NDI® SDK UNREAL ENGINE®

SDK VERSION 3.8

FOR USE WITH UNREAL ENGINE® 5.1 TO 5.6

Jun 11, 2025

1 OVERVIEW

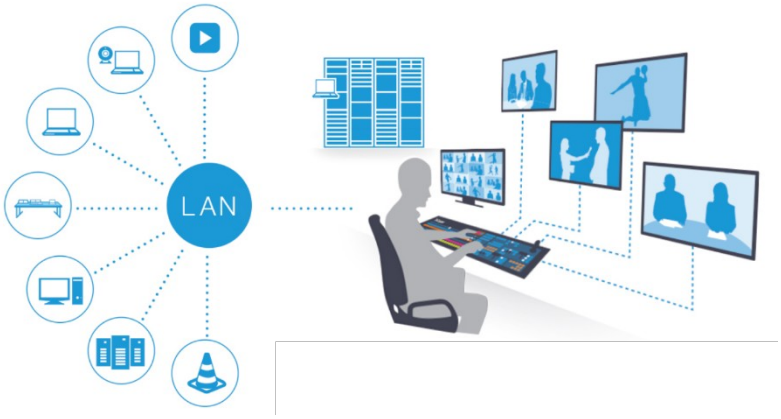
The NDI® (Network Device Interface) standard developed makes it easy to prepare products that share video on a local Ethernet network, and beyond. Its many additional features and capabilities have made it by far the world's most prolific IP protocol for video production and broadcast.

That a/v signals will be predominantly be carried over IP in future is no longer in doubt. All modern video rendering, graphics systems and switchers run on computers; cameras and most other production devices likewise use computer-based systems internally. The vast majority of all such systems are able to communicate via IP – and *NDI is serving this purpose far more often than any other protocol*.

NDI DOESN'T SIMPLY SUBSTITUTE NETWORK CABLES FOR SDI CABLES – IT CHANGES EVERYTHING!

Handling video over networks opens a world of new creative and pipeline possibilities. Consider a comparison: The Internet, too, *could* be narrowly described as a transport medium, moving data from point A to point B. Yet, by connecting *everyone and everything, everywhere*, it is much more than that. Likewise, introducing video into the IP realm with its endless potential connections delivers exponential creative possibilities and workflow benefits.

NDI allows multiple video systems to easily identify and communicate with one another and to encode, transmit and receive many streams of high quality, low latency, and audio in real time.



NDI can operate bi-directionally, and supports many video streams on a shared local network connection. It is resolution and frame-rate independent, supporting 4K and beyond, along with 16 channels and more of floating-point audio and 16-bit video.

NDI also includes tools to implement video access rights, grouping, bi-directional metadata, IP commands, routing, discovery servers, and more. Its superb performance over standard GigE networks makes it possible to transition facilities to an incredibly versatile IP video production pipeline without negating existing investments in SDI cameras and infrastructure, or requiring costly new high-speed network infrastructures. NDI t also revolutionizes ingest and post-production, by making fully time-synced capture on a massive scale a reality.

2 LICENSE

You may use this SDK in accordance with the "NDI SDK License Agreement", available for review from the root of the SDK folder in the SDK distribution. Your use of any part of the SDK for any purpose is acknowledgment that you agree to these license terms. Please note that, for distribution you must also respect the following requirements:

- You may use the NDI library within free or commercial Products (as defined by License) created using this SDK without paying any license fees.

- NDI is a registered trademark of NewTek Inc., a division of Vizrt Group, and should be used only with the ® (registration symbol) as follows:

  NDI®, along with the statement "NDI® is a registered trademark of NewTek, Inc." located on the same page near the mark where it is first used, or at the bottom of the page in footnotes. You are required to use the registered trademark designation only on the first use of the word NDI within a single document.

  Your application's About Box and any other locations where trademark attribution is provided should also specifically indicate that "NDI® is a registered trademark of NewTek, Inc." If you have any questions, please do let us know.

  Note that if you wish to use NDI within the name of your product then you should carefully read the NDI brand guidelines or consult with NewTek.

- Your application **must** provide a link to http://ndi.video/ in a location that is close to all locations where NDI is used/selected within the product, on your web site, and in its documentation. This will be a landing page that provides information about NDI and access to the tools we provide, any updates, and news.

- You may **not** distribute NDI software and tools developed or provided for exclusive distribution by Vizrt Group or any of its constituent divisions (including Vizrt, NDI Group, or NewTek Inc.). To make these accessible to your users you are permitted to provide a link to http://ndi.video/tools/. (For example, users can download "NDI Tools" software suites for various platforms directly from this site.)

- You should include the NDI dlls as part of your own application and keep them in your application folders, so that there is no chance that NDI DLLs installed by your application might conflict with other applications on the system that also use NDI. For this reason, please do not install your NDI dlls into the system path. If you are distributing the NDI dlls, you need to ensure that your application complies with the NDI SDK license terms as discussed in this document and the NDI SDK License Agreement, as well as such applicable third part licenses as may be included with this SDK.

We are interested in how our technology is being used and would like to ensure that we have a full list of applications that make use of NDI technology. Please let us know about your commercial application (or interesting non-commercial one) using NDI by emailing sdk@ndi.video.

If you have any questions, comments or requests, please do not hesitate to let us know. Our goal is to provide you with this technology and encourage its use, while at the same time ensuring that both end-users and developers enjoy a consistent high-quality experience.

3 NDI PLUGIN INSTALLATION

The NDI® Runtime is not included with the 'NDI SDK for Unreal Engine' plugin. To work properly with Unreal® Engine, the NDI® IO Plugin requires the NDI® Runtime to be installed on the host system. Please visit https://ndi.video/sdk and download the latest SDK).

The NDI® IO Plugin install folder includes pre-built versions of the plugin. These can be installed in one of two locations:

- Your Unreal Engine project, or
- Your Unreal Engine install folder

In either case, you must proceed by copying the "NDIIOPlugin" folder to the correct destination as described in one of the following sections.

3.1.1 INSTALLING TO YOUR UNREAL ENGINE PROJECT

Use this install option to make the plugin available selectively (on a project-by-project basis).

- Copy the "NDIIOPlugin" folder from the package matching your preferred Unreal Engine version.
- Paste this folder into the "Plugins" folder of your Unreal Engine project. (If your Unreal Engine Project does not have a "Plugins" folder yet, please create a folder with that name).

After re-starting your Unreal Engine project, the NDI® IO Plugin should appear in your Project's "Plugins" settings as activated.

3.1.2 INSTALLING TO THE UNREAL ENGINE INSTALL FOLDER

Alternatively, use this install procedure to make the NDI® IO Plugin available to **all** Unreal Engine projects on your system.

- Copy the "NDIIOPlugin" folder from the package corresponding to your Unreal Engine version.
- Paste that "NDIIO" folder into the "Plugins" folder of your Unreal Engine install (e.g., the Unreal Engine 5.3 "Plugins" folder is usually located at "C:\Program Files\Epic Games\UE_5.3\Engine".

With your project's editor open, use the file menu item 'Edit > Plugins' to make sure that the 'NDI IO Plugin' is enabled. Restart the editor as necessary. To learn about using the plugin within your application, see Section 9, Using the NDI Blueprints Nodes.

3.2 COMPILE FROM SOURCE

The plugin's full source code is distributed within each version-specific "NDIIO" plugin folder, as well as in the "NDIExamples" folder. Advanced users with Microsoft Visual Studio installed on their system can compile the plugin directly from the source code.

4 SIMPLE SETUP OF BROADCAST AND RECEIVERS

There are a number of ways to add NDI® broadcast of views of the level, and receiving of NDI® streams into the level. The easiest way is to add an "NDI Broadcast Actor" and "NDI Receive Actor" to the level.

4.1 SIMPLE BROADCAST ACTOR

In Place Actors search for "NDI Broadcast Actor". Drag&drop it into the level scene. A camera will appear in the level, and an "NDIBroadcastActor" is added to the World Outliner. The broadcast actor needs an "NDI Media Source" asset which contains details of the stream, such as the stream name and frame size.

Select the broadcast actor in the World Outliner, and look at its Details. In the "NDI IO" section there is the reference to the NDI Media Source. Initially it is set to "None". Use the dropdown menu next to it to create a new "NDI Media Sender" asset.

Play the level and the broadcast actor will begin streaming its view. By default the name of the stream is "Unreal Engine Output", at a resolution of 1920 x 1080 at 60fps. Open, for example, Studio Monitor to view the stream.

The details of the stream can be changed in the settings for the NDI Media Sender asset.

4.2 SIMPLE RECEIVER ACTOR

Setting up a simple NDI® stream receiver actor is similar to a broadcast actor. From “Place Actors” drag&drop an “NDI Receive Actor” into the level. It will be shown in the level as a black screen, and an “NDIReceiveActor” will appear in the World Outliner.

The receiver actor needs to be associated with an “NDI Media Receiver” asset containing the details of what stream to receive. Click on the added “NDIReceiveActor” in the World Outliner, and in the Details look for the “NDI IO” section. In the “Media” sub-section there is a reference to an “NDI Media Source” asset. Use the dropdown menu next to it to create a new “NDI Media Receiver” asset.

Open the settings for the new NDI media receiver asset and fill in the name of the source in the “Source Name”. Play the level, and after a few seconds the stream should appear on the receiver actor (make sure the actor is in view).

4.3 ACTIVE VIEWPORT BROADCASTING

The plugin has the ability to broadcast the active viewport. The settings for this are on the NewTek NDI page of the Plugins section of the Project Settings. By default the broadcast of the active viewport is started through a blueprint function. It is also possible to automatically start the broadcast by enabling “Begin Broadcast On Play” in the plugin's settings.

5 NDI BROADCAST ACTOR

A broadcast actor creates an NDI stream of the view as captured by the actor. It is represented in the level by a camera. Its view is independent of the viewport. Multiple broadcast actors can be used, each with its own stream, to stream multiple views of the scene at the same time.

By default a newly created broadcast actor does not have a media source set. The media source contains information about the stream, such as its name and resolution. The media source for a broadcast actor is an NDI Media Sender asset type. When setting up a broadcast actor, a new NDI Media Sender asset can be created and set, or an existing one can be selected.

The broadcast actor also comes with two components attached: the ViewportCaptureComponent containing the setting of how to capture the view, and a PTZController component that can be used to control the camera as a virtual PTZ device.

5.1 NDI VIEWPORT CAPTURE COMPONENT

The viewport capture component is responsible for capturing the view. It is derived from the standard USceneCaptureComponent2D, and has many settings to control how the view is captured. We shall go over a few that are particularly relevant for NDI® use.

5.1.1 CAPTURE SETTINGS

When overriding the broadcast settings, the view will be rendered at the given size, but scaled to fit the resolution of the stream (as set in the actor's media source).

If the stream outputs alpha values (the actor's media source has Output Alpha turned on), then Alpha Remap can be used to control how the alpha values in the view are translated to alpha values in the NDI® stream. The min

value is the Unreal Engine rendered alpha value that is mapped to an NDI® stream alpha value of 0, while max is the value that is mapped to a stream alpha of 1. All other alpha values are linearly interpolated from those. The min value can be set larger than the max value. The alpha remap values specified in the viewport capture component will overwrite the alpha remap values in the actor's NDI Media Sender asset.

5.1.2 SCENE CAPTURE

The Texture Target is optional. If not set (i.e. it is set to None) then a default transient texture target will be created internally. It should suffice for most uses.

The Capture Source sets how to capture the view. Unreal Engine renders views in a number of passes and can applies various post-processing steps for various effects. The capture source determines during what pass the view is captured. Depending on the capture source some information may or may not be available. For example, in Final Color modes alpha values are not available. While in Scene Color modes post-processing effects such as depth of field is not present.

When using a capture source with alpha (whether obtained from opacity or scene depth), the broadcast actor's media source should have Output Alpha turned on in order for NDI® to stream out an alpha channel. Care should also be taken with setting the Alpha Remap Min and Max value in the Capture Settings section. Notice that some Capture Source modes produce an inverted alpha channel. For those Alpha Remap Min should be set to 1 and Alpha Remap Max to 0 to undo the inversion.

When using a Capture Source mode with scene depth in the alpha channel, set Alpha Remap Max to the largest distance you want represented as 1 in the stream's alpha channel. The Alpha Remap Min can be set to 0 to capture the entire depth range. If you want to use the scene depth as an alpha mask, remember that an alpha value of 0 is fully transparent (and 1 fully opaque). In such cases set the Alpha Remap Min to the maximum depth (far away is transparent), and Alpha Remap Max to 0 (near is opaque). A hard distance cut-off can be achieved by setting Alpha Remap Max to slightly less than the Min.

5.2 NDI PTZ CONTROLLER

The PTZ Controller component allows the virtual camera to be oriented, zoomed, and focused through the NDI® PTZ interface. PTZ control commands are received from the associated broadcast actor's media sender asset, and applied to the viewport capture component.

By default PTZ control is enabled. PTZ control can be disabled and enabled for the actor through the toggle in the PTZ Controller settings. There is also an Enable PTZ toggle in the NDI® media sender, which can be used to disable PTZ control for all actors using that sender.

Minimum and maximum limits can be enabled and set for the pan, tilt, and field of view (zoom). If the minimum and maximum value are the same, then that effectively locks that movement. The directions in which pan and tilt move can also be inverted.

For focus control to have a visible effect the Capture Source for the viewport capture component should be set to a Final Color mode (the application of depth of field is a post-process). Enabling auto-focus sets the depth of field to zero, which Unreal Engine interprets as depth of field being disabled.

The PTZ controller can have a number of Presets. Keep in mind that PTZ controls such as those exposed through the NDI® Studio Monitor application are limited to nine presets. Presets can be pre-defined, or set and modified by receivers of the stream through PTZ controls.

When switching between presets the PTZ controller can be set to smoothly move the camera between current state and preset state. The Preset Recall Easing determines how many seconds it takes to complete the transition. If set to zero, the switch is done instantaneously.

When used with an NDI Broadcast Actor, the PTZ Controller usually uses the actor's media sender asset to get PTZ commands from. This can be overridden by explicitly specifying an NDI Media Sender asset in the PTZ Controller.

5.2.1 NDI PTZ CONTROL ON OTHER ACTORS

The NDI PTZ Controller component can be attached to actors other than the NDI Broadcast Actor. When doing so, a media sender asset should be specified in the PTZ component. Otherwise the controller will not know where to receive PTZ commands from. Only pan and tilt transformations will be applied to the actor. Zoom and focus will be ignored.

6 NDI RECEIVER ACTOR

An NDI® Receiver Actor acts as a screen on which an NDI® stream is played. The stream to be played is specified using an NDI Media Receiver asset.

By default the receiver actor does not have a media receiver asset set as media source. An existing NDI Media Receiver asset can be used, or a new one created, in the NDI IO section of the receiver actor's settings. Multiple receiver actors can share the same receiver asset.

6.1.1 NDI IO SETTINGS

The media source for the actor references an NDI Media Receiver asset. Multiple NDI Receiver Actors can share the same media receiver asset.

The Frame Width and Frame Height scale the display. By default it is set up to fit a 16:9 aspect ratio. If the frames in an NDI® stream have a different aspect ratio, the video will be stretched or squashed (the aspect ratio of the video will not be preserved).

What components of the stream to use for this actor can be controlled by enabling or disabling audio, color, and alpha channels. These settings only affect this actor, not any other actor sharing the same media source.

7 NDI MEDIA ASSETS

NDI streams are represented as assets in the Unreal Engine. This allows the same stream to be used by multiple NDI Receiver Actors. Furthermore, if a sending stream were to be a component or actor instead, problems can occur due to Unreal Engine creating multiple copies during gameplay.

7.1 NDI MEDIA SENDER ASSET

A media sender is an NDI stream going out of the Unreal Engine. It is typically used by an NDI Broadcast Actor to send out rendered views. Only one broadcast actor should be using a given NDI Media Sender asset.

7.1.1 BROADCAST SETTINGS

The Source Name is the name of the stream. It should be unique in order to avoid confusion.

The Frame Size is the desired width and height in pixels of the stream. If the stream is supplied by (for example) an NDI Broadcast Actor with frames of a size different than this, the frames will be scaled (preserving aspect ratio) to fit the specified size.

The Frame Rate specifies the desired framerate of the stream. Due to the nature of realtime rendering with Unreal Engine, it is not guaranteed that frames will actually be rendered and broadcast at this rate. Generally it should be set to no more than what Unreal Engine can manage for the scene.

The stream can optionally include an alpha channel. If the stream outputs alpha values, then Alpha Remap can be used to control how the alpha values in the Unreal Engine generated frame are translated to alpha values in the NDI® stream. The min value is the Unreal Engine rendered alpha value that is mapped to an NDI® stream alpha value of 0, while max is the value that is mapped to a stream alpha of 1. All other alpha values are linearly interpolated from those. The min value can be set larger than the max value.

PTZ control through the stream can be enabled and disabled for the NDI® Media Sender Asset. This has a slightly different effect than enabling or disabling the NDI® PTZ Controller on an NDI® Broadcast Actor. The toggle in the NDI® Media Asset tells the NDI® receivers of the stream whether or not PTZ is enabled or not (which in turn, for example, will cause Studio Monitor to display PTZ controls). The toggle in the broadcast actor PTZ controller component determines if the actor should make use of the PTZ commands.

7.1.2 CONTENT

The Render Target contains the texture that will be encoded and sent over NDI®. Typically it will be provided by the viewport capture component of the NDI® Broadcast Actor using this sender, so there is no need to set it manually.

7.2 NDI MEDIA RECEIVER ASSET

A media receiver asset is a stream coming into Unreal Engine. It is used by NDI Media Receiver Actors to receive decoded frames from an NDI® stream. The receiver asset can be used by multiple receiver actors.

7.2.1 SETTINGS

The Connection Information specifies how to connect to an NDI® stream. The name of the stream can be specified either in Source Name (using the “Machine Name (Stream Name)” format), or as an NDI Url.

The Bandwidth controls what components of the stream the receiver requests.

Audio and video can be (temporarily) muted. When muted the data is still received, but not processed by the receiver asset.

7.2.2 CONTENT

The Video Texture is the texture to which a received frame is decoded. It is optional. If not set, then an internally generated video texture is used.

7.3 NDI MEDIA RECEIVER AND MEDIA IO FRAMEWORK

NDI Media Receivers can be used with the Unreal Engine Media IO Framework. They can be added to Media Bundles, and the Media Player can play NDI® streams.

The Url for playing an NDI® stream in the Media Player is of the form: ndiio://MachineName (StreamName)

Existing NDI Media Receiver assets can also be used in the Media Player and added to playlists.

NDI Media Receivers can also be created in Media Bundles, which can then be used to create media bundle actors in the level. This provides an alternative to adding NDI Receiver Actors to the level directly.

7.4 NDI TIMECODE PROVIDER

The NDI® IO Plugin contains a timecode provider class, which can present the timecodes of an NDI stream to Unreal. The NDI Timecode Provider uses an NDI Media Receiver asset as the source of the timecodes.

In order to work, the video stream of the receiver must contain timecodes. NDI® does not guarantee that all video streams will have valid timecode values. The timecodes produced by the provider are those of the most-recently obtained frame.

8 GETTING STARTED WITH EXAMPLE BLUEPRINT PROJECTS

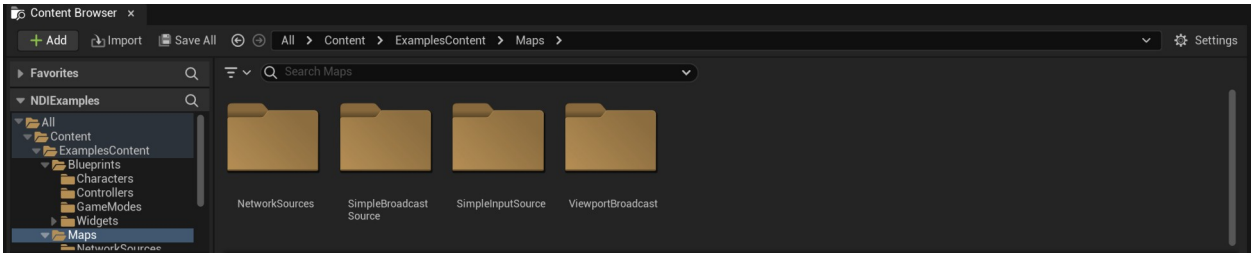
The NDI® IO Plugin install folder contains an example Unreal project containing maps for common use cases like creating NDI output streams of the Unreal viewport and showing NDI input on a texture within an Unreal map.

The example projects are C++ Unreal Engine projects, so you will need Visual Studio 2019 to be installed on your system (the Community Edition is available for download for free on the Microsoft website). It is recommended to copy the “NDIExamples” folder from the NDI SDK for Unreal installation directory to a folder of your choosing.

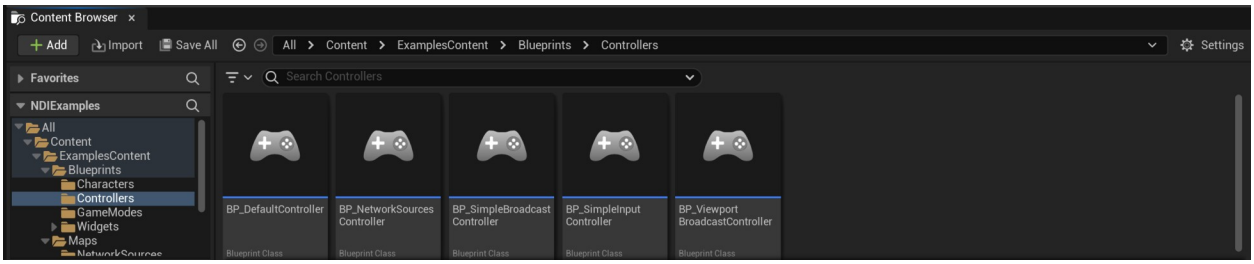
Double-click on the “NDIExamples.uproject” file in the “NDIExamples” folder to open the project in Unreal Engine.

Hint: You can switch between Unreal Engine versions by right-clicking on the “NDIExamples.uproject” file and selecting the “Switch Unreal Engine version…” entry from the context menu.

Afterward, you may load any of the example maps from the “Maps” folder in the Unreal Content Browser.

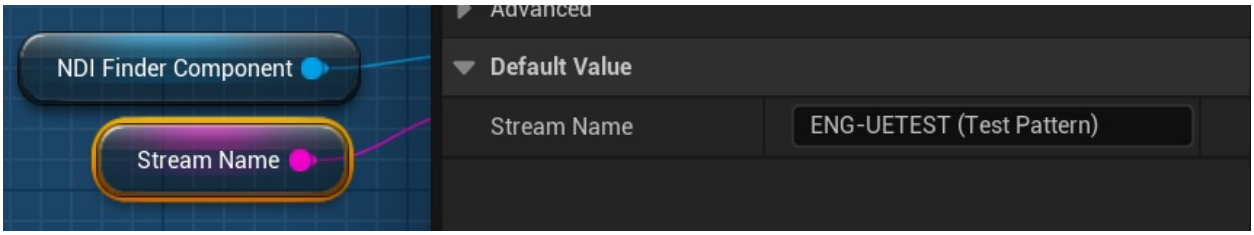


To open and experiment with NDI Blueprint setups, navigate to the Game Controller Blueprints in the “Blueprints/Controllers” folder using the Unreal Content Browser.



Some Blueprints require editing NDI stream names in order for a specific stream to be shown in the map (for example, in the "BP_SimpleInputController").

In such cases, select the "Stream Name" Blueprints node, and edit the Default Value to reflect the input NDI stream you would like to show.



Please consult the next section regarding using the NDI Blueprints nodes to set up such Blueprints from scratch.

9 USING THE NDI BLUEPRINTS NODES

This section provides a tutorial-style introduction to the features of the NDI® IO Plugin for Unreal Engine® using Blueprints.

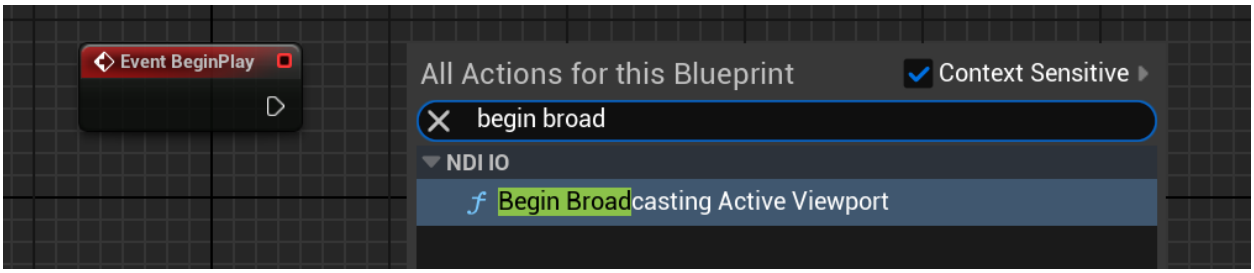
Completing these steps will allow you to broadcast the Active Viewport, create a simple Input Source to be viewed in the virtual world, and create a simple Broadcast Source to be viewed by an NDI® receiver of your choice.

NOTE: To work properly with Unreal® Engine, the NDI® IO Plugin requires you to have the NDI® Runtime installed on your system. If necessary, please consult the previous section of this document to confirm you have correctly installed the appropriate 'NDI IO Plugin'.

9.1 BROADCAST ACTIVE VIEWPORT

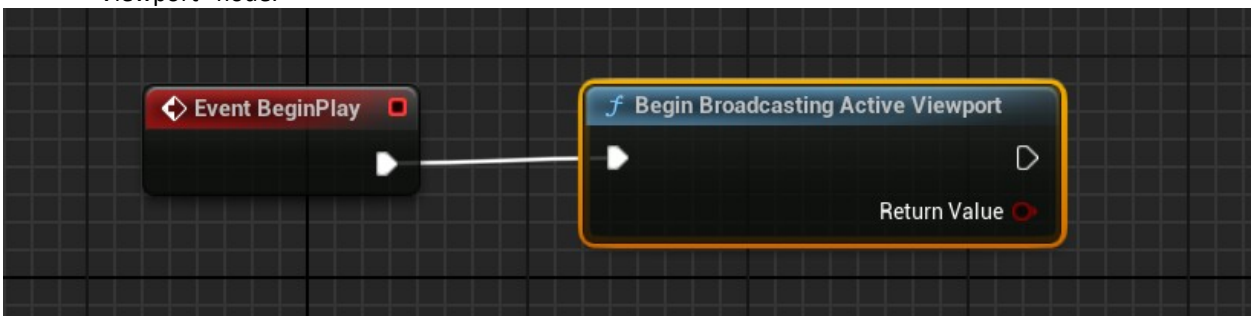
To begin:

- Click on Blueprints > Open Level Blueprint – to open the Blueprint Level Script for the current level.
- Then add a new Blueprint Node by right-clicking on the blueprint background, and searching for "Begin Broadcasting Active Viewport."



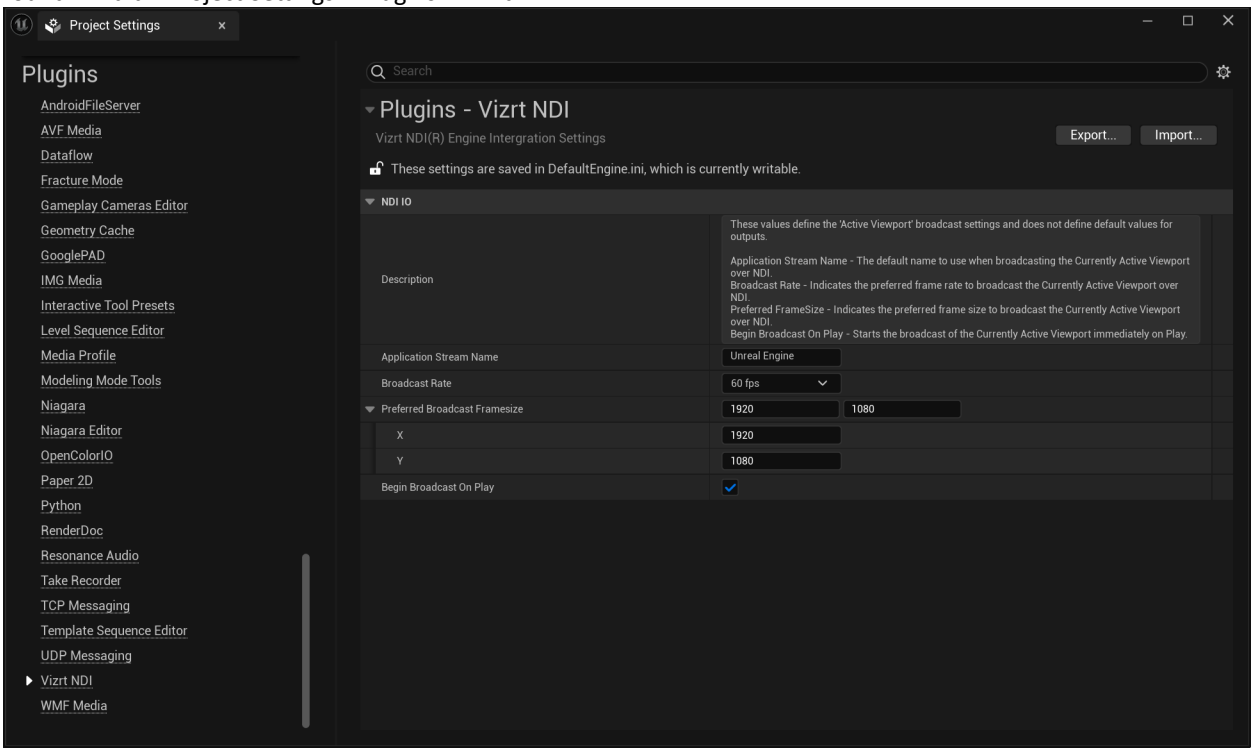
9.1.1 CONNECTING NODES

- After creating the blueprint node, connect the “Begin Play” node to the “Begin Broadcasting Active Viewport” node.



9.1.2 MODIFYING SETTINGS

The default broadcast settings for using “Begin Broadcasting Active Viewport” are set using the Plugin Settings found in Edit > Project Settings > Plugins > Vizrt NDI.



9.1.3 RUN PROJECT

Now that you have changed the plugin settings, it's time to start the project.

- Press the Play button. At this point you can expect to see your viewport output in your NDI® receiver of choice.

Congratulations – you have learned how to broadcast the active viewport using NDI® IO Plugin for Unreal Engine®.

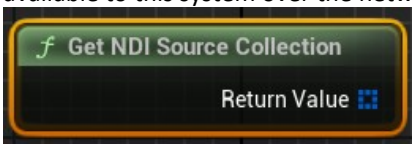
9.2 NETWORK SOURCES

The NDI® IO Plugin for Unreal Engine® enables you to receive audio and video from NDI® network sources, and use them in your Unreal Engine® scene.

This tutorial will show you how to use blueprint library functions to quickly get the NDI® Network Source Collection, and you will also learn about the NDI Finder Component and its functionality.

9.2.1 GET NDI SOURCE COLLECTION

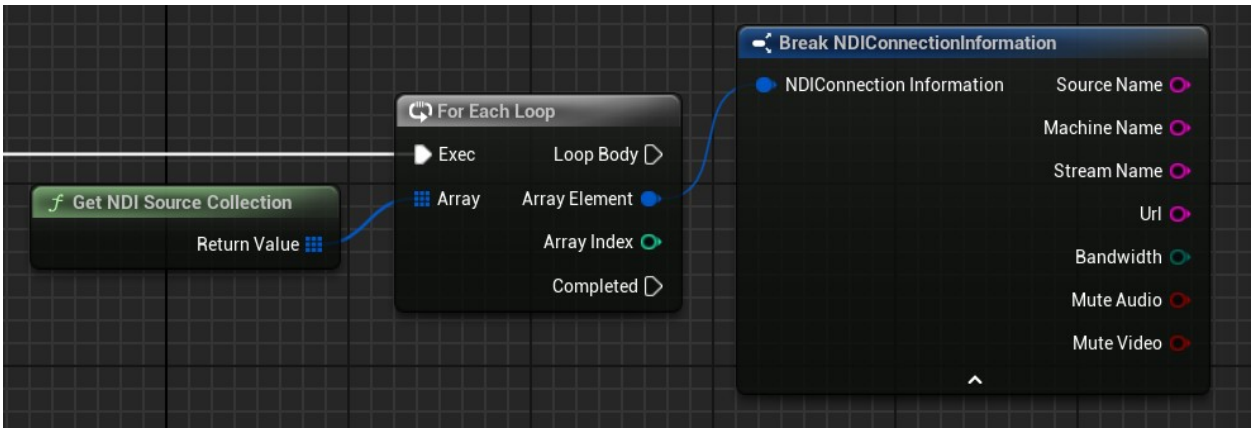
The blueprint node “Get NDI Source Collection” provides access to the list of NDI® network sources currently available to this system over the network.



9.2.2 NDI SOURCE INFORMATION

Iterating over the returning array will allow you to view the component properties.

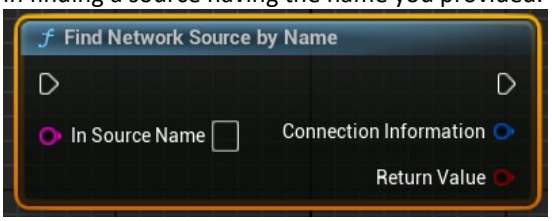
Hint: This is useful for populating lists of NDI® sources.



9.2.3 FIND SOURCE BY NAME

“Find Network Source by Name” is useful for connecting to a specific known source.

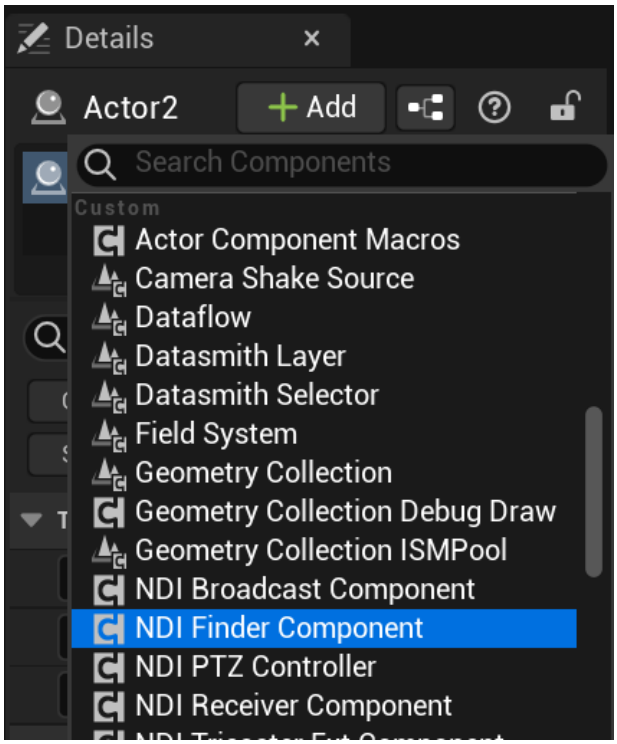
The function returns an “NDI Source Information” structure and a Boolean that describes whether it was successful in finding a source having the name you provided.



9.2.4 NDI FIND COMPONENT

To add the NDI Finder Component to an actor, first open the Actor you wish to add the component to. Once you are editing the Actor:

- Click on Add component
- Scroll to the NDI Finder Component.
- Click on NDI Finder Component.



ON NETWORK SOURCES CHANGED

After selecting the newly created NDI Finder Component, you will have access to create an event handler for “On Network Sources Changed”.

FIND NETWORK SOURCE BY NAME

The NDI Finder Component provides a useful function labeled “Find Network Source by Name”. This is helpful when you want to get the “Source Information” for a known source. The function returns an “NDI Connection Information” structure along with a Boolean indicating whether it was successful in finding the named source.

In this section you have learned how to use the NDI® IO Plugin for Unreal Engine® to get the current active network sources, find sources by name, create an NDI Finder Component, and handle network source change notifications.

9.3 SIMPLE INPUT SOURCES

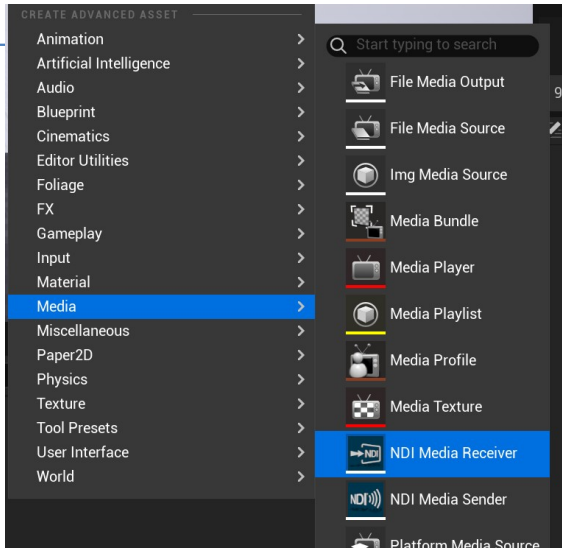
To get started using NDI Receiver Actor, you need to create an NDI Media Receiver media object, and optionally an NDI Video Texture.

Once you have created this content, add an NDI Receiver Actor to the scene. Afterward, you can call “Begin Receive” for your newly created NDI Input Source media.

9.3.1 NDI MEDIA RECEIVER

To create an NDI Media Receiver, open the Media Context Menu. To do this, proceed as follows:

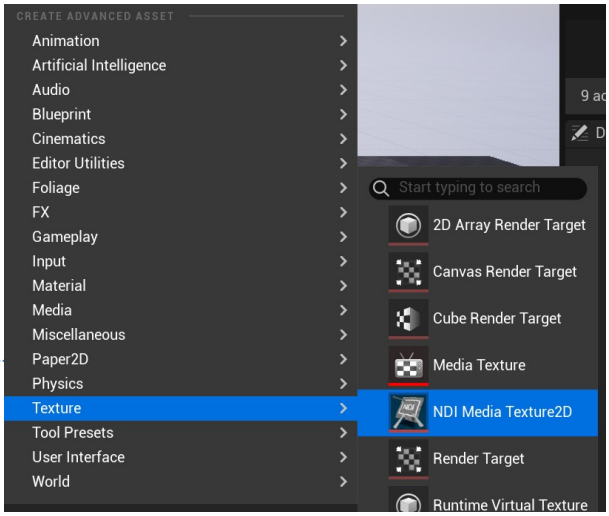
1. Right-click your Content Browser's folder view
2. And select Media > then NDI Media Receiver.
3. Finally, rename the newly created content to “NDI_SimpleInputSource.”



9.3.2 NDI MEDIA TEXTURE2D

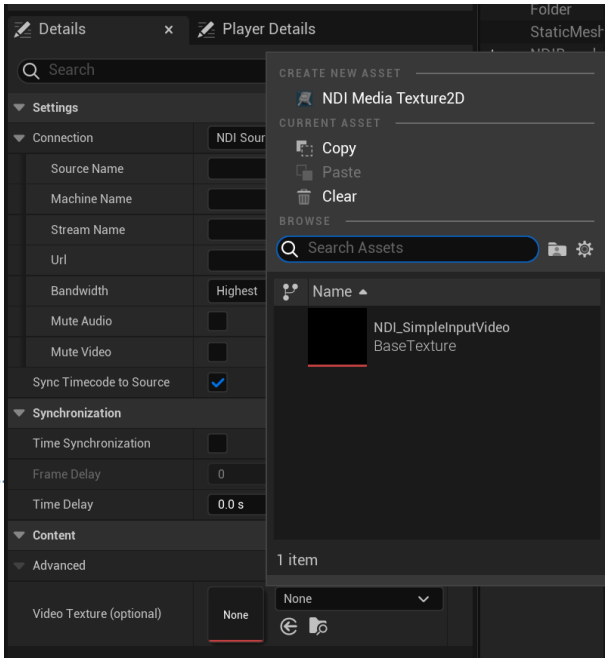
To create an NDI Media Texture2D, bring up the Media Context Menu, and proceed as follows:

1. Right-click your Content Browser's folder view.
2. Select Texture.
3. And choose NDI Media Texture2D.
4. Finally, rename the newly created content "NDI_SimpleInputVideo."



APPLY THE NDI VIDEO TEXTURE

1. Double-click on the newly created "NDI_SimpleInputSource" to open the property details for the NDI Input Source.
2. Select the Video Texture item drop-down and
3. Select the "NDI_SimpleInputVideo" created earlier.



The video texture is optional. If no texture is supplied, the NDI Media Receiver will automatically create an internal one when playing.

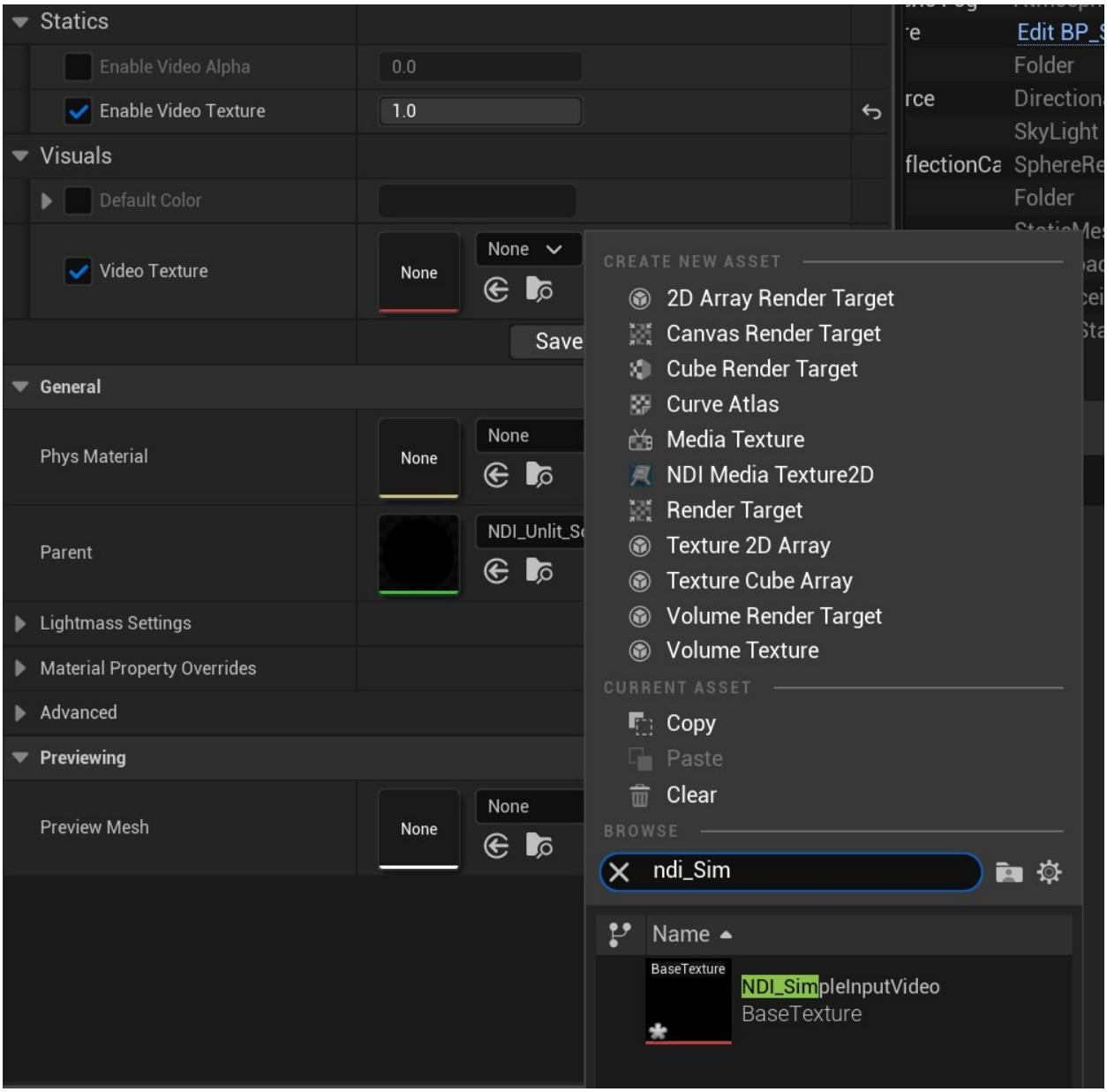
Supplying a video texture allows you to more easily reference the texture in other objects.

CREATE AN UNLIT VIDEO MATERIAL

A generic unlit material has been provided in the NDI® IO Plugin's content folder (as described in Section 3, the content folder location depends on whether you installed the NDI® IO Plugin to the engine or to the game. You may have to enable "Show Plugin Content in the Content Browser settings).

Create a Material Instance of the "NDI_Unlit_SourceMaterial", and name the new content "MI_NDISource_Unlit". Once opened, the Details panel of the Material Instance will allow you to set some configuration values.

1. Checkmark Enable Video Texture under the 'Statics' drop-down and set the parameter value to '1.0'.
2. Checkmark the Video Texture.
3. Select the drop-down for the Video Texture parameter, and find the "NDI_SimpleInputVideo" created earlier in this tutorial.



9.3.3 NDI RECEIVE ACTOR

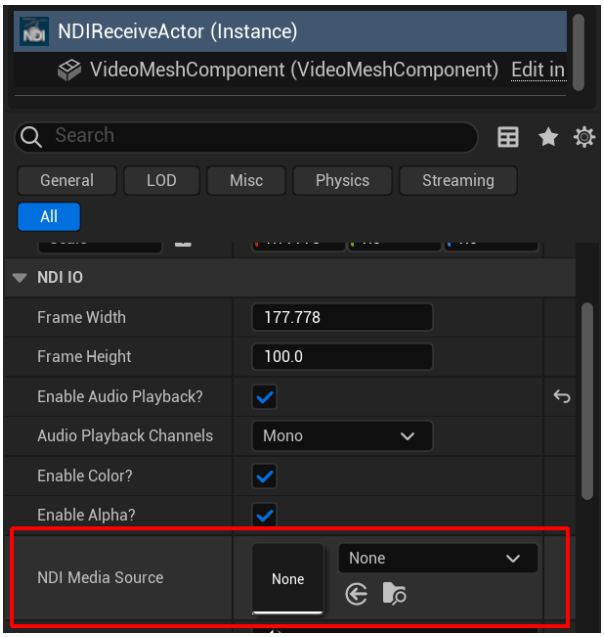
The NDI Receive Actor implemented by the NDI® IO Plugin for Unreal Engine® provides a simple way to display a network video source in your virtual world.

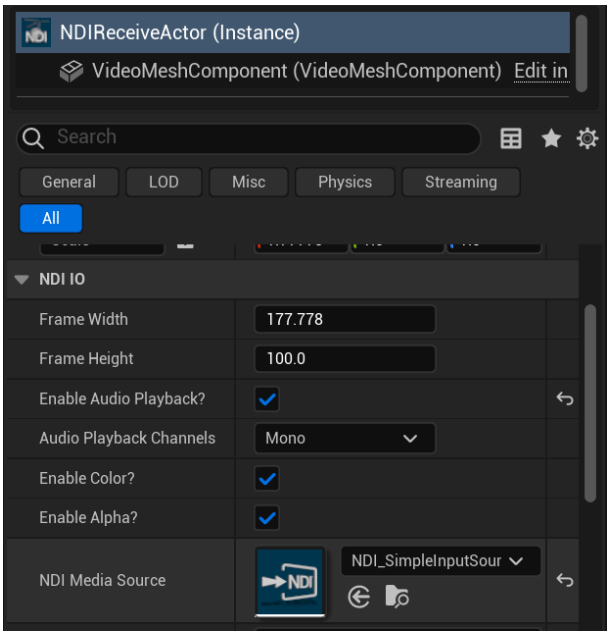
To begin, use the Search feature in the Place Actors panel and type “NDI Receive Actor”. After locating the actor, select and drag the content into the scene.

SETUP RECEIVE ACTOR

After adding the NDI Receive Actor to the scene, ensure it is selected, which will give you access to its properties in the Details panel.

With the NDI Receive Actor selected in the Details panel, scroll down to find the NDI Media Source' parameter located in the NDI IO category. Select the None property, then choose the “NDI_SimpleInputSource” media object you created earlier.

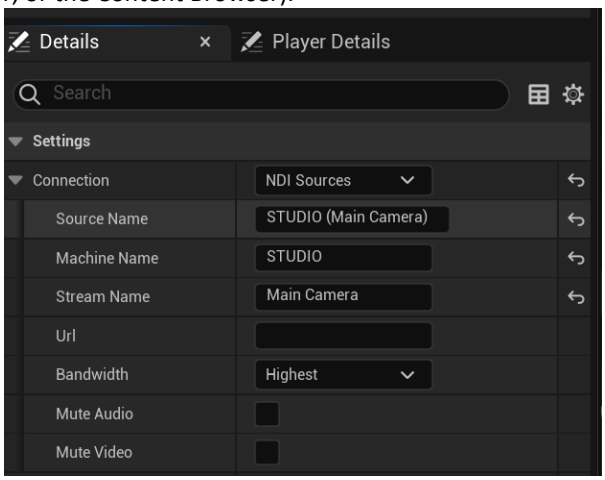




SET SOURCE TO RECEIVE

The source to receive is set in the NDI Media Receiver. Open the settings for NDI_SimpleInputSource by double-clicking its icon (either in the details of the NDI Receive Actor, or the Content Browser).

In its Settings the Connection subsection has an NDI Sources dropdown which is populated by the NDI sources currently visible. Picking one will fill in the Source Name field with the full name of the source. The source name will also be split up into its Machine Name and Stream Name components.



Alternatively any source name can be manually entered, even if the source currently does not exist or is not visible.

Other connectio settings are also available here, such as restricting what type of NDI data the receiver should receive.

The Media Details shown for the NDI Media Receiver will show up as zero or default values. This is normal, and the NDI Media Receiver will change them as it starts to receive the stream.

The NDI Media Receiver will automatically start to receive the NDI stream when the scene starts playing.

9.4 NDI BROADCAST ACTOR

To get started using NDI Broadcast Actor, you need to create an NDI Output Source media object, and optionally a Render Target. After the content has been created, add an NDI Broadcast Actor to the scene.

9.4.1 NDI MEDIA SENDER

To create an NDI Media Sender, bring up the Media Context Menu. To do this, right-click your content browser's folder view. Select Media, then NDI Media Sender. Finally, rename the newly created content to "NDI_SimpleBroadcastSource".

Double-clicking the newly created NDI Output Source opens a new tabbed window to shows the Details properties of the output source. Within the panel there are several valuable parameters, which if changed, modify the NDI source configuration and subsequently the information which is broadcast over NDI®.

9.4.2 CREATE RENDER TARGET

To create an optional Render Target, bring up the Media Context Menu. To do this, right-click your Content Browser's folder view. Select Textures, then Render Target. Finally, rename the newly-created content to "RT_SimpleBroadcastSource".

APPLY THE RENDER TARGET

With the Details panel open for the newly created NDI Output Source, find the category 'Content' with the property "Render Target". Select the drop-down on the 'Render Target' property, and set it to the render target created earlier, aptly named "RT_SimpleBroadcastSource".

The size of the render target will be changed by the sender to match the frame size set for the sender.

If no render target is supplied, a temporary one will automatically be created.

9.4.3 NDI BROADCAST ACTOR

The NDI Broadcast Actor implemented by the NDI® IO Plugin for Unreal Engine® provides a very simple way to provide a network video source to an external application.

To begin, use the Search feature in the Place Actors panel and type "NDI Broadcast Actor". After locating the actor, select and drag the content into the scene.

SETUP BROADCAST ACTOR

After adding the NDI Broadcast Actor to the scene, ensure that the actor is selected. Once selected, you will have access to the associated properties in the Details panel.

With the NDI Broadcast Actor selected, find the 'NDI IO' drop-down in the Details panel. Within this category, click the drop-down arrow of the "NDI Media Source" property, and select the previously created NDI Media Sender named "NDI_SimpleBroadcastSource".

The Broadcast Actor will begin to stream when the scene starts playing.

9.5 SENDING AND RECEIVING METADATA

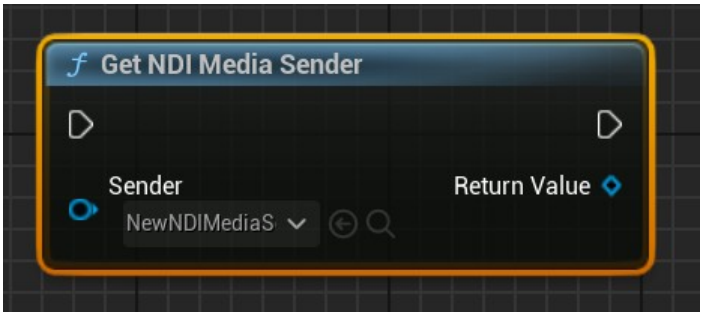
The NDI Media Sender Asset and NDI Media Receiver Asset can send and receive metadata. NDI metadata needs to be formatted as Xml data. The metadata support in the NDI® IO Plugin is focused on Xml data with a single element for ease of processing with Blueprints.

9.5.1 SENDING METADATA FROM AN NDI MEDIA SENDER

NDI® metadata can be sent with an NDI® Media Sender. The media sender must be broadcasting. The metadata can be sent as part of a video frame, or as a separate NDI® metadata frame.

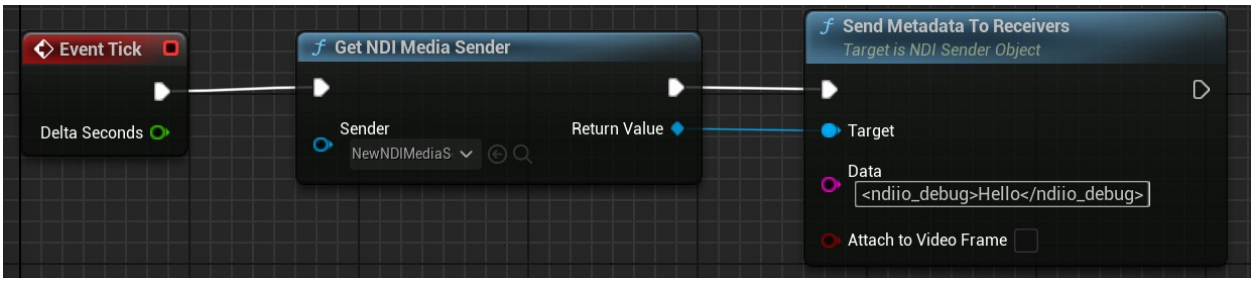
Create a level with an NDI Broadcast Actor, referencing an NDI Media Sender asset as its media source. Open the level blueprint.

There are a number of ways of referencing an NDI Media Sender in a blueprint. One way is to get a reference to the NDI Media Sender asset directly. In the level blueprint add the "Get NDI Media Sender" node, and set its Sender input to the same NDI Media Sender asset used for the previously created NDI Broadcast Actor.

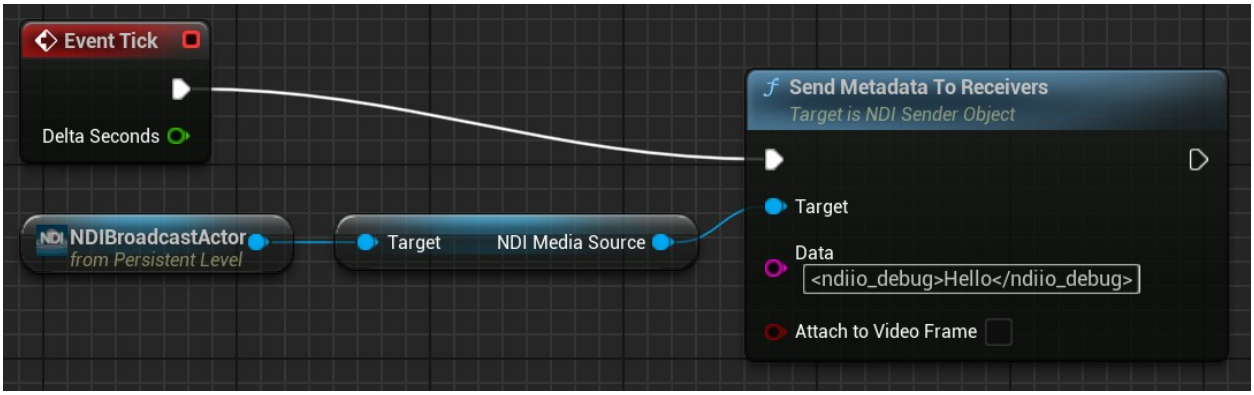


Next, add a "Send Metadata To Receivers" node, which will send a string of data to all receivers of an NDI Media Sender. The data needs to be properly formatted as XML. When "Attach to Video Frame" is enabled, the metadata will be included in the next video frame the sender transmits. Otherwise the metadata will be sent as a separate metadata frame.

Lastly, use an event to trigger the sending of the metadata. As an example, the standard "Event Tick" can be used to send the metadata every time Unreal Engine renders a frame.



An alternative setup is to obtain the media sender from a broadcast actor. Create a reference to the NDI Broadcast Actor in the blueprint, then use a "Get NDI Media Source" node to get a reference to the NDI Media Sender set for the actor. This can then be wired into the metadata sender.



The Send Metadata To Receivers node sends the data string as supplied. Two alternative nodes provide for more structured sending of metadata. The "Send Metadata To Receivers (Element + Data)" node forms the metadata from an element name and a data string. The resulting metadata will have the form `<Element>Data</Element>`.

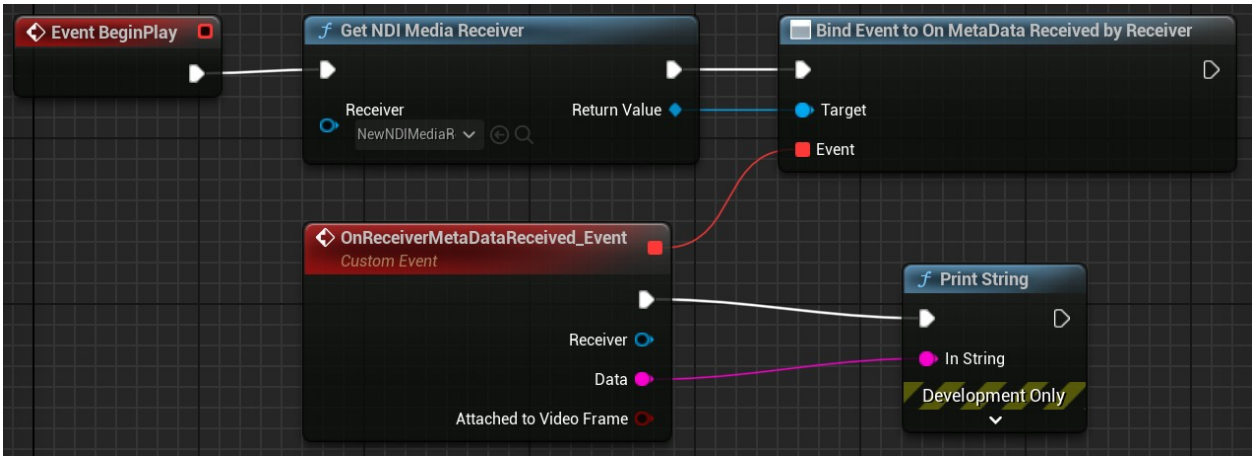
The "Send Metadata To Receivers (Element + Attributes)" node constructs the metadata from an element name and a set of key & value pairs stored in a map. The metadata will look like `<Element Key1="Value1" Key2="Value2" Keyn="Valuen"/>` containing all the key & value pairs in the attribute map.

9.5.2 SENDING METADATA FROM AN NDI MEDIA RECEIVER

NDI Media Receivers have the ability to transmit metadata back to the sender they are receiving audio or video from. The setup is analogous to sending metadata from NDI Media Senders, using "Get NDI Media Receiver" and "Send Metadata To Sender" nodes.

9.5.3 RECEIVING METADATA WITH AN NDI MEDIA RECEIVER

The metadata received by an NDI Media Receiver can come from a metadata frame in the NDI® stream, or from being attached to a video frame. The "On MetaData Received By Receiver" event dispatcher triggers on both. In the level blueprint use the "Assign On MetaData Received By Receiver" event dispatcher on an NDI Media Receiver.



Here the received metadata is printed to the screen for demonstration purposes. The generated events will contain a reference to the receiver that triggered the event, the metadata string, and a boolean indicating whether the metadata came from a video frame or a metadata frame.

9.5.4 RECEIVING METADATA WITH AN NDI MEDIA SENDER

Metadata can be sent back to a sender from any of the receivers of the NDI® stream. The setup for handling the receiving of this metadata is similar to that of receiving metadata in receivers, except for using "Assign On MetaData Received By Sender" instead.

Senders only check for incoming metadata every time Unreal Engine renders a frame.

9.5.5 PARSING METADATA

The NDI® IO plugin provides a utility blueprint function to parse NDI® metadata. The "Parse NDI MetaData" node takes in a string of NDI® metadata, and produces an array of metadata elements. Each element contains the name, the data, and a map of key & value attributes.

The NDI® metadata parser node does not support nested elements.

9.6 NDI BLUEPRINT EVENTS

The NDI® IO plugin exposes a number of events to blueprints, which can be used to trigger further actions. They are listed in the NDI Events category.

9.6.1 NDI MEDIA SENDER EVENTS

Event	
On Before Video Being Sent by Sender	Triggered just before the sender hands a video frame to NDI®. This can be used, for example, to call "Send MetaData To Receivers" with the metadata being attached to the frame.
On Video Sent by Sender	Triggered after the sender has delivered a video frame to NDI®.
On Before Audio Being Sent by Sender	Triggered just before the sender hands an audio frame to NDI®. Currently the sending of audio is not supported, so this event is currently never called.
On Audio Sent by Sender	Triggered after the sender has delivered an audio frame to NDI®. Currently the sending of audio is not supported, so this event is currently never called.
On Before MetaData Being Sent by Sender	Triggered just before the sender hands a metadata frame to NDI®. This does not apply to metadata attached to video frames.
On MetaData Sent by Sender	Triggered after the sender has delivered a metadata frame to NDI®. This does not apply to metadata attached to video frames.
On MetaData Received by Sender	Triggered when the sender has received a metadata frame from a receiver. The received metadata string is provided by the event.

9.6.2 NDI MEDIA RECEIVER EVENTS

Event	
On Video Received by Receiver	Triggered after the receiver has received a video frame from NDI®.
On Audio Received by Receiver	Triggered after the receiver has received an audio frame from NDI®.
On MetaData Received by Receiver	Triggered after the receiver has received a metadata frame from NDI®. The received metadata string is provided by the event.

9.6.3 NDI PTZ CONTROLLER EVENTS

The PTZ Controller component of an NDI Broadcast Actor exposes events so that PTZ control messages can be used in blueprints. These events occur even if "Enable PTZ" is turned off in the PTZ Controller component (but not if it is turned off in the media sender stream). This allows PTZ control to be customized, and even be used for control of other actors.

Event	
On PTZ Pan Tilt Speed	Triggered when pan and tilt speed commands are received from NDI®. The pan and tilt speed are provided by the event.
On PTZ Zoom Speed	Triggered when a zoom speed command is received from NDI®. The zoom speed is provided by the event.
On PTZ Focus	Triggered when a focus change command is received from NDI®. The event includes the new focus distance, as well as the autofocus state.
On PTZ Store	Triggered when a command to store the current PTZ state in a preset is received from NDI®. The index of the preset is provided. Note that the indices are counted from 0, not 1.
On PTZ Recall	Triggered when a command to recall a PTZ preset is received from NDI®. The index of the preset is provided. Note that the indices are counted from 0, not 1.

10 ADVANCED

The NDI SDK provides more advanced users with more powerful capabilities and features, as detailed in this section.

10.1 NDI BROADCAST CONFIGURATION

Essential properties used for modifying the broadcast configuration for an NDI Media Sender.

Properties	
FrameSize (FIntPoint)	The output size of the frame while sending video over NDI®.
FrameRate (FFrameRate)	The desired number of frames (per second) for video to be sent over NDI®.

The NDI Broadcast Configuration structure is used when initializing or changing the broadcast configuration of an NDI Media Sender object.

10.2 NDI CONNECTION INFORMATION

Defines a collection of properties which describe the Source Information for NDI® connections. This structure is returned by `Find Network Source by Name`. Obtaining the NDI Source Information is the first step before connecting to an NDI source on the network.

Properties	
SourceName (FString)	The full NDI name of the NDI source. SourceName is composed of the MachineName and the StreamName together, in the format: “MachineName (StreamName)”.
MachineName (FString)	The Network Machine Name of the NDI Source.
StreamName (FString)	A user-friendly name of the individual source, excluding the machine name.
Url (FString)	A network location for which this connection exists.
Bandwidth (ENDISourceBandwidth)	Indicates the current bandwidth mode used for this connection.
bMuteAudio (Boolean)	Sets whether or not to mute audio from this connection.

bMuteVideo (Boolean)	Sets whether or not to mute video from this connection.

To get NDI Source Information, call the "Blueprint Library Function" called `Find Network Source By Name`. To get a collection of multiple NDI Source Information structures, call `Get NDI Source Collection`.

The collection can be iterated and used to "Start Broadcasting" or "Change Connection Information" on an Input Source object.

10.3 NDI RECEIVER PERFORMANCE DATA

List of commands that provide detailed information on an NDI source and sender.

Properties	
AudioFrames (int64)	The number of audio frames received from the NDI® sender.
DroppedAudioFrames (int64)	The number of audio frames dropped in transit from an NDI® sender.
DroppedMetadataFrames (int64)	The number of metadata frames dropped in transit from an NDI® sender.
DroppedVideoFrames (int64)	The number of video frames dropped in transit from an NDI® sender.
MetadataFrames (int64)	The number of metadata frames received from the NDI® sender.
VideoFrames (int64)	The number of video frame received from the NDI® sender.

To get an NDI Receiver Performance Data structure, call the `GetPerformanceData` function on an NDI Receiver Component object in blueprints. This data can be useful for determining how strong/reliable the connection to the NDI® sender is.

10.4 NDI BROADCAST COMPONENT

Provides a wrapper around the NDI Media Sender to access functionality from Blueprints and perform simple broadcasting functionality.

Properties	

NDIMediaSource (UNDIMediaSender*)	A pointer to the Media Sender object representing the configuration of the network source to send audio, video, and metadata.

Functions	
StartBroadcasting(FString&)	Attempts to start broadcasting audio, video, and metadata via the NDIMediaSender object.
ChangeSourceName(const FString&)	Changes the name of the sender object as seen on the network for remote connections.
ChangeBroadcastConfiguration(const FNDIBroadcastConfiguration&)	Attempts to change the broadcast information associated with the NDIMediaSender object.
ChangeBroadcastTexture (UTextureRenderTarget2D*)	Attempts to change the RenderTarget property of the NDIMediaSender object.
GetTallyInformation(bool&, bool&)	Determines the current tally information.
GetNumberOfConnections(int32&)	Returns the current number of receivers connected to the NDIMediaSender object.
StopBroadcasting()	Attempts to immediately stop sending frames over NDI® to any connected NDI® receiver.

Adding the `NDI Broadcast Component` to an Actor in blueprints give your actor functionality for broadcasting audio, video, and metadata frames over NDI®.

10.5 NDI FINDER COMPONENT

Provides a component used for essential functionality when dealing with finding NDI® sources on the network allowing you to get a collection of sources as well as listen for events when the source collection has changed.

Functions	
FindNetworkSourcebyName (FNDIConnectionInformation&, FString)	Attempts to find a network source by the supplied name.

GetNetworkSources() (TArray<FNDIConnectionInformation>)	Returns the current collection of NDI® sources found on the network.

Delegates	
OnNetworkSourcesChanged (UNDIFinderComponet*)	A delegate which is broadcast when any change to the network source collection has been detected.

Adding the NDI Finder Component to an Actor in blueprints give your actor functionality for finding NDI® sources on the network. This retrieves a collection of network sources as well as notifies of changes to the NDI® source collection.

10.6 NDI RECEIVER COMPONENT

Provides a wrapper around the NDI Media Receiver to access functionality from Blueprints and perform simple receiver functionality.

Functions	
StartReceiver(const FNDIConnectionInformation&)	Begin receiving NDI® audio, video, and metadata frames from a connected source.
ChangeConnection(const FNDIConnectionInformation&)	Change the connection source to receive NDI® audio, video, and metadata frames.
SendMetadataFrame(const FString&)	Add a metadata frame and return immediately, having scheduled the frame asynchronously.
SendTallyInformation(const bool&, const bool&)	Setup the up-stream tally notifications. If no streams are connected, it will automatically send the tally state upon connection.
ShutdownReceiver()	Attempts to stop receiving audio, video, and metadata frames from the connected NDI® source.
GetCurrentFrameRate() (FFrameRate)	Returns the current framerate of the connected source.
GetCurrentTimecode()	Returns the current timecode of the connected source.

(FTimecode)	
GetCurrentConnectioninformation() (FNDIConnectionInformation)	Returns the current connection information of the connected NDI® source.
GetPerformanceData() (FNDIReceiverPerformanceData)	Returns the current performance data of the receiver while connected to an NDI® source.

Adding the NDI Receiver Component to an Actor in blueprints give your actor functionality for receiving audio, video, and metadata frames from a connected NDI® sender.

10.7 NDI VIEWPORT CAPTURE COMPONENT

Provides a component to be used to capture additional viewports for broadcasting over NDI®

Properties	
bOverrideBroadcastSettings (Boolean)	If true, will allow you to override the capture settings by ignoring the default Broadcast Settings in the NDI Media Sender.
CaptureSize (FIntPoint)	The size to set the viewport to capture.
CaptureRate (FFrameRate)	The desired number of frames (per second) to capture from the viewport.
NDIMediaSource (NDIMediaSender*)	A pointer to the NDI Media Sender object representing the configuration of the network source to send audio, video, and metadata.
AlphaMin (float)	When outputting an alpha channels, determines what source alpha value maps to an output alpha value of 0.
AlphaMax (float)	When outputting an alpha channels, determines what source alpha value maps to an output alpha value of 1.

Functions	
ChangeSourceName (const FString&)	Changes the name of the sender object as seen on the network for remote connections.

ChangeBroadcastConfiguration(const FNDIBroadcastConfiguration&)	Attempts to change the broadcast information associated with the NDIMedaSource object.
ChangeBroadcastTexture(UTextureRenderTarget2D*)	Attempts to change the RenderTarget used in sending video frames over NDI®.
ChangeCaptureSettings(FIntPoint, FFrameRate)	Change the capture settings of the viewport capture
GetTallyInformation(bool&, bool&)	Determines the current tally information.
GetNumberOfConections(int32&)	Returns the current number of connected NDI® receivers. This can be used to avoid rendering when nothing is connected to NDIMediaSource object.

Adding the NDI Viewport Capture Component to an Actor in blueprints give your actor functionality for rendering the viewport, while providing audio, video, and metadata frames from a connected NDI® sender.

10.8 NDI PTZ CONTROLLER COMPONENT

Provides a component for PTZ control from an NDI® sender. Typically this is part of an NDI® Broadcast Actor. It can also be applied to any other type of Actor, with some limited functionality.

Properties	
EnablePTZ (Boolean)	If true, enables PTZ control of the owning actor.
PTZWithPanLimit (Boolean) **PTZWithTiltLimit (Boolean)** **PTZWithFoVLimit (Boolean)**	Enable limiting the range of pan, tilt, and field of view values.
PTZPanMinLimit (float) **PTZPanMaxLimit (float)** **PTZTiltMinLimit (float)** **PTZTiltMaxLimit (float)** **PTZFoVMinLimit (float)** **PTZFoVMaxLimit (float)**	The limits on the range of pan, tilt, and field of view values. Used when the corresponding PTZWith*Limit is true.
bPTZPanInvert (Boolean)	Invert the direction in which pan and tilt rotate.

bPTZTiltInvert (Boolean)	
PTZPanSpeed (float) **PTZTiltSpeed (float)** **PTZZoomSpeed (float)**	How much to change the pan, tilt, and zoom (field of view) values per second. Internally the pan and tilt speeds are scaled by the field of view, so that apparent movemement remains constant with zoom.
PTZStoredStates (TArray<FPTZState>)	Stored PTZ state presets.
PTZRecallEasing (float)	Smooths out the transition between current PTZ state and a stored PTZ state being recalled over the given number of seconds.
NDIMediaSource (UNDIMediaSender*)	The NDI® media sender from which to receive the PTZ metadata.

Functions	
ReceiveMetaDataFromSender(UNDIMediaSender*, FString)	Parses an NDI metadata message from a sender for PTZ settings, and applies them to this PTZ controller.
Initialize(UNDIMediaSender*)	Initializes the component, setting it to listen for PTZ metadata coming from the supplied sender.
SetPTZPanTiltSpeed(float, float) **SetPTZZoomSpeed(float)**	Sets the pan, tilt, and zoom speeds.
SetPTZFocus(bool, float)	Sets the focus distance (for depth of field rendering). If the boolean is set to true, auto-focus is used and the distance ignored (currently this is implemented by turning off depth of field).
StorePTZState(int) **RecallPTZState(int)**	Store the current PTZ state as a preset. Recall the PTZ state from a preset. When recalling, recall easing is applied to interpolate between current and recalled state. Preset indices from 0 to 255 are supported, though be aware that Studio Monitor for example can only access from 0 to 9.
GetPTZStateFromUE() **SetPTZStateToUE(const FPTZState&)**	Get the state of the owning Actor as a PTZ state. Apply the given PTZ state to the owning Actor.

10.9 NDI MEDIA RECEIVER

Content used to provide functionality for receiving audio, video, and metadata frames over NDI® to a connected NDI® sender.

Properties	
Timecode (FTimecode)	The current frame count, seconds, minutes, and hours in time-code notation.
FrameRate (FFrameRate)	The desired number of frames (per second) for video to be displayed
Resolution (FIntPoint)	The width and height of the last received video frame.
bSyncTimecodeToSource (Boolean)	Indicates whether the timecode should be synced to the Source Timecode value *Default value is true
bPerformsRGBtoLinear (Boolean)	Indicates whether an sRGB to Linear color space conversion is performed after receiving an NDI® video frame from the connected NDI® sender.
ConnectionInformation (FNDIConnectionInformation)	A structure describing detailed information about the NDI® sender this object is connected to.
PerformanceData (FNDIReceiverPerformanceData)	A structure describing detailed information about the NDI® receiver object when connected to an NDI® sender
VideoTexture (NDIMediaTexture2D*)	An NDI® IO Plugin specific Texture2D object used to render video frame from a connected NDI® sender.

Functions	
Initialize (const FNDIConnectionInformation&)	Attempts to perform initialization logic for creating an NDI® receiver through the NDI® SDK API.
StartConnection()	Attempt to (re-)start the connection.
StopConnection()	Stop the connection.

ChangeConnection (const FNIDConnectionInformation&)	Attempts to change the connection to another NDI® sender object, for receiving audio, video, and metadata frames.
ChangeVideoTexture (UNDIMediaTexture2D*)	Attempts to change the VideoTexture object used to capture video frames from a connected NDI® sender.
GeneratePCMData(uint8*, int32) (int32)	Attempts to generate the pcm data required by the AudioWave property of this object.
GetAudioChannels()	Gets the number of audio channels provided by the NDI® source.
RegisterAudioWave(UNDIMediaSoundWave*)	Attempts to register a sound wave object with this object.
SendMetadataFrame(const FString&)	This will send a metadata frame to the sender. The data is expected to be valid XML.
SendMetadataFrameAttr(const FString&, const FString&)	This will send a metadata frame to the sender. The data will be formatted as: <Element>ElementData</Element>
SendMetadataFrameAttrs(const FString&, const TMap<FString,FString>&)	This will send a metadata frame to the sender. The data will be formatted as: <Element Key0="Value0" Key1="Value1" Keyn="Valuen"/>
SendTallyInformation(const bool&, const bool&)	This will set the up-stream tally notifications. If no streams are connected, it will automatically send the tally state upon connection.
Shutdown()	Attempts to immediately stop receiving frames from the connected NDI® sender.
UnregisterAudioWave(UNDIMediaSoundWave*)	Remove the AudioWave object from this object (if it was previously registered)
UpdateMaterialTexture(class UMaterialInstanceDynamic*, FString);	Updates the DynamicMaterial with the VideoTexture of this object.
CaptureConnectedVideo()	Attempts to capture a frame from the connected source.

CaptureConnectedAudio() **CaptureConnectedMetadata()**	
DisplayFrame(const NDIlib_video_frame_v2_t&)	Attempts to immediately update the 'VideoTexture' object with the captured video frame.
PerformsRGBToLinearConversion(bool)	Set whether or not a RGB to Linear conversion is made.
GetCurrentFrameRate()	Returns the current framerate of the connected source.
GetCurrentResolution()	Returns the current resolution of the connected source.
GetCurrentTimecode()	Returns the current timecode of the connected source
GetCurrentConnectionInformation()	Returns the current connection information of the connected source.
GetPerformanceData()	Returns the current performance data of the receiver while connected to the source.
GetIsCurrentlyConnected()	Returns a value indicating whether this object is currently connected to the sender source.

Creating an NDI Media Receiver will provide configuration details to an NDI Receiver Component. Using the context menu of the content browser, you can create an NDI Media Receiver using the menu “Media” and locating the “NDI Media Receiver” entry. This will allow you to create a virtually unlimited number of NDI® receiver objects.

10.10 NDI MEDIA SENDER

Content used to provide functionality for sending audio, video, and metadata frames over NDI® to connected NDI® receivers.

Properties	
SourceName (FString)	A user-friendly name of the output stream to differentiate from other output streams on the current machine.
FrameSize (FintPoint)	The size of the output frame (in pixels) while sending video frames to connected NDI® receivers

FrameRate (FFrameRate)	The desired number of frames (per second) for video to be sent to connected NDI® receivers.
OutputAlpha (Boolean)	Sets whether or not to output an alpha channel.
AlphaMin (float)	When outputting an alpha channels, determines what source alpha value maps to an output alpha value of 0.
AlphaMax (float)	When outputting an alpha channels, determines what source alpha value maps to an output alpha value of 1.
bEnableAudio (Boolean)	Sets whether or not to output audio.
bEnablePTZ (Boolean)	Sets whether or not to enable PTZ control through this sender.
RenderTarget (UTextureRenderTarget2D*)	A texture representing the current video frame data to be sent over NDI®
bPerformsRGBtoLinear (Boolean)	Indicates whether an RBG to Linear color conversion is performed on the video data prior to sending the frame to connected NDI® receivers.

Functions	
Initialize()	Performs initialization logic to create an NDI® sender through the NDI® SDK API.
ChangeSourceName(const FString&)	Changes the name of the sender object as seen on the network for remote connections.
ChangeBroadcastConfiguration(const FNDIBroadcastConfiguration&)	Attempts to change the broadcast information.
SendMetadataFrame(const FString&, bool)	Send a metadata frame to all receivers. The data is expected to be valid XML.
SendMetadataFrameAttr(const FString&, const FString&, bool)	Send a metadata frame to all receivers. The data will be formatted as: <Element>ElementData</Element>

SendMetadataFrameAttrs(const FString&, const TMap<FString,FString>&, bool)	Send a metadata frame to all receivers. The data will be formatted as: <Element Key0="Value0" Key1="Value1" Keyn="Valuen"/>
ChangeVideoTexture (UTextureRenderTarget2D*)	Attempts to change the RenderTarget object used in sending video frames over NDI®
ChangeAlphaRemap(float, float)	Change the alpha remapping settings.
GetTallyInformation(bool&, bool&)	Determines the current tally information.
GetNumberOfConnections(int32&)	Returns the current number of receivers connected to this source. This can be used to avoid rendering when nothing is connected to the video source, which can significantly improve the efficiency if you want to make a lot of sources available on the network.
Shutdown()	Attempts to immediately stop sending frames over NDI® to any connected NDI® receiver.
PerformLinearTosRGBConversion(bool)	Set whether or not a RGB to Linear conversion is made.
EnablePTZ(bool)	Set whether or not to enable PTZ support.
GetRenderTarget()	Returns the Render Target used for sending a frame over NDI.
GetFrameSize()	Returns the current frame resolution of the sender.
GetFrameRate()	Returns the current framerate of the sender.

Creating an NDI Media Sender will provide configuration details to an NDI Broadcast Component or NDI Viewport Capture Component.

Using the context menu of the content browser, you can create an NDI Media Sender using the Media menu and locating the “NDI Media Sender” entry. This allows you to create a virtually unlimited number of NDI® sender objects.

10.11 NDI MEDIA SOUND WAVE

An NDI® IO Plugin specific SoundWave object used to provide functionality for playback of NDI® audio frames from an NDI® sender.

Functions	
SetConnectionSource (UNDIMediaReceiver*)	Set a "NDI Media Receiver" object as the NDI® sender to provide audio frame to.

Creating an NDI Media Sound Wave will provide configuration details to an "NDI Reciever Component".

Using the context menu of the content browser, you can create an NDI Media Sound Wave using the menu Sounds > Source and locating "NDI Media Sound Wave".

10.12 NDI MEDIA TEXTURE 2D

An NDI® IO Plugin specific Texture2D object used to provide functionality for playback of NDI® video frames from an NDI® sender. Creating an "NDI Media Texture2D" will provide a texture object used by an NDI Media Receiver. Using the context menu of the content browser, you can create a NDI Media Texture 2D using the menu Materials and Textures, and locating the "NDI Media Texture 2D" option.

10.13 NDI BROADCAST ACTOR

An Actor which can be added to the scene that renders a new video texture from the viewpoint of the actor's transform. Provides support for broadcasting the texture as a video frame to connected to an NDI® receiver.

Properties	
NDIMediaSource (UNDIMediaSender*)	An "NDI Media Sender" object representing the configuration of the network source to send audio, video and metadata frames to a connected NDI® receiver.
ViewportCaptureComponent (UNDIViewportCaptureComponent*)	The component used to render an additional viewport for broadcasting a video frame to an NDI® receiver
PTZController (UPTZController*)	Component used for PTZ control.

Adding an "NDI Broadcast Actor" to a scene will provide an easy way to render a viewport to be sent over NDI® to connected NDI® receivers. You can find the "NDI Broadcast Actor" in the sub menu of "NDI®" within the Modes panel in the editor for your project.

10.14 NDI RECEIVE ACTOR

An Actor you can add to the scene to display NDI® sender audio, video, and metadata frames in the virtual scene.

Properties	
FrameHeight (float)	The desired height of the frame in centimeters, represented in the virtual scene.
FrameWidth (float)	The desired width of the frame in centimeters, represented in the virtual scene.
bEnableAudioPlayback (Boolean)	Indicates that this object should play the audio.
AudioPlaybackChannels (ENDIAudioChannels)	Sets how many channels to use to play the audio.
bEnableColor (Boolean)	Enable/disable the use of the color channels (if there are any).
bEnableAlpha (Boolean)	Enable/disable the use of the alpha channel (if there is one)
VideoMeshComponent (UStaticMeshComponent*)	A component which visually represents the video frame received from an NDI® sender.
NDIMediaSource (UNDIMediaReceiver*)	The NDI® receiver object used to get audio, video and metadata frames from an NDI® sender.

Functions	
SetFrameSize (FVector2D)	Attempts to set the desired frame size in cm, represented in the virtual scene.

GetFrameSize()	Returns the current frame size of the 'VideoMeshComponent' for this object.

Adding an “NDI Receive Actor” to a scene will provide an easy way to show video in a virtual environment. You can find the NDI Receive Actor in the “NDI®” sub menu within the “Modes” panel in the editor for your project.