Elias Software

Elias 3 plugin for Unreal 4

Plugin version 3.2

www.eliassoftware.com

Table of contents

1 Important Information

If you use Elias in a commercial product you will also need to buy a licence for the Elias Engine. For more information on engine licences and prices visit https://www.eliassoftware.com/licensing/.

A license for the Elias plugin for Unreal Engine 4 is for a company, and not bound to a specific project.

2 Basic Steps to Use an Elias Project in UE4

1. Import an Elias .mepro file via the content browser. You will get an *EliasProjectAsset*, and an accompanying *EliasComponent* blueprint will be created alongside it.
2. The *EliasComponent* blueprint is used to control playback, and has a property, *EliasAsset*, which must reference an imported *EliasProjectAsset*. This is set automatically on import.
3. The *EliasComponent* can be added to an actor in the world, or it can be created via a call to the *SpawnElias* blueprint node.
4. If the component's *AutoActivate* property is true (the default), it will immediately begin default playback when its containing actor begins play, or when it's created from a *SpawnElias* node.
5. Call functions such as *Play*, *Stop*, *SetLevel*, etc, either within your *EliasComponent* blueprint, or externally on a reference to it (retrieved from the actor, or the return value of *SpawnElias*).

3 Elias Project Asset

This is the UE4 asset representation of a .mepro Elias project file.

Opening the asset will give some basic configuration options, as well as list some simple information about themes, tracks and presets within the project.

From here you can also edit the source import path, and re-import the asset from the Asset menu.

Within the UE4 editor, the project asset references the audio files by their path, so they must remain at the location they were when first imported. This has been done to minimize the amount of bandwidth used for version control when making changes to audio and the .mepro file. When a project is packaged, the audio will be optionally be compressed and then cooked into the asset; there is no need to deploy the audio files themselves.
We suggest that the audio files as well as the mepro is placed under version control, but outside of the contents of the project, to avoid them being imported by UE4.

4 Elias Component

This is the interface to an instance of the Elias engine. It has an associated *EliasProjectAsset*, which is loaded into the engine when playback is requested on the component.

Playback defaults can be configured in the properties. Note that the *AutoActivate* property is used to determine if default playback should begin when a component is created. It defaults to true. If you wish to have more control over playback, set this to false and invoke one of the *Play* functions as required.

In the “Audio Render” section, the number of frames per buffer can be set. Under “Rendering Performance”, the cache page count and cache bytes per page can also be set. You can use these settings for performance fine tuning: High buffers will increase latency but also cause better stability, since larger buffers allow for more processing time.

An *EliasComponent* can be created in one of two ways - either added to an actor/blueprint components list, or spawned dynamically via the *SpawnElias* node.

5 Core Functions

5.1 Functions

The following is a brief overview of when the functions should be used. For information on parameters, see the *Blueprint Function Reference* in the documentation folder.

5.1.1 Play / PlayDefault / PlayWithActionPreset

One of these variants must be called before any other playback related functions. If *PlayDefault* is used, playback will begin using the default properties specified in the details panel for the component. This is also what happens when *AutoActivate* is enabled.

Note: successive calls to one of the *Play* functions without an intervening call to *Stop* will have no effect.

5.1.2 SetLevel

This is the function you should use to change the current theme, level or playback position.

5.1.3 SetLevelOnTrack

This is the function you should use to change the current theme, level or playback position but only wants to affect only one track.

5.1.4 PlayStinger

Starts playback of a stinger.

5.1.5 TriggerActionPreset

Activates an action preset.

5.1.6 Stop

Ends playback. After calling, one of the the *Play* variants must again be called to restart playback.

Important Note

Please take care to always call the functions in the *Elias|Playback* category, and not corresponding ones in the *Audio|Components|Audio* category that are inherited from the regular UE4 audio component. In particular, calling *Play* and *Stop* in *Audio|Components|Audio* will lead to incorrect behaviour due to the Elias engine state becoming desynced from the audio component state.

Unfortunately it is not currently possible to ensure these are hidden when calling functions on an *EliasComponent* from within another blueprint.

5.2 Urgent and wait for prior events flags

5.2.1 Urgent

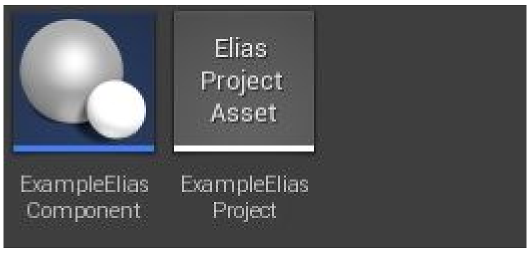
Using the Urgent flag will bypass the event queue to perform the action as fast as possible. For example, if you know that your project will have a lot of level changes and be very busy, the event you want to run can be delayed because of the overall queue of events in Elias. In those cases, be sure to check use urgent to prioritize this event in the queue.

5.2.2 Wait for Prior Event

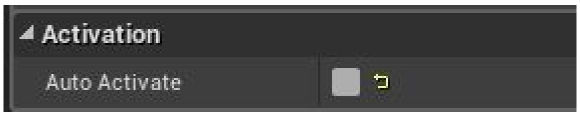
If the flag “Wait for Prior Events” is not used, this event can be skipped to reach the final destination of the Action Preset as fast as possible. If you want to make longer series of events in an and use multiple set level events, be sure to check use the “Wait for Prior Event” flag to make sure that the event will run before moving on to the next event in the chain.

6 Example Walkthrough

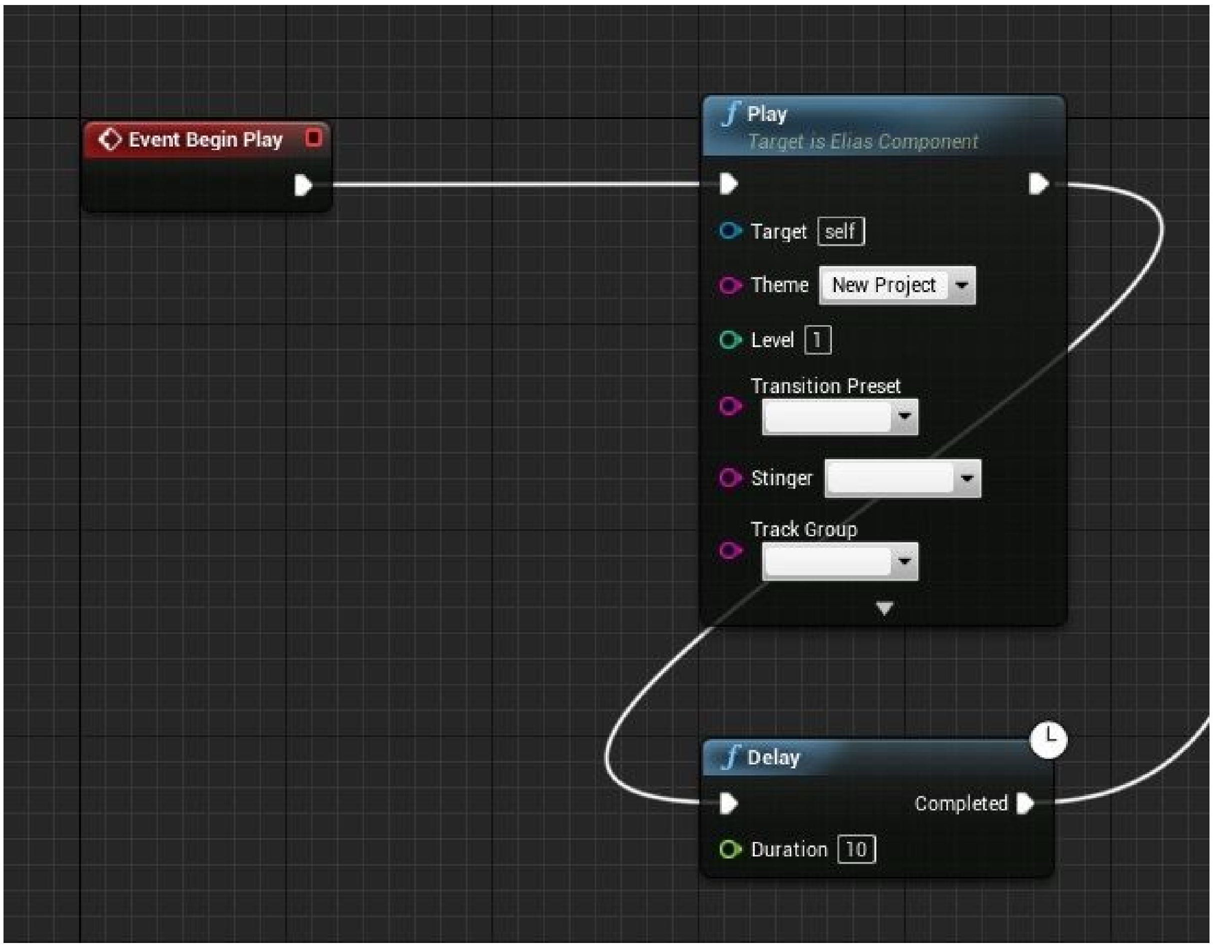
First we import a .mepro project into UE4, which will give us a project asset and a component blueprint to go with it.



Opening up the component blueprint, the first thing we'll do is disable *AutoActivate* in the properties view, since for this example we'll start playback explicitly ourselves.



In the component's *BeginPlay* event, we make a call to the *Play* function, giving it the name of the theme we want to play, and in this case beginning at level 1. We then add a 10 second delay.

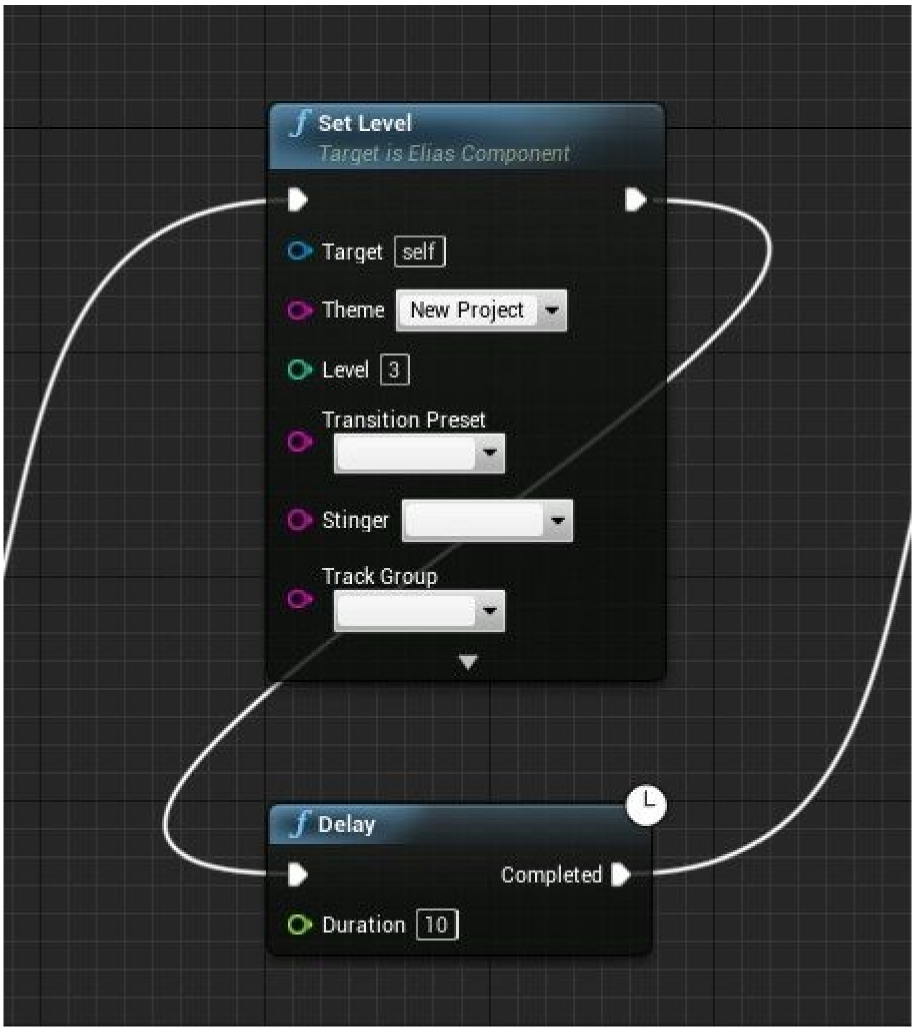


Event Begin Play

Play
Target is Elias Component
Target self
Theme New Project
Level 1
Transition Preset
Stinger
Track Group

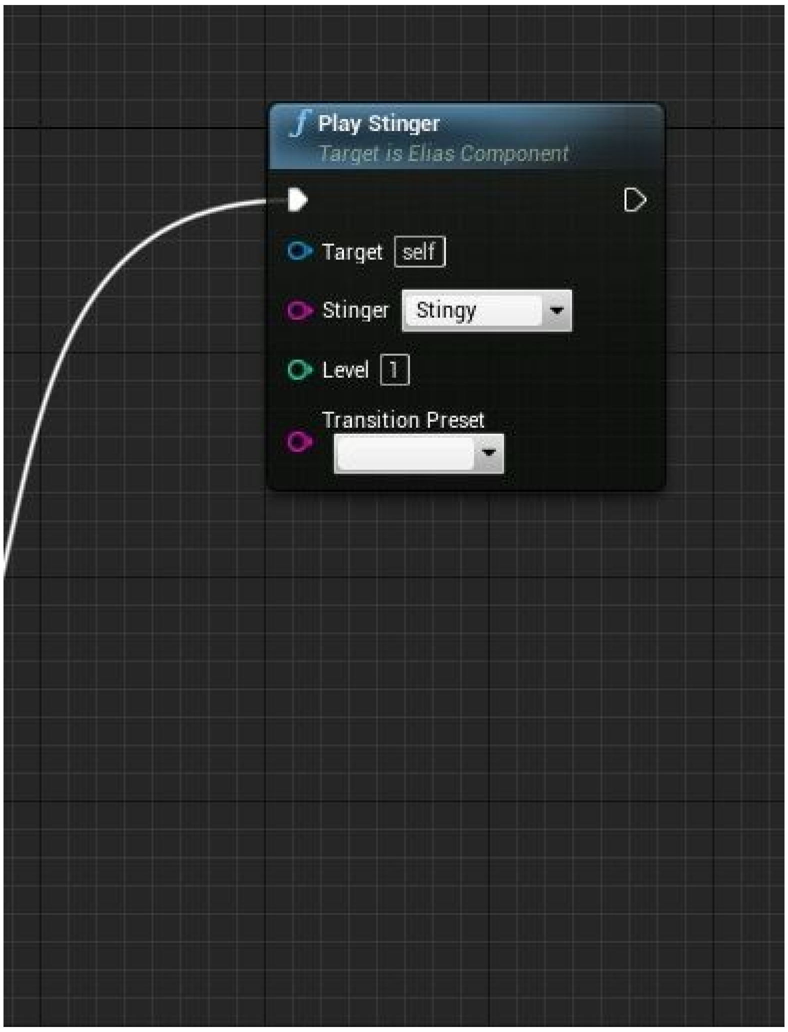
Delay
Completed
Duration 10

After that delay, we're going to change the level. We keep the same theme, but switch to level 3. After that we have another 10 second delay.

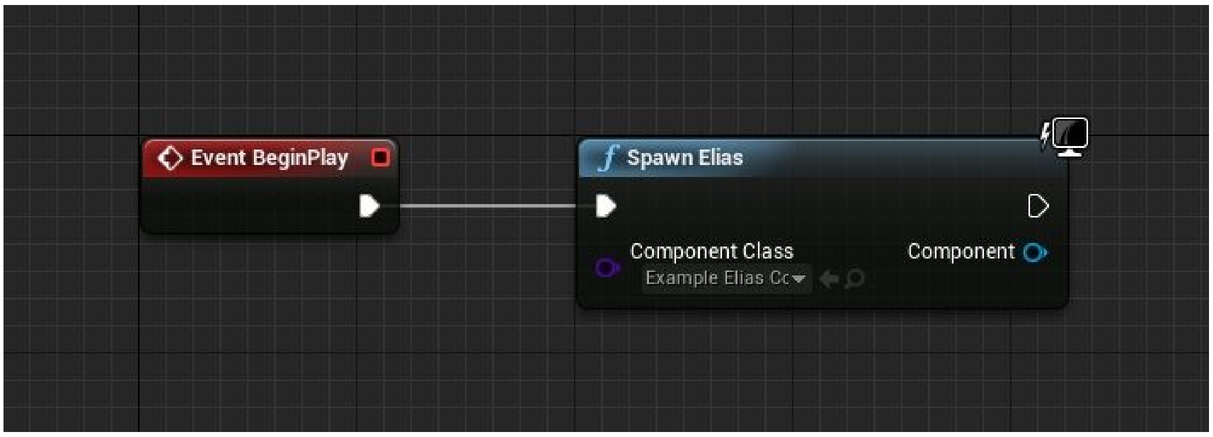

Set Level
Target is Elias Component
Target self
Theme New Project
Level 3
Transition Preset
Stinger
Track Group

Delay
Completed
Duration 10

Finally, we'll trigger a stinger.



All that's left to do now is create an instance of our component. Here, we do so by calling *SpawnElias* from the *BeginPlay* event of our level blueprint. This node can be called from anywhere.



Note that although in this example we put the controlling logic inside the *EliasComponent*, this is not required. If you prefer, you can invoke playback methods from an external blueprint, on a reference to your component.

7 C++ Interface

The Elias plugin can also be used at the C++ level. Use the static method *UEliasComponent::SpawnComponent* to create a component (you'll need a reference to an *EliasProjectAsset* and/or an *EliasComponent* blueprint). The C++ interface is listed at the top of the class declaration in *EliasComponent.h*. There is a one to one mapping to the blueprint interface, which is documented in the *Blueprint Function Reference* within the documentation folder.

For more low level control, *UEliasComponent::GetEliasHandle* can be called to retrieve a handle, which can be used to directly call into the Elias engine API.

8 Upgrade Instructions for Elias 3.0 UE4 Plugin

Note that the importing of the Elias project file (mepro) and the audio assets that are used is now delayed until cooking. This means, that existing projects where the source files were placed outside of version control need to either update, or prevent updating.

To update, simply move the mepro and audio files into version control, and update the import path for the Elias Project Asset.

9 Changelog

9.1 Updates in version 3.2

- Support for Unreal 4.20
- Made some significant performance optimizations in the SFZ player and the general mixer rendering chain in the engine.

9.2 Updates in version 3.1.2

- Support for Unreal 4.19
- Exposed cache settings in the component settings.
- “Get Theme” is now available in blueprints.
- The play function in blueprints now has same controls as in the Elias Studio.

9.3 Updates in version 3.1.1

- Added a new flag that can be set for events, called elias_event_flag_wait_for_prior. When this flag is set, the engine ensures that all prior events have finished processing before it proceeds.
- Improved performance in the Elias Sampler for midi playback.

9.4 Updates in version 3.1.0

- Significantly optimized the performance of the audio output rendering.

9.5 Fixes in version 3.0.0

- Fixed issue with stinger dropdown listing stingers incorrectly.
- Corrected calls to EnsureCompletion to always complete rendering on the dedicated thread.
- Bulk data cooking flag addition for recent UE4 versions.
- Removed defunct OriginalFilePath property from Elias project asset.
- Improvements to handle midi files and instrument patches.
- improved the engine for handling midi, sampler and instrument patches.

9.6 Changes in version 3.0.0

- Changed Elias project asset format. Within the UE4 editor, audio files are now loosely referenced by their path. Audio is baked into the asset during packaging. NOTE: There is an asset update process for converting existing assets to the new format. This process is destructive, please ensure your project is backed up/committed to version control before proceeding. You will need the raw audio files to be located where they were when the asset was imported in order to update assets in-place.

- Rendering buffer size can now be adjusted in Elias Component properties.
- Updates for compatibility with new UE4 audio mixer (-audiomixer).
- Added new Set Level On Track event.
- Added IsStopped query.
- Exposed some more Elias functions on the Elias Component for querying data about themes: GetThemeCursorInSeconds, GetThemeCursorInBarsAndBeats, GetThemeBasicInfo.

9.7 Elias Engine 3.0.0 fixes and features

- Fixed a bug in the MIDI file streaming implementation which resulted in some old notes continuing to play after transitioning away from the given level or variation.
- Fixed a bug where the engine would occasionally fail to clean up SFZ voices properly.
- Fixed a crash that would occur when attempting to import Ogg Vorbis files with invalid headers.
- Many internal optimizations in the SFZ player.
- Fixed a bug in the third party Ogg Vorbis decoder that caused certain seeking operations to fail.
- Added a temporary workaround for a bug in the third party Ogg Vorbis decoder which causes it to skip the last Vorbis frame in certain files. The engine currently mutes the remainder of the buffer when this problem occurs; a fix is in the works by the author of the decoder.
- Fixed a bug which resulted in lost MIDI events when sending data from multiple MIDI tracks to the same generator.
- Implemented so called "smart transitions" based on pre analysis of the audio files. When these are enabled, the engine will generate significantly smoother transitions between levels and variations by taking advantage of very quiet portions of the audio being rendered. Smart transitions can be disabled even if analysis data is available, by way of the elias_transition_option_disable_smart_transitions boolean transition option.
- The engine can now be instructed to automatically transition to the next variation on a level based on the given progression mode each time the loop wraps around. This is configurable by way of a new boolean transition option called elias_transition_option_enable_automatic_variation_transitions. This works both for audio and MIDI tracks, but only if the level in question contains no silent variations.
- Implemented a new "level selection strategy" called elias_level_selection_exact.