MPMP Manual

vers. 1.4

monoflow
new media experience

Contents

Chapter 1

Introduction

- **MPMP** (Multi Platform Media Player) is a high performance cross platform media player for Windows, Android and iOS/OSX.
  To get the best performance on each platform it uses different media frameworks.

    - **Windows: Media Foundation**

    - **Android: Media Player**

    - **iOS/OSX: AVFoundation**

- In Unity you have only a unified C# interface you work with, so you don't have to deal with all the platform differences. **MPMP** is a media player but at the moment it is only for playing video files. Depending on the platform it can play different kind of video formats. You have to look into the documentation of the media frameworks to check if your video is encoded in the right format.

Chapter 2

System Requirements

- **Unity** 5.2.2 or higher (Best is to use 5.3+ when on OSX)

- **Windows:**

  - Windows **8+**
  - **DX11 only!**
  - Visual C++ Redistributable Packages for Visual Studio 2013
    (https://www.microsoft.com/en-us/download/details.aspx?id=40784)
    (MPMP provides the installers in the vcredist folder)

- **Android:**

  - Android API level 15+ (Android 4.03+)
  - armeabi-v7a or x86 CPU
  - OpenGLES 2.0 as Graphic API

- **iOS:**

  - iOS 6+
  - OpenGLES 2.0 as Graphic API

- **OSX:**

  - 64 bit system

Chapter 3

Installation

1. Import the MPMP package from the Assetstore. You should now have a folder named MPMP with the following structure in your Unity project:
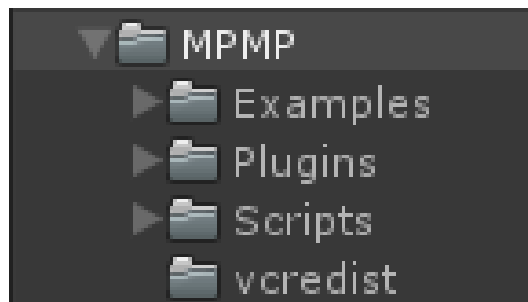
Figure 3.1: package structure

- Examples: MPMP comes with a demo scene to show all components in a ready setup.
- Plugins: All the native dlls and shared libraries
- Scripts: All the C# code
- vcredist: Installers from Microsoft (Visual C++ Redistributable Packages for Visual Studio 2013)

2. After installation you should see a menu item under ' GameObject/Create Others/MPMP. If there is no MPMP menu item you have to reimport the unitypackage or close & reopen Unity.

3. If you working on the Windows platform you need to install the Visual C++ Redistributable Packages for Visual Studio 2013 We provide the installers from Microsoft in the vcredist folder. If you don't want to install the full Redistributable Packages you need at least following dlls located beside your exe:
msvcp120.dll, msvcr120.dll, vcamp120.dll and vccorlib120.dll

Chapter 4

Usage

4.1   Setup

1. To play a video with MPMP you have to add an instance of the MPMP player to your scene. Just select the menu item under GameObject/Create Others/MPMP.

2. When you select the MPMP instance you can manage it with the component inspector.
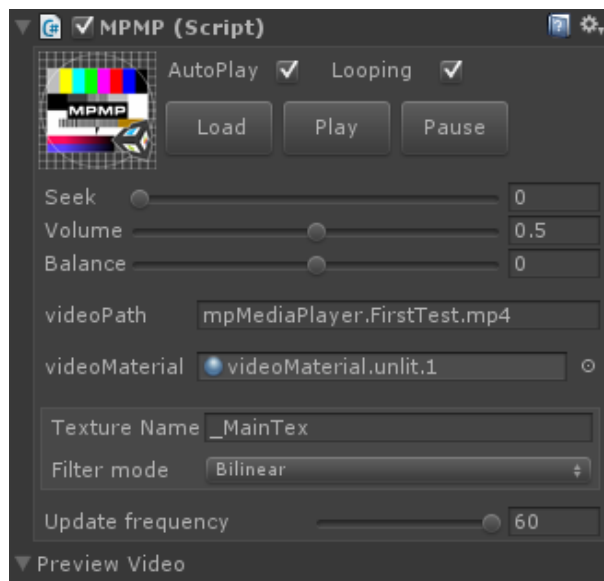
Figure 4.1: MPMP inspector

The inspector provides all the elements to interact with your media in the editor

- Load: Before you can play a media file/stream you have to load it into the player. MPMP loads the media file from the location you specified in the videoPath field.

- Play: When a media file is ready for playing(loaded) you can start playing it.

- Pause: If a media file is playing you can pause it. If you want to resume it just hit the play button again.

- Seek: You can seek to a position into the media file.(doesn't work on live streams). The time you seek the media file is paused.

- Autoplay: The media file starts to play automatically after it is loaded.

- Looping: When the media file reaches the end position it jumps to the start and plays again.

3. **videoPath:** the path/url to your video.
   You can play local files, remote files(progressive streaming) and streaming video (Media Foundation doesn't seems to support streaming right out of the box). You can change the content of MPMP at every time. Just change the videoPath and trigger a new loading. For local files you can specifiy an absolute path with the `file://` sheme or without a sheme (the StreamingAssets folder is the root) For remote files or streams you just use `http://`*yourURL* . Depending on you platform you can use progressiv streaming.(Your video file must be exported with the faststart option.) Examples:

   • myVideo.mp4 => StreamingAssetFolder/myVideo.mp4

   • file://C:/Folder1/myVideo.mp4 => C:/Folder1/myVideo.mp4

   • C:/Folder1/myVideo.mp4 => C:/Folder1/myVideo.mp4

   • http://www.myURL.com/Folder1/myVideo.mp4

4. **videoMaterial:**
   Here you specify a material that should display the texture of the video (the shader needs a '_MainTex' property!). At runtime the video is rendered into the materials mainTexture. Keep in mind that except of Android the raw video texture is flipped on the y-axis. We provide a script that flips the localScale.y of a gameobject (MPMPScaleFlipY.cs) on the affected platforms so you don't need to write your own flip script.

5. **Texture Name:**
   Name of the texture property within the materials shader, default is _MainTex to support almost every shader that comes with Unity. Change this only if you use some custom shader. For example if you have a shader that uses several video textures you can use multiple MPMP instances that have the same video material but every instance uses another texture name.

6. **Filter Mode:**(Windows only)
   You can choose between Point and Bilinear texture filtering. Bilinear filtering gives better visual result when looking from flat angles onto the video texture or when you scale your video.

7. **Update frequency:**
   You can adjust the interval how often the native plugin should update the texture.Default is 60 but in low performance situations you can lower the frequency to 30 without a huge visual impact.

8. **Preview Video:**
   When playing a video you can watch a little preview image of the video here.
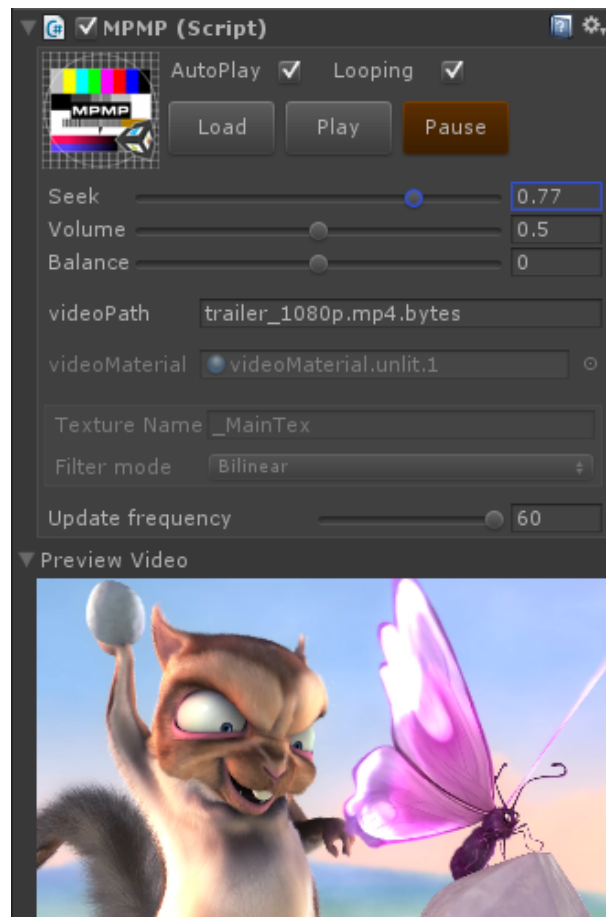
Figure 4.2: MPMP inspector

4.2 UGUI

MPMP comes with a preconfigured UGUI Prefab that you can use to control a MPMP instance at runtime.(MP↩
MP/Examples/GUI/MPMP.ugui) We provide a class MPMP_ugui_Element.cs that you can use for your own UGUI
elements. (MPMP/Scripts/GUI) As every instance of this class need a reference to a MPMP player you can use the
MPMP_ugui_Host.cs class in a parent gameobject as a single point to set a reference to a MPMP player. All child
gameobjects that are a MPMP_ugui_Element will inherit the MPMP reference from it.
Depending on the functionality you have to add the MPMP_ugui_Element to:

- Button (LOAD,PLAY,PAUSE,AUTOPLAY)

- Slider (SEEK,VOLUME,PLAYBACKRATE)

- RawTexture (TEXTURE)

- Text (TIME)

- InputField (PATH)

4.3 Miscellaneous

- **C# API**
  All the properties of the MPMP class are accessible via c#. For a small overview how you can work with

MPMP in your C# scripts we provide the MPMP_APITest.cs script. For more information you can read the Class Documentation chapter.

- **Events**
  The MPMP instance has events so you can add some callback methods. For example when the OnLoaded event is called you can check the new video size and can rescale your gameobject. For more information you can read the Class Documentation chapter.

- **Seeking**
  Seeking accuracy heavy depends on the codec and platform you use. The media framework only can seek exactly to keyframes. So you should test your video with different encoding settings to get the best results.(For example H264 is not the best codec for good seeking). When encoding your video you should make as much keyframes as possible. But this is always a difficult tradeoff between quality and filesize.
  On iOS/OSX there is a second seek mode available that specifies a tolerance value (default 0) for more precise seeking but could cause some decoding delay. To enable this seek mode you have to enable the SEEK_TOLERANCE definition at the top of the MPMP_API.cs script.

- **Graphics API**
  MPMP requires on mobile platforms OpenGLES 2.0 as Graphics API. When you publish your app and you forgot to adjust your player settings MPMP will automatically switch to OpenGLES 2.0 and write a warning to the console.

- **360° panorama viewer (Monoscopic/Stereoscopic)**
  MPMP comes with two demo scenes that has a camera setup for 360° panorama videos. You can use different layouts (side by side or top/bottom) for stereoscopic video depending on your video footage. For each eye a camera is inside a sphere that is textured with the video. To look around just rotate both cameras parent gameobject.

- **Preparing your videos for progressiv streaming**
  There are several ways to prepare your video to use progressiv streaming. The easiest is to use the Quicktime **PRO**.

  1. Open your movie with the Quicktime player **Pro**
  2. Choose File -> Export.
  3. Choose "Movie to QuickTime Movie" from the Export pop-up menu.
  4. Click Options and select video and sound compression options appropriate for web delivery.
  5. Make sure the "Prepare for Internet Streaming" checkbox is selected and Fast Start appears in the pop-up menu.

If you have already mp4 H.264 videos and don't want to re-encode them you could use the program **MP4 FastStart** to adjust the MP4's metadata.

Chapter 5

FAQ

- What shader can i use?
  You can use every shader that uses a texture. MPMP uses in default the '_MainTex' property but you can change in the MPMP inspector the name of the texture property you want to use.

- Can I use MPMP on Windows 7 or Vista?
  At the moment we only support Windows 8+ , but we are working on backward compatibility.

- Can i use APK Expansion Files on Android?
  Yes. MPMP checks if the Unity app is an obb file and loads the data in a different way. You can leave your videos in the StreamingAssets folder of Unity before you split your APK at built time.

- Can i use 4K movies?
  Yes but the performance depends on your graphics card. On older cards it is possible that there is only a CPU fallback (Windows). We tested a 4K movie on Windows 8.1 with a ATI 7850 (only CPU fallback) and with a NVIDIA GTX 970 (full GPU acceleration)

- 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. (For example at our tests on Android we had problems with a video that uses mp3 audio compression)

- I don't want to install the full redistributable packages on the target machine. What should i do?
  If you don't want to install the full redistributable packages you need at least following dlls from the packages: msvcp120.dll, msvcr120.dll, vcamp120.dll and vccorlib120.dll
  These dlls have to be located beside the main EXE of the application you publish

- Can i use MPMP for 360 videos or VR?
  Yes. It's not a problem of MPMP but more a problem of creating the right video footage and have a mesh with the right uv. We provide a 360° demo scene with a sphere that has inverted normals and and the right uv mapping. You only have to place your camera inside the sphere to watch the video. In the Links section we provide some links to video footage you can use.

Chapter 6

Known Issues

- There are some platform differences in terms of how remote files could be played or how they are cached on the system. Please check the video behaviour on all platforms you use. Some features don't work exactly the same way in the editor like on mobile platforms.

- On Android there is an issue with running multiple instances of MPMP at the same time and using streaming video. The native Media Player reacts on errors when your stream is disrupted. The problem is that all instances of the Media Player receive in this case a MEDIA_ERROR_SERVER_DIED error. This forces MPMP to reload the current media so all current instances of MPMP reload their content.

- Audio panning on iOS/OSX doesn't work

- Video loading on OSX 10.11 El Capitan sometimes hangs. AVFoundation on EL Capitan has some bugs so we don't recommend it at this time. We have added a small hack so if there arise a pixelbuffer error on loading we force a reload.

- On some graphics cards there could be a crash on application quit (Windows standalone). We provide a utility script where we delay the OnApplicationQuit event with a cleanup routine to shutdown all MPMP instances before the application closes.

- On NVIDIA cards the loading of a new video forces MPMP to destroy the actual video texture. So you will notice a short flicker.

Chapter 7

History

Version 1.4 - 2016.03.12

- Added an option for adjusting the refresh interval of the native texture update

- linear color space on NVIDIA cards is now supported

- Loading a new video with new dimensions don't cause a crash anymore on NVIDIA cards (Windows)

- API calls from mediaPlayer uGUI don't call the plugin directly (crash on ATI cards fix)

- Pause before load fix (Windows)

- Critical Section now with higher spincount to improve stability (Windows)

- SaveAndLoad method has now new action parameter for tracing the download progress

- Changed the initialisation of MPMP to Awake. Also added a script execution order manager that forces the MPMP.cs to be executed earlier.

- MPMP_DelayQuit.cs script for shutdown all MPMP instances before your application quits. (fixes a possible crash on quit)

- New Events:
  OnPixelbufferError (OSX/iOS) forces a reload for fixing some possible video refresh problem on OSX El Capitan.
  OnTextureChanged: called when the internal texture has changed the dimensions.(Windows,Android)

- API addition: GetVideoMaterial, SetUpdateFrequency, GetUpdateFrequency

- Documentation update

Version 1.3 - 2016.02.17

- video texture is now displayed correct in linear color space

- Texture filter mode(Point/Bilinear) option for Windows

- You can now specify the texture property name that should be used in the video texture.
  So every custom shader should now work without tweaking the MPMP code by hand.(default is '_MainTex')

- Added a time ugui element for displaying the position and duration in seconds

- Added a path ugui element for managing the video path

- Added a 360° demo scene for watching stereoscopic panorama videos

- Fixed exception bug on OSX when Windows is the target platform and you run the scene in the editor

- Added a log warning and auto switch to x86_64 when publishing for OSX

- Documentation update

Version 1.2 - 2016.02.11

- Events are now called from the main thread

- SetPlaybackRate implemented :
  You can change the playback speed of the video.(negative values for reverse playback)
  On Android you need API level 23+ (Android 6+)

- OnInit event implemented

- Fixed issue with OnError events on Windows

- SeekToWithTolerance for iOS/OSX :
  This seek mode is more accurate but could cause some decoding delay

- Fixed issue with GetDuration on Android and iOS/OSX. The values are now available at OnLoaded in the right unit (seconds).

- ScaleFlipY.cs renamed to ScaleFlip.cs. The script has now the axis mode property for selecting which axis to flip

- Added a 360° demo scene for watching panorama videos

- Documentation update

Version 1.1 - 2016.02.04

- Events implemented :
  OnLoad, OnLoaded, OnPlay, OnPause, OnError, OnDestroy, OnPlaybackCompleted

- Fixed OnApplicationPaused issue. The native media player pauses now when this event is called

- API test is now in a seperate scene

- API test improved

- Fixed issue with debug dll reference on Windows x86_64

- Fixed issue with missing zip_file.jar on Android

- Fixed issue with seeking problem when using AwesomePlayer on Android

- Documentation update

Version 1.0 - 2016.01.13

- Initial release

Chapter 8

Links

Platform specific supported media formats

- **Windows**: https://msdn.microsoft.com/en-us/library/windows/desktop/dd757927(v=vs.↩ 85).aspx

- **Android**: http://developer.android.com/guide/appendix/media-formats.html

- **iOS**: https://developer.apple.com/library/ios/documentation/Miscellaneous/↩ Conceptual/iPhoneOSTechOverview/MediaLayer/MediaLayer.html

Good overview over different formats: https://en.wikipedia.org/wiki/Comparison_of_video↩ _container_formats

Example 360° video footage: http://www.360heros.com/vr/

MP4 FastStart: http://www.datagoround.com/lab/

Chapter 9

Support

If you need support or have any question/suggestions please contact us.

- Email: `info@monoflow.org`
- Unity Forum: `http://forum.unity3d.com/threads/381894`

monoflow
new media experience

Chapter 10

Namespace Documentation

10.1 monoflow Namespace Reference

Classes

- class MPMP
- class ScriptOrder

Enumerations

- enum VRVideoMode { VRVideoMode.LEFT, VRVideoMode.RIGHT, VRVideoMode.TOP, VRVideoMode.B↩
  OTTOM }

  Enumeration for the VR Setup

10.1.1 Enumeration Type Documentation

10.1.1.1 enum **monoflow.VRVideoMode** `[strong]`
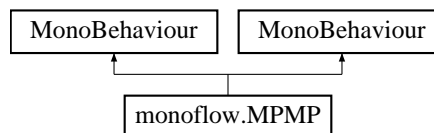
Enumeration for the VR Setup

Enumerator

    LEFT

    RIGHT

    TOP

    BOTTOM

Chapter 11

Class Documentation

11.1 monoflow.MPMP

Inheritance diagram for monoflow.MPMP:

```
┌──────────────┐  ┌──────────────┐
│ MonoBehaviour │  │ MonoBehaviour │
└──────────────┘  └──────────────┘
        └────────┬────────┘
         ┌──────────────┐
         │ monoflow.MPMP │
         └──────────────┘
```

Public Types

- enum Events {
  Events.LOAD, Events.LOADED, Events.PLAY, Events.PAUSE,
  Events.DESTROY, Events.ERROR, Events.PLAYBACKCOMPLETED, Events.AVF_PIXELBUFFER_ERR↩
  OR,
  Events.TEXTURE_CHANGED }

  Internal types of events from the native site
- enum FilterModeMPMP { FilterModeMPMP.Point, FilterModeMPMP.Bilinear }

  Texture filtering modes for Windows

Public Member Functions

- void Load ()

  Triggers a loading with the current videoPath
  See also

  > *monoflow.MPMP.videoPath*
- void Load (string path)

  Sets the current videoPath and triggers a loading
- void Play ()

  Start playing the media when loaded
- void Pause ()

  Pause the media
- void SeekTo (float t)

  Seek to a point in time
  The values are normalized (0 - 1)
  (Calling this method is the same as setting the seek property directly)

> *See also*
>
> > *monoflow.MPMP.seek*

- void SetSeeking (bool status)

  The seeking property should be set when seeking with a gui element on Android otherwise the video will not update while you seek

- double GetCurrentPosition ()

  Get the current position the media is playing
  values are normalized (0 -1)

- double GetDuration ()

  Get the duration of the media file in seconds

- bool IsPlaying ()

  Check if the media is playing

- bool IsPaused ()

  Check if the media is paused

- bool IsLoading ()

  Check if the media is loading

- Texture2D GetVideoTexture ()

  Get the raw video texture as Texture2D

- Vector2 GetNativeVideoSize ()

  Get the video size from the native plugin
  x: width
  y: height

- void SetVideoMaterial (Material mat)

  Set the video material.

- Material GetVideoMaterial ()

  Get the video material.

- void SetUpdateFrequency (float interval)

  Set the frequency how often per second the native plugin should update

- float GetUpdateFrequency ()

  Get the frequency how often per second the native plugin should update

- bool HasHadPixelBufferError ()
- IEnumerator LoadData (Uri loadUri, Action< byte[]> callBackAction, Action errorCallbackAction, Action< float > progressAction)
- IEnumerator SaveData (Uri saveUri, byte[] data, Action< bool > errorCallbackAction)
- IEnumerator DownloadAndSaveData (Uri loadUri, Uri saveUri, Action< bool > callbackAction, Action< float > progressAction)

  Use this method when you want to download a media file from a remote uri and store it local .

- IEnumerator CopyStreamingAssetData (string path)

  Copies a file from the Application.streamingAssetsPath to the Application.persistentDataPath

Static Public Member Functions

- static void MirrorUVY (MeshFilter meshf)

  Mirrors the uv.y of a mesh (1- uv.y)

- static void Set_VR_UV (MeshFilter meshf, VRVideoMode vr_mode)

Public Attributes

- Action< MPMP > OnInit

    Event that is called after the native part is initialized
- Action< MPMP > OnLoad

    Event that is called when a loading is triggert
- Action< MPMP > OnLoaded

    Event that is called when the loading is finished.
- Action< MPMP > OnPause

    Event is called when MPMP is pausing
- Action< MPMP > OnPlay

    Event is called when MPMP starts to play
- Action< MPMP > OnDestroyed

    Event is called when the MPMP instance is destroyed
- Action< MPMP > OnError

    Event is called when there arise an error on the native site
- Action< MPMP > OnPlaybackCompleted

    Event is called when the video has reached the end.
- Action< MPMP > OnPixelBufferError

    On OSX 10.11 El Capitan there is an issue with AVFoundation.
- Action< MPMP > OnTextureChanged

    Event is called when the dimension of the video texture has changed
- const string MENUITEM_NEW_MPMP = "GameObject/Create Other/MPMP"
- const string DEFAULT_TEXTURE_NAME = "_MainTex"
- const string LOGO64_NAME = "mpmp-logo.64x64"

Static Public Attributes

- static Color WARNING_COLOR = new Color(1f, 0.5f, 0f)

Properties

- bool autoPlay `[get, set]`

    Media starts playing automatically when loaded
- float seek `[get, set]`

    Seek to a point in time or just retrieve the current seek On iOS/OSX you can change the seek behaviour to a more precise version.
- float volume `[get, set]`

    get or set the current volume of the media
    values are normalized (0 - 1)
- float balance `[get, set]`

    Set or get the current audio output balance
    normalized values:
    -1 : just left channel
    0 : both channels
    1 just right channel
    (At the moment this is unsupported on OSX/iOS)
- bool looping `[get, set]`

    Get or set the looping of the media
- float rate `[get, set]`

    Get or set the current playback rate of the media
    negative values are reverse playback
    (On Android this is only supported at API level 23+ (Android 6+))
- FilterModeMPMP filtermode `[get, set]`

    On Windows you can choose between Point and Bilinear texture filtering

11.1.1 Member Enumeration Documentation

11.1.1.1 enum **monoflow.MPMP.Events** `[strong]`

Internal types of events from the native site

Enumerator

> ***LOAD***
> ***LOADED***
> ***PLAY***
> ***PAUSE***
> ***DESTROY***
> ***ERROR***
> ***PLAYBACKCOMPLETED***
> ***AVF_PIXELBUFFER_ERROR***
> ***TEXTURE_CHANGED***

11.1.1.2 enum **monoflow.MPMP.FilterModeMPMP** `[strong]`

Texture filtering modes for Windows

Enumerator

> ***Point***
> ***Bilinear***

11.1.2 Member Function Documentation

11.1.2.1 IEnumerator monoflow.MPMP.CopyStreamingAssetData (string *path*)

Copies a file from the Application.streamingAssetsPath to the Application.persistentDataPath

Parameters

path	

Returns

11.1.2.2 IEnumerator monoflow.MPMP.DownloadAndSaveData (Uri *loadUri,* Uri *saveUri,* Action< bool > *callbackAction,* Action< float > *progressAction*)

Use this method when you want to download a media file from a remote uri and store it local .

Yield as long as you download and save the data

Parameters

loadUri	
saveUri	
callbackAction	

Returns

11.1.2.3   double monoflow.MPMP.GetCurrentPosition ( )

Get the current position the media is playing

values are normalized (0 -1)

Returns

11.1.2.4   double monoflow.MPMP.GetDuration ( )

Get the duration of the media file in seconds

Returns

11.1.2.5   Vector2 monoflow.MPMP.GetNativeVideoSize ( )

Get the video size from the native plugin

x: width

y: height

Returns

11.1.2.6   float monoflow.MPMP.GetUpdateFrequency ( )

Get the frequency how often per second the native plugin should update

Returns

11.1.2.7   Material monoflow.MPMP.GetVideoMaterial ( )

Get the video material.

Returns

11.1.2.8   Texture2D monoflow.MPMP.GetVideoTexture ( )

Get the raw video texture as Texture2D

Returns

11.1.2.9 bool monoflow.MPMP.HasHadPixelBufferError ( )

11.1.2.10 bool monoflow.MPMP.IsLoading ( )

Check if the media is loading

Returns

11.1.2.11 bool monoflow.MPMP.IsPaused ( )

Check if the media is paused

Returns

11.1.2.12 bool monoflow.MPMP.IsPlaying ( )

Check if the media is playing

Returns

11.1.2.13 void monoflow.MPMP.Load ( )

Triggers a loading with the current videoPath

See also

> monoflow.MPMP.videoPath

11.1.2.14 void monoflow.MPMP.Load (string *path*)

Sets the current videoPath and triggers a loading

Parameters

path	

11.1.2.15 IEnumerator monoflow.MPMP.LoadData (Uri *loadUri,* Action< byte[]> *callBackAction,* Action *errorCallbackAction,* Action< float > *progressAction*)

11.1.2.16 static void monoflow.MPMP.MirrorUVY (MeshFilter *meshf*) [static]

Mirrors the uv.y of a mesh (1- uv.y)

Parameters

meshf	

11.1.2.17   void monoflow.MPMP.Pause (   )

Pause the media

11.1.2.18   void monoflow.MPMP.Play (   )

Start playing the media when loaded

11.1.2.19   IEnumerator monoflow.MPMP.SaveData (Uri *saveUri,* byte[] *data,* Action< bool > *errorCallbackAction*)

11.1.2.20   void monoflow.MPMP.SeekTo (float *t*)

Seek to a point in time

The values are normalized (0 - 1)

(Calling this method is the same as setting the seek property directly)

See also

monoflow.MPMP.seek

11.1.2.21   static void monoflow.MPMP.Set_VR_UV (MeshFilter *meshf,* VRVideoMode *vr_mode*) `[static]`

11.1.2.22   void monoflow.MPMP.SetSeeking (bool *status*)

The seeking property should be set when seeking with a gui element on Android otherwise the video will not update while you seek

Parameters

status	

11.1.2.23   void monoflow.MPMP.SetUpdateFrequency (float *interval*)

Set the frequency how often per second the native plugin should update

Depending on your system you normaly leave this at 60

Keep in mind that the real maximal frequency depends on the framerate of your app

Parameters

interval	

11.1.2.24   void monoflow.MPMP.SetVideoMaterial (Material *mat*)

Set the video material.

Parameters

mat	

11.1.3 Member Data Documentation

11.1.3.1 const string monoflow.MPMP.DEFAULT_TEXTURE_NAME = "_MainTex"

11.1.3.2 const string monoflow.MPMP.LOGO64_NAME = "mpmp-logo.64x64"

11.1.3.3 const string monoflow.MPMP.MENUITEM_NEW_MPMP = "GameObject/Create Other/MPMP"

11.1.3.4 Action<**MPMP**> monoflow.MPMP.OnDestroyed

Event is called when the MPMP instance is destroyed

11.1.3.5 Action<**MPMP**> monoflow.MPMP.OnError

Event is called when there arise an error on the native site

11.1.3.6 Action<**MPMP**> monoflow.MPMP.OnInit

Event that is called after the native part is initialized

11.1.3.7 Action<**MPMP**> monoflow.MPMP.OnLoad

Event that is called when a loading is triggert

11.1.3.8 Action<**MPMP**> monoflow.MPMP.OnLoaded

Event that is called when the loading is finished.

This is the best time to do some setup with regard to the actual video data like size, duration...

11.1.3.9 Action<**MPMP**> monoflow.MPMP.OnPause

Event is called when MPMP is pausing

11.1.3.10 Action<**MPMP**> monoflow.MPMP.OnPixelBufferError

On OSX 10.11 El Capitan there is an issue with AVFoundation.

To circumvent video refreshing errors you can catch this error.You should use this event to trigger a new Load.

11.1.3.11 Action<**MPMP**> monoflow.MPMP.OnPlay

Event is called when MPMP starts to play

11.1.3.12 Action<**MPMP**> monoflow.MPMP.OnPlaybackCompleted

Event is called when the video has reached the end.

When you are in Loop mode this event is not called!

11.1.3.13 Action<**MPMP**> monoflow.MPMP.OnTextureChanged

Event is called when the dimension of the video texture has changed

11.1.3.14 Color monoflow.MPMP.WARNING_COLOR = new Color(1f, 0.5f, 0f) `[static]`

11.1.4 Property Documentation

11.1.4.1 bool monoflow.MPMP.autoPlay `[get],[set]`

Media starts playing automatically when loaded

11.1.4.2 float monoflow.MPMP.balance `[get],[set]`

Set or get the current audio output balance

normalized values:

-1 : just left channel

0 : both channels

1 just right channel

(At the moment this is unsupported on OSX/iOS)

11.1.4.3 FilterModeMPMP monoflow.MPMP.filtermode `[get],[set]`

On Windows you can choose between Point and Bilinear texture filtering

If you set this property on non Windows platforms it has no effect on the video texture

11.1.4.4 bool monoflow.MPMP.looping `[get],[set]`

Get or set the looping of the media

11.1.4.5 float monoflow.MPMP.rate `[get],[set]`

Get or set the current playback rate of the media

negative values are reverse playback

(On Android this is only supported at API level 23+ (Android 6+))

11.1.4.6 float monoflow.MPMP.seek `[get],[set]`

Seek to a point in time or just retrieve the current seek On iOS/OSX you can change the seek behaviour to a more precise version.

(But could cause some decoding delay) Just enable the SEEK_TOLERANCE define at the top of the MPMP_API.cs file

The values are normalized (0 - 1)

11.1.4.7 float monoflow.MPMP.volume `[get],[set]`

get or set the current volume of the media

values are normalized (0 - 1)

11.2 monoflow.ScriptOrder

Inheritance diagram for monoflow.ScriptOrder:

```
┌─────────────────────────┐
│        Attribute        │
└─────────────────────────┘
             ▲
             │
┌─────────────────────────┐
│   monoflow.ScriptOrder  │
└─────────────────────────┘
```
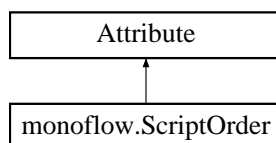
Public Member Functions

- ScriptOrder (int order)

Public Attributes

- int order

11.2.1 Constructor & Destructor Documentation

11.2.1.1 monoflow.ScriptOrder.ScriptOrder (int *order*)

11.2.2 Member Data Documentation

11.2.2.1 int monoflow.ScriptOrder.order

Index