NexPlayer TM Plugin for Unity

Version 1.0.1.21

Technical Reference Manual

Updated Feb 01th , 2019

Index

NexPlayerTM Plugin for Unity Engine

Legal Notices

Disclaimer for Intellectual Property

This product is designed for general purpose, and accordingly the customer is responsible for all or any of intellectual property licenses required for actual application NexStreaming Corp. does not provide any indemnification for any intellectual properties owned by third party

Copyright

Abstract

NexPlayerTM Plugin for Unity provides an interactive video playback for Android, iOS and through Unity
NexPlayerTM Plugin for Unity has been built to be reliable and robust without any sacrifice in performance, and has proven compatibility with international standards
NexPlayerTM for Unity works exclusively in conjunction with the native NexPlayer SDK and can benefit from all NexPlayerTM features, including intelligent ABR, PD/Local Playback, HTTP Live Streaming (HLS), DASH have a customizable feature set and API and many more

This documentation is a work in progress
Additional details and sample code will continue to be added

Please also be aware that for testing and development purposes, the Android/iOS emulator should not be used with the NexPlayerTM as there are known differences and issues between the emulator and actual devices, including the fact that the OpenGL renderer does not run properly within the emulator environment

Frequent testing on actual devices is strongly recommended during development and all apps should be tested on actual devices prior to release

NexPlayerTM Plugin for Unity Capabilities and Limitations

Protocols Summary

Platform	Supported Graphics APIs	HLS	PD	DASH	Local	Inside App (Streaming Assets)
Android(armeabi-v7a and x86)	OpenGL ES 2, OpenGLES3	O	O	O	O	O
iOS	OpenGLES2, OpenGL ES3, Metal	O	O	O	O	O

Features

- Support protocols for ABR algorithm, including HLS and DASH
- Support for progressive download (eg. online .mp4)
- Complete API including:
    - Play / Pause
    - Seek
    - Video resolution
    - Last millisecond buffered
- Useful callbacks including:
    - Information about the buffering state
    - State of the playback
- Widevine DRM on Android and iOS for DASH videos

Requirements

NexPlayerTM Plugin for Unity supports Android, iOS

In Android the following is required:

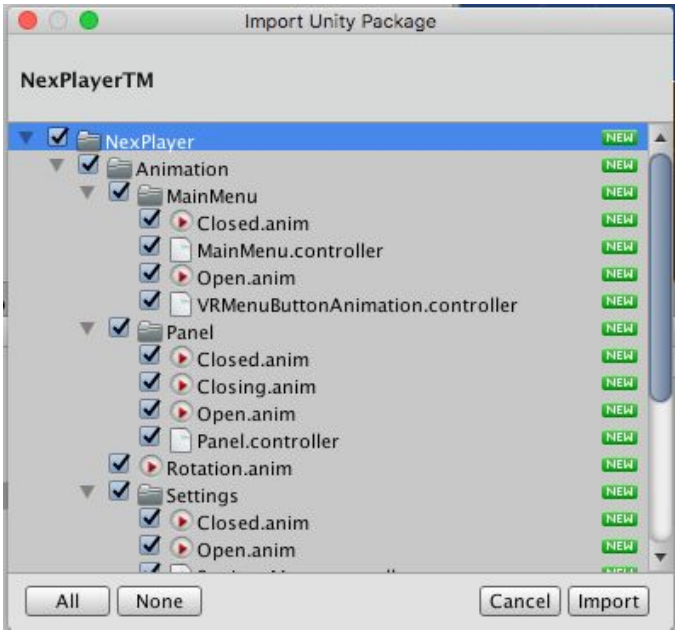
- Use of OpenGL ES version 2.0 or 3.0
- Use of the Android version 4.4.0 or above

In iOS the following is required:

- Use of OpenGL ES version 2.0
- Use of Metal
- Below iOS 9.0 ,Metal API is unstable(Recommended above 9.0 or above)

Integration Guide

A fully working example of NexPayerTM for Unity is provided in the Unity Package It can be imported into an Unity project by double clicking it



Example scenes are available in the example Unity project in the Unity folder ./Assets/NexPlayer/Scenes/ in the file NexPlayer.unity

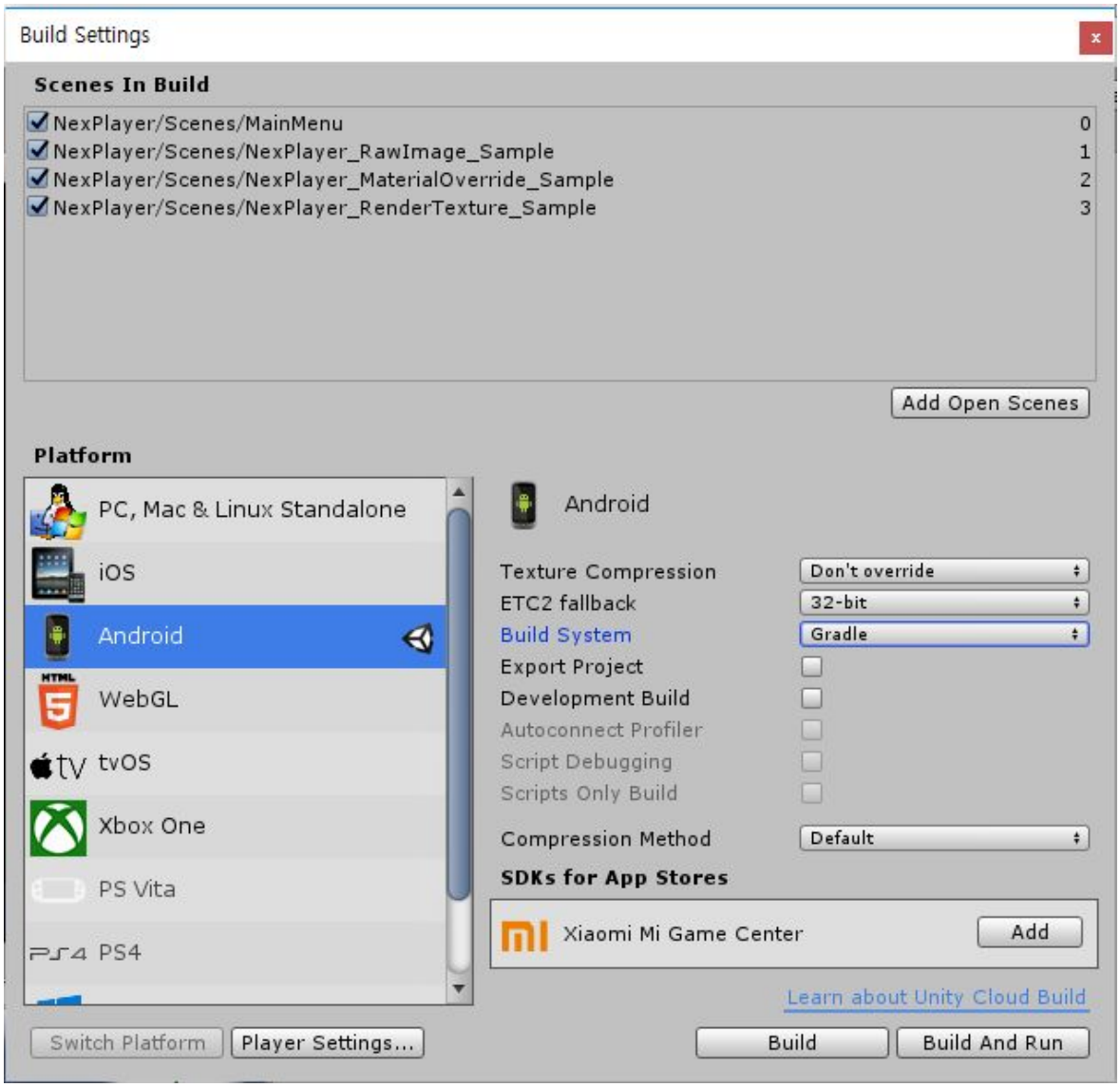
In order to integrate NexPlayerTM the compatible graphics APIs need to be selected That can done manually in the “Player Settings” section of Unity for each platform
If the helper component NexEditorHelper.cs is attached to any GameObject it will include a graphics UI to automatically detect any conflict regarding the graphics API, and it will promptly solve it

A sample way of integrating NexPlayerTM with any GameObject with a Unity Material can be used using the component NexPlayer.cs

To allow any remote video on Android “Internet Access” needs to be set to true in the Unity player settings
Also to view HTTP videos on iOS “Allow downloads over HTTP” needs to be enabled
A quick and easy way to enable these settings is using the helper component (NexEditorHelper.cs)

Build Configuration

Add the following scenes to the Unity build:
Select the platform which you want and Click Build And Run button



Sample scenes

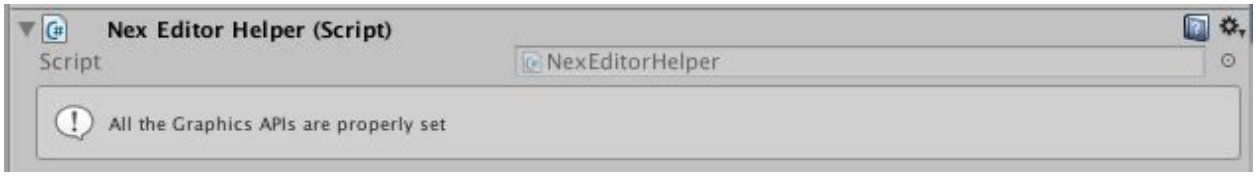
Description of each scene

Scene	Description
MainMenu.unity	The main screen
NexPlayer_RawImage_Sampler.unity	It is normal sample that plays video content using Raw Image component and that can **confirm that subtitles appear only in this scene**
NexPlayer_MaterialOverride_Sample.unity	The sample is to play video using Material Override component
NexPlayer_RenderTexture_Sample.unity	The sample is to play video using RenderTexture component

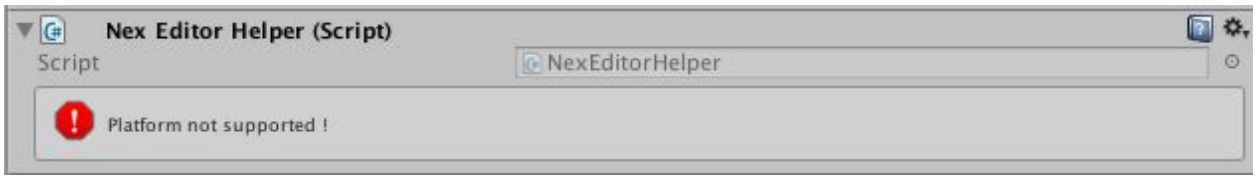
Playing a video

Editor Settings

An example of using NexPlayer can be found in the script NexPlayer
It's recommended to also attach the Nex Editor Helper script that will make sure the correct graphics APIs are selected



Example when select correct graphics API

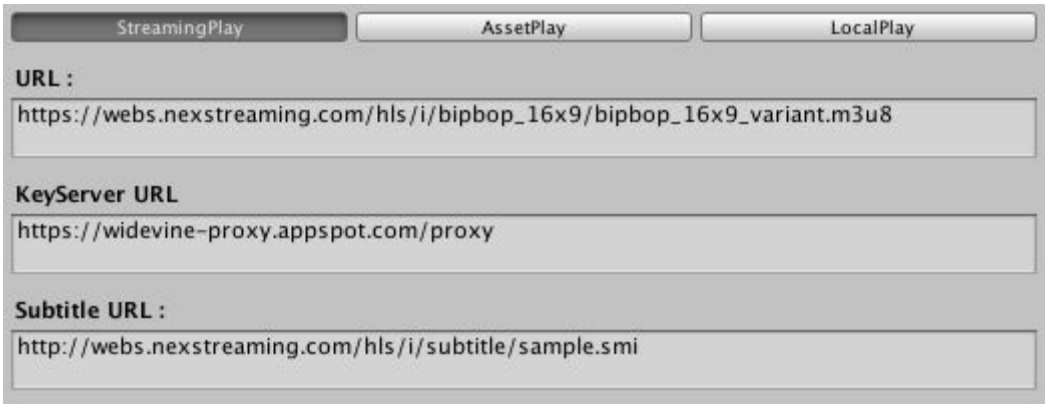


Example when select wrong graphics API

Playback Modes

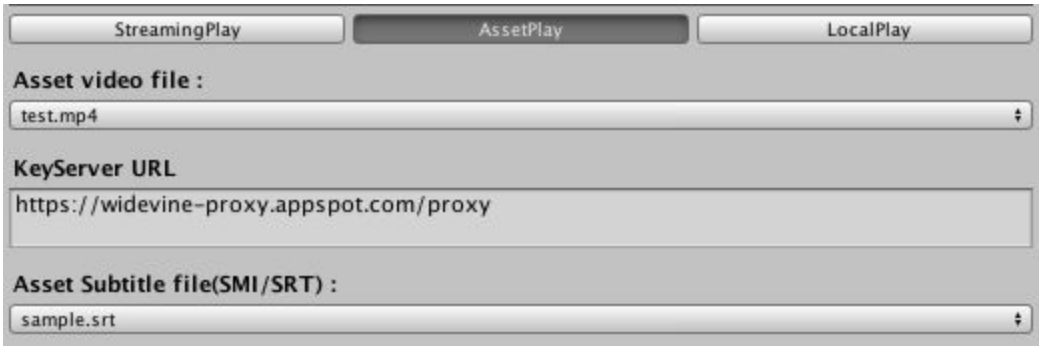
The NexPlayer plugin supports three playback modes

1) Streaming Play : When playing streaming content



URL	The URL of Streaming Content
KeyServer URL	The URL of Key Server
Subtitle URL	The URL of External Subtitle

2) AssetPlay : Plays the video file in the Streaming folder

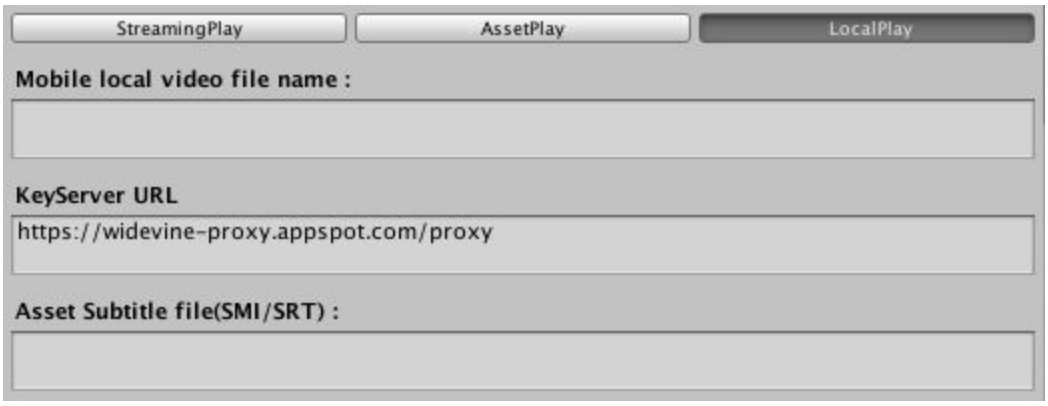


Example in StreamingAssets Directory

Example put subtitle file into StreamingAssets Directory

Asset video file	Selects the file to play among the mp4 files in the streaming folder (If there are no files in the folder, a warning message will be displayed)
KeyServer URL	The URL of the Key Server
Subtitle URL	Select the file to display subtitle among the SRT and SMI files in the streaming folder (If there are no files in the folder a warning message will be displayed)

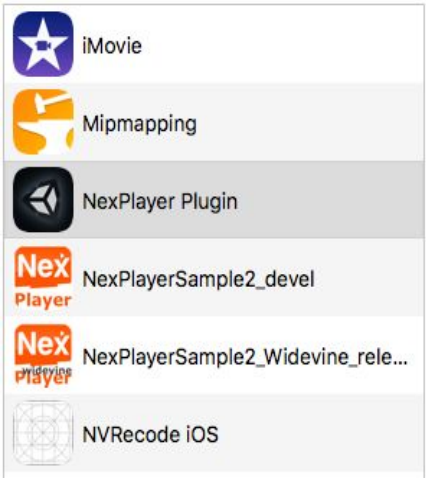
3) Local Play : Plays the video file in the Mobile phone

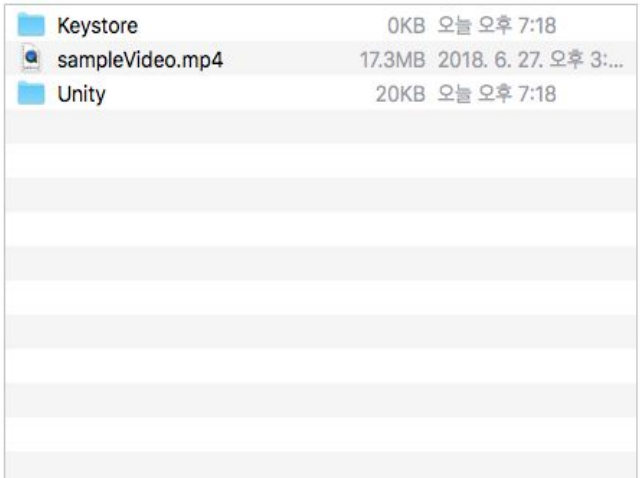


Mobile local video file name	The Name of the Video file
KeyServer URL	The URL of the Key Server

Subtitle file	The Name of the Subtitle file

Android
1.(SDCARD/Android/data/Bundle Identifier/files/FileName.mp4)
2.Bundle Identifier is different for each app
3.If the file exists in the above path based on the file name entered in the Inspector, it can be played
4.If you put a subtitle file in the same path as the video file the subtitle will be displayed on the screen
iOS
1.After building in unity, turn on the application supports iTunes file sharing in the info.plist file in the generated Xcode project
2.Using iTunes, put the file in the file name you entered in the inspector





3.Subtitles are the same way, Make sure that the file names you enter in the Inspector match

Creating the Player

First the Nexplayer needs to be created, an action should be registered to receive the callbacks, the rendermode should be set, the target renderer should be set, the player should be initialized, and the coroutine needs to be started.

```
At SetupNexPlayer() in Awake()
try
{
   // Creation of the NexPlayer instance

   player = NexPlayerFactory.GetNexPlayer();

   // Register to the events of NexPlayer
   player.OnEvent += EventNotify;

   //Default renderMode is RawImage

   switch (m_RenderMode)
   {
      case NexRenderMode.MaterialOverride:
         player.renderMode = NexRenderMode.MaterialOverride;
         player.targetMaterialRenderer = renderer;
         break;
      case NexRenderMode.RenderTexture:
         player.renderMode = NexRenderMode.RenderTexture;
         player.targetTexture = renderTexture;
         break;
      case NexRenderMode.RawImage:
         player.renderMode = NexRenderMode.RawImage;
         player.targetRawImage = rawImage;
         break;
   }

   SetProperties();

   // Initialize NexPlayer
      NexPlayerError initResult = player.Init(logLevel);

      URL = NexUtil.GetFullIUri(playType, URL);
      subtitleURL = NexUtil.GetFullIUri(playType, subtitleURL);

       if (initResult == NexPlayerError.NEXPLAYER_ERROR_NONE)
            {
                  OpenPlayer()

             }
```

```
            else
            {
                if (initResult ==
NexPlayerError.NEXPLAYER_INVALID_RENDERMODE_TARGET)
                    playerStatusText = "Render Fail";
                else if (initResult ==
NexPlayerError.NEXPLAYER_PLAYER_INIT_FAILURE)
                    playerStatusText = "Init Fail";
                else if (initResult ==
NexPlayerError.NEXPLAYER_TEXTURE_INIT_FAILURE)
                    playerStatusText = "Texture Fail";

                player = null;
            }
}
catch (System.Exception e)
{
    Debug.LogError("Error while initializing the player. Please check that your platform is
supported.");
    Debug.LogError("Exception: " + e);
    playerStatusText = “Error”;
}
```

The update method of the Nexplayer needs to be called at the Update callback of the MonoBehaviour object:

```
void Update()
{
    if (player != null)
    {
        player.Update();
    }
}
```

Releasing the Player

To release the Nexplayer, call the Release method and wait for the NEXPLAYER_EVENT_CLOSED callback:

```
public void ToogleQuit()
{
```

```
    if (this.gameObject.activeSelf == false)
    {
            return;
    }
   FinishGame();
}

 private void FinishGame()
        {
            if (player != null)
            {
                if (Application.platform == RuntimePlatform.WindowsEditor)
                {
                    player.Close();
                    player.Release();
                    player = null;
                    GoBack();
                }
                else
                {
                    GoBack();
                }
            }
            else
            {
                GoBack();
            }
        }


void EventNotify(NexPlayerEvent paramEvent, int param1, int param2)

{
            ...
            switch (paramEvent)
            {
                        ...
                        case NexPlayerEvent.NEXPLAYER_EVENT_CLOSED:

                                    {
                                                ResetPlayerUI();
                                    }
                                    break;
            }
}
```

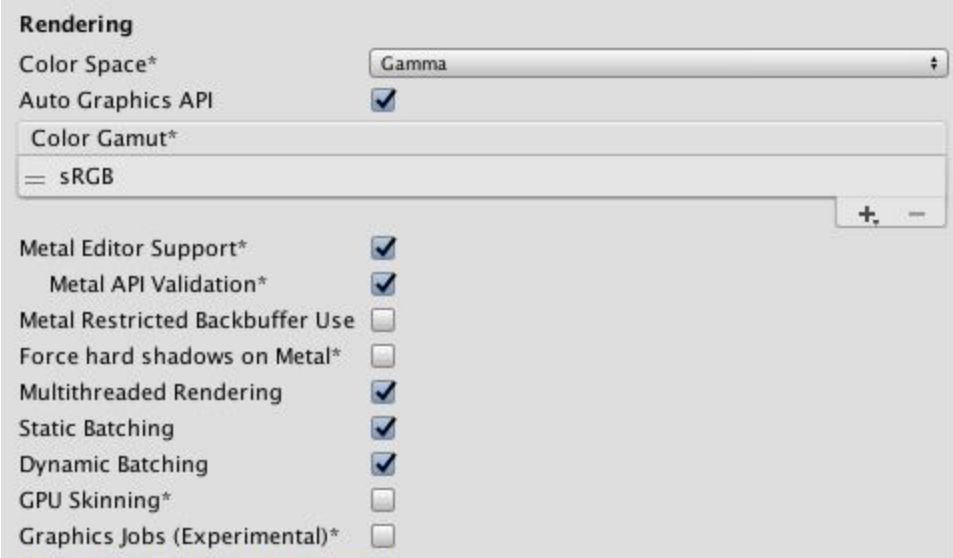
Background status handling

In Unity, Check the state change(back/foreground) via OnApplicationPause function's parameter value.
If the application state is background, Call Pause Function of the NexPlayer.
When the state of the application becomes foreground, calls the Resume Function of the NexPlayer.

```
void OnApplicationPause(bool pauseStatus)
{
    Log("OnApplicationPause(" + pauseStatus + ")");
    if (player != null)
    {
        //Go to Background
        if (pauseStatus)
        {
            // Save current player status
            playerStatus = player.GetPlayerStatus();
            bApplicationPaused = true;
            if (player.GetPlayerStatus() > NexPlayerStatus.NEXPLAYER_STATUS_STOP)
            {

                player.Pause();
            }
        }
        //Return to Foreground
        else
        {
            if (bApplicationPaused)
            {

                if (player.GetPlayerStatus() >
NexPlayerStatus.NEXPLAYER_STATUS_STOP && playerStatus ==
NexPlayerStatus.NEXPLAYER_STATUS_PLAY)
                {
                    player.Resume();
```

```
            }
            playerStatus = NexPlayerStatus.NEXPLAYER_STATUS_NONE;
            bApplicationPaused = false;
        }
    }
  }
}
```

iOS Graphic Settings



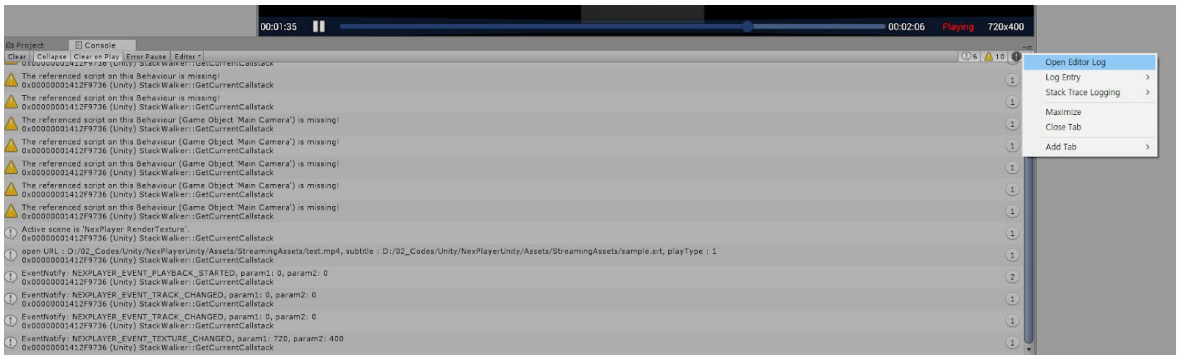
As set above, we recommend that enable the Auto Graphics API

Features supported by Unity Editor Play mode

Support features as below,

- Content Type
    - only MP4
- Render Mode
    - Render texture, Material override, raw Image
- Player features
    - Playback, Seek, resume/pause
- Subtitle
    - PC Player not supports subtitle.
- Streaming
    - PC Player not supports streaming play
- DRM
    - PC Player not supports DRM Contents
- Low Latency
    - PC Player not supports Low Latency
- Player properties
    - Set Mute : supported
    - Set Loop : supported
    - Autoplay : player plays automatically when scene is play
- Play Mode
    - Streaming Play **: <u>streaming play is not complete supported</u>**
    - AssetPlay : The files in the streamingAssets folder are displayed in drop-down list format.
    - LocalPlay : In windows, contents are located in C:\users\{$userName}\AppData\LocalLow\{$ComPanyName}\{$ProjectName}\
- Graphics API
    - Only support DirectX 11

How to get Debug Log

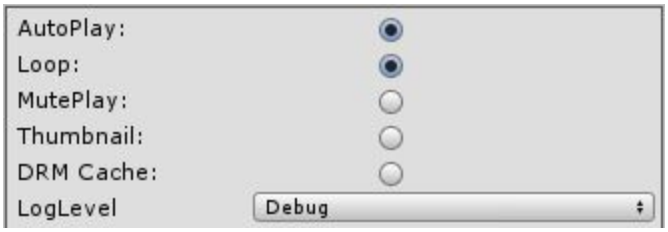
- To feedback, Like the picture below, open Editor Log and deliver log file to our (located in C:\Users\{$userName}\AppData\Unity\Editor\Editor.Log)
- 

Etc

- In Unity Editor, Not support go url button in “Main Menu” scene.

Documentation

NexPlayer Properties



Properties	Description
AutoPlay	Starts playback automatically after video is buffered
Loop	When the playback reaches the end position it jumps to the start and plays again
MutePlay	Turns off the Audio
Thumbnail	The Nexplayer waits for playback with first video frame
Volume	Adjust the NexPlayer s/w audio volume
DRM Cache	Enable to cache for license, but server-side should support it.
LogLevel	Debug Log Level

```
private void SetProperties()
{
```

```
        if (player != null) {
            player.mute = mutePlay;
            player.volume = volumeSize;
            player.loop = loopPlay;
            player.autoPlayback = autoPlay;
            player.firstFrame = thumbnail;
            player.bOnSubtitle = bOnSubtitle;
}
```

DRM

NexPlayer supports DRM contents

- **KeyServer URL**

  Input DRM KeyServer URL

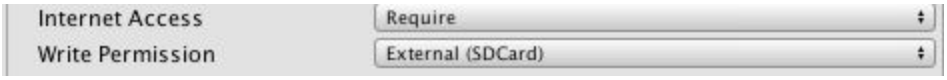
  

- **Content meta**

  Input DRM Request Header Data to drmHeaderData(Dictionary<string, string>

- **Player Settings(Android)**

  Please make the Android settings as shown below
  (Setup Process : File > Build Settings > Android(Player Settings) > Other Settings)

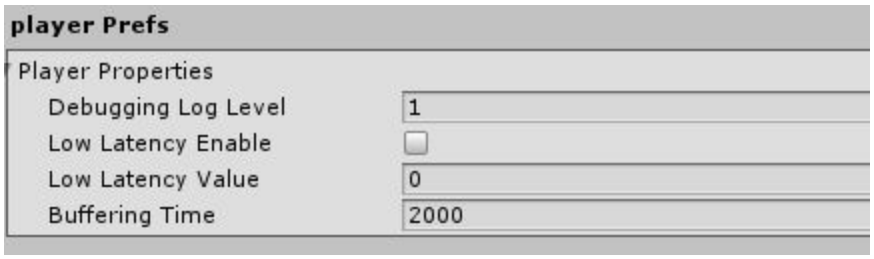
  

  Reason : Saving the DRM certification datas in Android SDCard

```
At OpenPlayer() in NexPlayer.cs
//DRM Setting
player.UseLicenseCacheMode(drmCache);
player.SetWVDRMConfig(KeyServerURI, drmHeaderData, licenseRequestTimeout);



.......

if (initResult == NexPlayerError.NEXPLAYER_ERROR_NONE)
{
    StartPlay(URL, subtitleURL);

    playerStatusText = "Opening...";
}
```

LowLatency

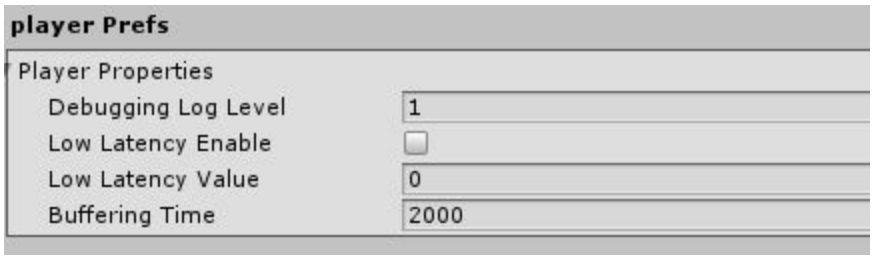
NexPlayer supports Low Latency



Properties	Description
Low Latency Enable	Set low-latency enabled/disabled
Low Latency Value	Set number of milliseconds of media to buffer if additional buffering is required during streaming playback for low-latency

```
At SetPlayerPrefs(NexPlayerPrefsHelper.NexPlayerPrefs playerPrefs) in OpenPlayer()
if (player != null && playerPrefs != null)
{
   //Set LowLatency
   if(playerPrefs.lowLatencyEnable)
   {
      player.SetLowLatency(playerPrefs.lowLatencyEnable,
playerPrefs.lowLatencyValue);
   }
}
```

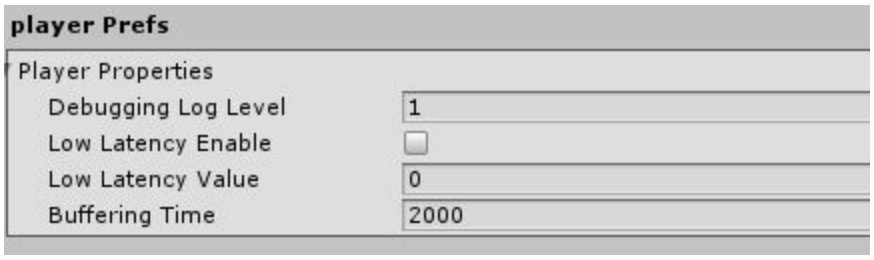
LogLevel for Debugging



Properties	Description
Debugging Log Level	Set Log level for debugging(protocol, Nex AL Libraries.). This value must be integer between -1~4.

```
At SetPlayerPrefs(NexPlayerPrefsHelper.NexPlayerPrefs playerPrefs) in OpenPlayer()
if (player != null && playerPrefs != null)
{
    //Set log level for debugging
    if(playerPrefs.lowLatencyEnable)
    {
        player.logLevelForDebugging = playerPrefs.debuggingLogLevel;
    }
}
```

Buffering Time



Properties	Description
Buffering Time	Set number of milliseconds of media to buffer if additional buffering is required during streaming playback(if this value is set to 0, then default value is set to 5000.)

```
At SetPlayerPrefs(NexPlayerPrefsHelper.NexPlayerPrefs playerPrefs) in OpenPlayer()
if (player != null && playerPrefs != null)
{
   //Set buffering time
   if(playerPrefs.bufferingTime > 0)
   {

         player.SetPlayerProperty(NexPlayerProperty.INITIAL_BUFFERING_DURATION
         , playerPrefs.bufferingTime);
         player.SetPlayerProperty(NexPlayerProperty.RE_BUFFERING_DURATION,
         playerPrefs.bufferingTime);

   }
}
```

RenderMode

NexPlayer supports three render modes

RenderMode	Description
RawImage	Plays the video with Raw Image component
Material Override	Plays the video with Renderer Material texture
RenderTexture	Plays the video with Render Texture

- **RenderMode.RawImage**

1. Set the Nexplayer renderMode to RenderMode.RawImage
2. Set target RawImage to play video

```
private void SetupNexPlayer() {
        try
        {
            .......

            player.renderMode = RenderMode.RawImage;
            player.targetRawImage = rawImage;
```

Example how to set Rendermode for RawImage in script and inspector

- **RenderMode.Material Override**

1. Set the Nexplayer renderMode to RenderMode.Material Override
2. Set target Material Renderer to play video

```
private void SetupNexPlayer() {
        try
        {
            .......

            player.renderMode = RenderMode.MaterialOverride;
            player.targetMaterialRenderer = renderer;
```

Example how to set Rendermode for Renderer in script and inspector

- **RenderMode.RenderTexture**

1. Set the Nexplayer renderMode to RenderMode.RenderTexture
2. Set target TargetTexture to play video

```
private void SetupNexPlayer() {
        try
        {
            .......

            player.renderMode = RenderMode.RenderTexture;
            player.targetTexture = renderTexture;
```

Example how to set Rendermode for RenderTexture in script and inspector

Log Property

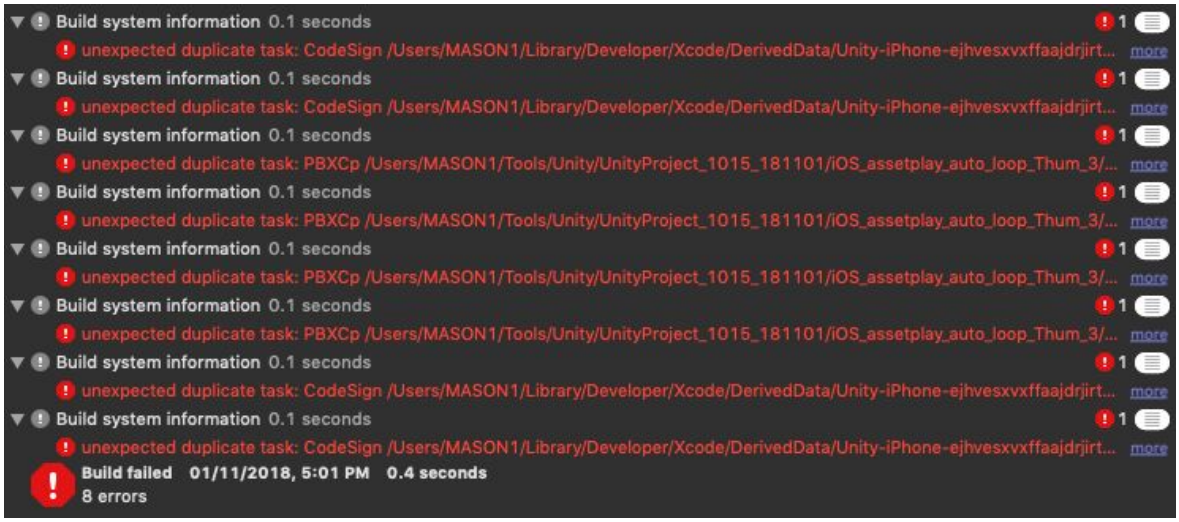
There are three log level types for Debugging

Log Level	Description
None	Do not use logs
0	Check only logs at C# level
Debug	View NexPlayer, NexAIFactory logs
RTP	Also, View RTP Logs(include Debug Logs)
RCP	Also, View RCP Logs(include RTP,Debug Logs)
Frame	Also, View Frame Logs(include RTP,Debug Logs)

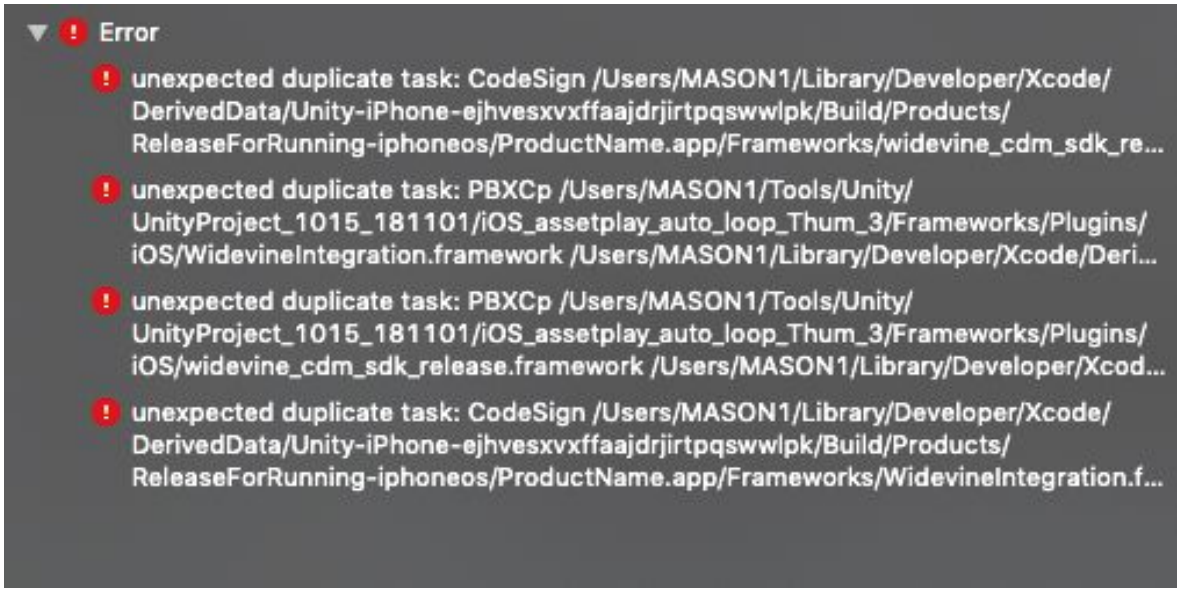
```
private void SetupNexPlayer() {
        try
        {
            .......

            player.Init(logLevel);
```

FAQ

1.If you encounter the following problems during Unity iOS build

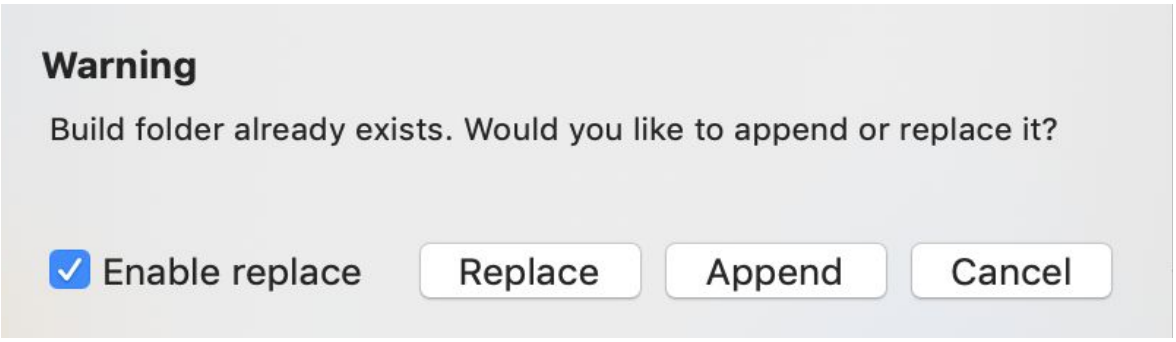


```
Build system information 0.1 seconds
    unexpected duplicate task: CodeSign /Users/MASON1/Library/Developer/Xcode/DerivedData/Unity-iPhone-ejhvesxvxffaajdrjirt...
Build system information 0.1 seconds
    unexpected duplicate task: CodeSign /Users/MASON1/Library/Developer/Xcode/DerivedData/Unity-iPhone-ejhvesxvxffaajdrjirt...
Build system information 0.1 seconds
    unexpected duplicate task: PBXCp /Users/MASON1/Tools/Unity/UnityProject_1015_181101/iOS_assetplay_auto_loop_Thum_3/...
Build system information 0.1 seconds
    unexpected duplicate task: PBXCp /Users/MASON1/Tools/Unity/UnityProject_1015_181101/iOS_assetplay_auto_loop_Thum_3/...
Build system information 0.1 seconds
    unexpected duplicate task: PBXCp /Users/MASON1/Tools/Unity/UnityProject_1015_181101/iOS_assetplay_auto_loop_Thum_3/...
Build system information 0.1 seconds
    unexpected duplicate task: PBXCp /Users/MASON1/Tools/Unity/UnityProject_1015_181101/iOS_assetplay_auto_loop_Thum_3/...
Build system information 0.1 seconds
    unexpected duplicate task: CodeSign /Users/MASON1/Library/Developer/Xcode/DerivedData/Unity-iPhone-ejhvesxvxffaajdrjirt...
Build system information 0.1 seconds
    unexpected duplicate task: CodeSign /Users/MASON1/Library/Developer/Xcode/DerivedData/Unity-iPhone-ejhvesxvxffaajdrjirt...
Build failed   01/11/2018, 5:01 PM   0.4 seconds
8 errors
```

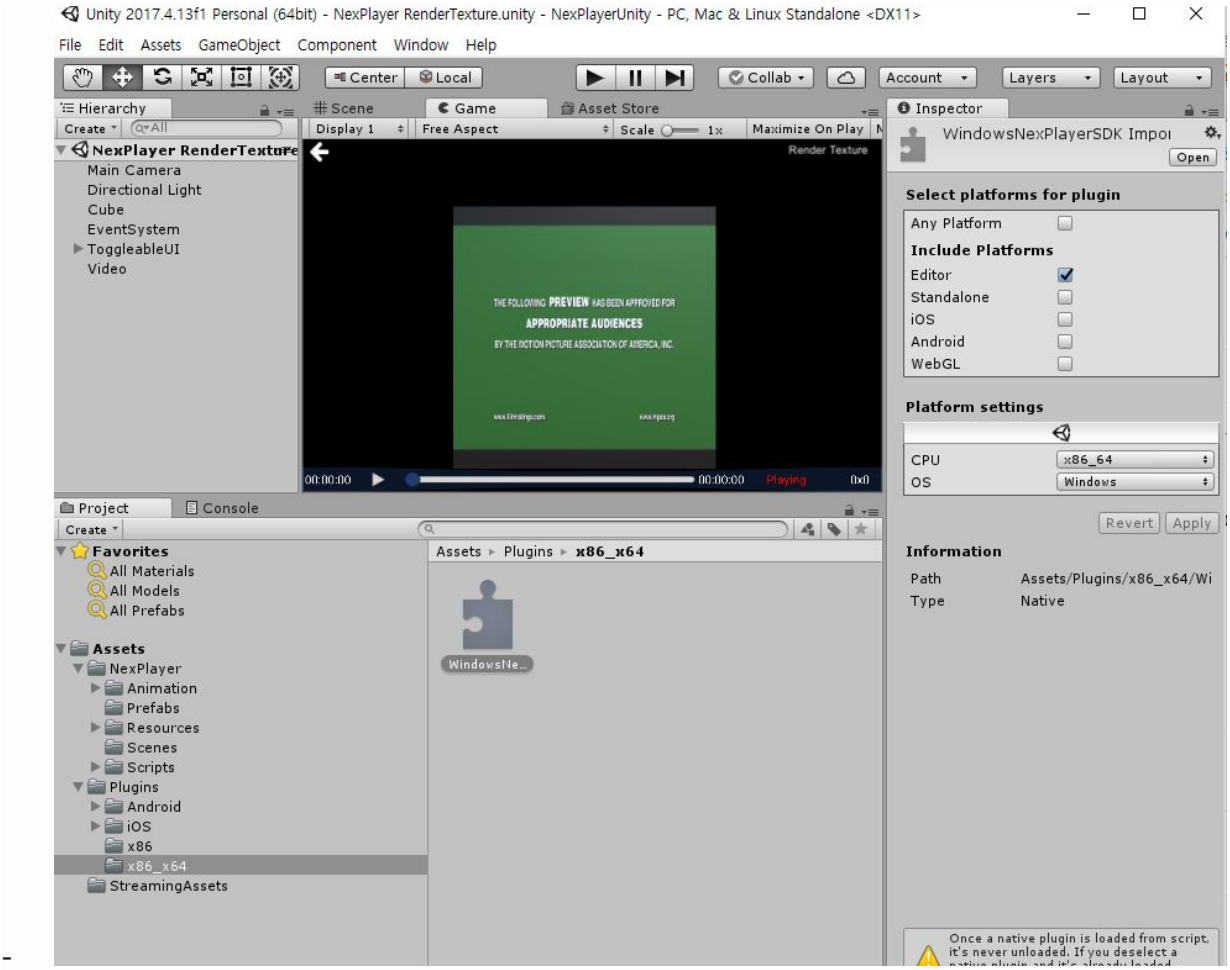


- When building the Unity iOS in the Unity Editor, creating a new Xcode project will solve the problem

- Or if you want to overlap on the same project, check enable replace and select replace



2.The Windows NexPlayerSDK DLL must be specified as Editor platforms as pictured below



-

3.What files are included in the NexPlayer Plugin for Unity ?

All the necessary files to integrate it with any Unity project , sample scenes for a normal video , this documentation and the release notes. The files are distributed in the following directories:

The red separated files are Player SDK API

```
./NexPlayer
        /NexPlayer_Plugin_for_Unity_Release_Notes.txt
        /NexPlayer_Plugin_for_Unity_Reference_Manual
                /index.html
        /License.txt

        /Prefabs
                /Main Camera.prefab
```

/Sample Controls.prefab
/SF Button.prefab
/SF Scene Elements.prefab
/SF Title.prefab
/Stereo Cameras.prefab

/Resource
/font-awesome_4-7-0_arrow-left_396_50_ffffff_none.png
/font-awesome_4-7-0_link_32_28_ffffff_none.png
/font-awesome_4-7-0_pause_256_100_d9d5d4_none.png
/font-awesome_4-7-0_play_256_100_d9d5d4_none.png

/Scenes
/NexPlayer.unity
/NexPlayer RenderTexture.unity
/NexPlayer Material Override.unity
/Main Menu.unity

/Scripts
/Utility
/StreamingAssets FileHelper.cs
/NexUtil.cs
/UI
/Tilt Window.cs
/StereoMode.cs
/PanelManager.cs
/NexVideo Object.cs
/NexUS Controller.cs
/NexSeekBar.cs
/NexMainCube.cs
/Main.cs

/Editor
/NexPlayerEditor.cs
/SunShafts Editor.cs
/iOS Build.cs

/**SDK**

/NexPlayeriOS.cs

/NexPlayerFactory.cs

/NexPlayerCommon.cs

/NexPlayerBase.cs

/NexPlayer Android.cs

./**Plugins**

/iOS

/Widevine Integration.framework

/Widevine Helper.m

Widevine Helper.h

/widevine_cdm_sdk_release.framework

/Android

/libs

/x86

/ so files *

/armeabi-v7a

/ so files *

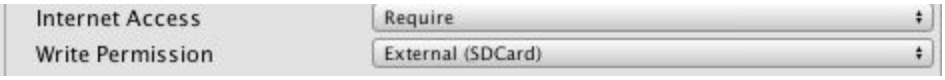
/classes.jar

The red separated files are Player SDK API

4.Can’t play DRM Content when build on Android platform

Please Check the Android Player Settings below

Setup Process : Build Settings > Android(Player Settings) > Other Settings



5.Detail API

See the Detail API to Assets/NexPlayer/NexPlayer_Plugin_for_Unity Reference Manual

6.Import NexPlayer component guide video

Import player component guide video