Tazman-Audio

Fabric Manual

FABRIC

Table of contents

Introduction

Fabric extends Unity's existing audio functionality and provides an extensive set of high level audio components that allows the creation of complex and rich audio behaviours.

Key benefits of Fabric are:

- **It allows the design of many types of audio behaviours quickly and easily.**
- **All game audio assets are located under one hierarchy making it easy to manage and locate.**
- **Quick integration with the game using an easy and intuitive event base system. Trigger audio in the editor or from code.**
- **Custom user interfaces allow to quickly iterate and focus on the audio functionality that is important.**
- **Multiple instances of the same audio behaviour can be triggered from different Game Objects.**
- **Written entirely with Unity's scripting language, doesn't require external native plug-ins and can be used on any platform Unity supports.**

Installation

To install Fabric you simple need to import the Fabric.package into your project.

The package comes with a number of DLL assemblies, documentation, a set of tutorials and the FdpToFtpConverter tool.

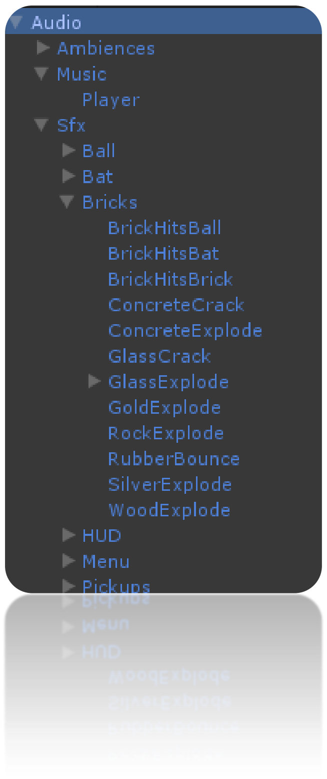
Getting started

Here we will outline some the key concepts and components of Fabric.

Fabric audio hierarchy

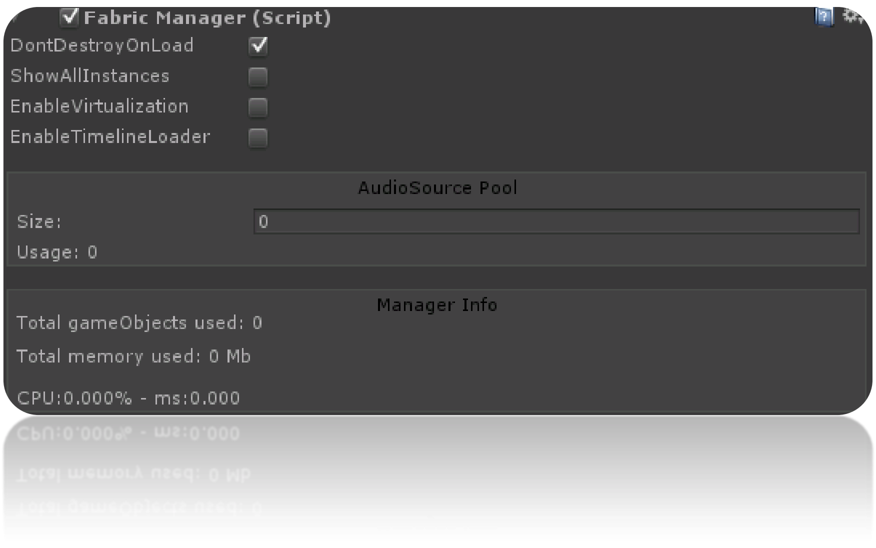
Fabric uses Unity’s object based hierarchical structure in order to build complex audio behaviours.

Each game object must have a Fabric component within it in order to be able to expand the hierarchy.



FabricManager

Fabric manager component must always be at the top of the hierarchy. Its main responsibility is to manage the component hierarchy.



Fabric Manager (Script)

DontDestroyOnLoad ✓
ShowAllInstances
EnableVirtualization
EnableTimelineLoader

AudioSource Pool

Size: 0
Usage: 0

Manager Info

Total gameObjects used: 0
Total memory used: 0 Mb
CPU:0.000% - ms:0.000

Audio Source Pool

The manager also provides a pool of audio sources that are used by the audio components instead of having to create one for each instance. For example if there is an audio component that has a very big number Fabric's default behaviour is to create an audio source for each instance. However, using the audio pool the number of audio sources is pre-allocated.

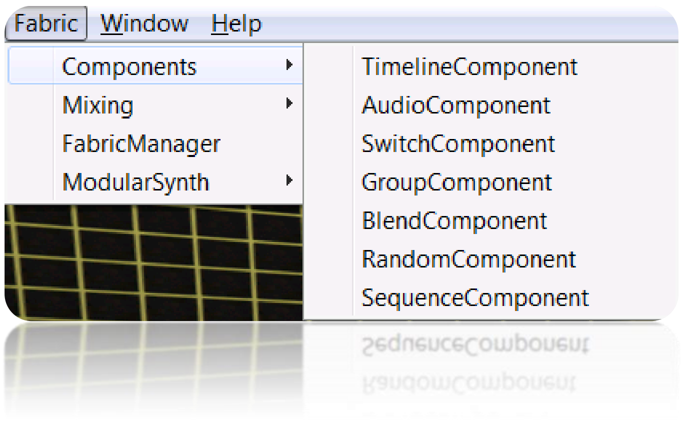
NOTE: At the moment if the maximum number of audio sources are used the pool will fail to allocate a new one.

Usage value shows the number of audio sources that are in use.

Components

Fabric components are the building blocks that allow you to create sophisticated audio designs using any combination of the following available components:

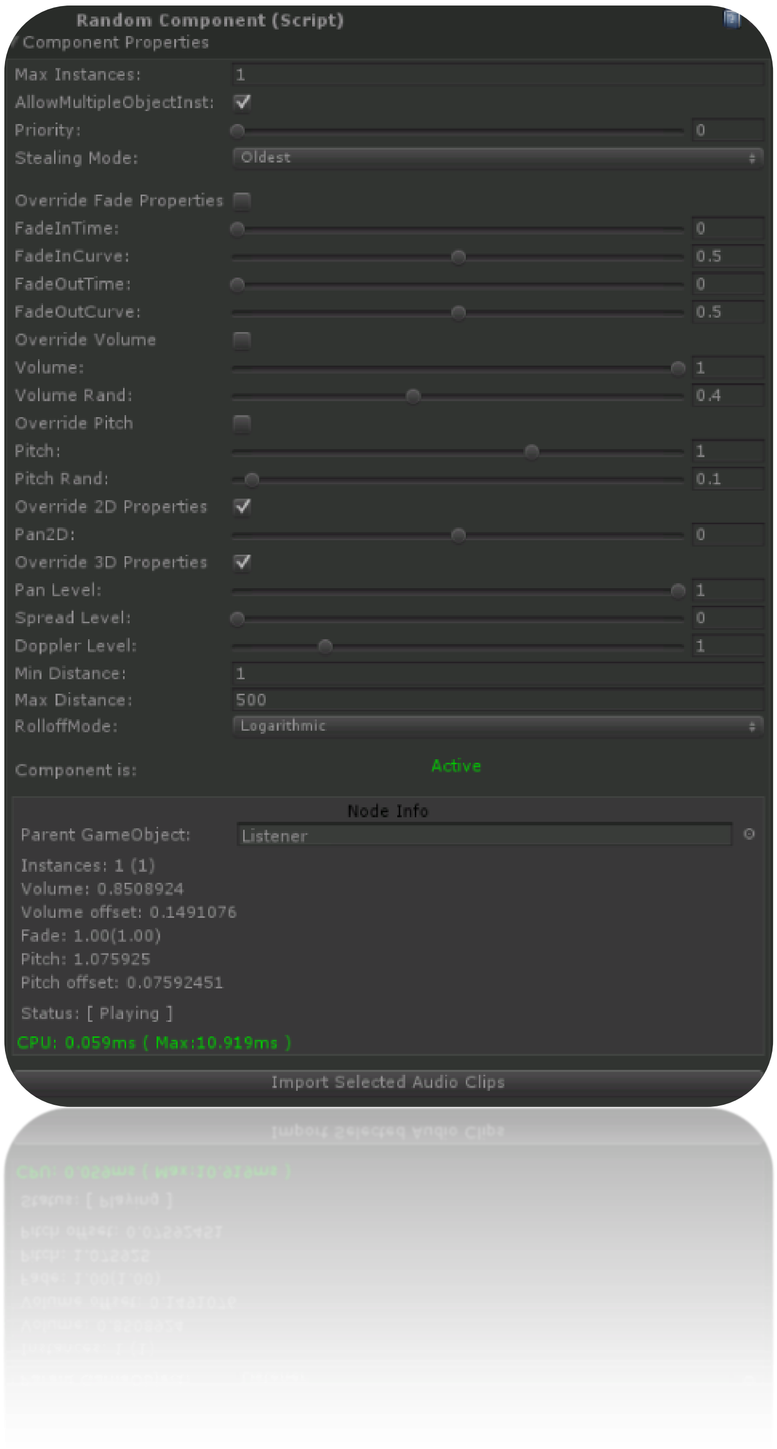
Fabric provides a menu option that allows to automatically create a new component node in the Fabric hiearchy saving time.



Component properties

All components come with a custom inspector UI that allows modifying their properties according to their position in the hierarchy.

A component displays different properties according to their position in the hierarchy or if they have a listener attached.



Random Component (Script)

Component Properties

Max Instances:	1
AllowMultipleObjectInst:	✓
Priority:	0
Stealing Mode:	Oldest
Override Fade Properties	☐
FadeInTime:	0
FadeInCurve:	0.5
FadeOutTime:	0
FadeOutCurve:	0.5
Override Volume	☐
Volume:	1
Volume Rand:	0.4
Override Pitch	☐
Pitch:	1
Pitch Rand:	0.1
Override 2D Properties	✓
Pan2D:	0
Override 3D Properties	✓
Pan Level:	1
Spread Level:	0
Doppler Level:	1
Min Distance:	1
Max Distance:	500
RolloffMode:	Logarithmic
Component is:	Active

Node Info

Parent GameObject: Listener

```
Instances: 1 (1)
Volume: 0.8508924
Volume offset: 0.1491076
Fade: 1.00(1.00)
Pitch: 1.075925
Pitch offset: 0.07592451

Status: [ Playing ]
CPU: 0.059ms ( Max:10.919ms )
```

The node info provides an overview of the component internal state and properties such as: the number of instances used/exist, volume and pitch with their randomisation offsets as well as CPU current/max usage.

Propagation of component properties

All of the component properties are propagated through their hierarchy and are either added or multiplied with their parent properties. However, it is possible to override some of the parent properties and use their own values instead.

Volume / Pitch properties:

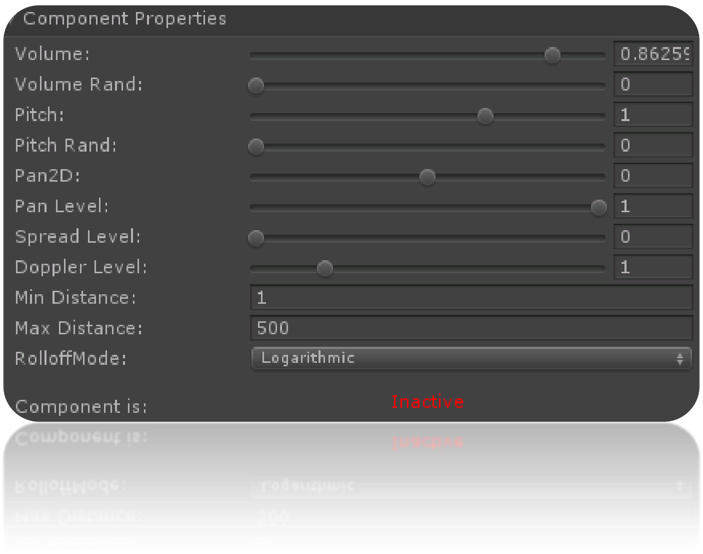
The component volume and pitch properties by default are always multiplied with its parent volume and pitch properties unless the override tick box is selected which then uses the node's properties.

3D/2D Properties:

The default behaviour of these properties is to passed down from their parent unmodified. When the override tick box is selected for either property then they become the properties that are going to be propagated down the hierarchy.

Top node:

All components that are just below the FabricManager are "top nodes" so they don't provide any override options.

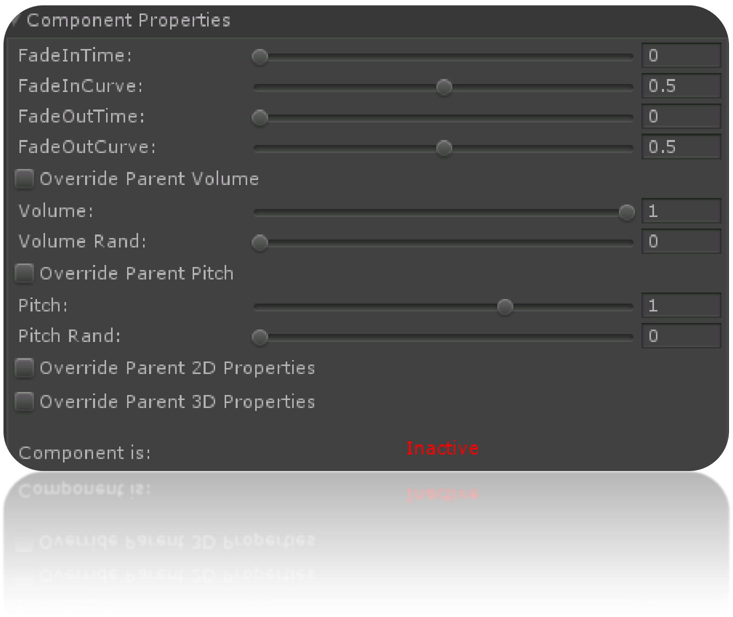


Component Properties

Volume:	0.8625s
Volume Rand:	0
Pitch:	1
Pitch Rand:	0
Pan2D:	0
Pan Level:	1
Spread Level:	0
Doppler Level:	1
Min Distance:	1
Max Distance:	500
RolloffMode:	Logarithmic
Component is:	Inactive

Child node:

When a component is located as a child of another component it allows to override most of the parent properties.

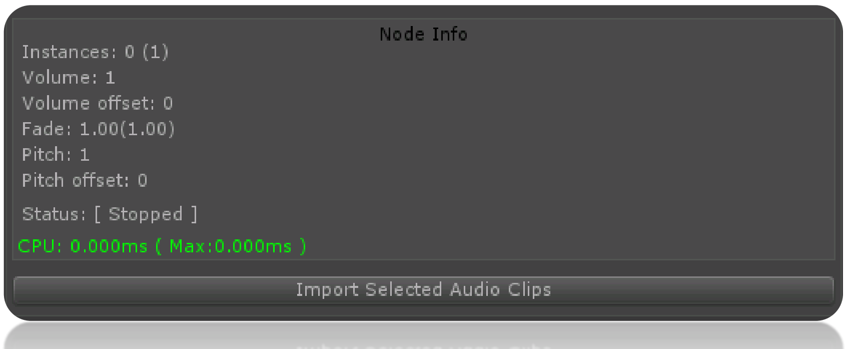


Component Properties

FadeInTime: 0
FadeInCurve: 0.5
FadeOutTime: 0
FadeOutCurve: 0.5
☐ Override Parent Volume
Volume: 1
Volume Rand: 0
☐ Override Parent Pitch
Pitch: 1
Pitch Rand: 0
☐ Override Parent 2D Properties
☐ Override Parent 3D Properties
Component is: Inactive

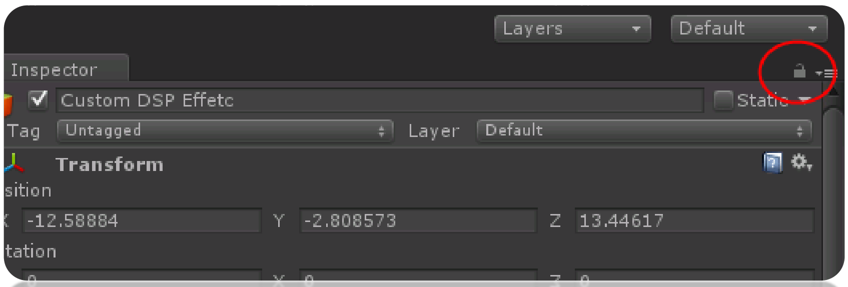
Import Selected Audio Clips:

Fabric provides a method to quickly import a selection of audio clips as audio components. This is particularly useful when there is a random component that needs to have a number of audio components.

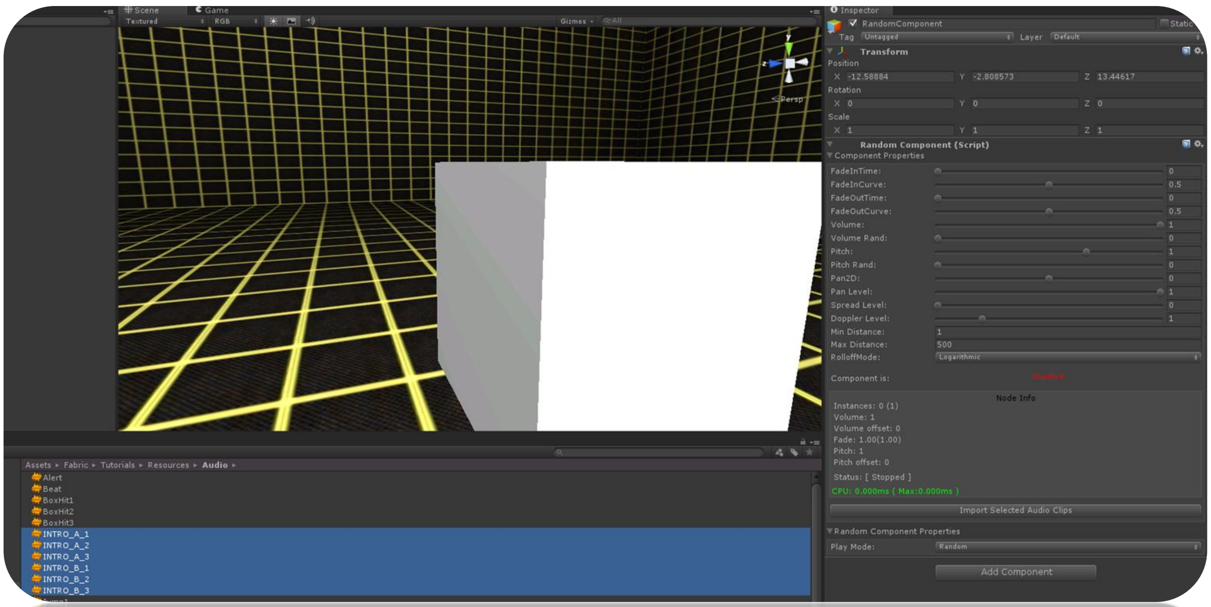


```
Instances: 0 (1)
Volume: 1
Volume offset: 0
Fade: 1.00(1.00)
Pitch: 1
Pitch offset: 0
Status: [ Stopped ]
CPU: 0.000ms ( Max:0.000ms )
```

In order to import multiple audio clips first the inspector UI needs to be locked by pressing the small lock icon located on the top right hand corner (red circle).

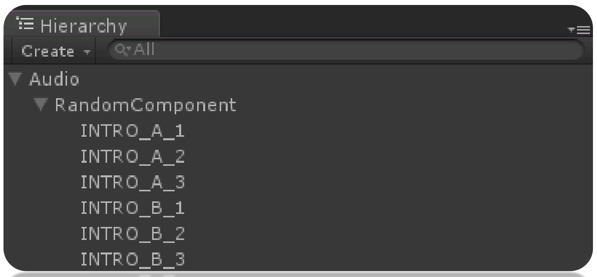


This allows the selection of multiple audio clips without the inspector view loosing focus.



```
RandomComponent                                    Static
Tag  Untagged                 Layer  Default
Transform
Position
  X -12.58884        Y -2.808573        Z 13.44617
Rotation
  X 0                Y 0                Z 0
Scale
  X 1                Y 1                Z 1
Random Component (Script)
Component Properties
FadeInTime:        0
FadeInCurve:       0.5
FadeOutTime:       0
FadeOutCurve:      0.5
Volume:            1
Volume Rand:       0
Pitch:             1
Pitch Rand:        0
Pan2D:             0
Pan Level:         1
Spread Level:      0
Doppler Level:     1
Min Distance:      1
Max Distance:      500
RolloffMode:       Logarithmic
Component is:      Inactive
Node Info
Instances: 0 (1)
Volume: 1
Volume offset: 0
Fade: 1.00(1.00)
Pitch: 1
Pitch offset: 0
Status: [ Stopped ]
CPU: 0.000ms ( Max:0.000ms )
Import Selected Audio Clips
Random Component Properties
Play Mode:         Random
```

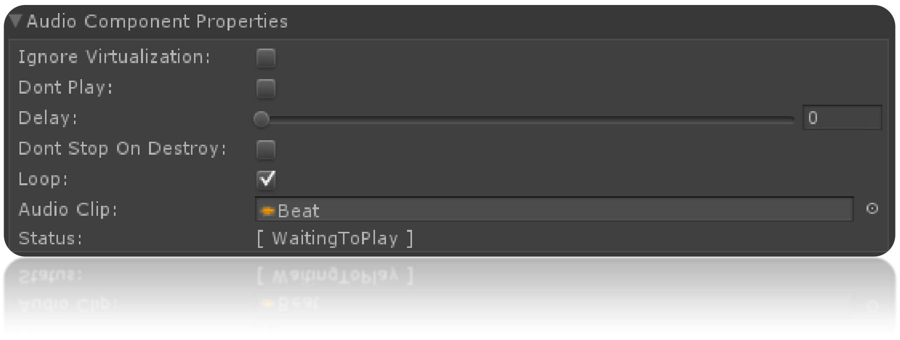
When the “Import Selected Audio Clips” button is pressed Fabric will automatically create game objects with audio components referencing the audio clips.



```
Audio
  RandomComponent
    INTRO_A_1
    INTRO_A_2
    INTRO_A_3
    INTRO_B_1
    INTRO_B_2
    INTRO_B_3
```

Audio Component

This is the lowest component in a hierarchy and extends Unity's existing audio source component functionality. An audio component cannot have any children components attached.



Audio Component Properties

Ignore Virtualization:	☐
Dont Play:	☐
Delay:	0
Dont Stop On Destroy:	☐
Loop:	✓
Audio Clip:	Beat
Status:	[WaitingToPlay]

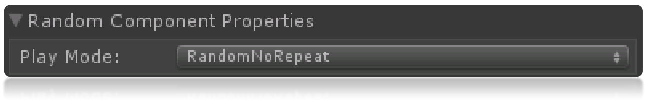
Timeline component

Timeline component provides a powerful multi-track editor functionality that allows layering and crossfading a number of sounds together. It also provides parameters driven by the game that can be linked with the timeline and volume/pitch curves.

A more detailed overview of the timeline features and how it works can be found in the timeline window section.

Random Component

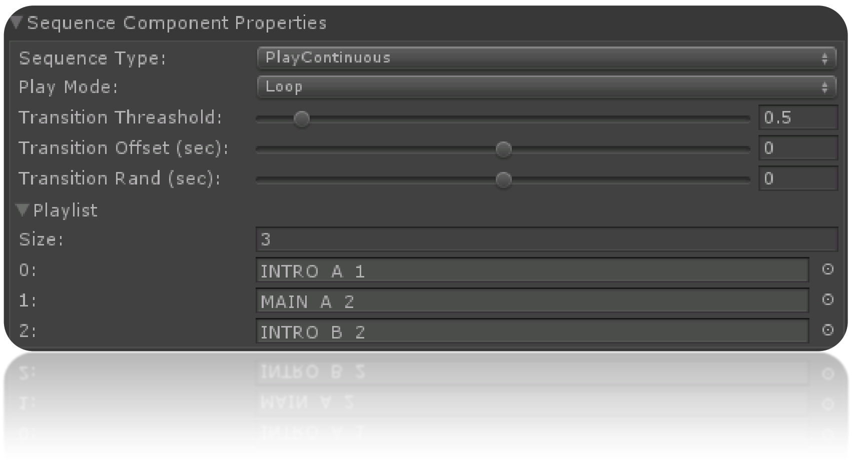
This component triggers a series of child components in a random order or in a way that avoids repetition.



Random Component Properties
Play Mode: RandomNoRepeat

Sequence Component

The sequence component triggers a series of child components defined by a playlist. It continuously advances (PlayContinuous) on the playlist in a sample-accurate way and loops back to the beginning or stops, depending on the play mode selected. It also allows advancing to the next entry when an "AdvanceSequence" event action is received (PlayAdvance).

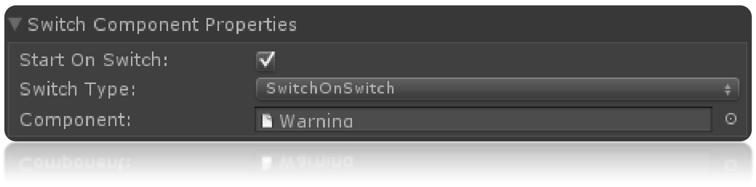


Sequence Component Properties
Sequence Type: PlayContinuous
Play Mode: Loop
Transition Threashold: 0.5
Transition Offset (sec): 0
Transition Rand (sec): 0
Playlist
Size: 3
0: INTRO_A_1
1: MAIN_A_2
2: INTRO_B_2

Finally, it is possible to set and randomize the Transition Offset to occur either before or after the end of the current track. This allows you to overlap two entries or introduce delays etc.

Switch Component

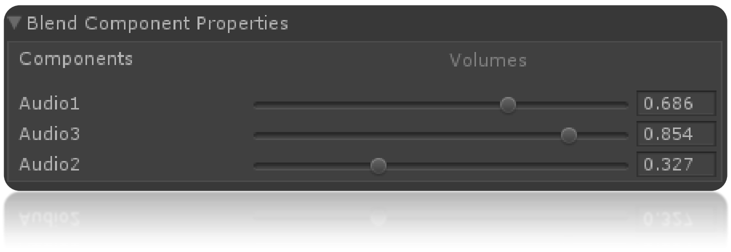
The switch component selects which child component to trigger from an option that is set by the game. The option is **ALWAYS** the name of the component that is required to trigger.


Switch Component Properties

Start On Switch:	✓
Switch Type:	SwitchOnSwitch
Component:	Warning

Blend Component

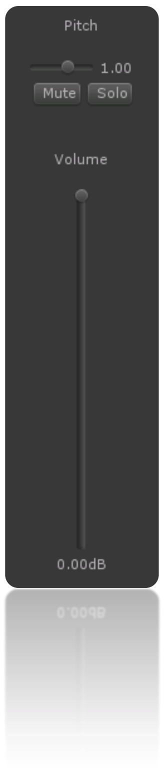
The Blend component is very similar to the group component, it triggers all its child components at the same time but it doesn't appear on the mixer view. Also through its inspector UI it is possible to set the children volumes, saving time from having to go to each node.


Blend Component Properties

Components	Volumes
Audio1	0.686
Audio3	0.854
Audio2	0.327

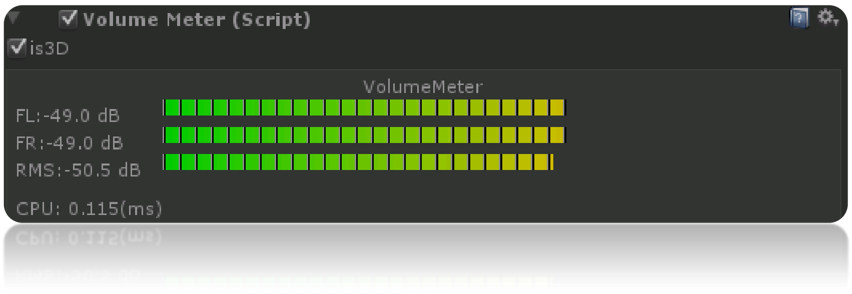
Group Component

The group component controls the volume and pitch properties of all the sounds in its hierarchy.



Volume Meter Component

The volume meter component displays the audio levels of the node in which the component is added. It will try and collect a mixed down version of ALL audio sources that exists below the node.



Volume Meter (Script)

is3D

VolumeMeter

FL:-49.0 dB

FR:-49.0 dB

RMS:-50.5 dB

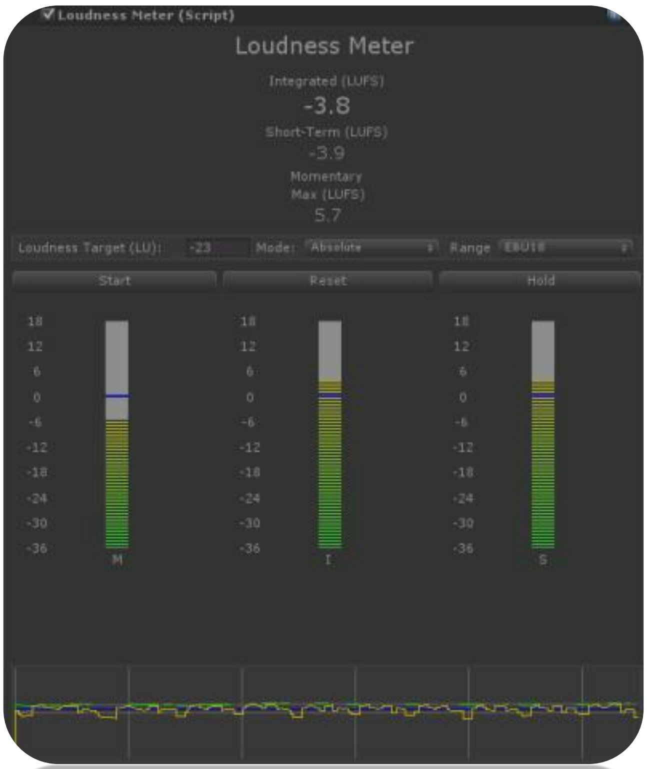
CPU: 0.115(ms)

Loudness Meter Component

Loudness meter offers EBU R 128-compliant loudness metering which measures the loudness levels that listeners will perceive.

There are three standard types of loudness metering available:

- Momentary (M) loudness with an integration time of 400ms.
- Short-term (S) loudness with an integration time of 3s.
- Integrated (I) loudness which is a long-term metering with a gating function.


✓ Loudness Meter (Script)

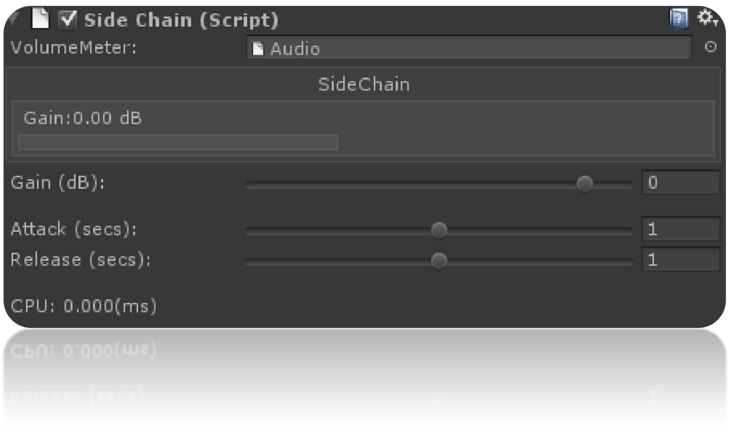
Loudness Meter

Integrated (LUFS)
-3.8
Short-Term (LUFS)
-3.9
Momentary
Max (LUFS)
5.7

Loudness Target (LU): -23 Mode: Absolute Range: EBU18

Side chain Component

This component uses the volume meter RMS property in order to reduce the volume of the component in which is added. A typical method using a side chain is reducing the music levels during speech.



Side Chain (Script)

VolumeMeter: Audio

SideChain

Gain:0.00 dB

Gain (dB): 0

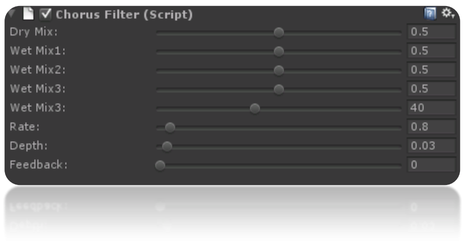
Attack (secs): 1

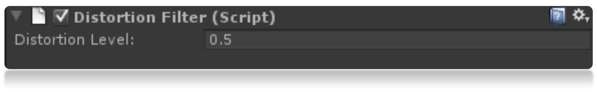
Release (secs): 1

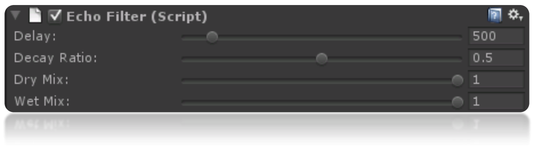
CPU: 0.000(ms)

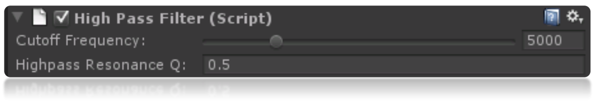
DSP Components

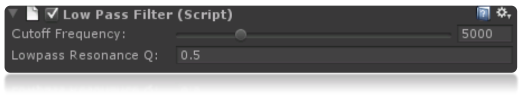
Fabric provides the same number of DSP components that Unity supports. A DSP component can be inserted anywhere in the Fabric hierarchy. At runtime it will collect all audio sources underneath that node and will add the effect on each source. This way it's possible to set an effect property at the same time for all instances either through the component inspector UI, Fabric's event system or in code using the API

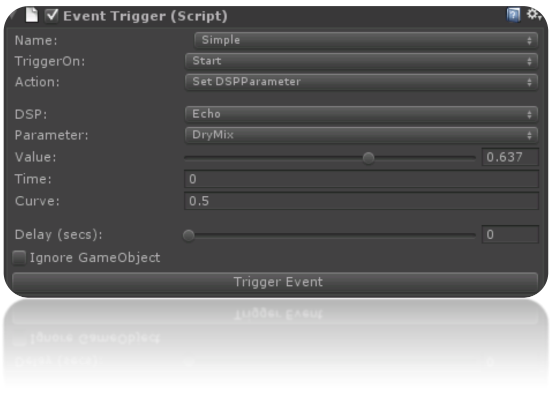

Chorus Filter (Script)
Dry Mix: 0.5
Wet Mix1: 0.5
Wet Mix2: 0.5
Wet Mix3: 0.5
Wet Mix3: 40
Rate: 0.8
Depth: 0.03
Feedback: 0


Distortion Filter (Script)
Distortion Level: 0.5


Echo Filter (Script)
Delay: 500
Decay Ratio: 0.5
Dry Mix: 1
Wet Mix: 1


High Pass Filter (Script)
Cutoff Frequency: 5000
Highpass Resonance Q: 0.5

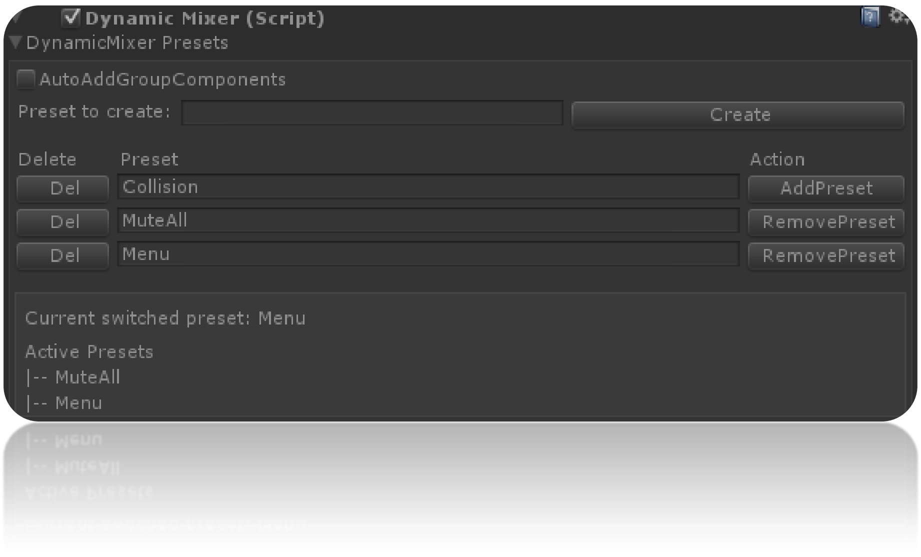

Low Pass Filter (Script)
Cutoff Frequency: 5000
Lowpass Resonance Q: 0.5


Event Trigger (Script)
Name: Simple
TriggerOn: Start
Action: Set DSPParameter
DSP: Echo
Parameter: DryMix
Value: 0.637
Time: 0
Curve: 0.5
Delay (secs): 0
Ignore GameObject
Trigger Event

Dynamic Mixer

The dynamic mixer component changes the audio balance of the game at runtime. Its strength lies in the ability to add multiple presets together, each changing the volume or pitch properties of a group component. The properties from each active preset are added together to produce the final volume and pitch values that are then passed to the group components. It is also possible to switch from one preset to another thereby giving a nice, smooth transition of the parameters.

In the dynamic mixer inspector it is possible to create or delete presets that can be called up at any time either from the event trigger component or programmatically using the API.


Dynamic Mixer (Script)
DynamicMixer Presets
AutoAddGroupComponents
Preset to create: Create

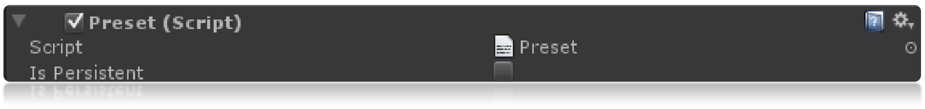
Delete	Preset	Action
Del	Collision	AddPreset
Del	MuteAll	RemovePreset
Del	Menu	RemovePreset

Current switched preset: Menu
Active Presets
|-- MuteAll
|-- Menu

Preset

Presets is simply a collection of group presets with a unique name. Presets can be activated at any moment either from the event trigger component or through the API.

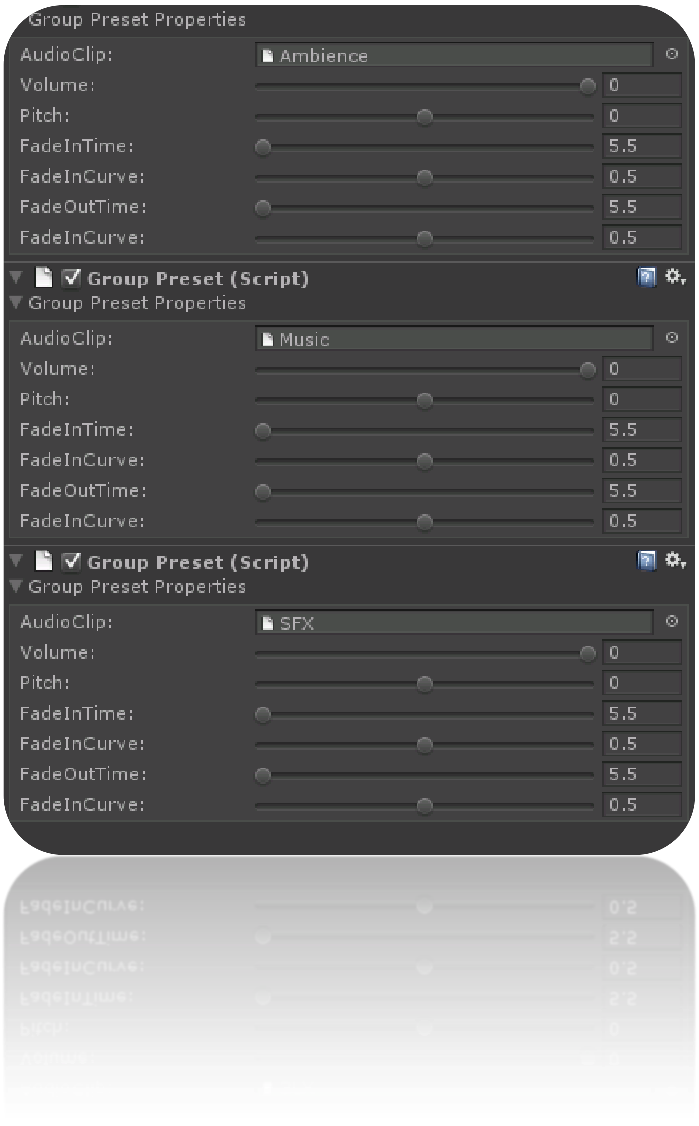
A preset can be persistent which allows it to remain active even when the dynamic mixer receives a reset event action.


Preset (Script)

Script	Preset
Is Persistent	☐

Group preset

A group preset defines properties that will make a change to an assigned group component. It is possible to change the volume, pitch and specify the duration and curve type (i.e. linear, log, exp) for each property change.



Group Preset Properties

AudioClip:	Ambience
Volume:	0
Pitch:	0
FadeInTime:	5.5
FadeInCurve:	0.5
FadeOutTime:	5.5
FadeInCurve:	0.5

Group Preset (Script)

Group Preset Properties

AudioClip:	Music
Volume:	0
Pitch:	0
FadeInTime:	5.5
FadeInCurve:	0.5
FadeOutTime:	5.5
FadeInCurve:	0.5

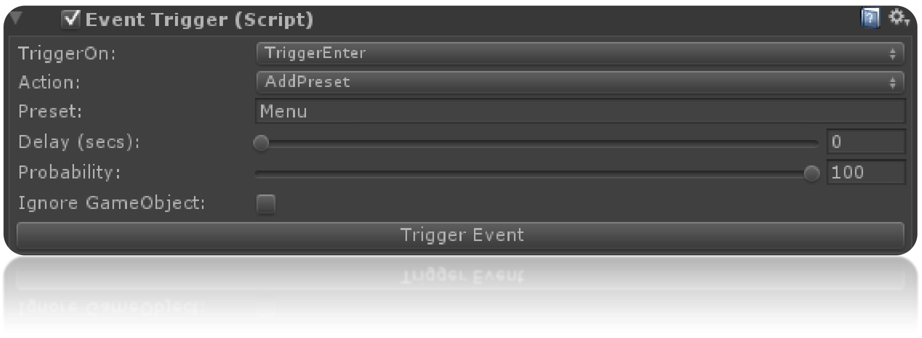
Group Preset (Script)

Group Preset Properties

AudioClip:	SFX
Volume:	0
Pitch:	0
FadeInTime:	5.5
FadeInCurve:	0.5
FadeOutTime:	5.5
FadeInCurve:	0.5

Adding presets at runtime

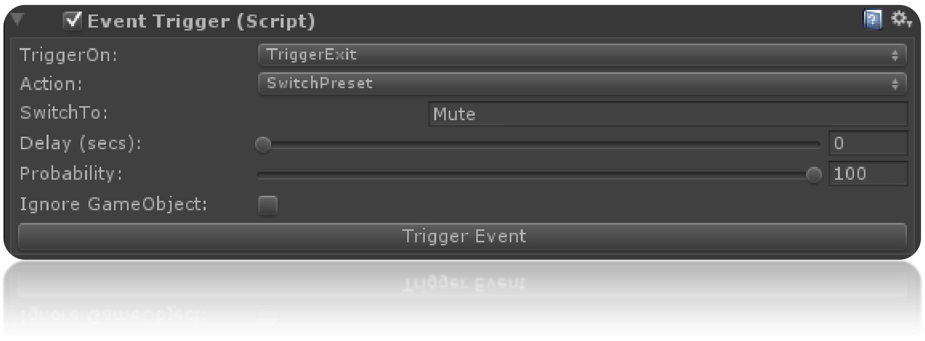
Adding a preset at runtime could be done either through the event trigger component or the API.


Event Trigger (Script)

TriggerOn: TriggerEnter
Action: AddPreset
Preset: Menu
Delay (secs): 0
Probability: 100
Ignore GameObject:

Switch preset at runtime

Sending a “SwitchPreset” event action will cause the mixer to switch from the currently active preset to the target preset.


Event Trigger (Script)

TriggerOn: TriggerExit
Action: SwitchPreset
SwitchTo: Mute
Delay (secs): 0
Probability: 100
Ignore GameObject:

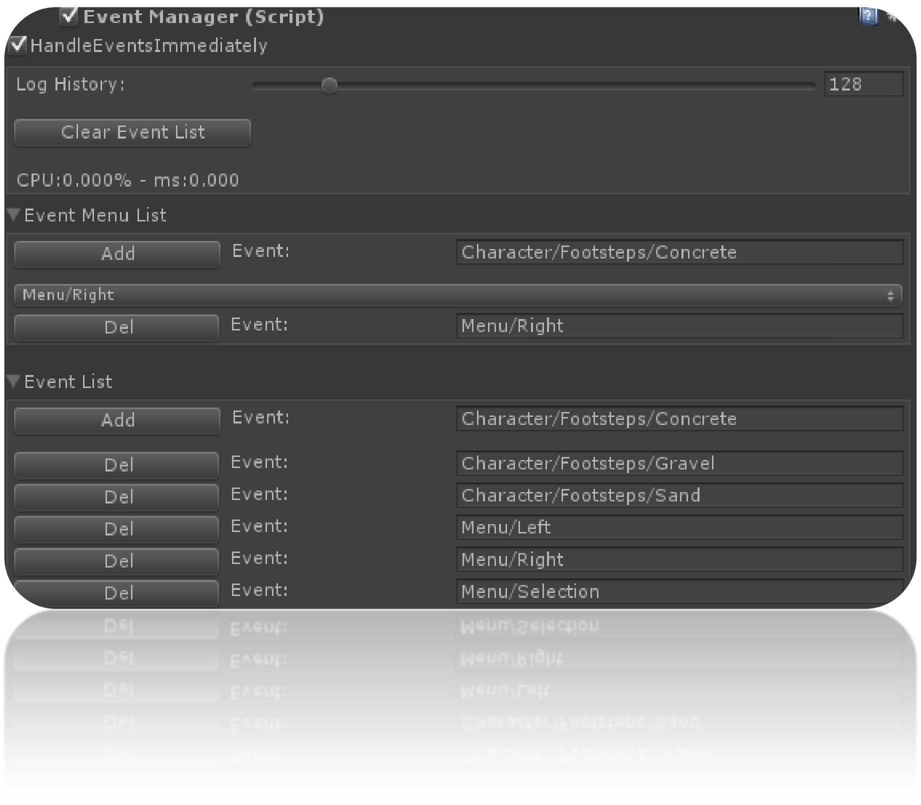
Events

Fabric's event based system allows events to be used by the game in order to drive the audio. Each event can be set to perform an action chosen from a list such as: play, stop, set volume, set pitch, set parameters. Events can be implemented very easily in a game either in the editor or by code even before any audio authoring is done. This method allows creating and re-iterating on the audio content of the game quickly and efficiently.

EventManager

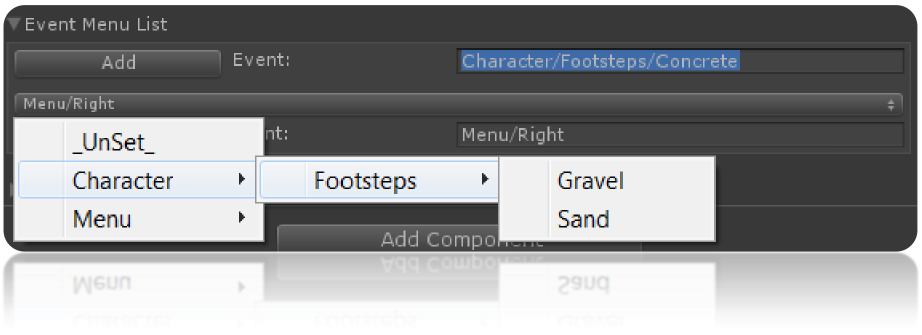
The event manager component is responsible for managing the list of events as well as the flow of events between the game and the fabric components.

NOTE: EventManager component needs to be located in the same game object as the Fabric Manager component. Fabric will automatically add it at runtime if one is not available but it will not be possible to store the event list.



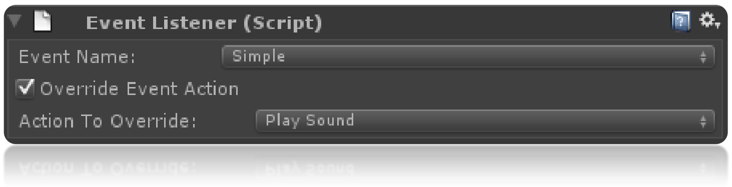
Event Manager (Script)
HandleEventsImmediately
Log History: 128
Clear Event List
CPU:0.000% - ms:0.000
Event Menu List
Add Event: Character/Footsteps/Concrete
Menu/Right
Del Event: Menu/Right
Event List
Add Event: Character/Footsteps/Concrete
Del Event: Character/Footsteps/Gravel
Del Event: Character/Footsteps/Sand
Del Event: Menu/Left
Del Event: Menu/Right
Del Event: Menu/Selection

The EventManager allows the creation of event sub-menu hierarchy by using the "/" when typing an event path. So for example the event with the name "Character/Footsteps/Gravel" will be shown as a sub-menu for both the event listener and trigger menu options.



Listener

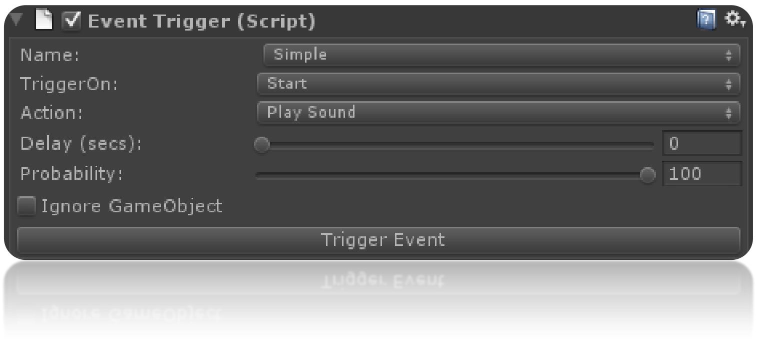
When a listener component is inserted in the same game object as a component component it allows for the component to be triggered by the game. The listener simply waits and listens for a specific event name and only when it receives that name it will respond and perform an action. Furthermore, a listener can accept and set parameters if they are supported by the component.


Event Listener (Script)
Event Name: Simple
☑ Override Event Action
Action To Override: Play Sound

It is possible to override the event action of the incoming event. This could be useful when one event can start a component but at the same time it can stop another component.

Trigger

The Event trigger component is responsible for sending an event into Fabric with a specific action (i.e. Play, Stop, Set Volume/Pitch) and parameter values. Every component that is listening for that event within the Fabric hierarchy will respond accordingly.


Event Trigger (Script)

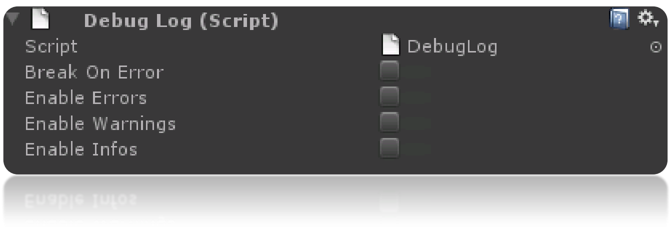
Name:	Simple
TriggerOn:	Start
Action:	Play Sound
Delay (secs):	0
Probability:	100
Ignore GameObject	☐

An even trigger can be set to trigger when certain game object function is called or by code through the event manager.

Debug log

DebugLog provides the option to choose which type of messages will be displayed in the console output therefore reduces the amount of information displayed.

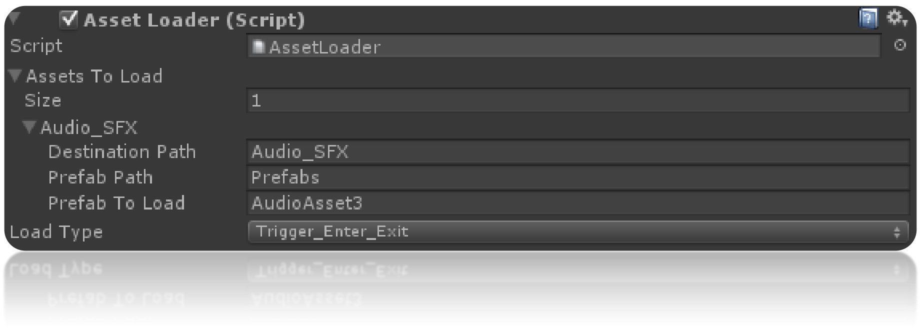
NOTE: The DebugLog component needs to be located in the same game object as the Fabric Manager component. Fabric will automatically add it at runtime if one is not available but it will not be possible to store it's properties.


Debug Log (Script)

Script	DebugLog
Break On Error	☐
Enable Errors	☐
Enable Warnings	☐
Enable Infos	☐

AssetLoader

AssetLoader is a component that loads and unloads prefab assets that contain Fabric components. It is possible to define when the asset will load the list of its prefabs such as: Start/Destroy, Enable/Disable, Trigger Enter/Exit, Collision Enter/Exit.



Asset Loader (Script)

Script	AssetLoader
Assets To Load	
Size	1
Audio_SFX	
Destination Path	Audio_SFX
Prefab Path	Prefabs
Prefab To Load	AudioAsset3
Load Type	Trigger_Enter_Exit

NOTE: On the destination path each node name MUST be separated with the "_" character.

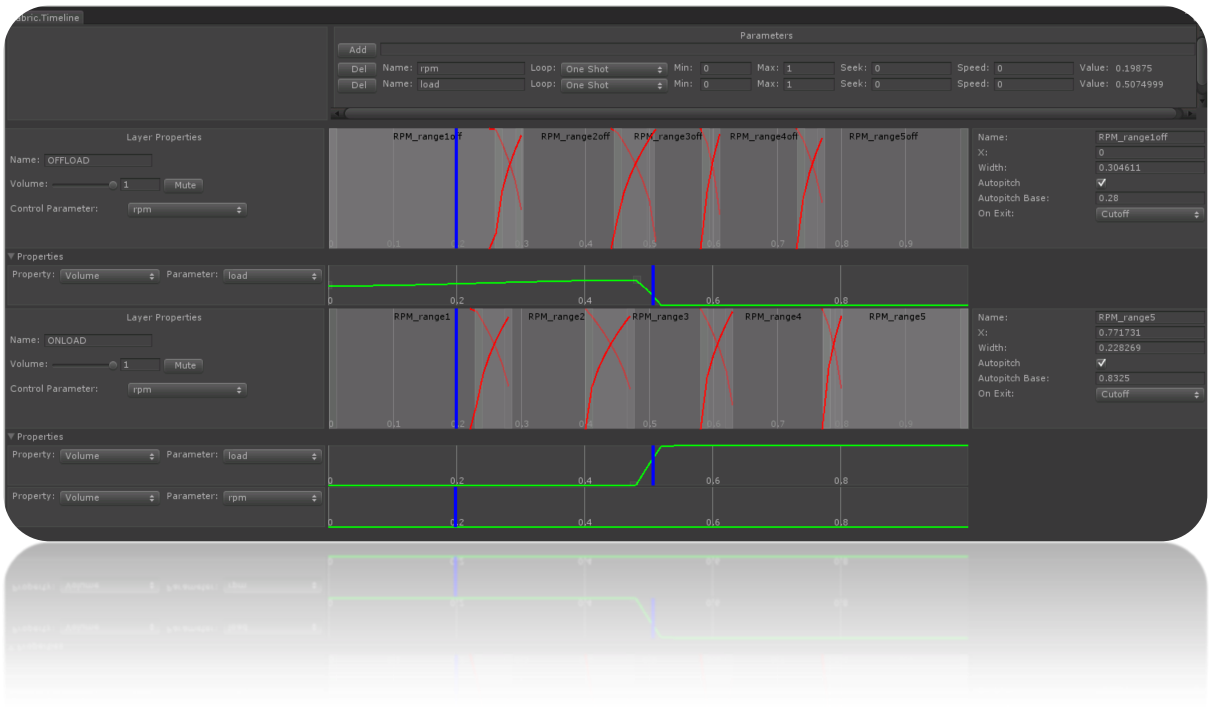
Fabric windows

Fabric provides a number of custom windows that allows quickly iterating and focusing on the audio functionality that is important.

Timeline

The timeline component is a powerful multi-track component that allows layering and crossfading a number of sounds together, linking the timeline and volume/pitch curves with game parameters.

A timeline component can be added in the hierarchy by selecting the Fabric->Components->Timeline menu option.



Name: rpm Loop: One Shot Min: 0 Max: 1 Seek: 0 Speed: 0 Value: 0.19875
Name: load Loop: One Shot Min: 0 Max: 1 Seek: 0 Speed: 0 Value: 0.5074999

Layer Properties
Name: OFFLOAD
Volume: 1
Control Parameter: rpm

Name: RPM_range1off
X: 0
Width: 0.304611
Autopitch ✓
Autopitch Base: 0.28
On Exit: Cutoff

Properties
Property: Volume Parameter: load

Layer Properties
Name: ONLOAD
Volume: 1
Control Parameter: rpm

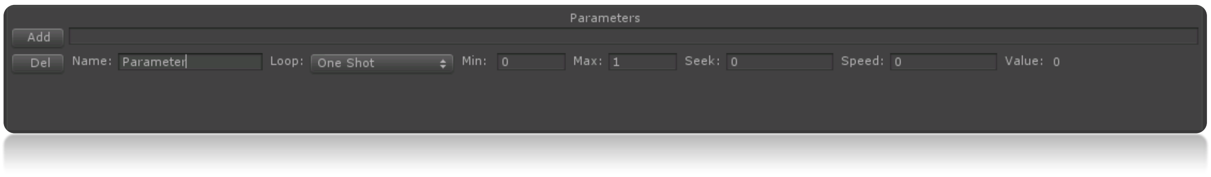
Name: RPM_range5
X: 0.771731
Width: 0.228269
Autopitch ✓
Autopitch Base: 0.8325
On Exit: Cutoff

Properties
Property: Volume Parameter: load
Property: Volume Parameter: rpm

Parameters

A timelime parameter provides the link between the game and the timeline internal functionality. With a parameter it is possible to trigger regions that are placed in a layer as well as control property/parameter graphs.

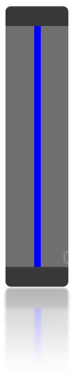
A timeline parameter can be controlled using the event trigger component or through the API.



Name: Parameter Loop: One Shot Min: 0 Max: 1 Seek: 0 Speed: 0 Value: 0

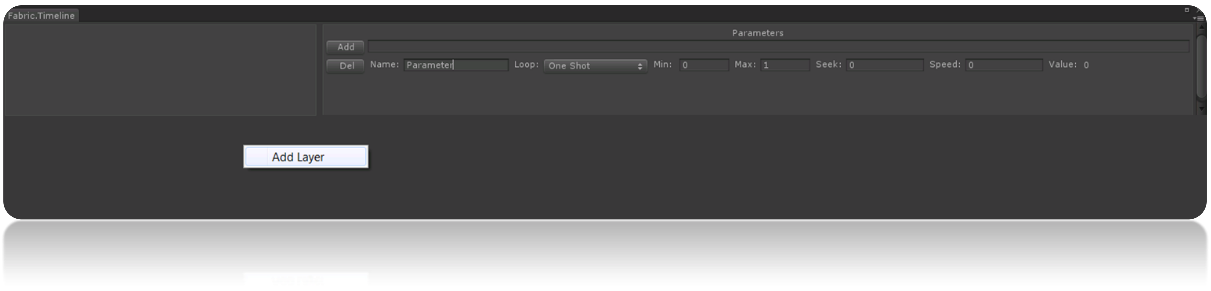
When a layer has a controlled parameter selected it displays a blue vertical bar in the timeline view. The length of the timeline is mapped to the range of the parameter.

It is possible to select the parameter bar and drag it left or right or use the slider displayed in the timeline inspector view.



Layer

A layer is added into the timeline by right clicking into an empty region and selecting the “Add Layer” option.

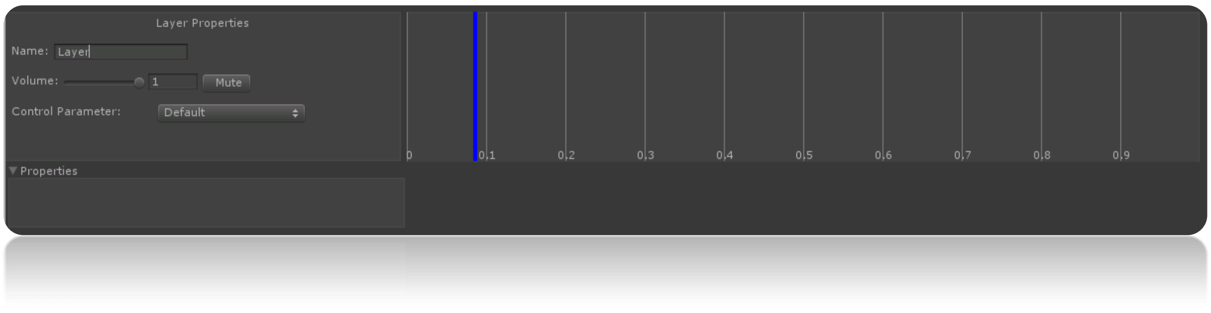

Fabric.Timeline

Parameters

Add

Del Name: Parameter Loop: One Shot Min: 0 Max: 1 Seek: 0 Speed: 0 Value: 0

A layer provides a timeline in which multiple regions can be inserted. When two or more regions overlap a crossfade is applied.


Layer Properties

Name: Layer

Volume: 1 Mute

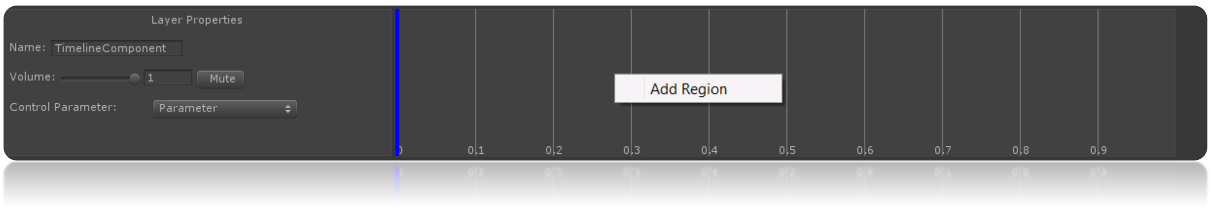
Control Parameter: Default

Properties

Region

A region defines an area in the timeline in which any component will be activated when the parameter cursor is entered, and stopped, according to the “On Exit” behaviour, when is exited.

A region can be added by selecting anywhere in the layer timeline and right clicking.

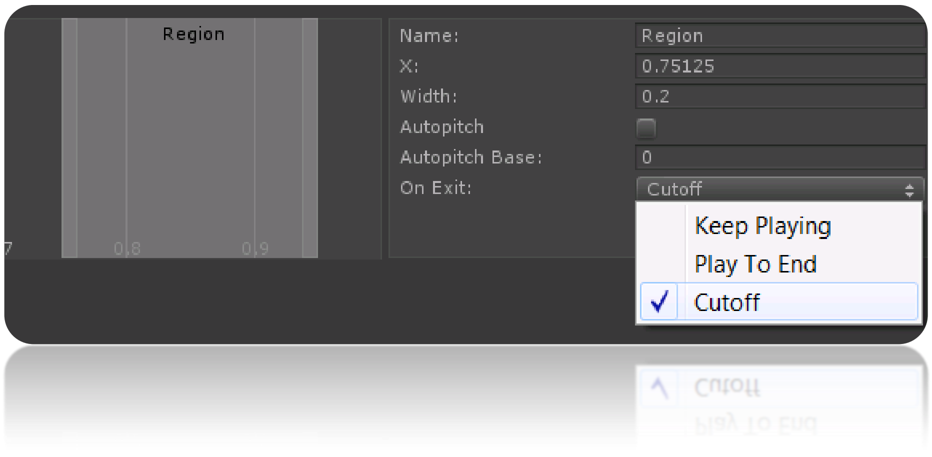

Layer Properties

Name: TimelineComponent

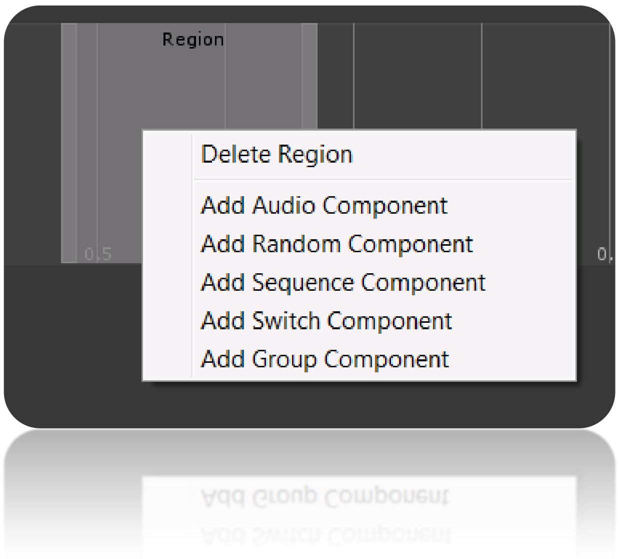
Volume: 1 Mute

Control Parameter: Parameter

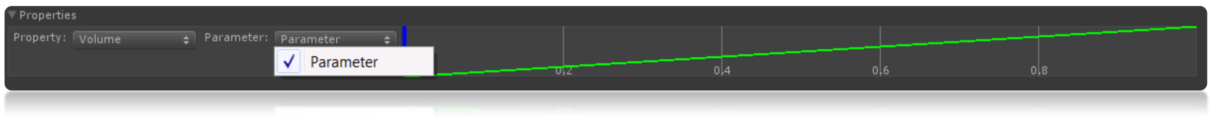
A region provides a number of properties. For a detailed description of each property please refer to the reference manual.


Name: Region
X: 0.75125
Width: 0.2
Autopitch
Autopitch Base: 0
On Exit: Cutoff

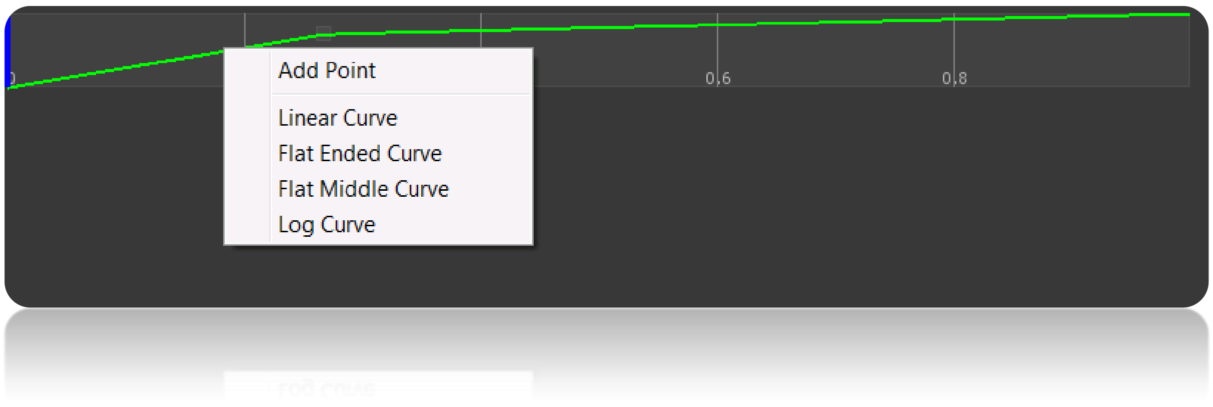
When a region is created it is possible to right click on it and add any of the Fabric components shown in the context menu.



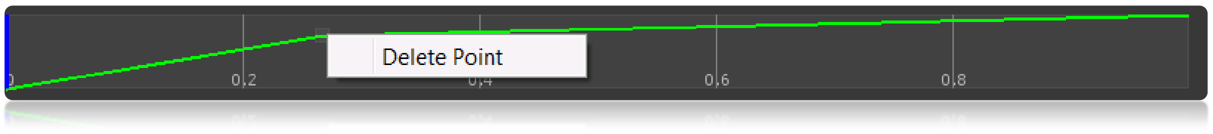
On each layer it is possible to add property/parameter graphs that allow to link and control a layer property from a timeline parameter such as volume and pitch.


Properties
Property: Volume
Parameter: Parameter

It is possible to add points in the graph by right click on the line. It is also possible to define the type of curve for the line between two points.

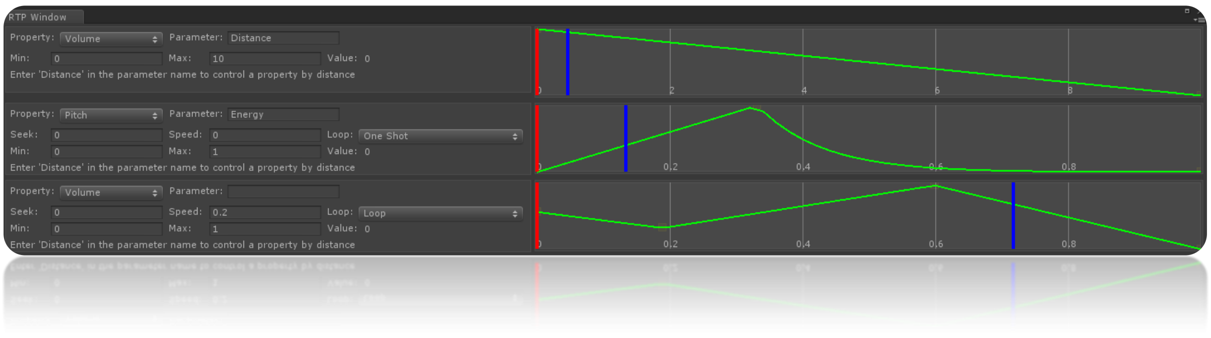


A point can be deleted by selecting on it and pressing the mouse right click.



Runtime parameter (RTP) Window

The runtime parameter (RTP) window allows you to link parameters from the game with internal component properties.

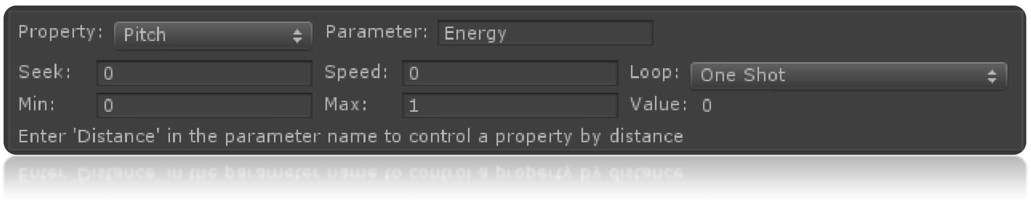

RTP Window

Property: Volume Parameter: Distance
Min: 0 Max: 10 Value: 0
Enter 'Distance' in the parameter name to control a property by distance

Property: Pitch Parameter: Energy
Seek: 0 Speed: 0 Loop: One Shot
Min: 0 Max: 1 Value: 0
Enter 'Distance' in the parameter name to control a property by distance

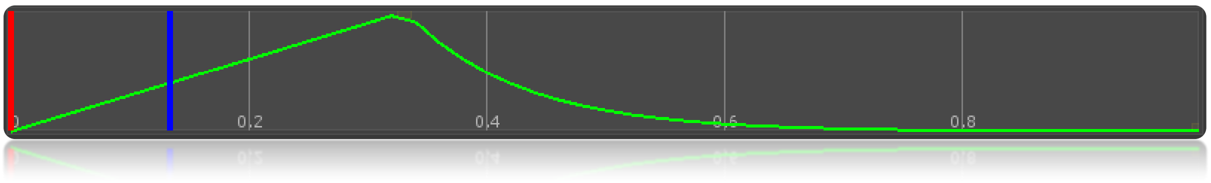
Property: Volume Parameter:
Seek: 0 Speed: 0.2 Loop: Loop
Min: 0 Max: 1 Value: 0
Enter 'Distance' in the parameter name to control a property by distance

The RTP parameter area allows setting which property a parameter will control. It is possible for a parameter to have a seek value that defines the amount of time it will take to reach the target value, speed which moves it continuously in one direction and loop type that determines if the parameter that has speed will loop when it reaches the end.


Property: Pitch Parameter: Energy
Seek: 0 Speed: 0 Loop: One Shot
Min: 0 Max: 1 Value: 0
Enter 'Distance' in the parameter name to control a property by distance

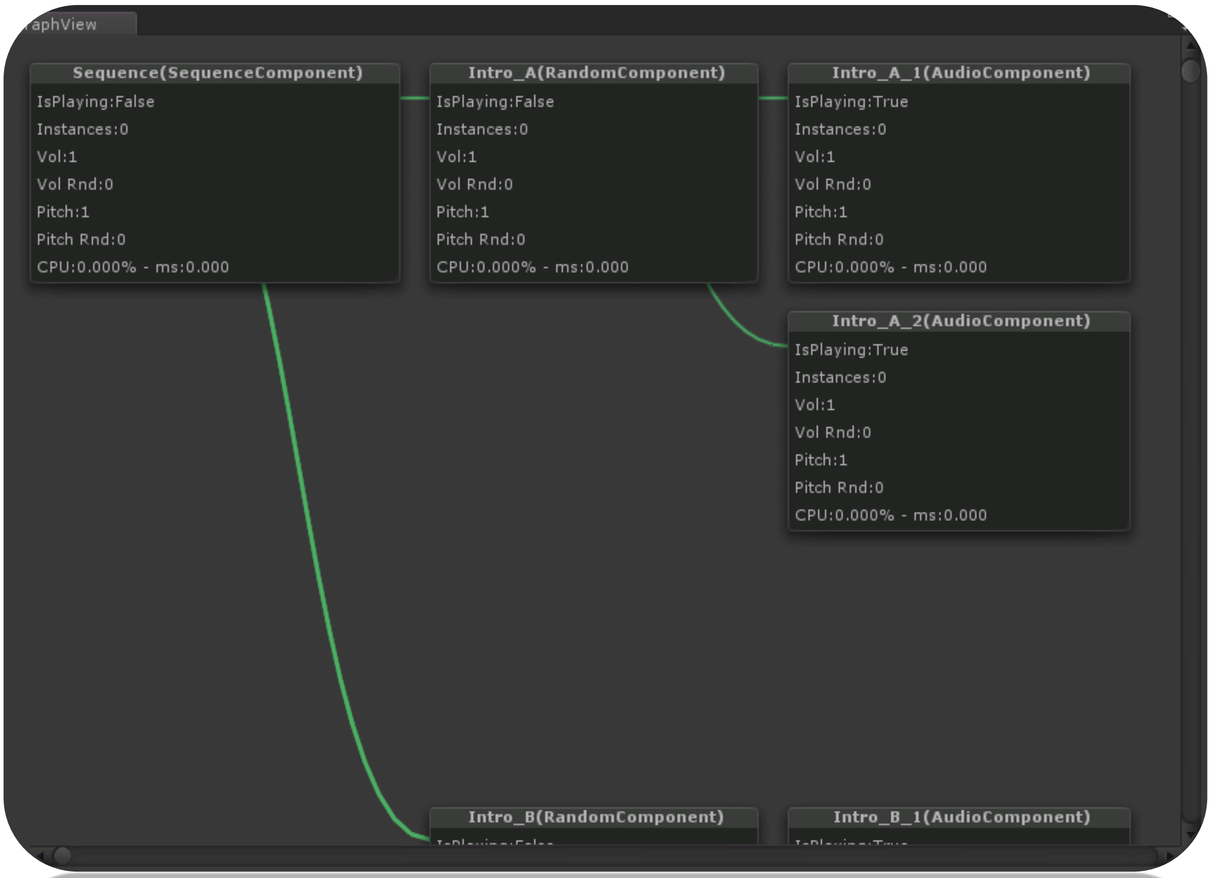
It is possible to get access to the distance value between the component and the camera if you type "Distance" in the parameter name. This way it is possible to control any of the component properties using the distance parameters.

The graph defines the mapping between the input parameter and the property that will be changed. Just like the timeline it is possible to add/delete points in a graph, change the curve line between points (i.e. linear, log, exp etc)



Graph View

Provides a flat view of the whole component hierarchy with runtime information of their state.

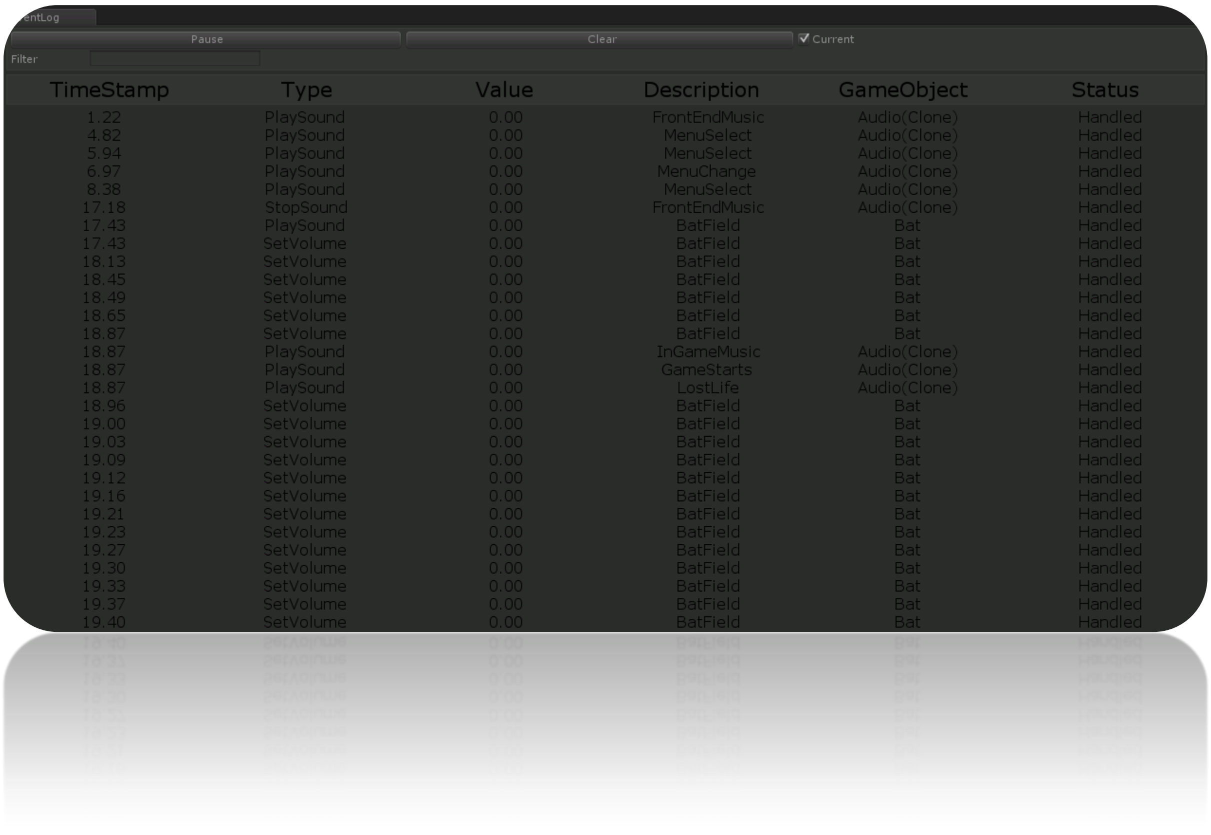


```
Sequence(SequenceComponent)
IsPlaying:False
Instances:0
Vol:1
Vol Rnd:0
Pitch:1
Pitch Rnd:0
CPU:0.000% - ms:0.000

Intro_A(RandomComponent)
IsPlaying:False
Instances:0
Vol:1
Vol Rnd:0
Pitch:1
Pitch Rnd:0
CPU:0.000% - ms:0.000

Intro_A_1(AudioComponent)
IsPlaying:True
Instances:0
Vol:1
Vol Rnd:0
Pitch:1
Pitch Rnd:0
CPU:0.000% - ms:0.000

Intro_A_2(AudioComponent)
IsPlaying:True
Instances:0
Vol:1
Vol Rnd:0
Pitch:1
Pitch Rnd:0
CPU:0.000% - ms:0.000

Intro_B(RandomComponent)

Intro_B_1(AudioComponent)
```

Event Log

Event log view allows monitoring the flow of events and identifying potential problems or missing events.

The event log view allows pausing or clearing the event log list, selecting to follow the current flow of events as well as filter event according to their description name.

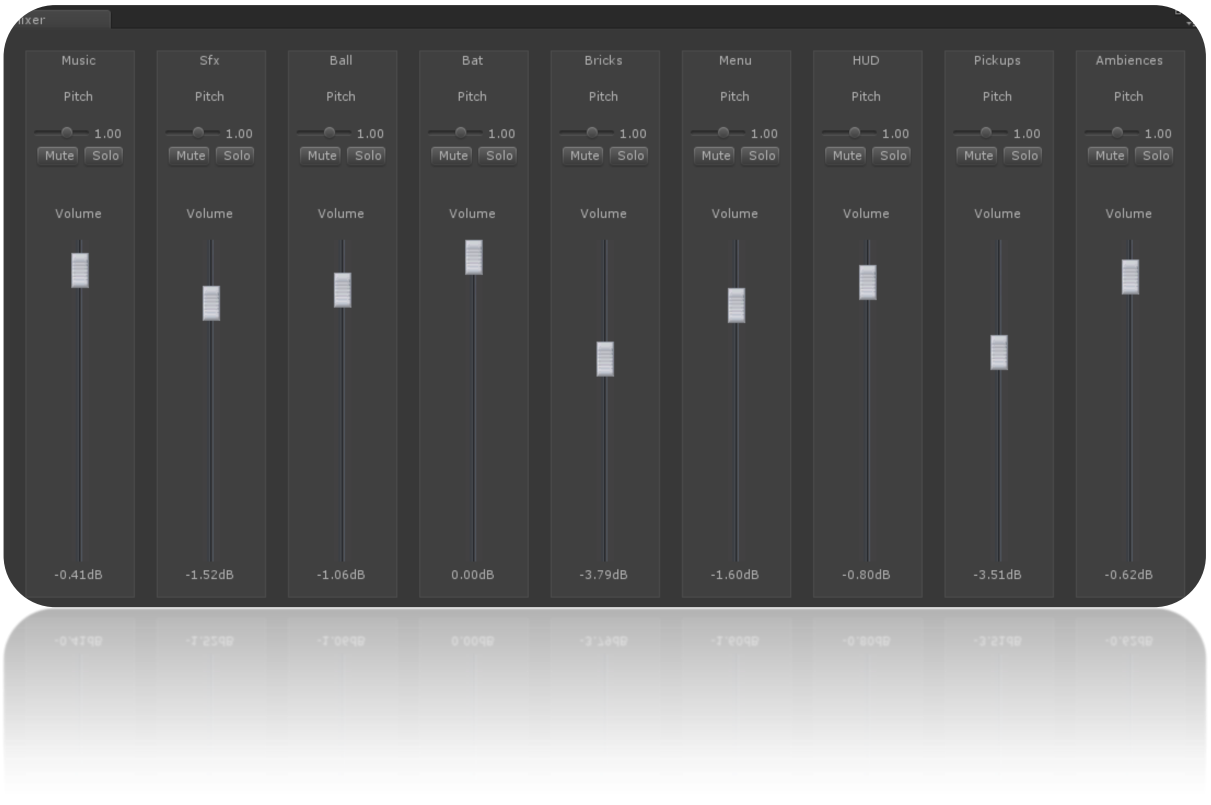


TimeStamp	Type	Value	Description	GameObject	Status
1.22	PlaySound	0.00	FrontEndMusic	Audio(Clone)	Handled
4.82	PlaySound	0.00	MenuSelect	Audio(Clone)	Handled
5.94	PlaySound	0.00	MenuSelect	Audio(Clone)	Handled
6.97	PlaySound	0.00	MenuChange	Audio(Clone)	Handled
8.38	PlaySound	0.00	MenuSelect	Audio(Clone)	Handled
17.18	StopSound	0.00	FrontEndMusic	Audio(Clone)	Handled
17.43	PlaySound	0.00	BatField	Bat	Handled
17.43	SetVolume	0.00	BatField	Bat	Handled
18.13	SetVolume	0.00	BatField	Bat	Handled
18.45	SetVolume	0.00	BatField	Bat	Handled
18.49	SetVolume	0.00	BatField	Bat	Handled
18.65	SetVolume	0.00	BatField	Bat	Handled
18.87	SetVolume	0.00	BatField	Bat	Handled
18.87	PlaySound	0.00	InGameMusic	Audio(Clone)	Handled
18.87	PlaySound	0.00	GameStarts	Audio(Clone)	Handled
18.87	PlaySound	0.00	LostLife	Audio(Clone)	Handled
18.96	SetVolume	0.00	BatField	Bat	Handled
19.00	SetVolume	0.00	BatField	Bat	Handled
19.03	SetVolume	0.00	BatField	Bat	Handled
19.09	SetVolume	0.00	BatField	Bat	Handled
19.12	SetVolume	0.00	BatField	Bat	Handled
19.16	SetVolume	0.00	BatField	Bat	Handled
19.21	SetVolume	0.00	BatField	Bat	Handled
19.23	SetVolume	0.00	BatField	Bat	Handled
19.27	SetVolume	0.00	BatField	Bat	Handled
19.30	SetVolume	0.00	BatField	Bat	Handled
19.33	SetVolume	0.00	BatField	Bat	Handled
19.37	SetVolume	0.00	BatField	Bat	Handled
19.40	SetVolume	0.00	BatField	Bat	Handled

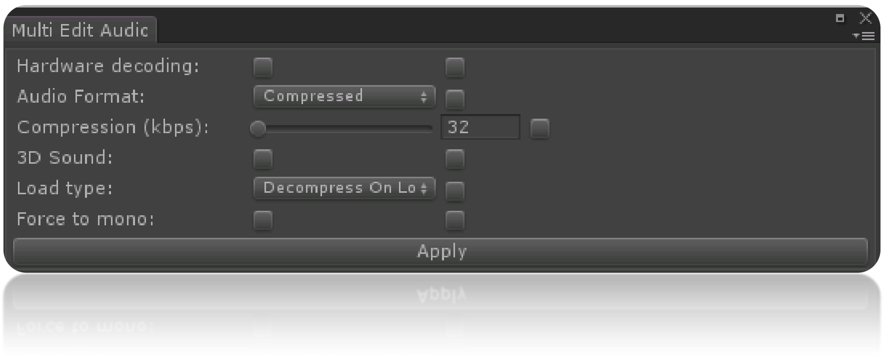
Mixer

The mixer window displays the group components in the hierarchy. From this view it is possible to change the volume or pitch properties of individual components.

The mixers Mute and Solo functionality allows to easily to mix and balance the audio levels in a game.



Multi edit Audio

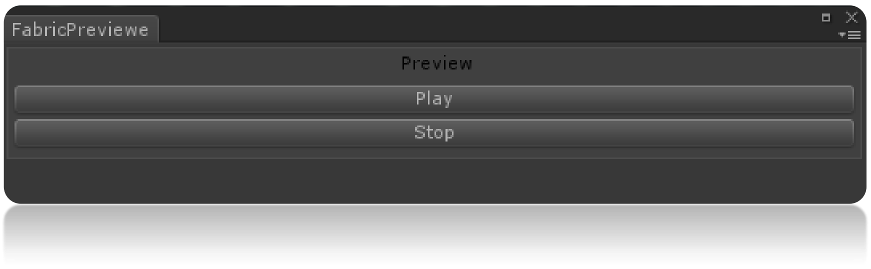


Hardware decoding:
Audio Format: Compressed
Compression (kbps): 32
3D Sound:
Load type: Decompress On Lo
Force to mono:

The multi edit audio window allows selecting any number of audio clips together and editing their properties collective with a touch of a button.

Previewer

If a component has an EventListener attached then the previewer will display the Play and Stop buttons allowing a preview version of the component in the editor.

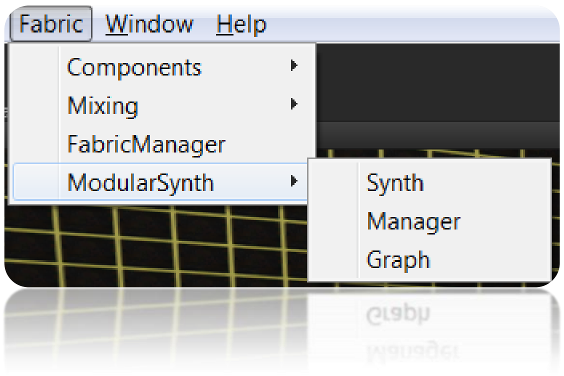


ConvertSamplesToSec

This utility script is used to convert all component properties that have been set as samples in projects created before Unity 4.1 version. Running this script once will be enough to convert the values to seconds.

Modular Synth Extension

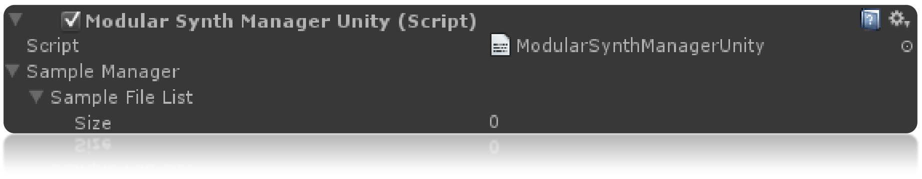
Fabric provides a modular synth which can link audio modules together in order to create complex audio effects or generators.



There are three key modular synth components

Synth manager

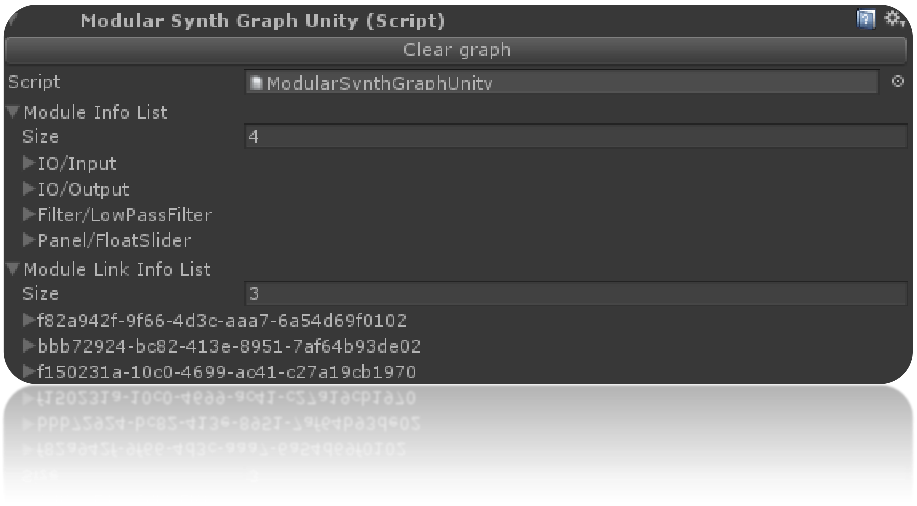
The manager is responsible for managing all the modular synths and also the samples used by the sample player module.



Modular Synth Manager Unity (Script)
Script ModularSynthManagerUnity
Sample Manager
Sample File List
Size 0

Synth Graph

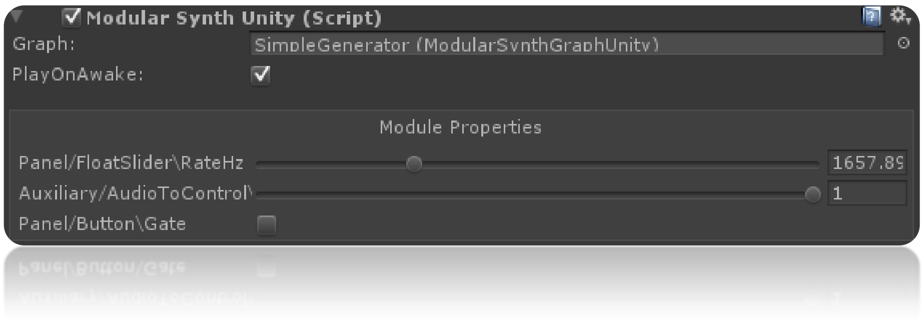
The synth graph stores all the modules and their connections



```
Modular Synth Graph Unity (Script)
Clear graph
Script                ModularSynthGraphUnity
▼Module Info List
  Size                4
  ▶IO/Input
  ▶IO/Output
  ▶Filter/LowPassFilter
  ▶Panel/FloatSlider
▼Module Link Info List
  Size                3
  ▶f82a942f-9f66-4d3c-aaa7-6a54d69f0102
  ▶bbb72924-bc82-413e-8951-7af64b93de02
  ▶f150231a-10c0-4699-ac41-c27a19cb1970
```

Synth

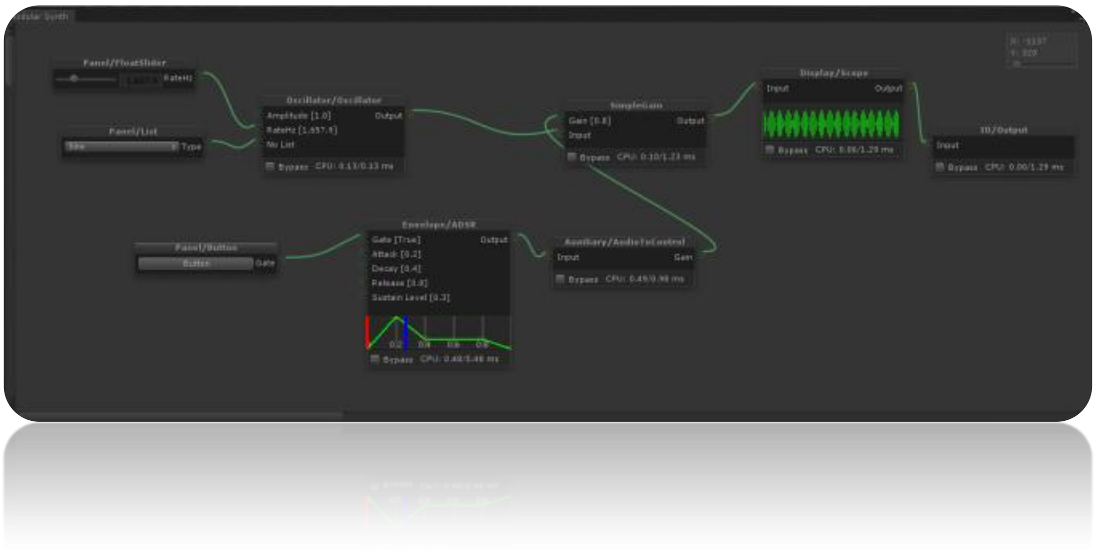
The synth component can accept any modular synth graph for processing. It also displays any panel module properties.



```
✓ Modular Synth Unity (Script)
Graph:                SimpleGenerator (ModularSynthGraphUnity)
PlayOnAwake:          ✓

                      Module Properties
Panel/FloatSlider\RateHz                1657.89
Auxiliary/AudioToControl\               1
Panel/Button\Gate
```

Synth window

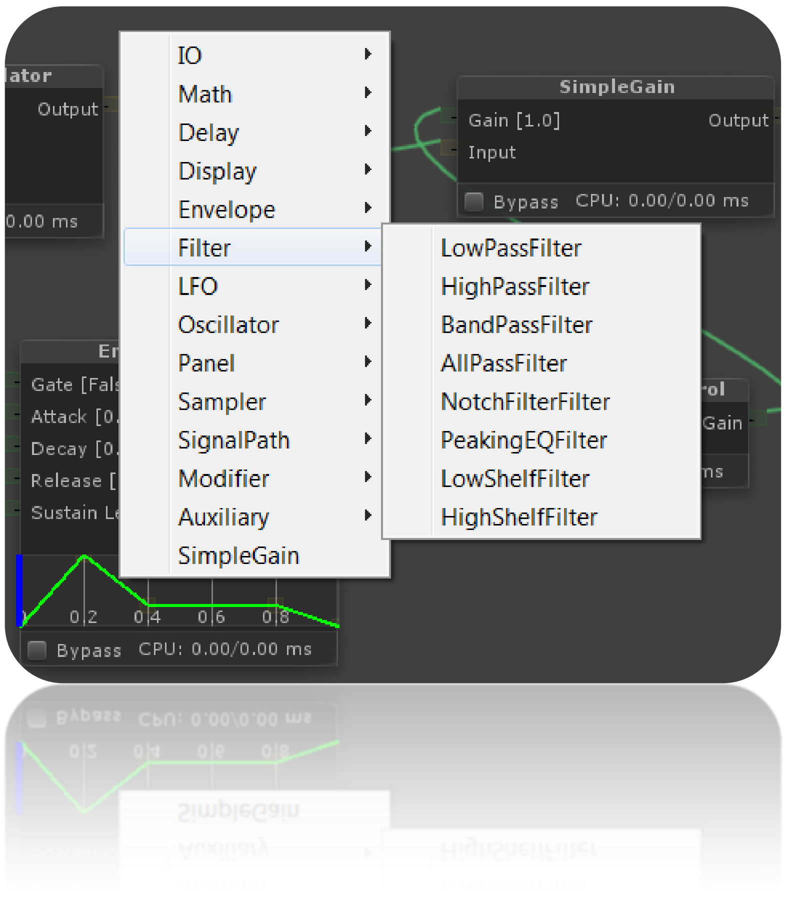
This is the main window that displays all the modules in a graph as well as allowing the addition of new ones, removing or linking them together



Fabric v2.1.6

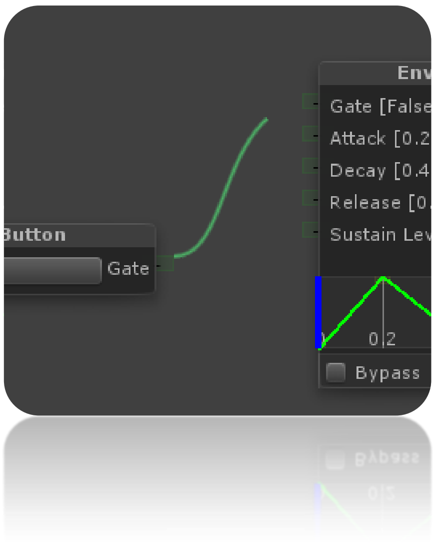
Adding modules to the graph

It is possible to add a new module by right clicking on the module window, this will show a drop down menu with the different modules available.



Linking module pins

Modules can be linked together by selecting an output pin and holding the left mouse button down; this will show a green line that can be dragged around into an input pin of a similar type on another module.

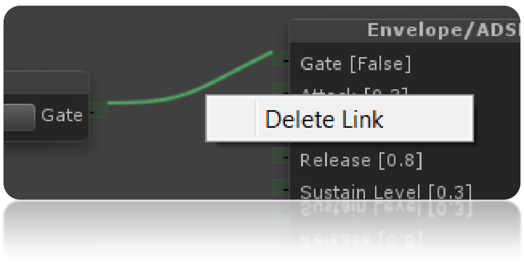


There are two different types of input/output pins:

Pins	Description
Audio input	Receives audio data from audio output pin
Audio output	Outputs audio data from module
Control input	Receives control input data (float, int, bool, string, list)
Control output	Outputs control data (float, int, bool, string, list)

Deleting module pins

Links can be deleted by hovering on top of a link line and clicking on the right mouse button.



Modules

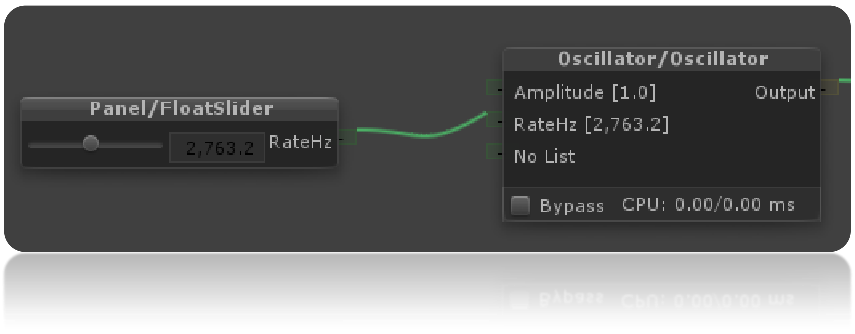
Modular synth provides two different types of module:

Audio processing modules

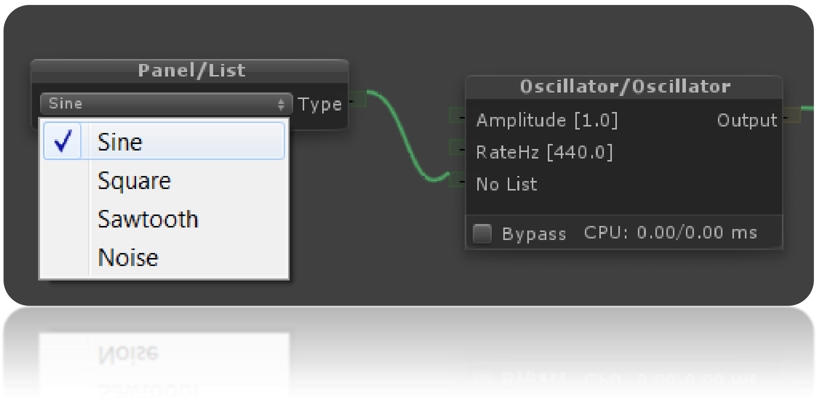
These are the processing modules that usually have input and/or output audio or control pins.

Panel modules:

These types of modules only have an output pin and are responsible for outputting parameter data such as float, integer, etc. They are automatically assigned a name and range when they are connected with a similar type input pin on another module.



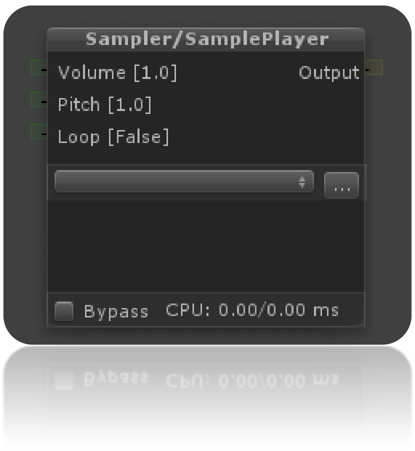
The list panel module by default is an empty list but when it is linked with an input pin it will automatically populate and provide the available options of that input pin. Below is an example of the panel list connected with the Oscillator input type.



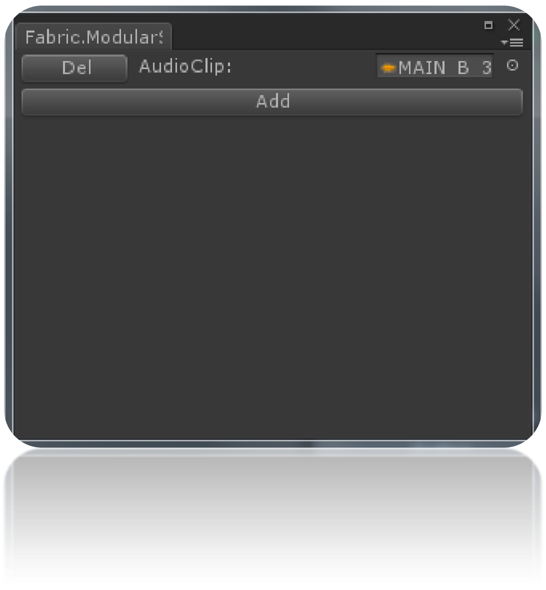
Sample manager

SamplerPlayer module imports audio clips into the modular synth.

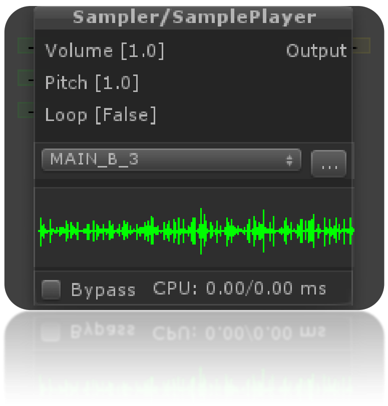
NOTE: The sample player can load wavfiles with markers.



In order for a sample player to see the audio clips they have to be added first to the sample manager. By selecting on the "…" button the manager view window will be displayed allowing addition or removal of audio clips.

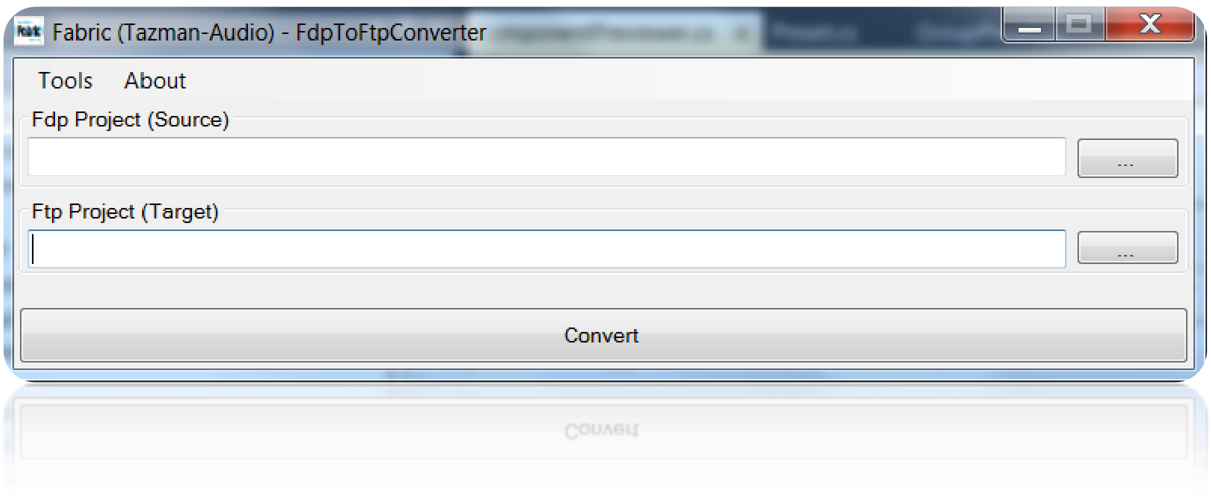


Once added to the SamplerManager, they will be available in all the SamplePlayer dropdown menu options.



FdpToFtpConverter Tool

This is a standalone tool that is capable of reading fdp project files and converting them into Fabric's timeline file format (*.ftp). It can also be run as a command line tool passing the projet name, project directory and build directory.



The following must be taken into consideration in order for the conversion to work.

- The audio source directory field must be empty.
- All wavfiles used by the project MUST be in the same folder as the fdp file.

Add Post-Build Command

This command is responsible for automatically adding a post build command into the project (PC configuration only) so that it is possible to automatically convert the project when it is built.

NOTE: The location of the FdpToFtpConverter executable must be added to the environment path.

Supported features

The tool supports a subset of the event properties:

- Max playbacks
- Volume
- Volume randomisation
- Pitch
- Pitch Randomisation
- Fade In Time
- Fade Out Time
- Priority
- Max Playbacks
- Max Playbacks Behaviour
- 3D Roll off
- 3D Min Distance
- 3D Max Distance
- Sound definitions
- Layers
- Parameters supporting seek speed, velocity and loop
- Effect properties: Volume and Pitch
- Effect curves

NOTE: Sound definition is supported as a Random Component so it provides limited support.

Not supported features

- Categories
- Sound definition folders
- DSP effects
- Music
- Sound Banks

Quick walkthrough

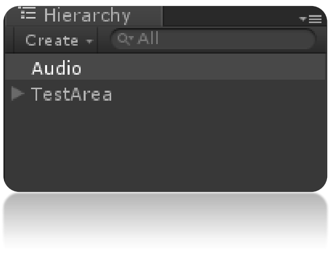
This is a quick walkthrough explaining how to create and trigger a simple audio component in Fabric.

Setting up the scene

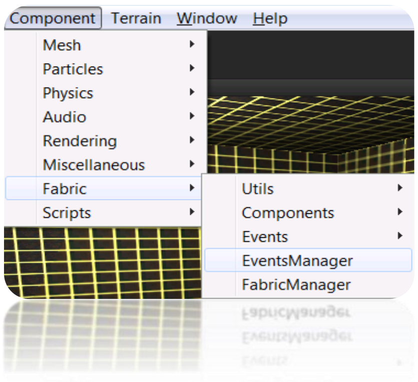
First we are going to start with a new empty Unity project in which we are going to place our audio assets, scene and Fabric hierarchy.

Creating the audio hierarchy

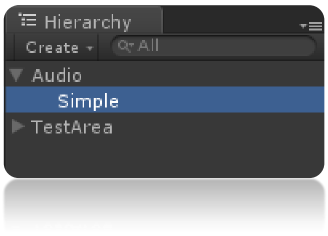
- Create a game object which is going to be the root of Fabric and give it a meaningful name (i.e. Audio, GameAudio etc.)



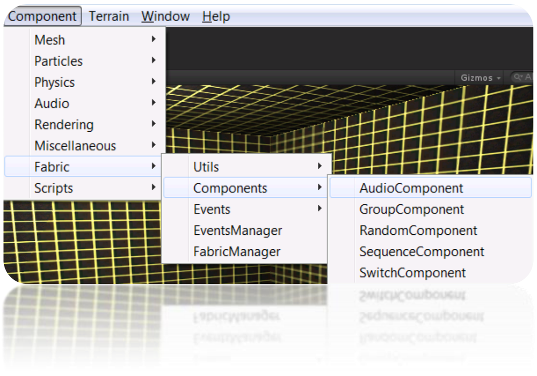
- Highlight the game object and from the Components->Fabric menu add FabricManager and EventManager components.



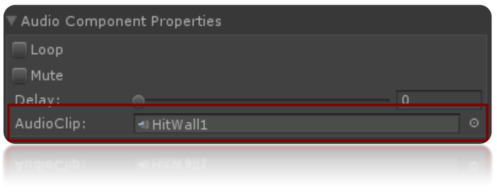
- Create another game object below the root and call it Simple.



- From the Components->Fabric->Components menu selection add the AudioComponent component.



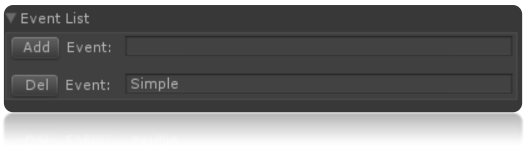
- Next step is to assign the audio clip property of the audio, this could be done either by dragging and dropping the audio file in the AudioClip property or by clicking on it.



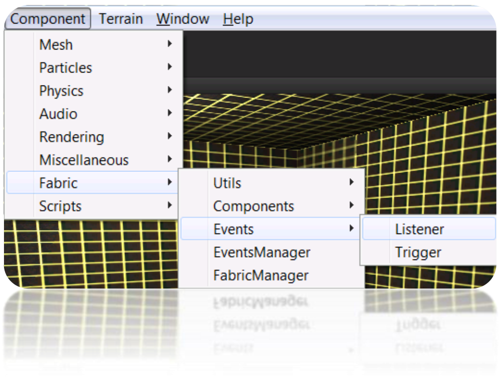
Creating the events

Each component needs to have a listener with a specific event name that it will listen to and respond accordingly.

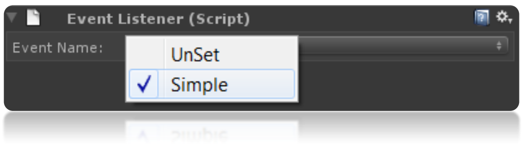
- In the EventManager enter the name of the event and click on the Add button to add it into the list of available events.



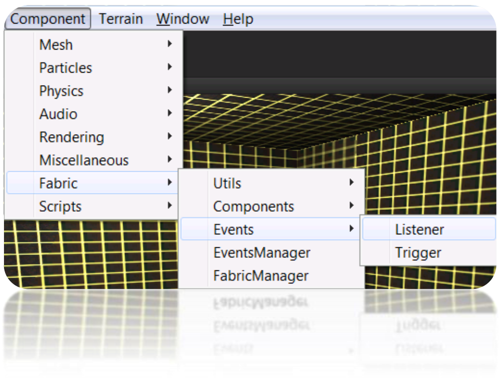
- Highlight the Simple game object and from the Components->Fabric->Events add the EventListener component.



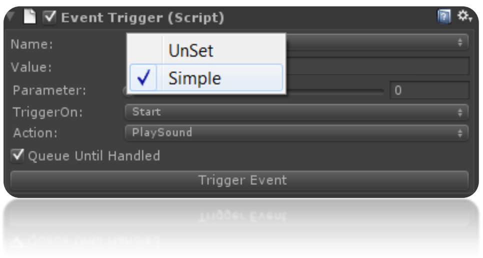
- Next step is to select the event name from the drop down menu.



- Now create another empty game object, outside of the audio hierarchy, and from the Components->Fabric->Events menu selection add an EventTrigger.



- In the event name drop down menu select the same event name as the listener.



- And that’s it, when you run the game it will automatically trigger the audio component and you should be able to hear a sound. It is also possible to trigger the audio again by clicking on the trigger button.