
Audio Features

User Manual and Reference

DATE	19/04/2023
PRINCIPAL AUTHOR	Peter Eszlari, Steve Hart
SECONDARY AUTHORS	
VERSION	V 1.5
UPDATE	Updated from 4.6.5 01/11/2024
MISC	

Overview

This document explores the audio capabilities of the **Swift** technology brand and explains how to use all audio features. It is written based on the assumption that the reader is already familiar with **Swift CG** and **Swift CG+**, particularly with Nodes, Timeline editor, Animators, video input, opening video files, using Shaders and Textures.

Introduction

Swift handles audio signals gathered from several different sources. These comprise embedded audio in video streams passing through Swift from the video inputs such as SDI, network based NDI and 2110. Internally video files opened in a project and sound samples that are added to scripts as design elements.

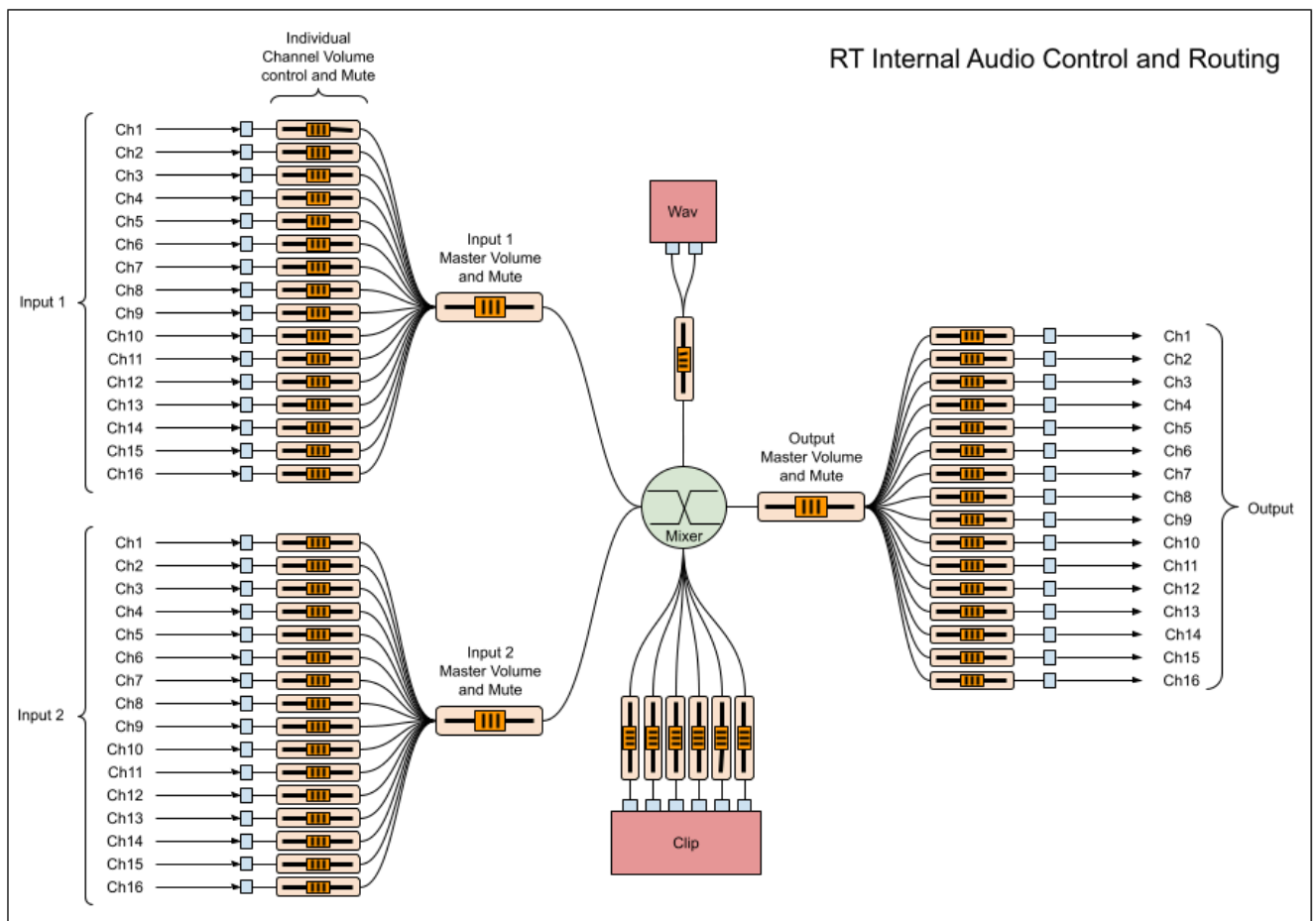
These are all synchronised in time with the running video frames from the point when Swift is launched. Sound samples can be opened and processed in a script by using the Audio Node dedicated for this purpose. Opening sound files, start and stop playback, processing volume and mute can all be controlled by animators and other input methods, in sync with video frames.

Audio controls for video files are also accessible in the Shader Node when a video file is opened as an image. The current audio processing state such as channel routing, volume levels and mute can all be saved in the script by using Audio Nodes and Shader Nodes.

The audio streams from 64 inputs are distributed through 16 output channels that can share multiple sources by mixing streams. These can share outputs with streams passed through from the inputs or other Shader and Audio Nodes. In the following article these features will all be explained in detail.

Audio features in brief

- Playing back any number of audio “wave” format files with multiple channels
- Playing back video files with mono, stereo or Dolby 5.1 sound
- Play back audio from SDI video input streams on 64 audio channels
- Play back audio from network via NDI or 2110
- Process all audio, route to output channels and mix
- Automate audio processing such as volume levels and mute, in sync with video frames using animators or other input methods
- Output audio on 16 individual channels both as embedded SDI and AES/EBU
- Drive graphic parameters with sound amplitude values to build a signal graph
- Layoff video with graphics and sound



Audio Signal Path

Hardware

Swift uses the audio interface of the installed video I/O boards to handle audio streams. This includes getting audio signals into the computer and into Swift from external sources and to output all audio signals after processing. 16 mono audio streams are carried on each video SDI inputs and outputs.

Video board compatibility

AJA video boards

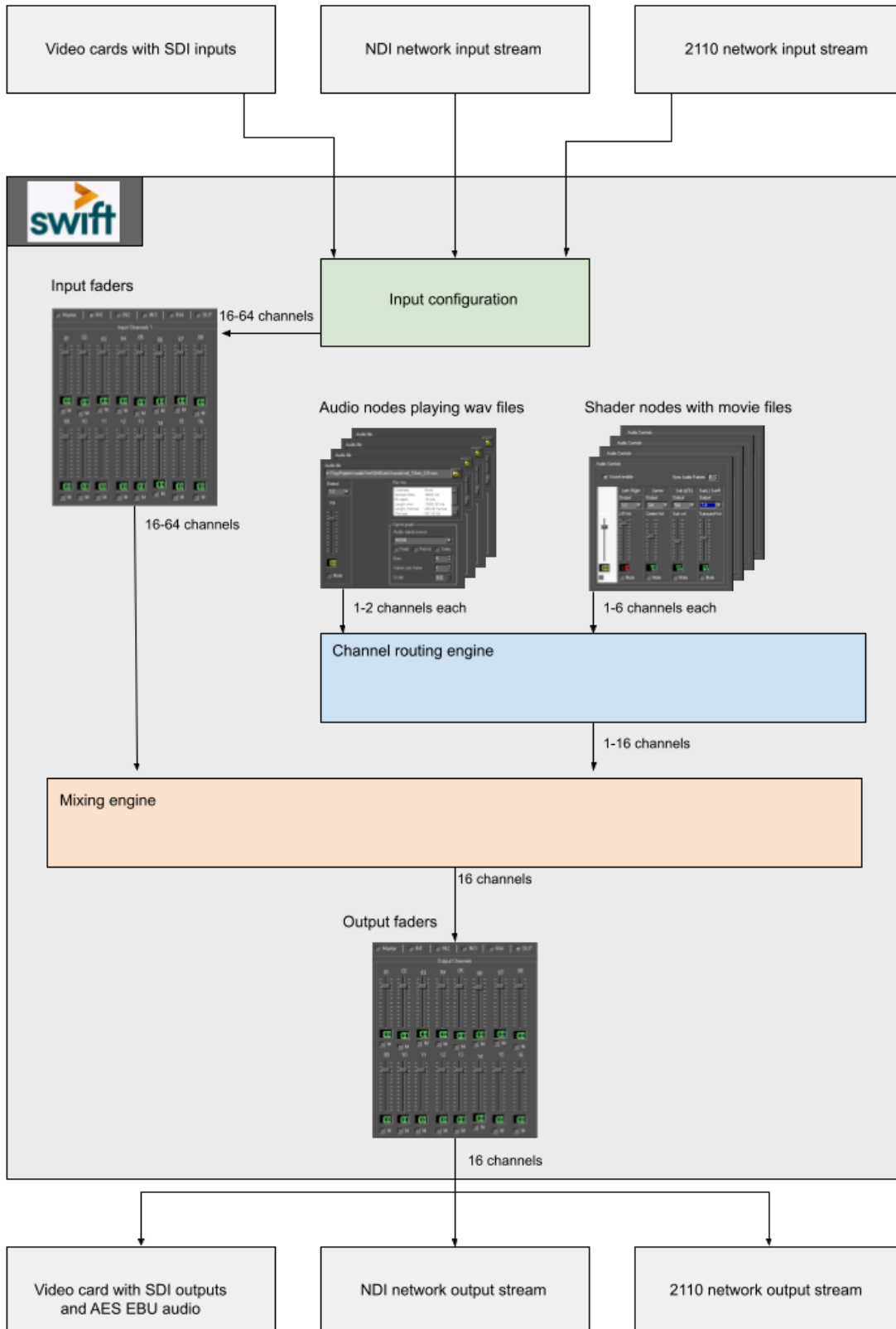
64 channel input on 4 SDI video and 16 channel output on one SDI. Simultaneously 16 channel output on AES/EBU if the installed AJA card supports this.

Connections

SDI : the audio signal is embedded in the SDI video stream that is connected to one of the SDI inputs on the video board. The audio output channels appear on the SDI output sockets (BNC type).

NOTE: Refer to the [Audio Preferences](#) section of this document on how to change audio settings and how to access input and output controls.

Audio signal path diagram



Input section

There are various audio input sources that Swift handles, some external such as SDI and internal such as a Wav file.

External sources:

There are two ways to input audio into Swift and they cannot be used simultaneously: SDI video streams or network streams based on the NDI standard or SMPTE 2110.

SDI AIV (Audio embedded In Video):

The audio signal is embedded in the video stream on the SDI inputs of the video input board. 16 mono audio channels can be carried this way on each SDI.

AJA video boards support up to 64 audio channels on 4 SDI video input streams, depending on AJA model and configuration.

NDI and SMPTE 2110

The audio signal is received via network together with video frames. In every aspect it is handled the same way as the SDI source.

NOTE: All of the unused input channels can be switched to mute to avoid any unwanted signals or low level noise, to enter into the signal path. This is important because even a very low level noise can cause digital clipping (distortion) on the outputs when mixed with other high level audio signal.

Internal sources:

Internal audio sources are audio and video files that are opened in Swift. They are accessible by using nodes.

- Audio files are opened and played back through the Audio Node. See the Audio Node section below.
- Video files are opened and played back as an “image” through the Shader Node. See the Audio in video image – Shader Node section below.

Output section

SDI AIV

Swift outputs 16 mono channels of audio on the SDI video output streams and. The output channel controls can be accessed in the Preferences window under the Audio tab.

AES/EBU

By default AJA boards also support AES/EBU audio output simultaneously to the SDI output. This does not require any configuration in Swift.

AES connection is accessible via break-out cables on selected AJA products, mainly across the Kona range.

NDI and SMPTE 2110

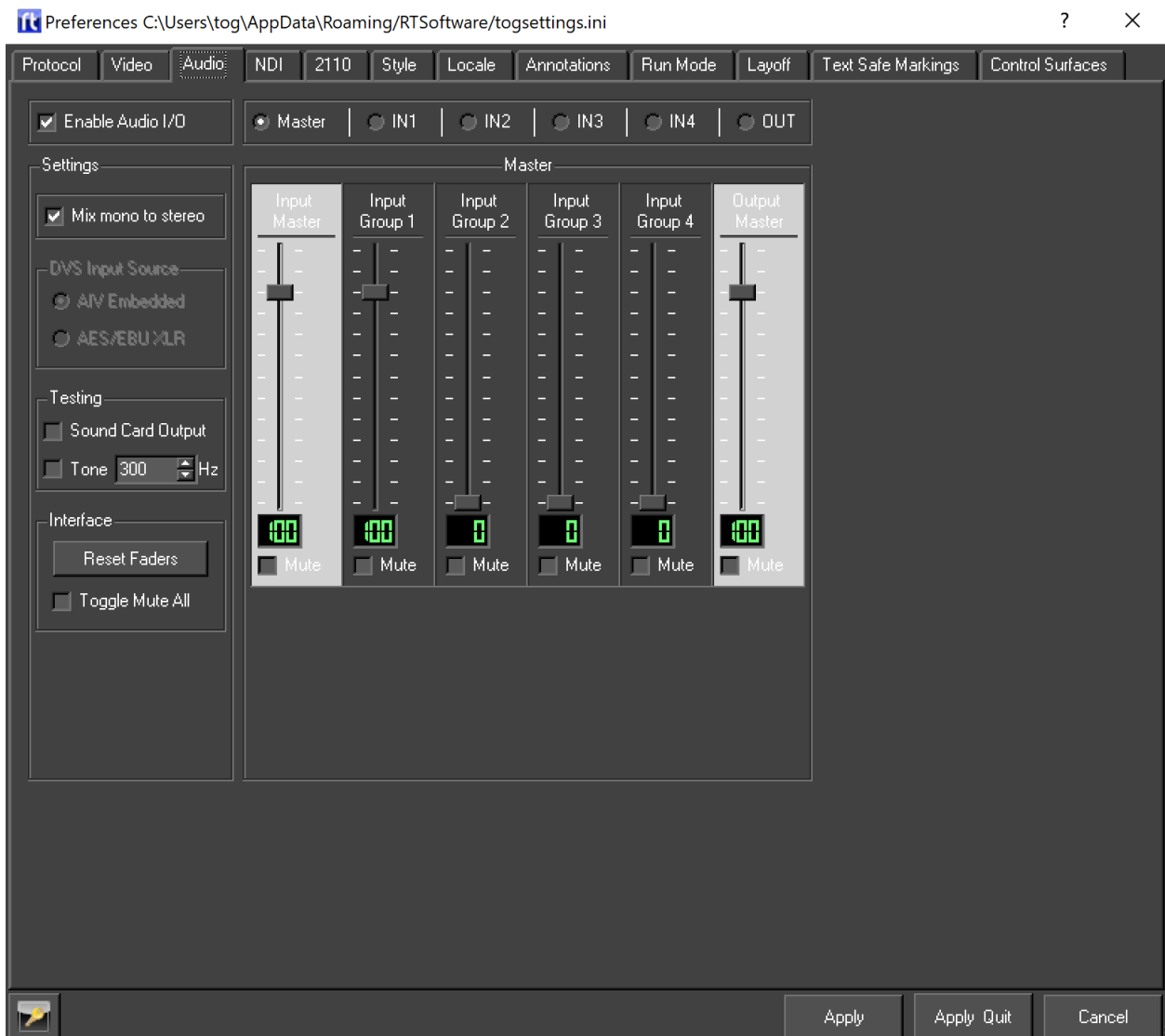
The audio signal is transferred via network together with video frames. In every aspect it is handled the same way as the SDI audio output.

Audio preferences

Audio Tab

The Audio tab in the Preferences editor allows the user to access the audio settings and to control audio input and output levels.

NOTE: *Enable Audio I/O requires Swift to be re-started*



Enable Audio I/O

When deselected, all audio features are bypassed across the application.

Settings section

Mix mono to stereo: Opening a mono wav or video, will playback as **Dual Mono** on two selected consecutive channels on the output instead of playing on one side (one channel) only. This is enabled by default.

Testing section

Soundcard Output: enables domestic level audio output on a machine running a Windows operating system. This is used only to be able to monitor sound output while building a project, testing, etc. The sound engine will output through the default sound device of the PC, sound card, headphone output, etc.

NOTE: Soundcard option currently only supports two outputs, channel 1 and 2 and MS Windows only.

Tone: provides generated test tones at the input stage. Its purpose is to test audio output instantly or to help with building SDI input audio level automation via animator graphics. The tone is affected by the following input volume controls:

- Input Master
- Input Group1
- Input Channels 1: volumes 1-16. - Scroll down to see where to find these at the *Input and Output Pages* section.

In case of configuring **2 video inputs** another pitch will be added to the second input that is somewhat higher than the pitch on input 1. This second tone will be affected by the *Input Group 2* volume control and *Input Channels 2: 1-16*.

- Change the pitch of the tone next to the tickbox, default is 300 Hz.
- Mind setting pitches under 100 Hz with speakers or headphones that are sensitive to low frequencies and prone to damage by them.

Interface section

- **Reset faders:** re-set all channel-faders to unity gain (100). Master faders excluded.
- **Toggle Mute all:** mute / un-mute all channels at once

Master page

The Master page shows six control channel strips. These are independent from the individual channel controls and placed sequentially in the signal chain.

- **Input Master:** This allows control over all the inputs (64) at once.
- **Input Group 1-4:** These each control 16 channels at once for each video input.
- **Output Master:** This allows control over all the outputs (16) at once.

Each channel has a volume fader, a mute button and a digital display showing the fader levels.

The faders are scaled from 0 – 120, 100 being the default value at which setting the volume is not changed. Values 0 – 99 mean attenuation and 101 – 120 mean amplification. The value 0 results in silence.

Input and Output pages

Sharing the same layout, each page has controls for 16 individual channels. Each channel has a volume fader, a mute button and a digital display showing fader levels.



IN1, IN2, IN3, IN4 pages:

These correspond to 4 SDI video inputs on the video board. Each has 16 faders and mute buttons, one for each channel.

OUT page

This allows access to the 16 audio output channels. These controls are the last before the audio streams are routed to the physical outputs. They control the 16 mono channels separately regardless whether their source is mono or part of a stereo or surround source.

NOTE: it is possible to overload the audio signal path, which causes distortion in the signal. Make sure you adjust the input, output and local volume levels correctly, especially if you share output channels between several audio sources. For more information see the [Mixing audio](#) section of this document.

Apply / Apply Quit

Click on this to save audio settings. If the audio settings have been changed since the preferences window was opened a window pops up with three options:

- Save changes: The audio settings are saved to disk and will be re-loaded when Swift starts next time.
- Use changes: Keep the changes for now but do not save them. (Same as the Esc key or closing the preferences window).
- Undo changes: Revert to the settings that were before the preferences window was opened.

Cancel

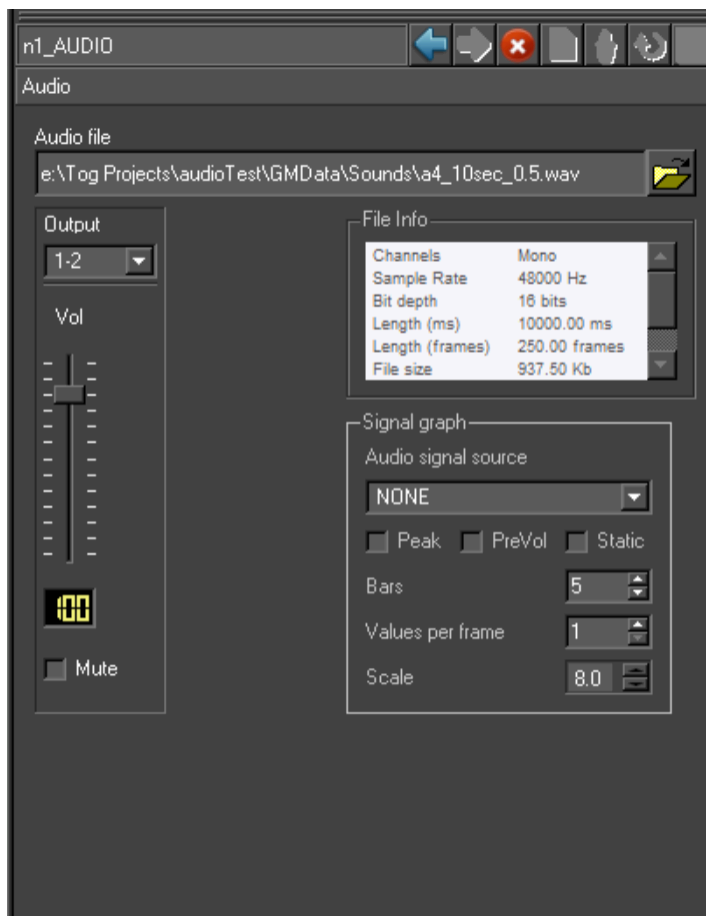
Revert to the settings that were before the preferences window was opened.

Audio Node

The Audio Node allows the user to open and play back audio files in sync with the events of the graphic scripts.

Audio file format requirements

File format	wave (PCM), .wav extension (e.g.: somesound.wav)
Channels	mono or stereo or multiple channels
Bit depth	16, 24 or 32 bit
Sample rate	48000 Hz (48 KHz)



Usage

- Place an Audio Node under an object anywhere into the scene.
- After selecting the Audio Node in the scenegraph, open an audio file using the file chooser in the audio editor window.
- Drag and drop the Audio Node from the scene graph to the timeline and set the StartSound AField as a step animator.

NOTE: For further information on how to use the timeline editor, refer to the [Timeline section of the Design and Edit document](#).

Interface

Audio file

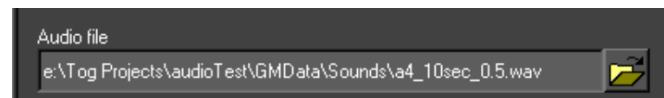
File compatibility:

File format: Wave PCM files (.wav)

Channels: number of channels in the wav

Sample rate: 48000 Hz

Bit depth: 16, 24 or 32 bit

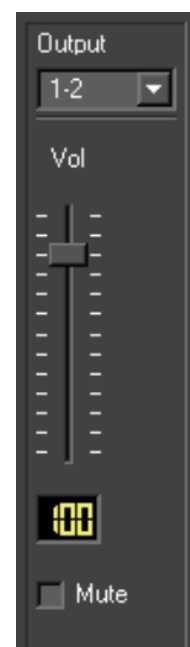


Output

Select output channel

- For mono audio files the channels are selected as one channel for each Audio Node.
- For stereo or multichannel sources the output channels are selected as channel pairs. Each channel is paired with a consecutive channel. (1-2, 2-3, 3-4,...). If there are more channels they will also play in consecutive order to this, all controlled by one volume fader.

There is no limitation on how many Audio Nodes share the same output channels. Refer to the Mixing audio section of this document on how to work with several different audio signal paths in Swift.



Vol

Local volume fader for this audio source, one controlling all included channels at once.

NOTE: When sharing output channels between several sources the volume levels must be set carefully for each source to avoid overloading the signal path.

Mute

Mute or un-mute this Audio Node.

File info

Audio file details are printed here.

Channels: number of channels

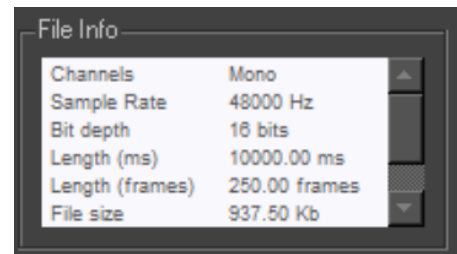
Sample rate: Sampling frequency

Bit depth: Bit resolution

Length (Ms): The length of the file in milliseconds.

Length (frames): The length of the file in video frames.

File size: Size of the opened file in Kbytes



File Info	
Channels	Mono
Sample Rate	48000 Hz
Bit depth	16 bits
Length (ms)	10000.00 ms
Length (frames)	250.00 frames
File size	937.50 Kb

Signal graph

Audio driven graphic parameters.

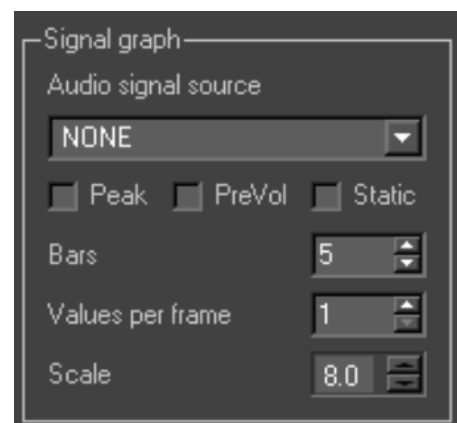
Bars: this must correspond to the number of graphic elements (rectangles , etc) driven by the signal. Not matching these will result in unexpected visual effects..

Values per frame: number of values taken from each frame-length audio content ,taken at equal intervals.

Scale: Scales the graphic in case the driving value is too low/high. It actually scales the value...Default is 10.

Static: enabled by default, this will use Static bars. In

this case the Values per frame should match the number of bars, less values would make some bars freeze or disappear.



Disabling this option will result in a crawling soundwave effect if the Values per Frame is less than the number of bars.

Audio signal source: this is where you select which available audio source should be the basis of this signal graph. Opening a movie or wav file will add to a list of sources by their file name with a pre-tag.

Pre-tags:

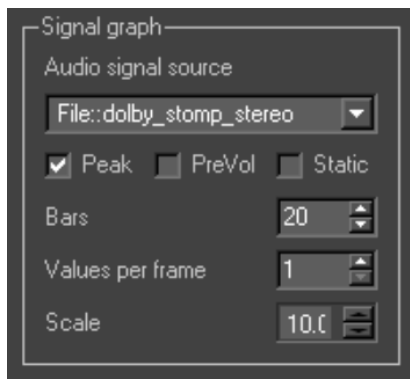
"File::" the graph is driven by an audio file

"Movie::" a video clip

"InputGroup1::" embedded audio on one of 4 video inputs

Peak: the graph takes the highest values per interval while disabling this takes average values in equal periods. Depending on sound content one might work better than the other.

Pre-volume: takes values regardless of the volume fader position or animators.



Wave file playback

Timeline editor

Crop beginning

The Field value in the Step Animator for the Audio Node represents the start of playback of the sound in seconds relative to the beginning of the audio file. By this method the user can virtually crop the beginning of a sound.

e.g.: if the Field value is set to 0.5 the audio playback will start half a second into the audio file when the step animator is hit on the timeline.

Crop end

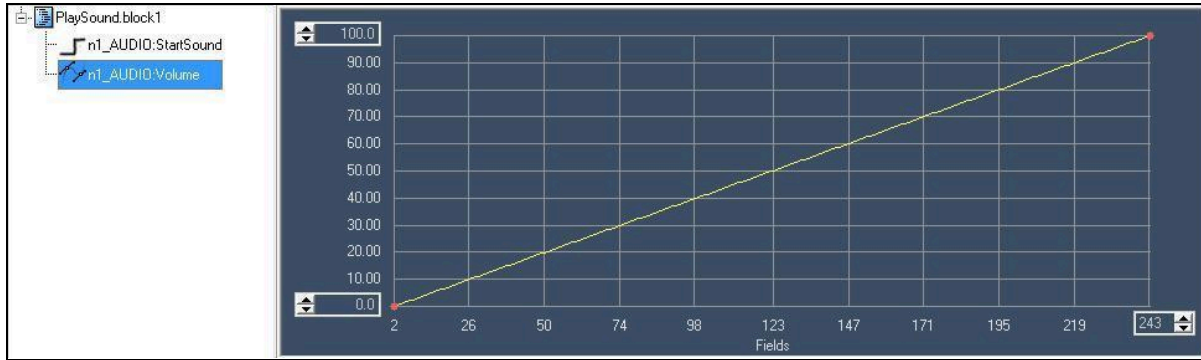
The user can virtually crop the end of the animated sound by setting up a StopSound field as a step animator by dragging the same Audio Node from the scene graph to the timeline.

Volume animation

It is best to use path animators to animate audio volume changes in Swift. The possible values set here by the path correspond to the values set by the volume faders: 0 - 120. (See below)

Volume levels can be animated for all the 16 inputs, outputs and any Audio Nodes by using the corresponding AField property.

NOTE: the resolution of the path animator is "1 value/video field", however the volume changes gradually for each audio sample by automatically calculating the values between the beginning and the end of a field. The volume fader has a logarithmic response curve but the values that the animator path shows represent the movement of the volume fader. Therefore a linearly increasing path will result in a logarithmic volume change.



Mute animation

The easiest way to animate the mute and un-mute functions is to use path animators.

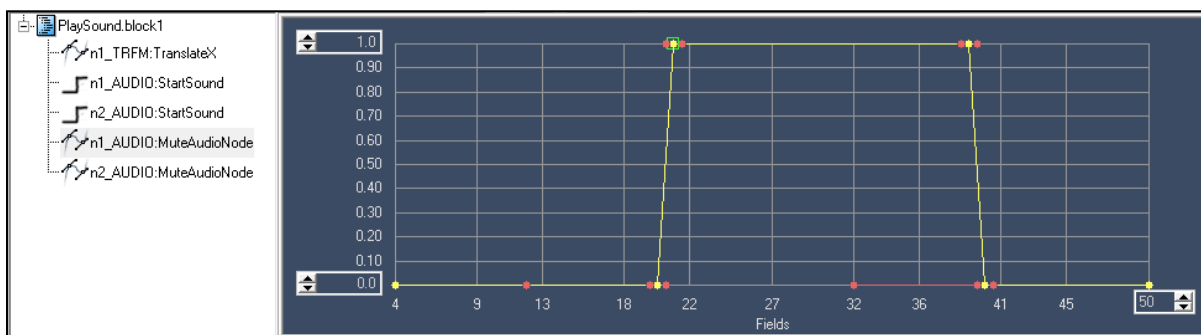
Mute source

Set the value to 1 or bigger

Un-mute source

Set the value to smaller than 1

The mute function can be animated for all the 16 inputs, outputs and any Audio Nodes (Figure 1.1).



Save Audio Node settings

The output channel, volume and mute settings for the Audio Node are saved in the current graphic script (.rb).

Signal graph guide

The signal graph is accessible via Audio Nodes, each Audio Node can drive one graph. One graph can drive 20 bars (or some other type of graphic parameters/elements).

The AField values that transfer sound sample values to graphics are called AudioGraphValue1 to AudioGraphValue20. In this example, values 1-20 each are linked to one of the 20 bars' (rectangles) ScaleY values.



Building a signal graph of 20 bars

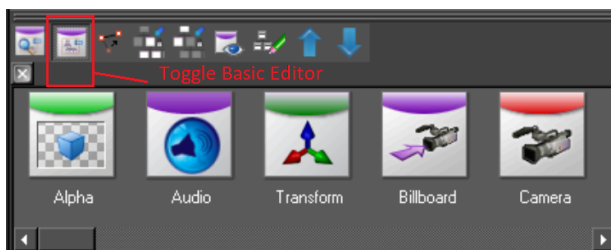
Create 20 narrow rectangles with equal width, laid horizontally in a sequence with an arbitrary gap in-between (or no gap). In this example:

width (ScaleX): 0.05

height: (ScaleY): 0

gap: 10

When one of the transform nodes is selected in the scene graph click on Toggle Basic Editor.

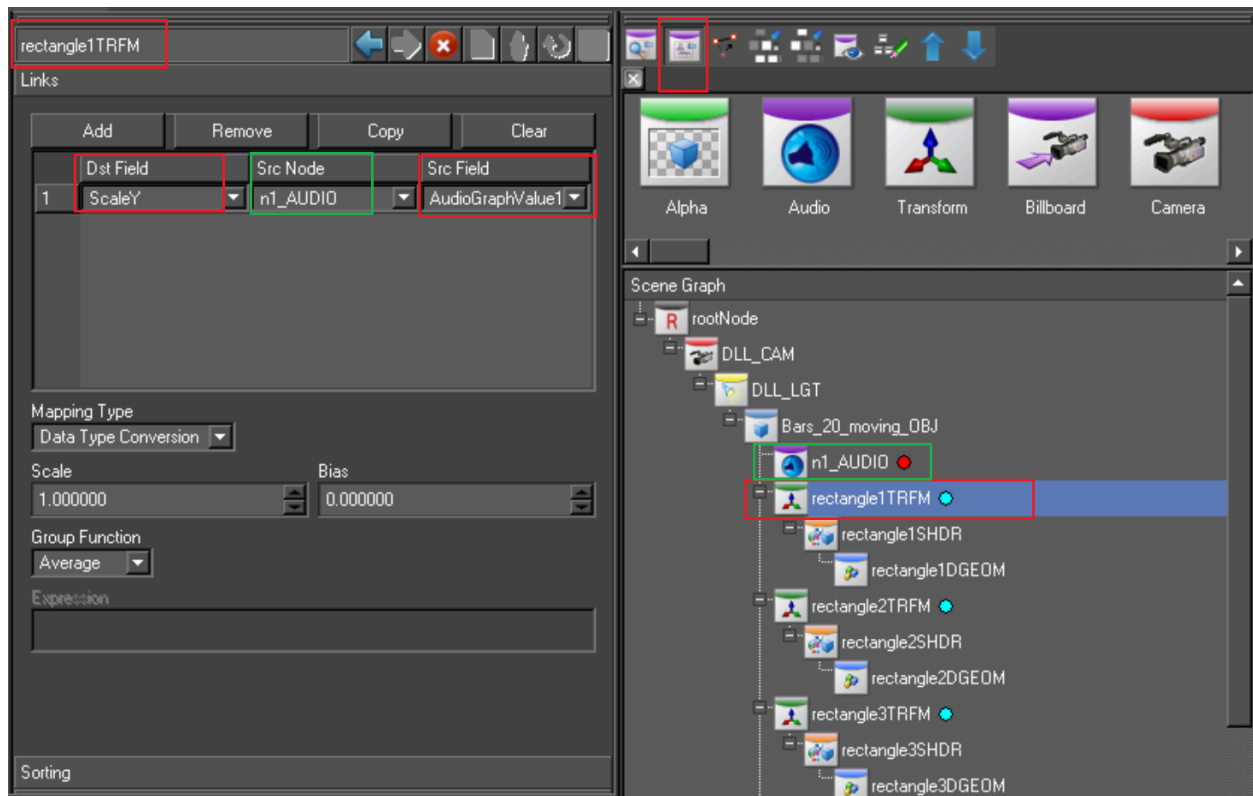


In the editor window Add a new line and set the parameters accordingly:

Dst Field: ScaleY - the destination field to drive the height of the rectangle.

Src Node: n1_AUDIO, it's the default name of an audio node, the source of the graph.

Src Field: AudioGraphValue1 - source field that will drive the Dst Field value above.



Do this for each of the 20 rectangles in the scene graph by clicking on each transfer node, add a new line and set the values above. **Dst Field** and **Src Node** will be the same for all the nodes while **Src Field** will correspond to the position of the rectangle on the x axis going from left to right in this example. Typically, transform node 1-20 will correspond to AudioGraphValue1-20 respectively.

Setting up the graph in Audio Node

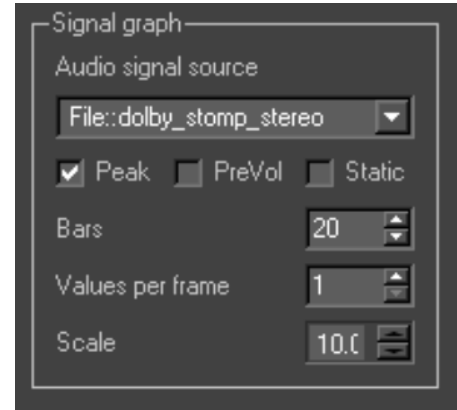
Clicking on the Audio Node in the scene graph reveals the interface for the signal graph

- Set the **Bars** value to **20**. This value must correspond to the number of graphic elements animated.
- Disable **Static**.
- **Values per frame** will determine not only the resolution of the graph but the speed

of movement across the 20 bars. Value 1 will result in the slowest movement. In this setting each value travels across the bars from the highest number to the lowest, typically right to left, taking 20 frames to get across. This gives the visual effect of a soundwave crawling from the right to the left.

The maximum value that is practical to set is the maximum number of bars, in this case 20, when each bar will get a new value at each frame, so none of the values will move to other bars.

- Set the source of the sound, in this case a wav file. Make sure the file is opened first via the Audio File chooser.



Beyond this basic guide there is room to explore and it's up to the users' creativity how to achieve their visual goals with the help of this tool.

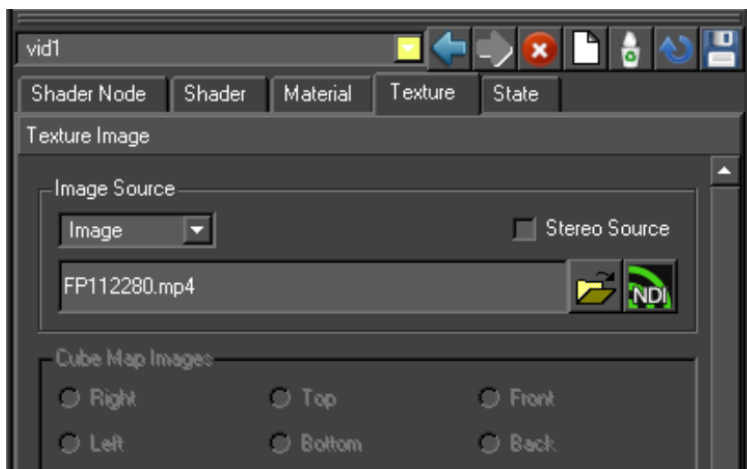
Audio in video image

Shader Node

The Shader Node allows the user to open and play back video files with mono, stereo and Dolby 5.1 PCM audio channels.

Usage

1. Select a Shader Node on the scene graph
2. Select a shader or create new
3. Select the shader editor
4. In the Texture tab select Texture Image and set Image as Image Source
5. With the file browser select a video file saved on your system
6. In the Texture tab open up the Audio controls page to access audio settings.



For more information on how to open and view video files using the Shader Node refer to the Shader Editor section of the Design and Edit documentation.

Audio controls for video

The audio controls here work much like in the Audio Node except that we can control multiple channels depending on the audio content of the video file. The controllers are also enabled or disabled depending on audio content.

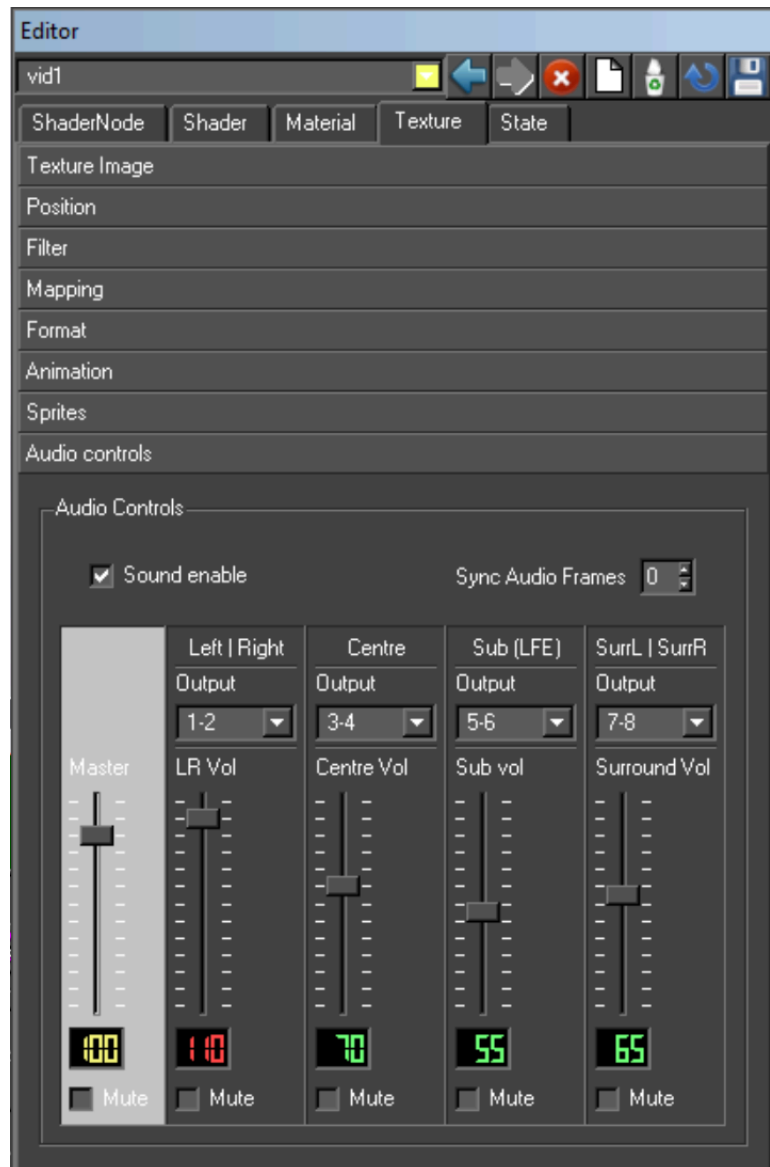
Sound enable

This checkbox is to enable / disable sound playback from this video file. When enabling this option it might take a few seconds for the sound to re-sync with video frames.

Sync Audio Frames

Shifts the audio samples by frames relative to the video frames. This is a number scaled from (-10) to 10.

Various video file formats may need some adjustment to get the audio in sync with the video frames.



Channel strips

There are five set of controls:

- Master to adjust the four streams at once
- Left | Right: for stereo or mono sound
- Centre: for the Centre channel in Dolby 5.1
- Sub (LFE): for the Sub channel (a.k.a LFE) in Dolby 5.1
- SurrL | SurrR: for the Rear Left and Rear Right channels in Dolby 5.1

For each set of controls there is a channel router, a volume fader and a mute check-box.

All channels are mapped out to **4 tracks** that are routed to some of the 16 outputs in consideration of the “**dual mono**” (**Mono To Stereo**) settings in preferences:

1: Left/Right : stereo

2: Centre: mono or dual mono

3: Sub: mono or dual mono

4: SurrL/SurrR: stereo

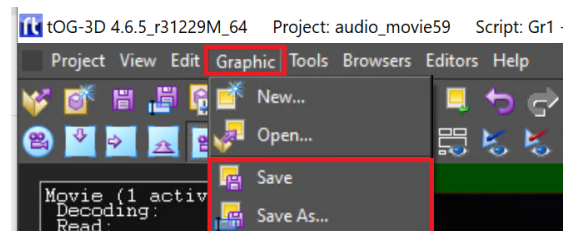
Left–Right and SurroundLeft-SurroundRight are always paired, therefore there are only four output channel routers for the six channels.

Saving audio control settings

These audio control parameters are saved at the following places:

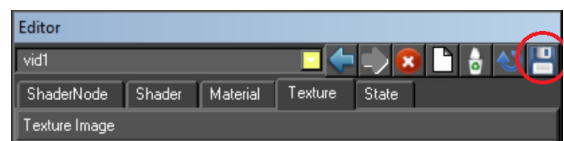
Output channels, volume, mute:

saved in the graphic script (.rb)



Sound Enable :

saved in the texture file when the shader is saved



Mixing audio

All audio streams can be mixed with one another in Swift, i.e. the 16 audio outputs can be shared between several audio sources. This is done by selecting the desired output channels for the source and then setting the balance between them by their local volume faders. When several sources share the same outputs they also share the corresponding output channel volume faders in the preferences editor that control their volume all at once.

Routing options

The audio inputs are “hardwired” to the correspondingly numbered output channels and cannot be changed:

Input 1	Output 1
Input 2	Output 2
~	~
Input 16	Output 16

Volume level can be set for each input by the input volume faders in the Preferences window under the Audio tab.

- Both the Audio Node and the Shader Node can be routed to any of the 16 outputs. Also each output can be shared between any number of sources.
- A stereo source always uses two outputs and a mono source uses one or two when the two are identical in Dual Mono mode.
- The volume levels can be set by local volume faders found at each audio source. When sharing outputs between several audio signals it is crucial to set all signal levels carefully to avoid digital clipping.

E.g.: Suppose you use two audio files on two Audio Nodes. Their peak levels are on the maximum possible level before clipping occurs (0dB). If they share output channels their fader levels have to be set to approximately 80 to avoid clipping. This means an approximate 6 dB attenuation for each. Alternatively, reduce the output level with the clipping channel’s output fader or the master output fader.

Audio via RTDLL implementations

Audio processing within the RT DLL provides the following capabilities:

- 1) Audio in: it is the audio content associated with a video input or program stream. It is constrained to 16 channels (8 stereo pairs) per program stream.
- 2) Wave file sound effect files. These are audio files on disk that may be played and synchronised for a particular graphic template. The volume, channel assignment and mute may be specified as part of the template through either direct set commands or via animators. Wave audio may be mixed in with existing audio inputs or other wave or movie audio channels
- 3) Movie files. These are movie files on disk that contain audio streams. These may be played and synchronised for a particular graphic template. The volume, channel assignment and mute may be specified as part of the template through either direct set commands or via animators. Movie audio may be mixed in with existing audio inputs or other wave or movie audio channels
- 4) Audio out. The 16 channels (8 stereo pairs) combined or mixed audio is passed back as a buffer of interleaved samples. The channel distribution and sample interleaving methods are explained in detail in RT_OEM_DLL Audio Implementation Guide