AVPro DeckLink

for Windows Desktop

Unity plugin for broadcast standard video input and output via BlackMagic DeckLink hardware

Version 1.6.6

Released 9 January 2019

Contents

1. Introduction
    1.1 Features
    1.2 Trial Version
2. System Requirements
    2.1 Platforms not Supported
3. Installation
    3.1 Trial Version & Watermark Notes
        3.1.1 Watermark
        3.1.2 Updating from Trial Version
    3.2 Platform Notes
4. Quick Start and Demos
    4.1 Quick Start: Fastest Start for Unity Experts
    4.2 Quick Start: Video streaming using Prefabs
5. Usage
    5.1 Getting Started
    5.2. Components
        5.2.1 DeckLinkInput Component
        5.2.2 DeckLinkOutput Component
        5.2.3 ApplyToMaterial Component
        5.2.4 IMGUIDisplay Component
        5.2.5 uGUIDisplay Component
    5.3 Scripting
        5.3.1 Namespace
        5.3.2 DeckLink Input Scripting
        5.3.3 DeckLink Output Scripting
        5.3.4 Obtaining device and mode information
        5.3.5 Using Device class directly
        5.3.6 Documentation
6. Asset Files
    6.1 Demos
    6.2 Prefabs
    6.3 Scripts
7. Supported Input and Output Formats
8. Support
    8.1 Bug Reporting
9. About RenderHeads Ltd
    9.1 Services
    9.2 Our Unity Plugins

Appendix A - Release History
Appendix B - Roadmap
Appendix C - Document Version History

1. Introduction

AVPro DeckLink is the latest plugin from RenderHeads, the developer of a variety of Unity plugins including AVPro Video, AVPro Live Camera, and AVPro Movie Capture. This plugin aims to provide quick and easy to use DeckLink capture card support in Unity so developers can easily integrate high quality video streaming functionality into their projects.

1.1 Features

- *NEW* DeckLink 8K Pro support
- *NEW* Timecode support
- *NEW* Ancillary data support
- Free watermarked trial version available
- Broadcast standard video input and output
- Up to 8K video input and output
- Automatic video input mode detection
- Alpha channel output supported
- Internal and external keyer supported
- IMGUI and uGUI support
- Material mapping support
- Stereo (3D) support
- HLG support
- High performance
- Easy to use
- Includes example scenes
- Includes drag and drop components
- Unity 5.1, 2017.x, 2018.x and above supported
- Genlock pixel offset support
- Unity audio source and audio listener support for audio input and output
- Full duplex card support
- Configurable duplex options
- Supports both capture and playback from/to multiple devices.

1.2 Trial Version

We offer an unlimited trial version of AVPro DeckLink for download from our website at http://renderheads.com/product/avpro-decklink/. The trial version has no missing features or time restrictions but it does apply a watermark to the rendered output. The watermarking does have a small performance impact which is only really noticeable on very high resolution videos.

2. System Requirements

- Unity 5.1 and above
- Unity 2017.x Unity 2018.x
- Windows XP and above (32-bit and 64-bit)
- BMD Desktop Video 10.11.4

2.1 Platforms not Supported

- Linux
- Android
- macOS
- iOS
- tvOS
- Game Consoles (XBox, PS4 etc)

3. Installation

1. Open up a fresh Unity session (to clear any locked plugin files)
2. Import the **unitypackage** file into your Unity project. If prompted to upgrade some scripts, click Yes.

3.1 Trial Version & Watermark Notes

3.1.1 Watermark

If you are using a trial version of the plugin then you will see a watermark displayed over the video. The watermark is in the form of a thick horizontal bar that moves around the screen.

3.1.2 Updating from Trial Version

If you are upgrading from the trial version, make sure you delete the old /Assets/Plugins folder as this contains the trial plugin and could conflict. You may need to close Unity first, delete the files manually and then restart Unity and re-import the package (because Unity locks native plugin files once they are loaded).

You can check which version you have installed by adding a DeckLinkInput or DeckLinkOutput component to your scene and clicking on the 'about' button in the Inspector for that component. The version number is displayed in this box.

3.2 Platform Notes

- Only Direct3D11 is currently supported
- Multi-threaded rendering is supported
- Supports automatic conversions from the following formats for input:
    - 8-bit ARGB
    - 8-bit BGRA
    - 8-bit UYVY 4:2:2
    - 10-bit UYVY 4:2:2
- Supports automatic conversions to the following formats for output:
    - 8-bit ARGB
    - 8-bit BGRA
    - 8-bit UYVY 4:2:2
    - 10-bit UYVY 4:2:2
    - 10-bit RGBX big endian
    - 10-bit RGBX little endian
    - 10-bit RGB big endian

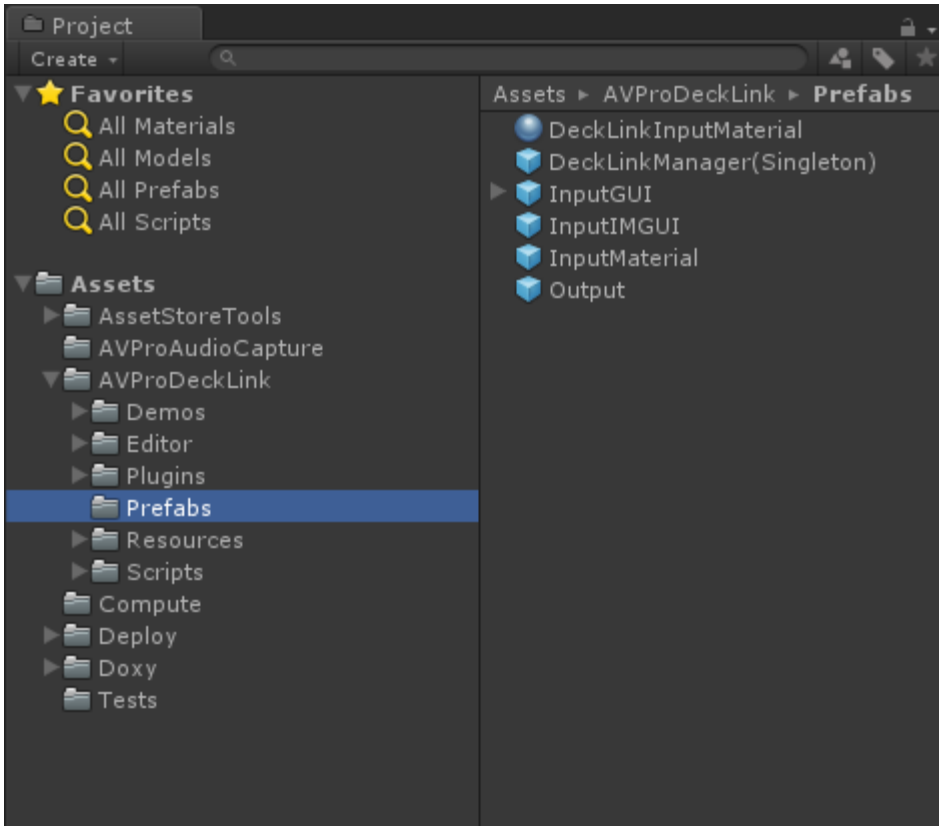
4. Quick Start and Demos

4.1 Quick Start: Fastest Start for Unity Experts

1. Use the DeckLinkInput component to capture a video stream from a DeckLink device.
2. Use the DeckLinkOutput script to send a video stream from a specified camera to a DeckLink device.
3. In the case where you are streaming input, use one of the display scripts (IMGUIDisplay, uGUIDisplay, and ApplyToMaterial) to display to the relevant surface

4.2 Quick Start: Video streaming using Prefabs

1. Create a new Unity project
2. Import the AVProDeckLink package
3. Open the AVProDeckLink/Prefabs folder in your project window to see the various prefabs, and add the desired prefab to your scene.



The prefabs are as follows:

- **InputIMGUI:** Streams video input and displays it to the screen using Unity's legacy IMGUI system.

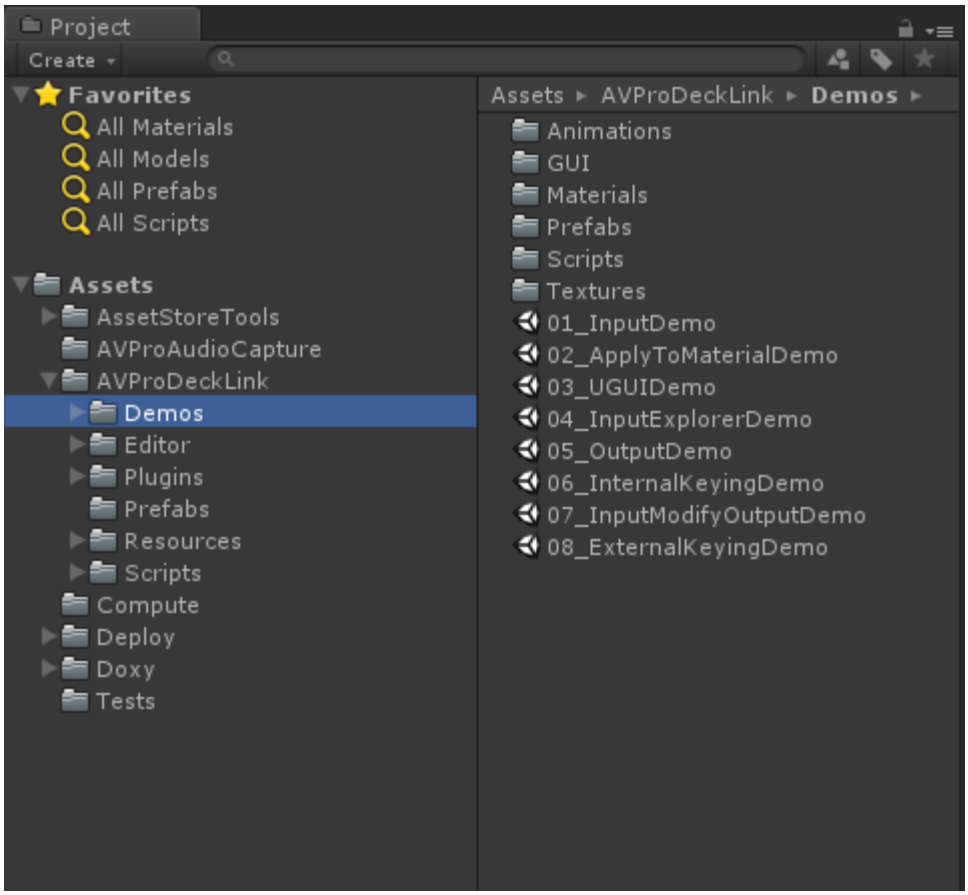
- **InputGUI:** Streams video input and displays it using Unity's uGUI system. If you wish to have multiple displays, simply duplicate the display object, and move/resize/rotate it to your desired location.
- **InputMaterial:** Applies input video stream to a material, which can then be used in conjunction with 3D objects in order to display your video stream on meshes. To use a 3D object together with the applied material, simply set the material property of the object's renderer to "DeckLinkInputMaterial".
- **Output:** Streams video output to desired DeckLink device. Move/Rotate/Scale it in order to get different viewing angles.
- **DeckLinkManager:** Exactly one is required in the scene in order to handle tasks such as sending signals to the native plugin. The user does not have to explicitly create this as it gets auto created if one is not found. There also should be no more than one in the scene as it is a singleton and errors will be thrown if otherwise.
- **DeckLinkSettings:** This is used to configure the plugin. There are currently only two tasks that require this component. It can be used to enable multi-output mode, which allows for multiple devices to simultaneously playback from different cameras using different modes, and it can also be used to configure the duplex mode of devices. There are plans to expand this to allow users to configure far more aspects of their devices. There should be a maximum of one of these in any given scene.

4. After the desired prefab is added, you need to configure either the DeckLinkInput or the DeckLinkOutput component to correspond with your input or output devices (leaving them as is will result in them finding the first available device). Full information on the various options of these components can be found in the sections 5.2.1 DeckLink Input Component and 5.2.2 DeckLink Output Component.

5. Usage

5.1 Getting Started

The easiest way of getting started is to have a look at the included demos in order to see the various script components required to perform various tasks.



For streaming video from an input device to unity, the following is required:

- A DeckLink capture card installed on your computer
- An input device (such as a HD camera) connected to the input slot of your DeckLink card.
- A DeckLinkInput script added to your scene, and configured correctly for your specific DeckLink card and input device. Section 5.2.1 DeckLink Input Component discusses in depth how to configure this component.
- A display component, for displaying the input stream being captured by the DeckLinkInput component. There are three display components included, namely ApplytoMaterial, IMGUIDisplay, and uGUIDisplay. The usage of these components are discussed fully in sections 5.2.3 ApplytoMaterial Component, 5.2.4 IMGUIDisplay Component, and 5.2.5 uGUIDisplay Component respectively.

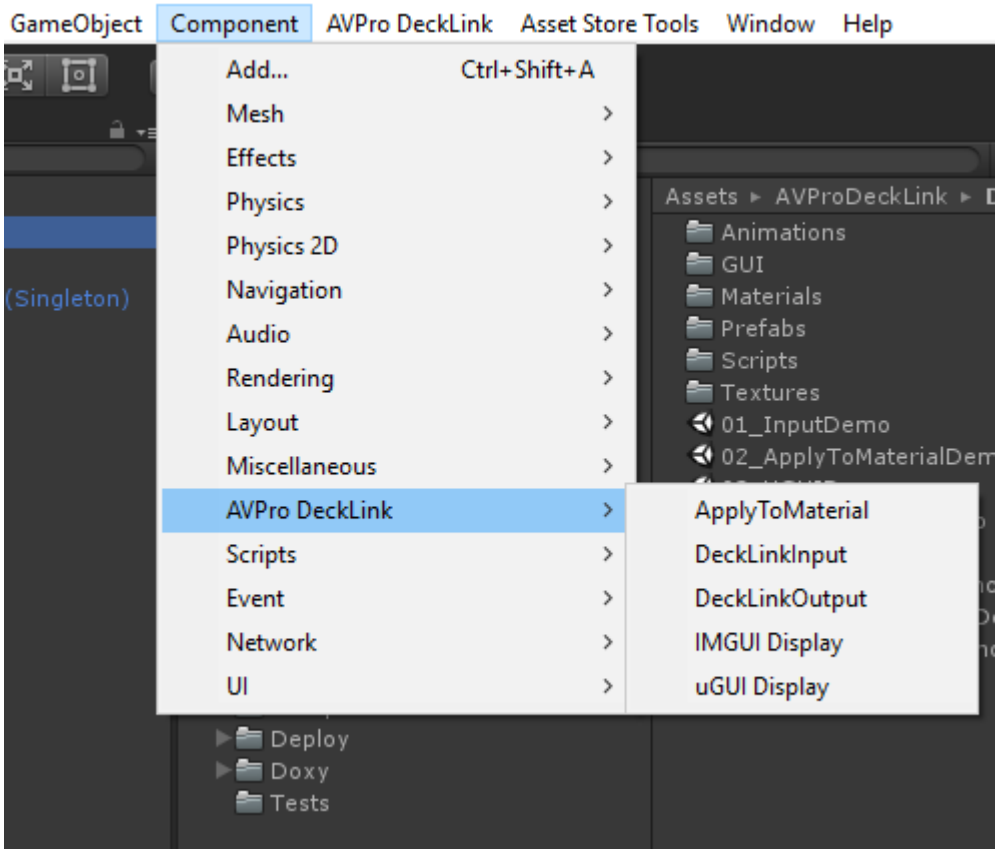
For streaming video from unity to a decklink capture card, the following is required:

- A DeckLink capture card capable of output installed on your computer
- An output device (such as a monitor, or another capture card) connected to the output slot of your DeckLink card.
- A DeckLinkOutput component added to your scene, and configured correctly for your specific DeckLink card and output device. Section 5.2.2 DeckLink Output Component provides an in depth discussion on how to configure this component.
- A camera object in your unity scene (in addition to any other cameras you may want to use). The rendered frames of this camera are sent to the DeckLink capture card. Information on how to couple the camera with the DeckLinkOutput component is provided in section 5.2.2 DeckLink Output Component.

5.2. Components

There are a total of five components included in this plugin that the user can use to easily achieve video streaming with a capture card. These components can be split into two groups: the DeckLink scripts, consisting of the DeckLinkInput and DeckLinkOutput components, and the display scripts, which consist of the ApplyToMaterial, IMGUIDisplay, and uGUIDisplay components.

The DeckLink scripts serve as a high level representation of a DeckLink capture card, allowing the users to read or send video streams to an installed DeckLink capture card. As these scripts represent the actual DeckLink capture cards, each capture card is limited to one of these components. If multiple DeckLink components are assigned to the same physical DeckLink card, only one of the components will work, whereas the rest will issue warnings.

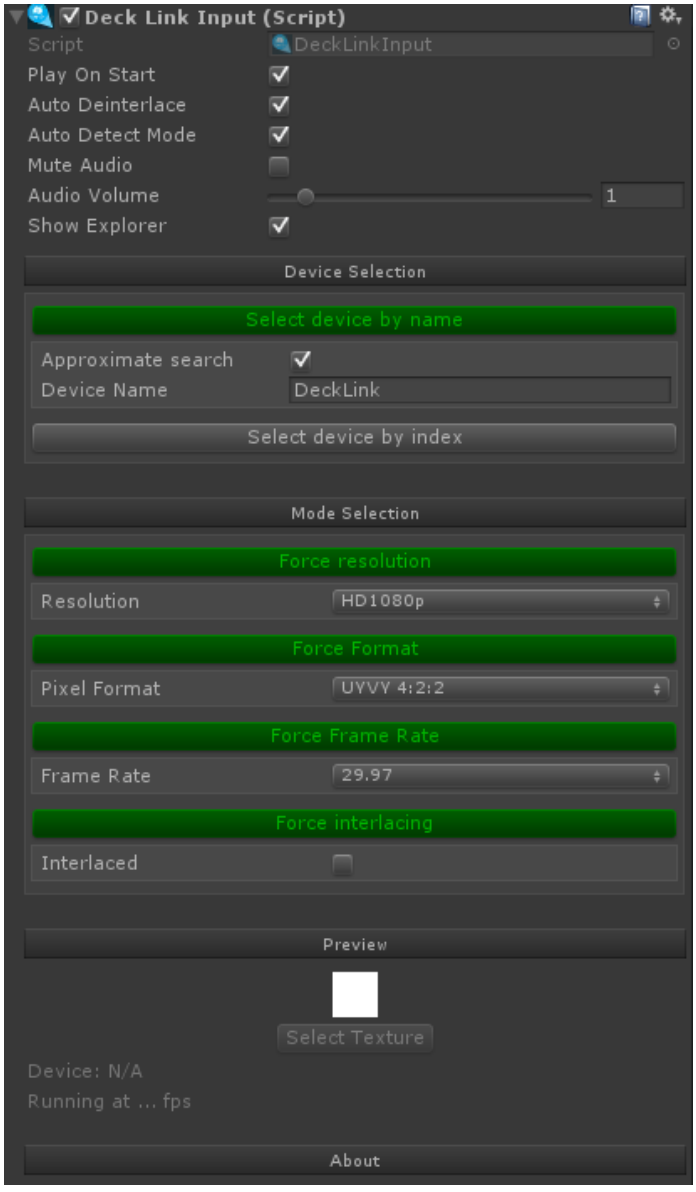


The display scripts serve to display input video information obtained from the DeckLinkInput component in a variety of ways. For example, the uGUIDisplay components displays the video stream onto a uGUI gameObject, whereas the ApplyToMaterial script writes the video stream to a material, which can then be used on various GameObjects.

These components are located in AVProDeckLink/Scripts/Components, or alternatively can be added via the components menu.

5.2.1 DeckLinkInput Component

The DeckLinkInput component is responsible for receiving an input video stream from a DeckLink device. This input video stream can then be displayed using one of the display components. The main task that this component performs behind the scenes is format conversion between the acquired raw video data from the DeckLink capture card, and the native Unity texture format.



In order for this component to correctly read and convert input from a DeckLink capture card, it has to be configured correctly. The configuration settings are as follows:

- **Play On Start:** Whether or not to start reading the input stream when the scene launches. If this is set to false, the input stream can be started by acquiring the DeckLinkInput object in script, and calling Begin().
- **Auto Deinterlace:** Will automatically deinterlace interlaced input when this is checked.
- **Auto Detect Input:** Certain DeckLink cards are capable of automatically detecting the format of the input device. Specifically, it is able to detect if a YUV or RGB input stream is being received. If this option is checked, the mode of the DeckLinkInput is set to that of either YUV 4:2:2 or ARGB 32 on supported cards.
  It should also be noted that if auto detect input is checked and the capture card is capable of auto detection, the input format will automatically revert to the automatic detected format if changed to an incompatible one.

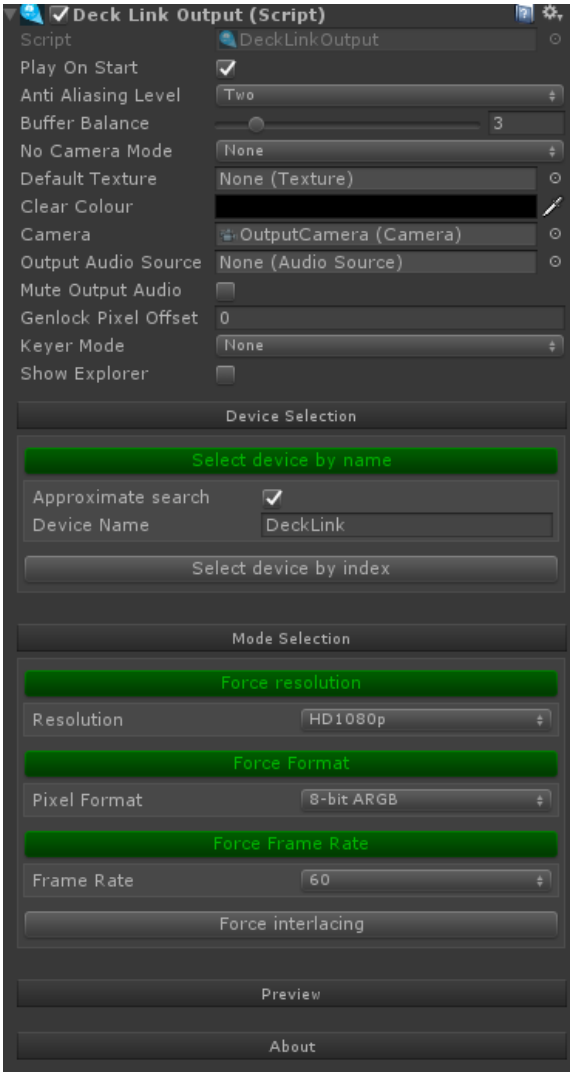
- **Mute Audio:** Mutes the audio source attached to the DeckLinkInput object (this is automatically created and hidden. It can be accessed via script using the InputAudioSource property of the DeckLinkInput component).
- **Audio Volume:** Sets the volume of the input audio. 1 is the default audio level captured.
- **Show Explorer:** Whether or not to show the device explorer for the current object at runtime. This allows the users to easily switch between devices and modes when the system is running.
- **Device Selection:** Provides preference options for the DeckLink device. These preference options are as follows
    - Select device by name - Finds a device that is the exact inputted string, or containing the inputted string string if “Approximate search” is checked.
    - Select device by index - Finds the nth device, where n is the inputted number. If not found or the device is busy, it will try find the closest free device. If no device is free, none will be selected.
    - It should be noted that the index is not the global device index, but rather the index of the already filtered device list. For example, selecting with name “Intensity Pro” and index 10 will find the 10th DeckLink card with Intensity Pro in its name.
    - The system will always look to find free devices, rather than already running ones. The reason for this is because there can only be one running component for each DeckLink device. If no valid device is found, the component is not run.
- **Mode Selection:** Allows user to specify mode settings, such as resolution, pixel format, frame rate, and interlacing. The selected mode will then match the specified parameters. If no mode matching the specified options is found, no mode will be chosen and will have to be manually set. It should also be noted that if Auto detect is selected and is supported by the DeckLink card, the mode selection options are ignored.
- **Preview:** Shows preview of input, as well as provides some information about the device and modes running. Also allows the user to pause the received input video stream.
- **About:** Provides miscellaneous information about RenderHeads, guidelines for submitting bugs, and links to the Unity store page, online documentation, and email support.

As mentioned previously, a one-to-one mapping is enforced between the DeckLink components and the available DeckLink capture cards. In the case that you want to display the input video stream in multiple instances, you can simply use either multiple display components, or in the case of the ApplyToMaterial component, have multiple objects use the updated material.

An exception to the above mentioned rule is if the DeckLink device supports full-duplex mode, in which case the same device can be used simultaneously for both capture and playback.

5.2.2 DeckLinkOutput Component

The DeckLinkOutput component is responsible for sending an input video stream rendered by a Unity camera, converting it to the desired format, and streaming it to the selected DeckLink card.



Much like the DeckLinkInput component, the DeckLinkOutput component needs to be properly configured in order for the output device to correctly read it. The configurable options for this component are as follows:

- **Play On Start:** Whether the component will stream output to the DeckLink device when the app starts. If this is unchecked, you can start sending data by acquiring the DeckLinkOutput object, and calling Begin().
- **Anti Aliasing Level:** Sets the anti-aliasing level of the output stream. The output video stream will have less aliasing with a higher number.
- **Camera:** The camera that you wish to stream the output of. In the case that this is property is not set and the DeckLinkOutput component is attached to a camera, it will assume that the camera it is attached to is intended as the output video stream provider. If you wish to change the output camera during execution, simply assign the

desired output camera's TargetTexture property to the InputTexture property of the DeckLinkOutput device.
One should note that you still need a camera that is not associated with the DeckLinkOutput script should you wish to see the rendered scene on your computer (ie. not sent to the DeckLink card).

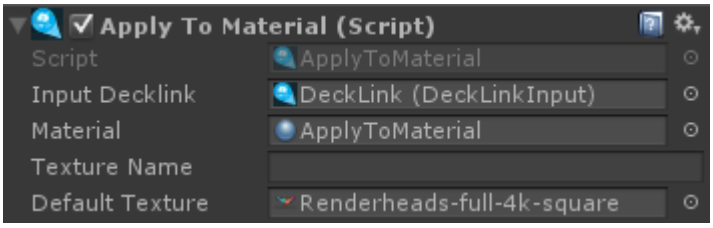
- **Buffer Balance:** How many frames in the DeckLink output queue the plugin will try to balance around at. A lower number will result in less latency but possibly result in hiccups now and then, whereas a higher number will have no frame drops but will have higher latency. We recommend the balance to be 3-4.
- **No Camera Mode:** Specify what the playback should output when no camera is connected. Users can choose between None (output is whatever the last proper frame was), Colour, and DefaultTexture.
- **Default Texture:** Texture to output if no camera is connected and No Camera Mode is set to DefaultTexture
- **Default Colour:** Default colour to output if no camera is selected and No Camera Mode is set to Colour.
- **Genlock Pixel Offset:** Offset of genlock in pixels. This number is clamped between -511 and 511 unless the device selected supports full frame genlock pixel offset, in which case the number is clamped according to half the number of pixels in the frame.
- **Output Audio Source:** Audio source that you wish to output. If no audio source is selected, the scene's audio listener is used. If no audio listener is found, an audio listener will be added to the scene and used.
- **Mute Output Audio:** Mutes the audio being outputted.
- **Keying Mode:** Choose whether to use one of the in-built keying modes of the DeckLink card. These modes are Internal (alpha-blends output data with an input video stream) and External (separate video and key signals, which can then be subsequently used on a third-party keyer). Some care should be taken when doing keying with UI elements, as the assets will in most cases be either interpolated or anti-aliased, resulting in the alpha channel being larger than the original asset. This will result in a halo around your asset if the background colour is not the same as the foreground colour.
- **Show Explorer:** Same as DeckLinkInput
- **Device Selection:** Same as DeckLinkInput
- **Mode Selection:** Same as DeckLinkInput, with the exception that the output modes are searched instead, and that there is no auto detect for output.
- **Preview Explorer:** Same as DeckLinkInput, except for the fact that the output texture is shown, and that there is no pause button as output streams cannot be paused.
- **About:** Same as DeckLinkInput.

NB: It should noted that if multiple playback devices are used, the user is required to turn on multi-output mode for seemless playback. See 5.2.6 DeckLinkSettings Component for more information on this.

Another very important note is that in order for the DeckLinkOutput to work without dropping frames, vsync is automatically turned off when DeckLinkOutput components are in the scene. A reference counter is used so once the last DeckLinkOutput component is removed from the scene, the previous vsync options are restored.

5.2.3 ApplyToMaterial Component

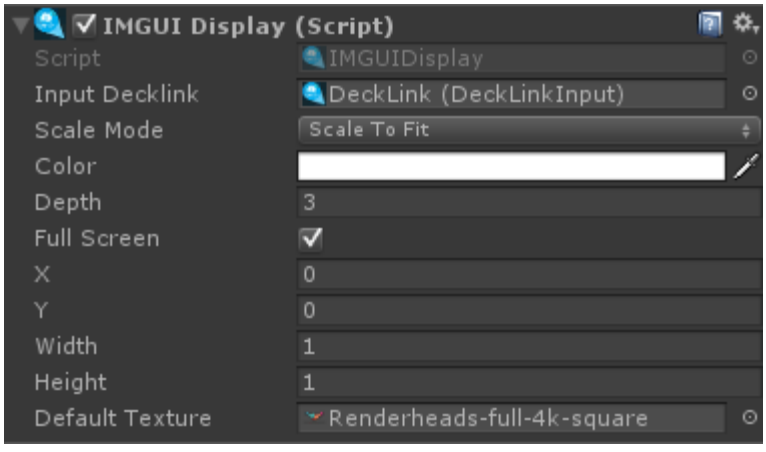
The apply to material component is used to display the video stream obtained from a DeckLinkInput component onto a material. This material can then be used on 3D objects within the scene in order to display the input video.



The properties for the ApplyToMaterial component are as follows:

- **Input Decklink:** Gameobject containing the InputDeckLink component that you wish to apply to a material.
- **Material:** Material that you wish to apply the input video stream to.
- **Texture Name:** Name of the texture on the material. See Material.SetTexture for information about possible names. If this property is left blank, the texture set defaults to the main texture of the material.
- **Default Texture:** Default texture to be shown in the case of an error or a not yet started stream.

5.2.4 IMGUIDisplay Component



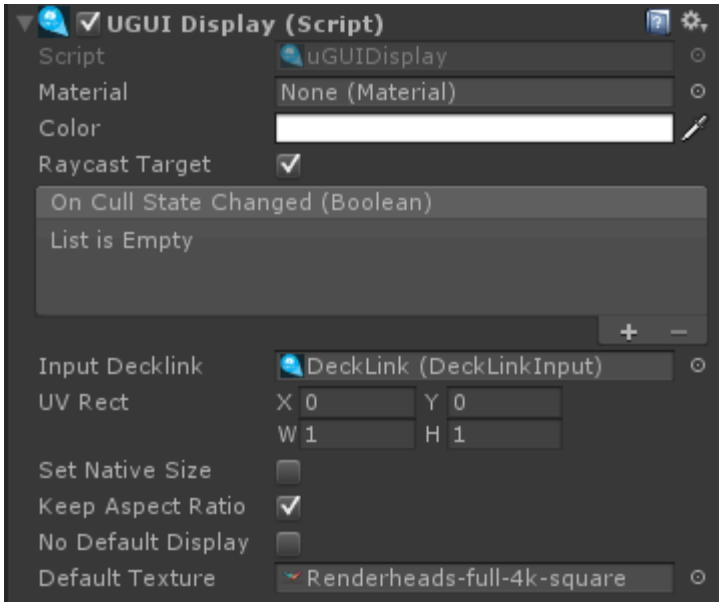
The IMGUIDisplay component displays the video obtained from a DeckLinkInput component using Unity's legacy IMGUI system. This component can be configured in the following ways:

- **Input Decklink:** The gameobject containing the InputDeckLink component that you wish to display.
- **Scale Mode:** There are three options for scale mode: Scale to Fit, which preserves the video aspect ratio by padding the sides; Stretch to Fill, which stretches the video according to the desired dimensions; and Scale and Crop, which maintains the video aspect ratio by cropping the edges.
- **Color:** Color that the input video is blended with.
- **Depth:** GUI depth of the video displayed. If this depth is higher than other GUI elements, then it will be drawn behind them, and if lower, will be drawn in front of them.
- **Full Screen:** Whether or not the video should take up the full screen. If this is unchecked, the dimensions of the display will be determined by the x, y, width, and height properties.
- **X, Y:** The starting coordinates of the display rectangle, with (0, 0) being the top left of the window, and (1, 1) being the bottom right. These coordinates will only be used if Full Screen is unchecked.
- **Width, Height:** The width and height of the display rectangles, with (0, 0) being 0 width and height, and (1, 1) being the width and height of the window. These dimensions will only be used if Full Screen is unchecked.

Unlike the uGUI and ApplyToMaterial components, this component only allows for a single display.

5.2.5 uGUIDisplay Component

The uGUIDisplay component displays the video from a DeckLinkInput onto a uGUI object, such as a canvas.



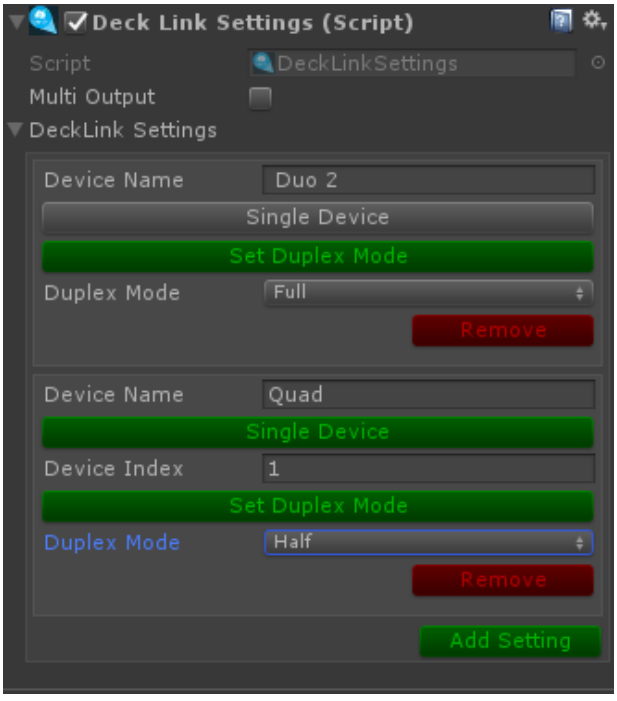
The parameters for this component are as follows:

- **Material:** The Unity UI.Graphic material set by user.
- **Color:** The base color that is blended with the video stream.
- **Input Decklink:** Gameobject containing the DeckLinkInput component that you wish to display.
- **UV Rect:** Determines the offset and tiling of the video onto the display. X and Y determine the offset, whereas width and height determines tiling.
- **Set Native Size:** If checked, the video will be run at native resolution, otherwise it will be scaled according to the UV Rect.
- **Keep Aspect Ratio:** If checked, resizes the display so that it is the same as the video stream's aspect ratio.
- **No Default Display:** Does not show default texture if checked.
- **Default Texture:** Texture shown when the input stream has no signal or is not running.

This component can be added to as many uGUI objects as desired, with each uGUI object acting as a separate display. Furthermore, if one wishes to resize or reposition the display, resizing or repositioning the uGUI element will achieve the desired effects.

5.2.6 DeckLinkSettings Component

The DeckLinkSettings component is currently quite barebones in its functionality, and serves only as a method to configure the duplex mode of devices, as well as a toggle for the output mode of the playback.



This component currently has 2 general properties:

-Multi Output: Toggles multi output mode. When multi output mode is on, multiple DeckLinkOutput components are able to playback simultaneously on arbitrary modes, at the cost of some smoothness in the output video stream.
-DeckLink Settings: Currently, this property is only used to configure the duplex mode of devices. One searches for devices in a similar manner to the DeckLinkInput and DeckLinkOutput devices, and then specifies the duplex mode rule for those devices. There are plans to expand this property to support more configurations in the future.

5.3 Scripting

5.3.1 Namespace

All scripts use the namespace RenderHeads.Media.AVProDeckLink so be sure to add “using RenderHeads.Media.AVProVideo” to the top of your source files.

5.3.2 DeckLink Input Scripting

Controlling the input stream occurs through the DeckLinkInput component. This class exposes the following interface:

- *Begin()* - establishes a connection to the DeckLink device specified by DeviceIndex, and starts reading with the mode specified by ModeIndex.
- *Pause()* - pauses reading from the current input device.
- *Unpause()* - resumes reading from the current input device.
- *StopInput()* - stops reading from the specified DeckLink device, closes the connection, and frees up additional resources.

5.3.3 DeckLink Output Scripting

Like the DeckLinkInput component, the DeckLinkOutput component can be interacted with in script to control the output video stream. This class exposes a very similar interface to DeckLinkInput, with the following being exposed:

- *Begin()* - establishes a connection to a DeckLink device specified by DeviceIndex, and starts sending output to it with the mode specified by ModeIndex.
- *StopOutput()* - stops outputting to specified DeckLink device, closes the connection, and frees up additional resources.

5.3.4 Obtaining device and mode information

The DeckLinkOutput and DeckLinkInput classes all take device and mode indices in order to determine what device and mode they are reading from or sending to. As these indices may not be very descriptive, AVProDeckLink provides methods for obtaining device and mode information.

The DeckLink class provides a couple of static methods in the form of:

- *int GetNumDevices()*, which queries the number of available DeckLink devices, and
- *Device GetDevice(int deviceIndex),* which gets a specific device when given the corresponding device index.

After obtaining a device, you can then query its properties such as Name and ModelName for device specific information. You can also query its available modes through the following methods and properties:

- *NumInputModes:* total number of input modes for the device.
- *NumOutputModes:* total number of output modes for the device.
- *DeviceMode GetInputMode(int index):* Gets the input mode associated with the index
- *DeviceMode GetOutputMode(int index):* Gets the output mode associated with the index

After obtaining a mode, you can query its properties for further details such as Width, Height, FrameRate, and PixelFormat.

5.3.5 Using Device class directly

Although not recommended for all but very advanced users, it is possible to use the Device class directly, instead of using the various components provided. This may be desired in order to achieve a greater amount of customisation and control.
To achieve this, one needs to acquire the device using an index, set the input or output texture for the device by calling `DeckLinkPlugin.SetTexturePointer(int deviceIndex, System.IntPtr texturePointer)` or `DeckLinkPlugin.SetOutputTexturePointer(int deviceIndex, System.IntPtr texturePointer)` respectively, and then starting the device. The output texture should be converted to the relevant format for the selected mode.

For input, the user needs to perform two tasks every frame, namely sending the input event to the plugin via GL.IssuePluginEvent with the signal `PluginID | PluginEvent.UpdateAllInputs`, and converting the data obtained from the sent input texture to a Unity readable format.

For output, the user is required to convert the desired output texture to the desired format, and then copy that data to the passed through texture. The system also needs to send the event `PluginID | PluginEvent.UpdateAllOutputs` at the correct framerate to the plugin using GL.IssuePluginEvent at the correct framerate in order to achieve smooth output.

5.3.6 Documentation

This section only provided a very rough outlook on the available scripts. Full documentation on the various classes within the plugin can be found here.

6. Asset Files

6.1 Demos

- **01_InputDemo.unity**
  - Demo demonstrates how to receive input using a DeckLinkInput component, and display it using an IMGUIDisplay component. It also shows how to obtain device and mode information from a DeckLinkInput.
- **02_ApplyToMaterialDemo.unity**
  - Demo demonstrates how to map the input received from a DeckLinkInput component to a material using the ApplyToMaterial component.
  - Spawns multiple objects in the scene that use the mapped material.
- **03_UGUIDemo.unity**
  - Demonstrates how to display input received from a DeckLinkInput component onto a uGUI object using the uGUIDisplay component.
- **04_InputExplorerDemo.unity**
  - Shows how to get a list of devices, their modes, and how to switch modes in script.
  - Creates an explorer window for every device
- **05_OutputDemo.unity**
  - Demonstrates how to output video to a specified DeckLink device.
  - Has a device explorer that allows users to customise the device and modes at runtime. Also shows buffer information to check for possible framedrops.
- **06_InternalKeyingDemo.unity**
  - Demonstrates the unity automatic internal keying feature that is available on supported hardware.
  - Overlays the RenderHeads logo on top of a received inputstream.
  - Has the same device and mode explorer in OutputDemo.
- **07_InputModifyOutputDemo.unity**
  - Demonstrates how to read input into a unity scene, and then output that scene to a DeckLink card.
  - Requires separate devices for input and output in order to work.
  - Provides an explorer for selecting device and mode
- **08_ExternalKeyingDemo.unity**
  - Same as 07_InternalKeyingDemo, except with the keying mode set as external.

6.2 Prefabs

- **InputIMGUI.prefab**
  - Prefab containing a DeckLinkInput component and an IMGUIDisplay component that displays input obtained from it.
- **InputMaterial.prefab**

    - Prefab containing a DeckLinkInput component together with an ApplyToMaterial component that applies the input video stream onto a the material DeckLinkInputMaterial.
- **InputuGUI.prefab**
    - Prefab containing a DeckLinkInput component, as well as an uGUI component that displays the input video stream on a canvas object.
- **Output.prefab**
    - Prefab is a camera that has a DeckLinkOutput component, which will be used to sent the camera's rendered data to an output DeckLink device.
- **DeckLinkManager.prefab**
    - Prefab for manager class that deals with sending signals to native plugins, and loading devices.
- **DeckLinkSettings.prefab**
    - Prefab for configuring both the multi-output mode, as well as the duplex modes of the plugin.

6.3 Scripts

- **Components**
    - **ApplyToMaterial.cs**
        - Applies input from a DeckLinkInput to a material
    - **DeckLinkInput.cs**
        - Reads input video data from a DeckLink device
    - **DeckLinkOutput.cs**
        - Sends output (typically rendered from a camera) to a DeckLink device
    - **IMGUIDisplay.cs**
        - Displays input obtained from a DeckLinkInput component using Unity's IMGUI system
    - **uGUIDisplay.cs**
        - Displays input obtained from a DeckLinkInput component using Unity's uGUI system.
    - **DeckLinkSettings.cs**
        - Used to configure the plugin and devices
- **Internal**
    - **DeckLink.cs**
        - Parent class for DeckLinkInput and DeckLinkOutput. Implements common functionalities.
    - **DeckLinkAudioOutput.cs**
        - Script to grab the output from an audio source or listener, and send it to the specified DeckLink devices.
    - **DeckLinkManager.cs**
        - Singleton class that is responsible for handling shaders and sending signals to the native plugin.
    - **Device.cs**
        - High level representation of a DeckLink card. Used to control input and output.

    - **DeviceExplorerManager.cs**
        - Singleton script used to render device explorers.
    - **DeviceMode.cs**
        - Script representation of a Device's mode. Contains information such as resolution, pixel format, and fps.
    - **FormatConverter.cs**
        - Responsible for converting the pixel format from an DeckLinkInput stream.
    - **PrefabSingleton.cs**
        - Prefab version of singleton class. Used by DeviceExplorerManager.
    - **Singleton.cs**
        - Singleton class. Used by DeckLinkManager and DeckLinkSettings.
- **Editor**
    - **DeckLinkEditor.cs**
        - Parent editor class for DeckLinkInputEditor and DeckLinkOutputEditor. Contains common logic such as mode and device displays.
    - **DeckLinkInputEditor.cs**
        - Editor script for DeckLinkInput components.
    - **DeckLinkOutputEditor.cs**
        - Editor script for DeckLinkOutput components
- **DeckLinkPlugin.cs**
    - Wrapper for the native plugin.
- **Helper.cs**
    - Stores plugin version

7. Supported Input and Output Formats

Supported input formats:

- 8-bit UYVY 4:2:2
- 10-bit UYVY 4:2:2 (V210)
- 8-bit ARGB
- 8-bit BGRA

Supported output formats:

- 8-bit UYVY 4:2:2
- 10-bit UYVY 4:2:2 (V210)
- 8-bit BGRA
- 8-bit ARGB
- 10-bit RGBX big endian
- 10-bit RGBX little endian
- 10-bit RGB big endian

*Other formats may also be supported through hardware conversion

8. Support

If you are in need of support or have any comments/suggestions regarding this product please contact us.

Email: unitysupport@renderheads.com
Website: http://renderheads.com/product/avpro-decklink/
Unity Forum: http://goo.gl/E8YHU8

8.1 Bug Reporting

If you are reporting a bug, please include any relevant files and details so that we may remedy the problem as fast as possible.

Essential details:

- Error message
  - The exact error message
  - The console/output log if possible
- Hardware
  - DeckLink device model
  - Input / output device information
- Development environment
  - Unity version
  - Development OS version
  - AVPro DeckLink plugin version
- Video details:
  - Resolution
  - Codec
  - Frame rate
  - Interlaced / Non-interlaced

Better still, send us a full or reduced copy of your Unity project

9. About RenderHeads Ltd

RenderHeads Ltd is an award-winning creative and technical company that has been designing and building cutting edge technology solutions since its formation in 2006. We specialise in creating interactive audio-visual software for installations at auto shows, museums, shows and expos.

11.1 Services

- Unity plugin development and consulting
- Bespoke software development:
    - High-end technology for events and product launches
    - Interactive installations and educational museums
    - Video walls
    - Virtual reality experiences
    - Augmented reality apps

9.2 Our Unity Plugins

Many of the apps and projects we develop require features that Unity doesn't yet provide, so we have created several tools and plugins to extend Unity which are now available on the Unity Asset Store. They all include a **free trial or demo version** that you can download directly from the website here:
http://renderheads.com/product-category/for-developers/

9.2.1 AVPro DeckLink

Integrates DeckLink capture card functionality into Unity, allowing users to send and receive high-definition uncompressed video data to and from these capture cards.

9.2.2 AVPro Video

Powerful cross-platform video playback solution for Unity, featuring support for Windows, OS X, iOS, Android and tvOS.

9.2.3 AVPro Movie Capture

Video capture to AVI files direct from the GPU and encoded to files using DirectShow codecs. Features include 4K captures, lat-long (equirectangular) 360 degree captures, off-line rendering and more. Windows only.

9.2.4 AVPro Live Camera

Exposes high-end webcams, tv cards and video capture boards to Unity via DirectShow. Windows only.

Appendix A - Release History

- **Version 1.6.6 - 9 January 2018**
  - Features
    - Added support for input frame timecodes
    - Added support for input frame ancillary data
    - Improvements made to input frame buffering
    - Upgraded from DeckLink SDK 10.11.1 to 10.11.4
    - Added color tint to IMGUI component
  - Fixes
    - Improved accuracy of skipped frame detection
    - Fixed issue where some ARGB sources output 0.0 in the alpha channel

- **Version 1.6.3 - 7 October 2018**
  - Fixes
    - Fixed an issue introduces in Unity 2018.2.0 where the timing behaviour has changed which caused the DeckLinkOutput component to generate too many frames when the multi-output option was disabled
    - Fixed an issue that could cause stereo output rendering to sometimes not set the right eye texture correctly
    - Fixed an issue where RGB input modes would never be auto-detected

- **Version 1.6.2 - 23 July 2018**
  - Features
    - Minor UI and garbage collection improvements
    - Upgraded from DeckLink SDK 10.11 to 10.11.1
  - Fixes
    - Fixed a major memory leak when using stereo (3D) mode
    - Fixed some bugs related to 8K resolution selection

- **Version 1.6.0 - 25 June 2018**
  - Features
    - Added support for stereo (3D) input and output
    - Added support for DeckLink 8K Pro capture card
    - Upgraded from DeckLink SDK 10.9.11 to 10.11
  - Fixes
    - Reduced garbage generation in many components and demos
    - Fixed “synced to input” option, by making it wait for the auto-detection input mode to resolve

- **Version 1.5.1 - 5 April 2018**
  - Features

        - Upgraded from DeckLink SDK 10.9.5 to 10.9.11

- **Version 1.5.0 - 4 April 2018**
    - Features
        - Added support for HLG HDR (high dynamic range) and BT.2020 colour-space, including 10-bit output and internal floating point processing
        - Added support for internal and external keying using YUV and other non-alpha modes
        - Added option to synchronise output to input
        - Massive performance improvements
        - Added support for new video modes
        - Added support for Unity 2018
    - Fixes
        - Fixed the device model name always being null
        - Fixed some memory not being released
        - Fixed RGBA shaders to preserve alpha channel

- **Version 1.3.2 - 21 July 2017**
    - Features
        - Added ability to set default textures/colours when no camera is assigned to DecklinkOutput component
        - Improved efficiency of playback slightly
        - Added support for Unity 2017
    - Fixes
        - Fixed nullpointerexception when no camera is assigned to DecklinkOutput
        - Fixed linear space bugs where the conversions were incorrect for both capture and playback.
        - Changed workflow for linear to gamma conversions in order to minimize banding
    - Misc
        - Updated native sdk to 10.9.3

- **Version 1.3.1 - 1 February 2017**
    - Features
        - Added ability to perform device and mode search with values specified at runtime
    - Fixes
        - Fixed issue where output-only card (such as DeckLink Mini Monitor) failed to initialise
        - Fixed audio output deadlock issue
        - Fixed bug where the mode was being displayed incorrectly in the DeckLinkOutput component's preview section

- **Version 1.3.0 - 12 December 2016**

    - Features
        - RGBX10, RGBX10_LE, and RGB10 formats are now supported for playback.
        - Added the ability to flip the X and Y axis on DeckLinkInput components. 04_InputExplorerDemo.unity also updated to reflect this.
    - Fixes
        - Fixed access violation crash when frame drops occurred (typically when pausing the unity system, or stepping through code).
        - Fixed access violation crash that sometimes occurred when shutting down Unity
        - V210 pixel format capture now flips correctly on later versions of Unity
        - Fixed capture on some previous Unity versions where only 1 pixel was being displayed

- **Version 1.2.1 - 18 November 2016**
    - Fixes
        - Moved audio input logic from c# script to native plugin. This should both fix the glitchy audio input, as well as improve performance slightly.
        - DeckLinkSettings now refreshes the plugin before each run so that changes get reflected.

- **Version 1.2.0 - 4 November 2016**
    - Features
        - Support for playback on multiple devices
        - Configurable duplex
    - Fixes
        - Fixed audio buffering bug that would cause exceptions to be thrown at times.
        - Output demo now correctly shows target frame rate

- **Version 1.1.0 - 11 October 2016**
    - Features
        - Audio input and output supported, integrated into Unity's audio system.
        - Support for full duplex cards, allowing for simultaneous capture and playback from a single device.
        - Multi-object editing
        - Updated DeckLink SDK to 10.8
    - Fixes
        - Fixed native plugin settings
        - Fixed colour space build bug
        - Fixed warning messages when switching devices and modes using the explorer
        - Buffer info now shows correctly in output and keying demos
        - Switching devices in output and keyer demos now works properly

        - Fixed output demo explorer to now correctly display the mode and device currently in use

- **Version 1.0.0 - 13 September 2016**
    - Initial release
        - Reduced latency for outputting video streams
        - Device explorer option for DeckLink components
        - Improved device and mode selection
        - Icon update
        - Added support to offset genlock
        - Added support for linear colour spaces
        - Minor bugfixes

- **Version 0.99 - 05 August 2016**
    - Second beta release

- **Version 0.98 - 29 July 2016**
    - Initial beta release

Appendix B - Roadmap

- **Version X.X.X**
    - Timecode & Ancillary data embedding
    - Expand on DeckLinkSettings component
    - Linux port (May move to earlier version depending on interest)
    - macOS port (May move to earlier version depending on interest)
    - Facelift demos
    - Streaming support
    - nVidia GPUDirect support

- **Version X**
    - ← Your suggestions here, let us know :)

Appendix C - Document version history

- **Version 1.5.0**
    - Added FAQ
    - Added new feature notes
    - Reformatting

- **Version 1.3.2**
    - Updated Roadmap
    - Updated Release History
    - Updated DeckLinkOutput component section
    - Added note about external keying and texture filtering

- **Version 1.3.1**
    - Updated Release History
    - Updated Roadmap

- **Version 1.3.0**
    - Updated format support section
    - Updated Release History
    - Updated Roadmap

- **Version 1.2.1**
    - Updated Release History

- **Version 1.2**
    - Updated roadmap
    - Updated Release History

    - Added descriptions for the new DeckLinkSettings component

- **Version 1.1**
    - Updated roadmap
    - Updated Release History
    - Updated descriptions for DeckLinkInput and DeckLinkOutput to include audio information.

- **Version 1.0**
    - Updated roadmap
    - Updated some descriptions and fixed some grammar and spelling mistakes.
    - Updated screenshots
    - Added information on new functionality such as buffer balance and genlock.

- **Version 0.99**
    - Updated certain images and descriptions.

- **Version 0.98 - Initial document**

Appendix D - FAQ

1. **Why isn't DeckLinkOutput maintaining frame rate?**

   Check that you don't have any IMGUI components active. We have noticed that especially on Unity 2017.X this seems to cause frame rate issues that affect the DeckLinkOutput component.