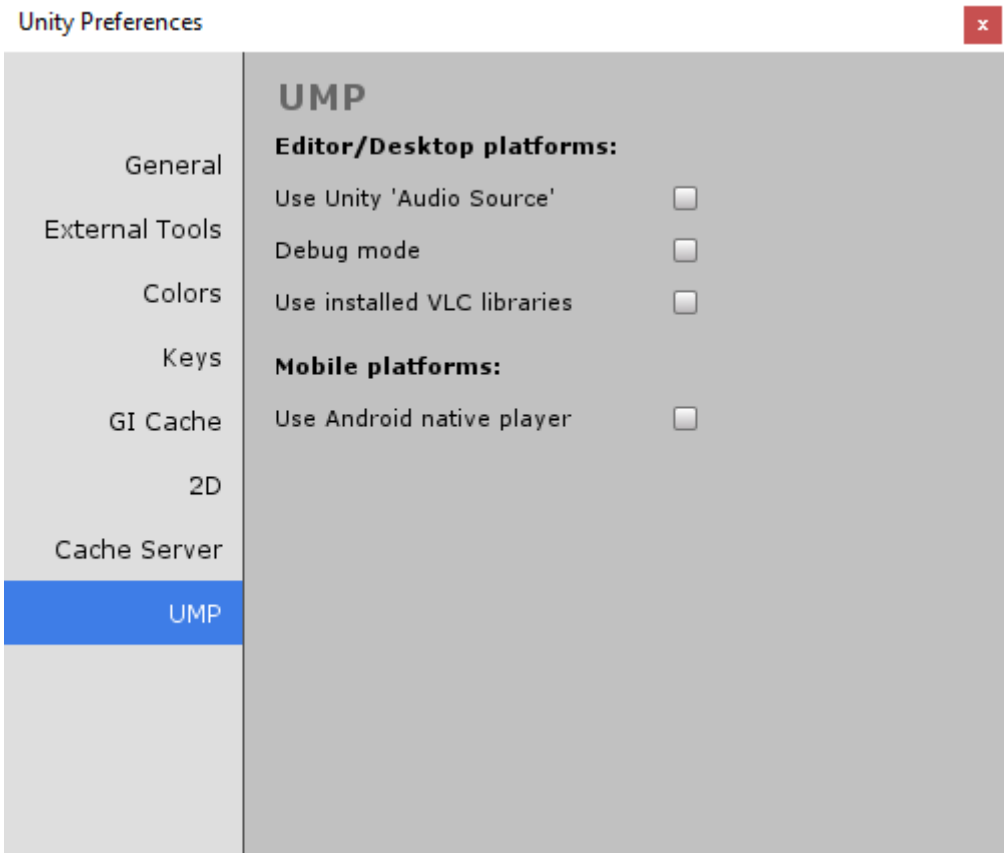

Figure 3.2: UMP additional settings

In short:

- **Use Unity 'Audio Source'**: will be using Unity 'Audio Source' component for audio output for all UMP instances (global) by default (works only on Windows, OSX and Linux platforms);
- **Debug mode**: allows you to use debug mode in your IDE with UMP asset in Unity Editor (it's still experimental and will work only in Unity Editor, also some default UMP functions will not work in this mode, because it's conflicting with IDE debug mode, (works only on Windows, OSX and Linux platforms);

- **Use installed VLC libraries**: will be using external/installed VLC player libraries for all UMP instances (global, works only on Windows, OSX and Linux platforms). Path to install VLC directory will be obtained automatically (you can also setup your custom path):

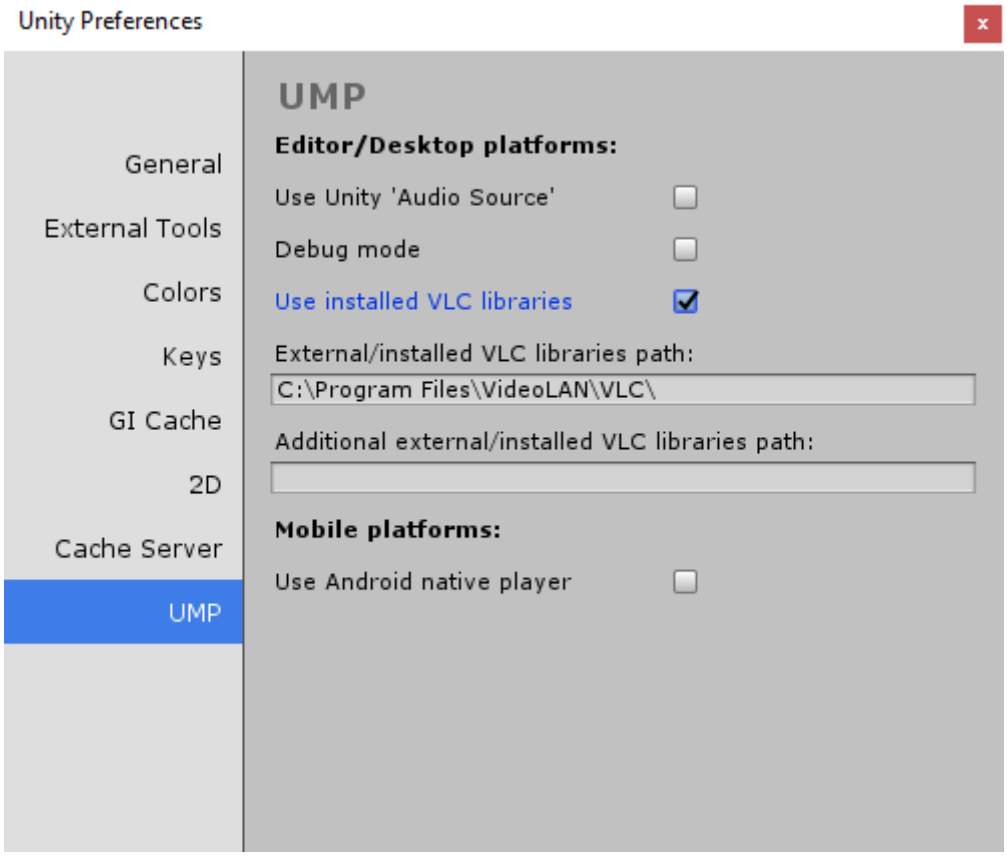

Figure 3.3: UMP "Use installed VLC libraries" setting

  - **External/installed VLC libraries path**: Default path to installed VLC player libraries, it's directory will be obtained automatically if you will have installed regular VLC player;
  - **Additional external/installed VLC libraries path**: Additional path to installed VLC player libraries. Will be used if path to libraries can't be automatically obtained.

- **Use Android native player**: will be using Android native media player for all UMP instances (global, works only for Android platform).
  - **Android native player**: has better compatibility with old devices and better support of playback video files from "StreamingAssets" folder. So, if your videos/stream work with this player it's better to use it by default;
  - **LibVLC player (by default)**: has better codecs support, but low devices support. So, if your videos/stream don't work with Android native player, use this one by default.

UMP Prefab Usage Tutorial

You need to find the special UMP prefab in "Prefabs" folder that give you possibility to manage all media player functionality and move it to your Unity scene:

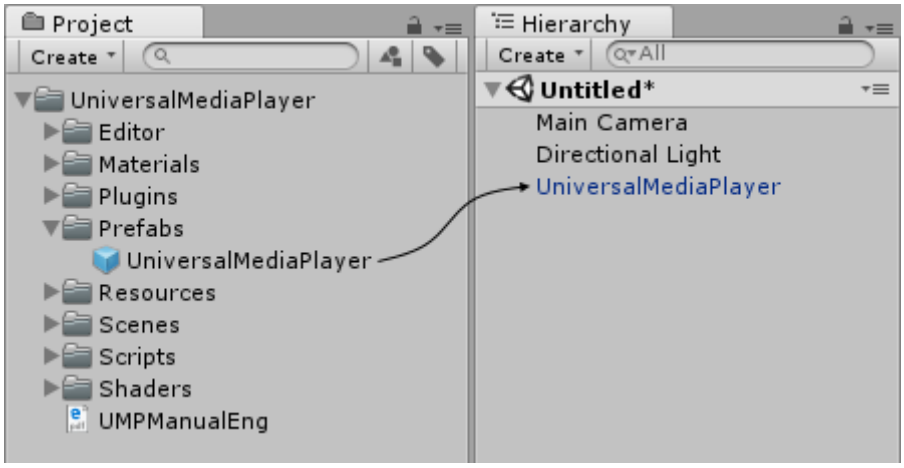

Figure 4.1: add new UMP instance

When you select the UMP instance in your "Hierarchy" window you can manage it with the component inspector:

Figure 4.2: UMP inspector view

In short:

- **Rendering GameObjects**: simple array that consist with Unity "GameObjects" that have "Mesh Renderer" with some material or "Raw Image" component:

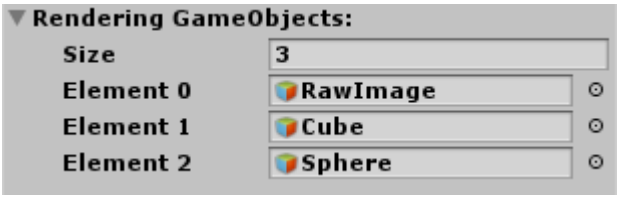

Figure 4.3: UMP rendering objects

- **Audio Sources**: simple array that consist with Unity "Audio Source" components. Gives you possibility to redirect output of current audio track into "Audio Source" component (works only on Windows, OSX and Linux platforms):

Figure 4.4: UMP audio sources

- **Path to video file**: url link or local path to your video. Both of this will be work correctly and you don't need to worry about video file location. For local files you can specify an absolute path with the file:/// sheme on all supported platforms. For remote files or streams you just use your url link. Usage examples:
    - Play video from "StreamingAssets" folder (StreamingAssets\myVideoFile.mp4)- file:///myVideoFile.mp4;
    - Play video from local storage space (C:\MyFolder\Videos\video1.mp4) - file:///C:\MyFolder\Videos\video1.mp4 or C:\MyFolder\Videos\video1.mp4 or file:///DCIM/100ANDRO/MyVideo.mp4 (example for Android platform, so you can ignore root folder – in my case is "/storage/emulated/0/"). Also, if you want to play video on Android platform, don't forget to set 'Write Access' to 'External (SDCard) in your player settings;
    - Play video from remote space (play streams) - rtsp://wowzaec2demo.streamlock.net/vod/mp4:BigBuckBunny_115k.mov.
- **AutoPlay**: start playback automatically after video is buffered;
- **Looping**: When the playback reaches the end position it jumps to the start and plays again. Also, you have possibility to use "**Smooth Loop**":

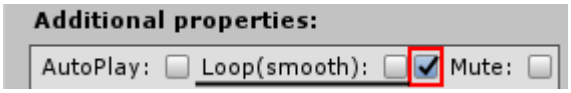

Figure 4.5: UMP smooth loop

Differences between this features is:

- **Loop**: will be have buffering stage, but use lees devices resources;
- **Loop(smooth)**: don't have buffering stage, but use more devices resources.

- **Mute**: set mute status for current video playback;
- **Advanced**: special properties that give you possibility to have additional setup for your video playback (file, live capture, disc and network caching):

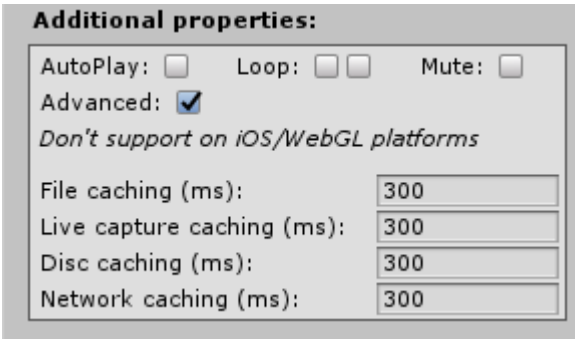

Figure 4.6: UMP advanced properties

So, if you will have not smooth playback (included some freezes) in your desktop platforms or you want to decrease buffering time, just try to change this properties, it's can help to solve your issue.

- **Volume**: set current software audio volume (by default you can change this value from "0" to "100");
- **Play rate**: set the requested movie play rate (by default you can change this value from "0.5" to "5", also it's don't support negative playback).
- **Position**: set movie position. This has no effect if playback is not enabled. This might not work depending on the underlying input format and protocol;
- **Load**: prepare video playback (wait, until we can't get first frame of current video);
- **Play**: start video playback;
- **Pause**: if a video is playing you can pause it. To continue playback you should press "Play" button again;
- **Stop**: stops the media and seeks to the first frame. You need to press "Play" button to start playback again;
- **Shot**: Take a snapshot of the current video playback and save it by default in "Application.persistentDataPath" folder (works only on Windows, OSX and Linux platforms);

- **Log Detail**: gives you possibility to getting log messages from native library (works only on Windows and OSX platforms):

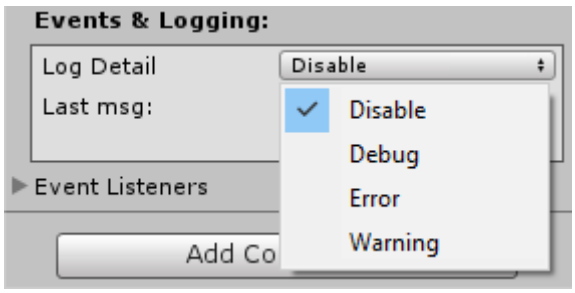

Figure 4.7: UMP log detail filters

- **Last msg**: display last playback event or playback error;
- **Event listeners**: gives possibility to attach to the special callbacks (Opening, Buffering, Playing, Paused, Stopped, End Reached, Encountered Error, Time Changed, Position Changed, Snapshot events):

Figure 4.8: UMP log detail filters

UMP Patches

The newest versions of UMP asset for desktop platforms start using the latest versions of libVLC libraries, in short - “nightly builds”. This libVLC libraries can be unstable and sometime will be work worse than released libraries, but they contains many fixed issues with support some new codecs and other important implementations, from the latest release of stable versions of libVLC libraries (the latest released version in current time is 2.2.4 version). So, by default UMP asset will be contains the latest “nightly builds” libVLC libraries, but you have possibility to use the stable released libraries, just download this UMP patch and import into your project: UMP Stable Libraries (2.2.4).

Linux Platform Setup

INSTRUCTION FOR STANDALONE BUILD

If you don't have installed regular VLC player (or don't wan’t install it) on your Linux OS, you need to make additional installation of some packages that gives you possibility to have correct video playback. Without this additional installation, UMP asset probably will not work properly on your Linux OS.

So, first of all, please add "Execute" permission to special UMP shell scripts "UMPLauncher.sh" and "UMPRemover.sh" - this scripts should be generated automatically when you complete building process of your Unity project. They give you possibility to automate installing/removing of necessary additional packages:

- Right click on the file -> Permissions -> Allow executing file as program:

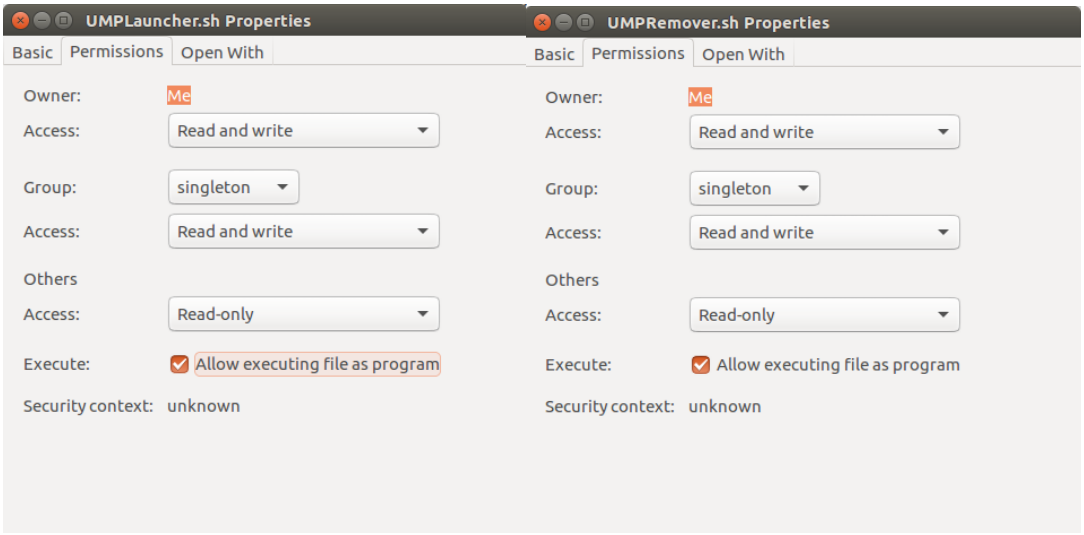

Figure 4.9: UMP setup special Linux shell scripts

Open terminal and go to the folder where you extract your build or make "Right click in build folder window" and click to the "Open in Terminal" option:

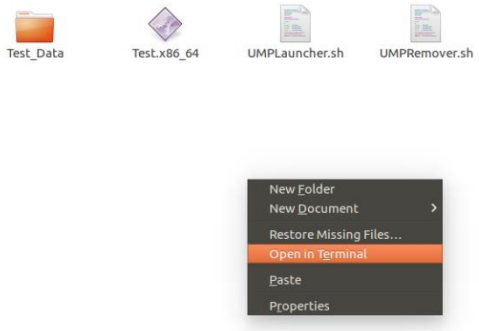

Figure 4.10: Open "Terminal" in build folder

In terminal you need to run special UMP shell script:

- To correct launch and install necessary packages: **./UMPLauncher.sh**
- To correct remove installed packages: **./UMPRemover.sh**

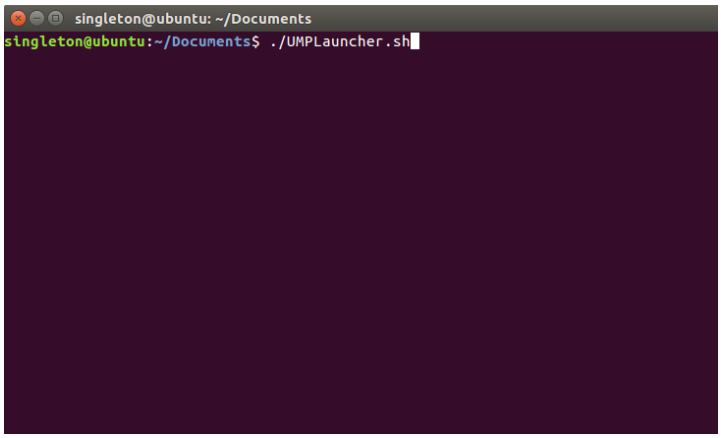

Figure 4.11: Command example in "Terminal"

If you have some problems to correctly run UMP shell scripts, please open it with default file editor and press "Save" button or "Ctrl+S" and try again. After first correct launch in future you can simply run your executable file without any additional manipulation.

INSTRUCTION FOR UNITY EDITOR

At first you need to find the special UMP shell script “UMPEditor.sh” in your UMP asset in this folder “UniversalMediaPlayer\Plugins\Linux” and add "Execute" permission like in instruction for standalone build. After it, open this file and change: ‘Path_to_your_libvlc_libraries_x86_64’ and ‘Path_to_your_libvlc_libraries_x86’ to the path to your current project where you store all libVLC libraries, example:

‘/home/pcname/Documents/MyProject/Assets/UniversalMediaPlayer/Plugins/Linux/x86_64’.

When you do this, run it in the same way like you do it in instruction for standalone build.

IOS Platform Setup

When you completely export iOS project from Unity Engine, please check if all needed frameworks is available in "Build Phases" tab in "Link Binary With Libraries" phase:

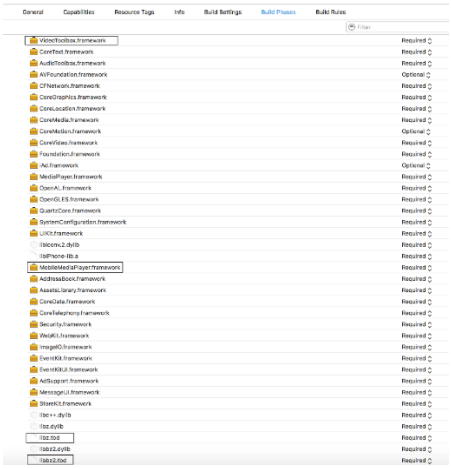

Figure 4.12: iOS build frameworks

If this frameworks and libraries (VideoToolbox, libz, libbz2) don't exist in your list, please add them manually or try to make Unity build.

FAQ

What shader can I use?

You can use every shader that uses a texture property.

Can I change stream buffering size for iOS platform

Yes you can do this by editing "UniversalMediaPlayer.mm" file that exist in "Libraries/Plugins/iOS/" and add new option "max-buffer-size" when create new media player object in "initMediaPlayer" method:

#define MAX_QUEUE_SIZE (15 * 1024 * 1024)

//...

[options setPlayerOptionIntValue:1 forKey:@"start-on-prepared"];

[options setPlayerOptionIntValue:1 forKey:@"videotoolbox"];

[options setPlayerOptionIntValue:MAX_QUEUE_SIZE forKey:@"max-buffer-size"];

_mediaPlayer = [[FFMoviePlayerController alloc] initWithOptions:options];

All possible options you can find in this file: Additional Options

Can't run Linux application correctly, what I am doing wrong?

If you see this message when you try to launch your build:

Could not display “appname”.

There is no application installed for “executable” files. Do you want to search for an application to open this file?

You need to make the file executable, just right click on the file -> Properties -> Permissions -> Allow executing file as program or from terminal: chmod +x appname

How to set up the audio output for Oculus Rift Headphones?

When you create new instance of media player object, just add additional defined argument to set new audio output device:

```
        _mediaPlayer = new MediaPlayer(this, _renderingObjects, new
PlayerArguments(null)
        {
            AudioOutputDevice = PlayerArguments.GetAudioOutputDevice("Rift Audio")
        });
```

Why is my video playing in the editor but not on my mobile device?

Every platform has is own restrictions in terms of video codecs or audio formats. So it is most likely a problem of your video encoding.

How can I setup my custom GameObjecs with UMP (for example NGUI component):

You can use special callback to get video texture that will be created for current video and make some additional manipulation with it (how to use it you can find in “UniversalMediaPlayer.cs” script):

```
public void OnPlayerPrepared(Texture2D videoTexture)
{
    //apply video output texture to NGUI component
}
```

Can I remove certain codecs such as .mpg (or any other that don't be used) from "Plugins" folder to decrease size on application?

Yes, you can remove some of them. For example you can run video that you want to support in your project and try to delete all "plugins" folder - so, all .dlls that used in current video playback you can't delete, but other .dlls will be deleted.

Is there any way to specify the resolution of the youtube videos?

You have possibility to get video with resolution that you want, just use this function when you get youtube video link from youtube service:

```
var videoInfo = _videoHostingParser.GetBestQualityVideo(videoInfos, 240);//240, 360, 480, 720, 1080...
```

CLASSES/METHODS DESCRIPTION

Classes

- **MediaPlayer** – media player class;
- **MediaPlayerHelper** – adds possibility to get some additional stuff from media player;
- **DefinedArgs** – used to setup new media player instance with some additional parameters (advanced usage).

Interfaces

- IMediaListener (combine all main video playback interfaces)
- IPlayerOpeningListener: void OnPlayerOpening()
- IPlayerBufferingListener: void OnPlayerBuffering(float percentage)
- IPlayerPreparedListener: void OnPlayerPrepared (UnityEngine.Texture2D videoTexture)
- IPlayerPlayingListener: void OnPlayerPlaying()
- IPlayerPausedListener: void OnPlayerPaused()
- IPlayerStoppedListener: void OnPlayerStopped()
- IPlayerEndReachedListener: void OnPlayerEndReached()
- IPlayerEncounteredErrorListener: void OnPlayerEncounteredError()
- IPlayerTimeChangedListener: void OnPlayerTimeChanged(long time)
- IPlayerPositionChangedListener: void OnPlayerPositionChanged(float position)
- IPlayerSnapshotTakenListener: void OnPlayerSnapshotTaken(string path)
- ILogMessageListener: void OnLogMassage(LogMessage message)

MediaPlayer

```
public MediaPlayer(MonoBehaviour monoObject, GameObject[] videoOutputObjects)
```

- create and initialize a MediaPlayer instance:
    - monoObject – MonoBehaviour instanse
    - videoOutputObjects – Objects that will be rendering video output

---

```
public MediaPlayer(MonoBehaviour monoObject, GameObject[] videoOutputObjects, PlayerArguments playerArgs)
```

- create and initialize a MediaPlayer instance with special arguments:
    - monoObject – MonoBehaviour instanse

- videoOutputObjects – Objects that will be rendering video output
- playerArgs - Additional player arguments (more information about arguments, you can find by clicking the link)

```
public void AddMediaListener(IMediaListener listener)
```

- add to MediaPlayer new main group of listeners

```
public void RemoveMediaListener(IMediaListener listener)
```

- remove from MediaPlayer the main group of listeners

```
public Uri DataSource { get; set; }
```

- get/set path to video source that will be use do play

```
public bool SetSubtitleFile(Uri path)
```

- set new video subtitle file
  - path – path to new video subtitle file

```
public GameObject[] VideoOutputObjects {get; set; }
```

- objects that will be rendering video output

```
public PlayerManagerEvents EventManager { get; }
```

- adds possibility to attach/detach of media player listeners

---

```
public PlayerState State { get; }
```

- get current video playback state

```
public int VideoWidth { get; }
```

- get current video width in pixels

```
public int VideoHeight { get; }
```

- get current video height in pixels

```
public Vector2 VideoSize { get; }
```

- get the pixel dimensions of a video:
    - new Vector2(the video width in pixels, the video height in pixels)

```
public byte[] FramePixels { get; }
```

- get pixels of current frame

```
public float Fps { get; }
```

- get movie fps rate:
    - frames per second (fps) for this playing movie

```
public long FrameCount { get; }
```

- get movie frame counter

```
public bool Prepare()
```

- prepare video playback

```
public bool Play()
```

- play or resume:
    - true, if playback started (and was already started), or false on error

---

```
public void Pause()
```

- toggle pause (no effect if there is no media)

```
public void Stop()
```

- stop (no effect if there is no media)

```
public void Stop(bool resetTexture)
```

- stop (no effect if there is no media):
    - resetTexture – clear the last frame of current movie. Used for looping video playback

---

```
public void Release()
```

- release a MediaPlayer instance

```
public long Length { get; }
```

- get the current movie length (in ms)

```
public string GetFormattedLength(bool detail)
```

- get the current movie formatted length (hh:mm:ss[:ms]):
  - detail – is formatted length will be with [:ms]

```
public MediaTrackInfo[] SpuTracks { get; }
```

- gets the available spu tracks

```
public MediaTrackInfo SpuTrack { get; set; }
```

- get/set the current spu track

```
public MediaTrackInfo[] AudioTracks { get; }
```

- gets the available audio tracks

```
public MediaTrackInfo AudioTrack { get; set; }
```

- get/set the current audio track

```
public int Volume {get; set; }
```

- get/set current software audio volume:
  - the software volume in percents (0 = mute, 100 = nominal / 0dB)

```
public bool Mute { get; set; }
```

- get/set current mute status:
  - the mute status

```
public long Time { get; set; }
```

- get/set the current movie time (in ms). This has no effect if no media is being played. Not all formats and protocols support this:
  - the movie time (in ms), or 0 if there is no media

```
public float Position { get; set; }
```

- get/set movie position. This has no effect if playback is not enabled. This might not work depending on the underlying input format and protocol:
    - movie position, or 0 in case of error

```
public bool IsPlaying { get; }
```

- is media is currently playing:
    - true if the player is playing, false otherwise

```
public bool IsReady { get; }
```

- is media is ready to play (first frame available):
    - true if the player is playing, false otherwise

```
public bool AbleToPlay { get; }
```

- is the player able to play:
    - true if the player is able to play, false otherwise

```
public float PlaybackRate { get; set; }
```

- get/set the requested movie play rate

```
public object PlatformPlayer { get; }
```

- get player for current running platform

```
public float BufferingValue {get; }
```

- get buffering value for current video playback

MediaPlayerHelper

```
public static void ApplyTextureToRenderingObjects(Texture2D texture, GameObject[] renderingObjects)
```

- apply texture to Unity game objects that has 'RawImage' or 'MeshRenderer' component:
    - texture - texture video output
    - renderingObjects - Game objects where will be rendering video output

---

```
public static Color GetAverageColor(byte[] frameBuffer)
```

- getting average color from frame buffer array (FramePixels)

```
public static Color32[] GetFrameColors(byte[] frameBuffer)
```

- Getting colors from frame buffer array