SoundManagerPro User Manual

3.5.5+

Generated by Doxygen 1.8.7

Sun Aug 10 2014 14:41:44

Contents

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 Package antilunchbox

Classes

- class Singleton< T >

  Singleton base class that will cause any inheriting class to create itself when referenced in any way at all.

Enumerations

- enum ClipType { ClipType.AudioClip, ClipType.ClipFromSoundManager, ClipType.ClipFromGroup }

  Specifies how to load `AudioClip`*s.*
- enum AudioSourceAction { AudioSourceAction.None, AudioSourceAction.Play, AudioSourceAction.PlayLoop, AudioSourceAction.PlayCapped, AudioSourceAction.Stop }

  Specifies what an AudioSubscription should do when an event is fired.
- enum AudioSourceStandardEvent { AudioSourceStandardEvent.OnStart, AudioSourceStandardEvent.OnVisible, AudioSourceStandardEvent.OnInvisible, AudioSourceStandardEvent.OnCollisionEnter, AudioSourceStandardEvent.OnCollisionExit, AudioSourceStandardEvent.OnTriggerEnter, AudioSourceStandardEvent.OnTriggerExit, AudioSourceStandardEvent.OnMouseEnter, AudioSourceStandardEvent.OnMouseClick, AudioSourceStandardEvent.OnEnable, AudioSourceStandardEvent.OnDisable, AudioSourceStandardEvent.OnCollisionEnter2D, AudioSourceStandardEvent.OnCollisionExit2D, AudioSourceStandardEvent.OnTriggerEnter2D, AudioSourceStandardEvent.OnTriggerExit2D, AudioSourceStandardEvent.OnParticleCollision }

  Standard events to bind to that are automatically provided by the Unity Engine.

5.1.1 Enumeration Type Documentation

5.1.1.1 enum antilunchbox.AudioSourceAction

Specifies what an AudioSubscription should do when an event is fired.

Enumerator

None

Play

PlayLoop

PlayCapped

Stop

5.1.1.2 enum antilunchbox.AudioSourceStandardEvent

Standard events to bind to that are automatically provided by the Unity Engine.

Enumerator

OnStart

OnVisible

OnInvisible

OnCollisionEnter

OnCollisionExit

OnTriggerEnter

OnTriggerExit

OnMouseEnter

OnMouseClick

OnEnable

OnDisable

OnCollisionEnter2D

OnCollisionExit2D

OnTriggerEnter2D

OnTriggerExit2D

OnParticleCollision

5.1.1.3 enum antilunchbox.ClipType

Specifies how to load `AudioClip`s.

Enumerator

AudioClip

ClipFromSoundManager

ClipFromGroup

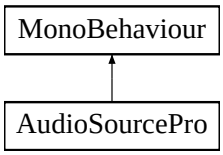
Chapter 6

Class Documentation

6.1 AudioSourcePro Class Reference

SoundManagerPro's version of an AudioSource with additional features.

Inheritance diagram for AudioSourcePro:



Public Member Functions

- void Bind ()

  Bind all events.
- void Unbind ()

  Unbind all events.
- void BindStandardEvent (AudioSourceStandardEvent evt, bool activated)

  Binds or unbinds an AudioSourceStandardEvent.
- void GetOutputData (float[] samples, int channel)

  Returns a block of the currently playing source's output data.
- void GetSpectrumData (float[] samples, int channel, FFTWindow window)

  Returns a block of the currently playing source's spectrum data.
- void Pause ()

  Pauses playing the clip.
- void Play (ulong delay=0)

  Plays the clip with an optional certain delay.
- void PlayDelayed (float delay)

  Plays the clip with a delay specified in seconds.
- void PlayOneShot (AudioClip clip, float volumeScale)

  Plays an `AudioClip`*, and scales the AudioSourcePro volume by volumeScale.*
- void PlayScheduled (double time)

  Plays the clip at a specific time on the absolute time-line that `AudioSettings.dspTime` *reads from.*
- void SetScheduledEndTime (double time)

  Changes the time at which a sound that has already been scheduled to play will end.
- void SetScheduledStartTime (double time)

Changes the time at which a sound that has already been scheduled to play will start.

- void Stop ()

  Stops playing the clip.

Static Public Member Functions

- static void PlayClipAtPoint (AudioClip clip, Vector3 position, float volume=1f)

  Plays an `AudioClip` at a given position in world space.

Public Attributes

- AudioSource audioSource

  The underlying `AudioSource`
- ClipType clipType = ClipType.AudioClip

  Specifies how the `AudioClip` should be loaded.
- string clipName = ""

  The name of the `AudioClip` if you are loading by clip name.
- string groupName = ""

  The name of the SFXGroup if you are loading by group name.
- bool OnStartActivated = false

  Is there a trigger on start?
- bool OnVisibleActivated = false

  Is there a trigger on visible?
- bool OnInvisibleActivated = false

  Is there a trigger on invisible?
- bool OnCollisionEnterActivated = false

  Is there a trigger on collision enter?
- bool OnCollisionExitActivated = false

  Is there a trigger on collision exit?
- bool OnTriggerEnterActivated = false

  Is there a trigger on trigger enter?
- bool OnTriggerExitActivated = false

  Is there a trigger on trigger exit?
- bool OnMouseEnterActivated = false

  Is there a trigger on mouse enter?
- bool OnMouseClickActivated = false

  Is there a trigger on mouse click?
- bool OnEnableActivated = false

  Is there a trigger on enable?
- bool OnDisableActivated = false

  Is there a trigger on disable?
- bool OnCollision2dEnterActivated = false

  Is there a trigger on 2D collision enter?
- bool OnCollision2dExitActivated = false

  Is there a trigger on 2D collision exit?
- bool OnTriggerEnter2dActivated = false

  Is there a trigger on 2D trigger enter?
- bool OnTriggerExit2dActivated = false

  Is there a trigger on 2D trigger exit?
- bool OnParticleCollisionActivated = false

Is there a trigger on particle collision?

- bool ShowEditor3D = false

  Editor variable – IGNORE AND DO NOT MODIFY
- bool ShowEditor2D = false

  Editor variable – IGNORE AND DO NOT MODIFY
- bool ShowEventTriggers = false

  Editor variable – IGNORE AND DO NOT MODIFY
- int numSubscriptions = 0

  The number of AudioSubscription's are on this instance.
- List< AudioSubscription > audioSubscriptions = new List<AudioSubscription>()

  The AudioSubscriptions on this instance.

Properties

- bool componentsAreValid `[get]`

  Gets a value indicating whether this AudioSourcePro components are valid.
- bool audioIsValid `[get]`

  Gets a value indicating whether this AudioSourcePro audio setup is valid.
- bool bypassEffects `[get, set]`

  Bypass effects (Applied from filter components or global listener filters).
- AudioClip clip `[get, set]`

  The default `AudioClip` to play.
- float dopplerLevel `[get, set]`

  Gets or sets the Doppler scale for this `AudioSource`.
- bool ignoreListenerPause `[get, set]`

  Allows `AudioSource` to play even though `AudioListener.pause` is set to true.
- bool ignoreListenerVolume `[get, set]`

  This makes the `AudioSource` not take into account the volume of the `AudioListenter`.
- bool isPlaying `[get]`

  Gets a value indicating whether this AudioSourcePro is playing.
- bool loop `[get, set]`

  Gets or sets a value indicating whether this AudioSourcePro is set to loop.
- float maxDistance `[get, set]`

  (Logarithmic rolloff) MaxDistance is the distance a sound stops attenuating at.
- float minDistance `[get, set]`

  Within the Min distance the AudioSourcePro will cease to grow louder in volume.
- bool mute `[get, set]`

  Un- / Mutes the AudioSource.
- float pan `[get, set]`

  Sets a channels pan position linearly.
- float panLevel `[get, set]`

  Sets how much the 3d engine has an effect on the channel.
- float pitch `[get, set]`

  The pitch of the AudioSourcePro.
- bool playOnAwake `[get, set]`

  If set to true, the AudioSourcePro will automatically start playing on awake.
- int priority `[get, set]`

  Gets or sets the priority.
- AudioRolloffMode rolloffMode `[get, set]`

  Gets or sets the rolloff mode, how the AudioSourcePro attenuates over distance.

- float spread [get, set]

  Gets or sets the spread angle a 3D stereo or multichannel sound in speaker space.
- float time [get, set]

  Gets or sets the time.
- int timeSamples [get, set]

  Gets or sets the time samples.
- AudioVelocityUpdateMode velocityUpdateMode [get, set]

  Gets or sets the velocity update mode.
- float volume [get, set]

  Gets or sets the volume.

6.1.1 Detailed Description

SoundManagerPro's version of an AudioSource with additional features.

6.1.2 Member Function Documentation

6.1.2.1 void AudioSourcePro.Bind ( )

Bind all events.

Fires automatically on activate.

6.1.2.2 void AudioSourcePro.BindStandardEvent (AudioSourceStandardEvent *evt*, bool *activated*)

Binds or unbinds an AudioSourceStandardEvent.

Parameters

evt	The AudioSourceStandardEvent to bind or unbind.
activated	Whether to bind or unbind.

6.1.2.3 void AudioSourcePro.GetOutputData (float[] *samples*, int *channel*)

Returns a block of the currently playing source's output data.

Parameters

samples	Samples.
channel	Channel.

6.1.2.4 void AudioSourcePro.GetSpectrumData (float[] *samples*, int *channel*, FFTWindow *window*)

Returns a block of the currently playing source's spectrum data.

Parameters

samples	Samples.
channel	Channel.
window	Window.

6.1.2.5 void AudioSourcePro.Pause ( )

Pauses playing the clip.

6.1.2.6 void AudioSourcePro.Play (ulong *delay* = `0`)

Plays the clip with an optional certain delay.

Parameters

delay	Delay.

6.1.2.7 static void AudioSourcePro.PlayClipAtPoint (AudioClip *clip*, Vector3 *position*, float *volume* = `1f`) `[static]`

Plays an `AudioClip` at a given position in world space.

Parameters

clip	Clip.
position	Position.
volume	Volume.

6.1.2.8 void AudioSourcePro.PlayDelayed (float *delay*)

Plays the clip with a delay specified in seconds.

Users are advised to use this function instead of the old Play(delay) function that took a delay specified in samples relative to a reference rate of 44.1 kHz as an argument.

Parameters

delay	Delay.

6.1.2.9 void AudioSourcePro.PlayOneShot (AudioClip *clip*, float *volumeScale*)

Plays an `AudioClip`, and scales the AudioSourcePro volume by volumeScale.

Parameters

clip	Clip.
volumeScale	Volume scale.

6.1.2.10 void AudioSourcePro.PlayScheduled (double *time*)

Plays the clip at a specific time on the absolute time-line that `AudioSettings.dspTime` reads from.

Parameters

time	Time to play.

6.1.2.11 void AudioSourcePro.SetScheduledEndTime (double *time*)

Changes the time at which a sound that has already been scheduled to play will end.

Notice that depending on the timing not all rescheduling requests can be fulfilled.

Parameters

time	Time to play.

6.1.2.12 void AudioSourcePro.SetScheduledStartTime (double *time*)

Changes the time at which a sound that has already been scheduled to play will start.

Parameters

time	Time to play.

6.1.2.13 void AudioSourcePro.Stop ( )

Stops playing the clip.

6.1.2.14 void AudioSourcePro.Unbind ( )

Unbind all events.

Fires automatically on deactivate.

6.1.3 Member Data Documentation

6.1.3.1 AudioSource AudioSourcePro.audioSource

The underlying `AudioSource`

6.1.3.2 List<AudioSubscription> AudioSourcePro.audioSubscriptions = new List<AudioSubscription>()

The AudioSubscriptions on this instance.

The editor modifies this value. It is not recommended to modify this unless you know what you're doing.

6.1.3.3 string AudioSourcePro.clipName = ""

The name of the `AudioClip` if you are loading by clip name.

The clip should live on the SoundManager.

6.1.3.4 ClipType AudioSourcePro.clipType = ClipType.AudioClip

Specifies how the `AudioClip` should be loaded.

6.1.3.5 string AudioSourcePro.groupName = ""

The name of the SFXGroup if you are loading by group name.

The group should live on the SoundManager.

6.1.3.6 int AudioSourcePro.numSubscriptions = 0

The number of AudioSubscription's are on this instance.

The editor modifies this value. It is not recommended to modify this unless you know what you're doing.

6.1.3.7 bool AudioSourcePro.OnCollision2dEnterActivated = false

Is there a trigger on 2D collision enter?

6.1.3.8 bool AudioSourcePro.OnCollision2dExitActivated = false

Is there a trigger on 2D collision exit?

6.1.3.9 bool AudioSourcePro.OnCollisionEnterActivated = false

Is there a trigger on collision enter?

6.1.3.10 bool AudioSourcePro.OnCollisionExitActivated = false

Is there a trigger on collision exit?

6.1.3.11 bool AudioSourcePro.OnDisableActivated = false

Is there a trigger on disable?

6.1.3.12 bool AudioSourcePro.OnEnableActivated = false

Is there a trigger on enable?

6.1.3.13 bool AudioSourcePro.OnInvisibleActivated = false

Is there a trigger on invisible?

6.1.3.14 bool AudioSourcePro.OnMouseClickActivated = false

Is there a trigger on mouse click?

6.1.3.15 bool AudioSourcePro.OnMouseEnterActivated = false

Is there a trigger on mouse enter?

6.1.3.16 bool AudioSourcePro.OnParticleCollisionActivated = false

Is there a trigger on particle collision?

6.1.3.17 bool AudioSourcePro.OnStartActivated = false

Is there a trigger on start?

6.1.3.18 bool AudioSourcePro.OnTriggerEnter2dActivated = false

Is there a trigger on 2D trigger enter?

6.1.3.19 bool AudioSourcePro.OnTriggerEnterActivated = false

Is there a trigger on trigger enter?

6.1.3.20 bool AudioSourcePro.OnTriggerExit2dActivated = false

Is there a trigger on 2D trigger exit?

6.1.3.21 bool AudioSourcePro.OnTriggerExitActivated = false

Is there a trigger on trigger exit?

6.1.3.22 bool AudioSourcePro.OnVisibleActivated = false

Is there a trigger on visible?

6.1.3.23 bool AudioSourcePro.ShowEditor2D = false

Editor variable – IGNORE AND DO NOT MODIFY

6.1.3.24 bool AudioSourcePro.ShowEditor3D = false

Editor variable – IGNORE AND DO NOT MODIFY

6.1.3.25 bool AudioSourcePro.ShowEventTriggers = false

Editor variable – IGNORE AND DO NOT MODIFY

6.1.4 Property Documentation

6.1.4.1 bool AudioSourcePro.audioIsValid [get]

Gets a value indicating whether this AudioSourcePro audio setup is valid.

`true` if audio setup is valid; otherwise, `false`.

6.1.4.2 bool AudioSourcePro.bypassEffects [get], [set]

Bypass effects (Applied from filter components or global listener filters).

`true` if bypass effects; otherwise, `false`.

6.1.4.3 AudioClip AudioSourcePro.clip [get], [set]

The default `AudioClip` to play.

The clip.

6.1.4.4 bool AudioSourcePro.componentsAreValid [get]

Gets a value indicating whether this AudioSourcePro components are valid.

`true` if components are valid; otherwise, `false`.

6.1.4.5 float AudioSourcePro.dopplerLevel [get], [set]

Gets or sets the Doppler scale for this `AudioSource`.

The doppler level.

6.1.4.6 bool AudioSourcePro.ignoreListenerPause [get], [set]

Allows `AudioSource` to play even though `AudioListener.pause` is set to true.

This is useful for the menu element sounds or background music in pause menus.

`true` ignoring listener pause; otherwise, `false`. This property can only be set via the script and is not serialized.

6.1.4.7 bool AudioSourcePro.ignoreListenerVolume [get], [set]

This makes the `AudioSource` not take into account the volume of the `AudioListenter`.

`true` if ignore listener volume; otherwise, `false`.

6.1.4.8 bool AudioSourcePro.isPlaying [get]

Gets a value indicating whether this AudioSourcePro is playing.

`true` if is playing; otherwise, `false`.

6.1.4.9 bool AudioSourcePro.loop [get], [set]

Gets or sets a value indicating whether this AudioSourcePro is set to loop.

`true` if looping; otherwise, `false`.

6.1.4.10 float AudioSourcePro.maxDistance [get], [set]

(Logarithmic rolloff) MaxDistance is the distance a sound stops attenuating at.

The max distance.

6.1.4.11 float AudioSourcePro.minDistance [get], [set]

Within the Min distance the AudioSourcePro will cease to grow louder in volume.

The minimum distance.

6.1.4.12 bool AudioSourcePro.mute [get], [set]

Un- / Mutes the AudioSource.

Mute sets the volume=0, Un-Mute restore the original volume.

`true` if mute; otherwise, `false`.

6.1.4.13 float AudioSourcePro.pan [get], [set]

Sets a channels pan position linearly.

Only works for 2D clips.

The pan.

6.1.4.14 float AudioSourcePro.panLevel [get], [set]

Sets how much the 3d engine has an effect on the channel.

The pan level.

6.1.4.15 float AudioSourcePro.pitch [get], [set]

The pitch of the AudioSourcePro.

The pitch.

6.1.4.16 bool AudioSourcePro.playOnAwake [get], [set]

If set to true, the AudioSourcePro will automatically start playing on awake.

`true` if play on awake; otherwise, `false`.

6.1.4.17 int AudioSourcePro.priority [get], [set]

Gets or sets the priority.

The priority.

6.1.4.18 AudioRolloffMode AudioSourcePro.rolloffMode [get], [set]

Gets or sets the rolloff mode, how the AudioSourcePro attenuates over distance.

The rolloff mode.

6.1.4.19 float AudioSourcePro.spread [get], [set]

Gets or sets the spread angle a 3D stereo or multichannel sound in speaker space.

The spread.

6.1.4.20 float AudioSourcePro.time [get], [set]

Gets or sets the time.

Playback position in seconds.

6.1.4.21 int AudioSourcePro.timeSamples [get], [set]

Gets or sets the time samples.

Playback position in PCM samples.

6.1.4.22 AudioVelocityUpdateMode AudioSourcePro.velocityUpdateMode [get],[set]

Gets or sets the velocity update mode.

Whether the Audio Source should be updated in the fixed or dynamic update.

6.1.4.23 float AudioSourcePro.volume [get],[set]

Gets or sets the volume.

The volume of the audio source (0.0 to 1.0).

The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/AudioSourcePro.cs

6.2 AudioSourceTools Class Reference

Extending SoundManager SFX functions to regular AudioSources.

Static Public Member Functions

- static void PlaySFX (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name, bool loop, float delay, float volume, float pitch)
- static void PlaySFX (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name, bool loop, float delay, float volume)
- static void PlaySFX (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name, bool loop, float delay)
- static void PlaySFX (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name, bool loop)
- static void PlaySFX (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name)
- static void PlaySFX (ref AudioSource theAudioSource, AudioClip clip, bool loop, float delay, float volume, float pitch)
- static void PlaySFX (ref AudioSource theAudioSource, AudioClip clip, bool loop, float delay, float volume)
- static void PlaySFX (ref AudioSource theAudioSource, AudioClip clip, bool loop, float delay)
- static void PlaySFX (ref AudioSource theAudioSource, AudioClip clip, bool loop)
- static void PlaySFX (ref AudioSource theAudioSource, AudioClip clip)
- static void StopSFX (ref AudioSource theAudioSource)
- static void PlaySFXLoop (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name, bool tillDestroy, float volume, float pitch, float maxDuration)
- static void PlaySFXLoop (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name, bool tillDestroy, float volume, float pitch)
- static void PlaySFXLoop (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name, bool tillDestroy, float volume)
- static void PlaySFXLoop (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name, bool tillDestroy)
- static void PlaySFXLoop (ref AudioSource theAudioSource, bool fromGroup, string clipOrGroup_Name)

6.2.1 Detailed Description

Extending SoundManager SFX functions to regular AudioSources.

6.2.2 Member Function Documentation

6.2.2.1 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name,* bool *loop,* float *delay,* float *volume,* float *pitch*) [static]

6.2.2.2 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name,* bool *loop,* float *delay,* float *volume*) [static]

6.2.2.3 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name,* bool *loop,* float *delay*) [static]

6.2.2.4 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name,* bool *loop*) [static]

6.2.2.5 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name*) [static]

6.2.2.6 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* AudioClip *clip,* bool *loop,* float *delay,* float *volume,* float *pitch*) [static]

6.2.2.7 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* AudioClip *clip,* bool *loop,* float *delay,* float *volume*) [static]

6.2.2.8 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* AudioClip *clip,* bool *loop,* float *delay*) [static]

6.2.2.9 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* AudioClip *clip,* bool *loop*) [static]

6.2.2.10 static void AudioSourceTools.PlaySFX (ref AudioSource *theAudioSource,* AudioClip *clip*) [static]

6.2.2.11 static void AudioSourceTools.PlaySFXLoop (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name,* bool *tillDestroy,* float *volume,* float *pitch,* float *maxDuration*) [static]

6.2.2.12 static void AudioSourceTools.PlaySFXLoop (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name,* bool *tillDestroy,* float *volume,* float *pitch*) [static]

6.2.2.13 static void AudioSourceTools.PlaySFXLoop (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name,* bool *tillDestroy,* float *volume*) [static]

6.2.2.14 static void AudioSourceTools.PlaySFXLoop (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name,* bool *tillDestroy*) [static]

6.2.2.15 static void AudioSourceTools.PlaySFXLoop (ref AudioSource *theAudioSource,* bool *fromGroup,* string *clipOrGroup_Name*) [static]

6.2.2.16 static void AudioSourceTools.StopSFX (ref AudioSource *theAudioSource*) [static]

The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Extensions/AudioSourceTools.cs

6.3 AudioSubscription Class Reference

Class that contains all the data needed to bind an AudioSourceAction to an event.

Public Member Functions

- void Bind (AudioSourcePro sourcePro)

  Binds event and action on the specified AudioSourcePro.
- void Unbind ()

  Unbind this instance.

Public Attributes

- AudioSourcePro owner

  The owner.
- bool isStandardEvent = true

  Whether this is a standard event binding.
- AudioSourceStandardEvent standardEvent

  The standard event to bind to.
- Component sourceComponent

  The source component if it's a custom event binding.
- string methodName = ""

  The name of the method if it's a custom event binding.
- AudioSourceAction actionType = AudioSourceAction.None

  The action to take when the event is triggered.
- string cappedName

  If the action is to play a capped SFX, then this is the cap name.
- bool filterLayers

  Whether triggers will be filtered by layer.
- bool filterTags

  Whether triggers will be filtered by tags.
- bool filterNames

  Whether triggers will be filtered by gameObject name.
- int tagMask

  Editor variable – IGNORE AND DO NOT MODIFY
- int nameMask

  Editor variable – IGNORE AND DO NOT MODIFY
- string nameToAdd = ""

  Editor variable – IGNORE AND DO NOT MODIFY
- List< string > allNames = new List<string>()

  Editor variable – IGNORE AND DO NOT MODIFY
- int layerMask

  If filterLayers is `true`, only trigger for this layer mask.
- List< string > tags = new List<string>() { "Default" }

  If filterTags is `true`, only trigger for these tags.
- List< string > names = new List<string>()

  If filterNames is `true`, only trigger for these gameObject names.

Properties

- bool componentIsValid `[get]`

  Gets a value indicating whether this AudioSubscription component is valid.
- bool standardEventIsValid `[get]`

  Gets a value indicating whether this AudioSubscription standard event is valid.

6.3.1 Detailed Description

Class that contains all the data needed to bind an AudioSourceAction to an event.

AudioSubscriptions are bound at run-time to keep serialization intact.

6.3.2 Member Function Documentation

6.3.2.1 void AudioSubscription.Bind (AudioSourcePro *sourcePro*)

Binds event and action on the specified AudioSourcePro.

Parameters

sourcePro	The AudioSourcePro.

6.3.2.2 void AudioSubscription.Unbind ( )

Unbind this instance.

6.3.3 Member Data Documentation

6.3.3.1 AudioSourceAction AudioSubscription.actionType = AudioSourceAction.None

The action to take when the event is triggered.

6.3.3.2 List<string> AudioSubscription.allNames = new List<string>()

Editor variable – IGNORE AND DO NOT MODIFY

6.3.3.3 string AudioSubscription.cappedName

If the action is to play a capped SFX, then this is the cap name.

6.3.3.4 bool AudioSubscription.filterLayers

Whether triggers will be filtered by layer.

6.3.3.5 bool AudioSubscription.filterNames

Whether triggers will be filtered by gameObject name.

6.3.3.6 bool AudioSubscription.filterTags

Whether triggers will be filtered by tags.

6.3.3.7 bool AudioSubscription.isStandardEvent = true

Whether this is a standard event binding.

6.3.3.8 int AudioSubscription.layerMask

If filterLayers is `true`, only trigger for this layer mask.

6.3.3.9 string AudioSubscription.methodName = ""

The name of the method if it's a custom event binding.

6.3.3.10 int AudioSubscription.nameMask

Editor variable – IGNORE AND DO NOT MODIFY

6.3.3.11 List<string> AudioSubscription.names = new List<string>()

If filterNames is `true`, only trigger for these gameObject names.

6.3.3.12 string AudioSubscription.nameToAdd = ""

Editor variable – IGNORE AND DO NOT MODIFY

6.3.3.13 AudioSourcePro AudioSubscription.owner

The owner.

6.3.3.14 Component AudioSubscription.sourceComponent

The source component if it's a custom event binding.

6.3.3.15 AudioSourceStandardEvent AudioSubscription.standardEvent

The standard event to bind to.
Only used if isStandardEvent is `true`.

6.3.3.16 int AudioSubscription.tagMask

Editor variable – IGNORE AND DO NOT MODIFY

6.3.3.17 List<string> AudioSubscription.tags = new List<string>() { "Default" }

If filterTags is `true`, only trigger for these tags.

6.3.4 Property Documentation

6.3.4.1 bool AudioSubscription.componentIsValid [get]

Gets a value indicating whether this AudioSubscription component is valid.

`true` if component is valid; otherwise, `false`.

6.3.4.2 bool AudioSubscription.standardEventIsValid [get]

Gets a value indicating whether this AudioSubscription standard event is valid.

`true` if standard event is valid; otherwise, `false`.

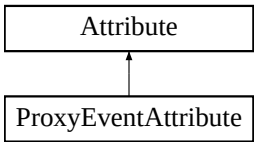
The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/AudioSubscription.cs

6.4 ProxyEventAttribute Class Reference

Proxy event attribute for custom event binding.

Inheritance diagram for ProxyEventAttribute:



6.4.1 Detailed Description

Proxy event attribute for custom event binding.

The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/ProxyEventAttribute.cs

6.5 SFXGroup Class Reference

Used to group SFX together with certain attributes to share.

Public Member Functions

- SFXGroup (string name, int capAmount, params AudioClip[] audioclips)
  Initializes a new instance of the SFXGroup class.
- SFXGroup (string name, params AudioClip[] audioclips)
  Initializes a new instance of the SFXGroup class.

Public Attributes

- string groupName
  The name of the group.
- int specificCapAmount
  The specific cap amount.
- List< AudioClip > clips = new List<AudioClip>()
  The clips in the group.
- bool independentVolume
  NOT IMPLEMENTED YET.
- bool independentPitch

NOT IMPLEMENTED YET.

- float volume

  NOT IMPLEMENTED YET.

- float pitch

  NOT IMPLEMENTED YET.

6.5.1 Detailed Description

Used to group SFX together with certain attributes to share.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 SFXGroup.SFXGroup (string *name*, int *capAmount*, params AudioClip[] *audioclips*)

Initializes a new instance of the SFXGroup class.

Parameters

name	Name of the group.
capAmount	The specificCapAmount.
audioclips	Audioclips in the group.

6.5.2.2 SFXGroup.SFXGroup (string *name*, params AudioClip[] *audioclips*)

Initializes a new instance of the SFXGroup class.

Parameters

name	Name of the group.
audioclips	Audioclips in the group.

6.5.3 Member Data Documentation

6.5.3.1 List<AudioClip> SFXGroup.clips = new List<AudioClip>()

The clips in the group.

6.5.3.2 string SFXGroup.groupName

The name of the group.

6.5.3.3 bool SFXGroup.independentPitch

NOT IMPLEMENTED YET.

6.5.3.4 bool SFXGroup.independentVolume

NOT IMPLEMENTED YET.

6.5.3.5 float SFXGroup.pitch

NOT IMPLEMENTED YET.

6.5.3.6 int SFXGroup.specificCapAmount

The specific cap amount.

If set to -1, it will use the default global cap amount. If set to 0, the group will not use a specific cap amount at all. This amount will only be respected when using SoundManager.PlayCappedSFX

6.5.3.7 float SFXGroup.volume

NOT IMPLEMENTED YET.

The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/SFXGroup.cs

6.6 SFXPoolInfo Class Reference

Contains information on SFX Pools.

Public Member Functions

- SFXPoolInfo (int index, int minAmount, List< float > times, List< GameObject > pool, float baseVol=1f, float volVar=0f, float pitchVar=0f)

  Initializes a new instance of the SFXPoolInfo class.

Public Attributes

- int currentIndexInPool = 0

  The current index in pool.
- int prepoolAmount = 0

  The prepool amount.
- float baseVolume = 1f

  The base volume.
- float volumeVariation = 0f

  The volume variation.
- float pitchVariation = 0f

  The pitch variation.
- List< float > timesOfDeath = new List<float>()

  The times of death for SFX objects over the prepoolAmount.
- List< GameObject > ownedAudioClipPool = new List<GameObject>()

  The owned audio clip pool.

6.6.1 Detailed Description

Contains information on SFX Pools.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 SFXPoolInfo.SFXPoolInfo (int *index*, int *minAmount*, List< float > *times*, List< GameObject > *pool*, float *baseVol* = `1f`, float *volVar* = `0f`, float *pitchVar* = `0f`)

Initializes a new instance of the SFXPoolInfo class.

Parameters

index	Index.
minAmount	Prepool amount.
times	Times of death.
pool	Pool of SFX objects.
baseVol	Base volume.
volVar	Volume variation
pitchVar	Pitch variation.

6.6.3 Member Data Documentation

6.6.3.1 float SFXPoolInfo.baseVolume = 1f

The base volume.

6.6.3.2 int SFXPoolInfo.currentIndexInPool = 0

The current index in pool.

6.6.3.3 List<GameObject> SFXPoolInfo.ownedAudioClipPool = new List<GameObject>()

The owned audio clip pool.

6.6.3.4 float SFXPoolInfo.pitchVariation = 0f

The pitch variation.

6.6.3.5 int SFXPoolInfo.prepoolAmount = 0

The prepool amount.

6.6.3.6 List<float> SFXPoolInfo.timesOfDeath = new List<float>()

The times of death for SFX objects over the prepoolAmount.

6.6.3.7 float SFXPoolInfo.volumeVariation = 0f

The volume variation.

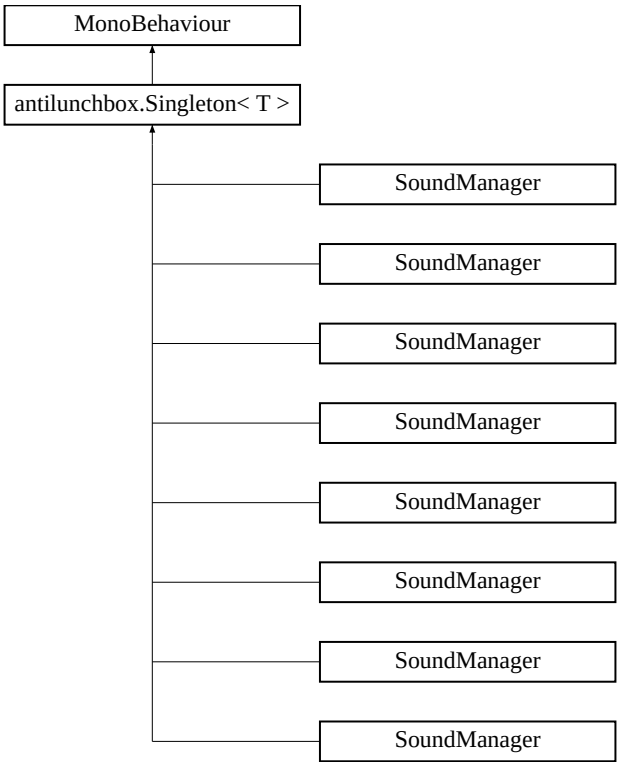
The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/SFXPoolInfo.cs

6.7 antilunchbox.Singleton< T > Class Template Reference

Singleton base class that will cause any inheriting class to create itself when referenced in any way at all.

Inheritance diagram for antilunchbox.Singleton< T >:



Properties

- static Singleton< T > mInstance `[get, set]`

  Gets or sets the instance.

6.7.1 Detailed Description

Singleton base class that will cause any inheriting class to create itself when referenced in any way at all.

Type Constraints

***T* : *Singleton*<*T*>**

6.7.2 Property Documentation

6.7.2.1 Singleton<T> antilunchbox.Singleton< T >.mInstance `[static], [get], [set], [protected]`

Gets or sets the instance.

The instance.

The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/Singleton.cs

6.8 SoundConnection Class Reference

Contains information on sound connections, meant for background music.

Public Member Functions

- SoundConnection (string lvl, params AudioClip[] audioList)

  Initializes a new instance of the SoundConnection class.
- SoundConnection (string lvl, SoundManager.PlayMethod method, params AudioClip[] audioList)

  Initializes a new instance of the SoundConnection class.
- SoundConnection (string lvl, SoundManager.PlayMethod method, float delayPlay, params AudioClip[] audio↩ List)

  Initializes a new instance of the SoundConnection class.
- SoundConnection (string lvl, SoundManager.PlayMethod method, float minDelayPlay, float maxDelayPlay, params AudioClip[] audioList)

  Initializes a new instance of the SoundConnection class.
- void SetToCustom ()

  Sets this SoundConnection to a Custom SoundConnection.

Public Attributes

- string level

  Name of the scene the SoundConnection is attached to, or the name of the custom SoundConnection.
- bool isCustomLevel

  Whether it is a custom SoundConnection.
- List< AudioClip > soundsToPlay

  The clips in the SoundConnection.
- List< float > baseVolumes = new List<float>()

  The base volumes for each clip in the SoundConnection.
- SoundManager.PlayMethod playMethod

  The play method.
- float minDelay

  The minimum delay for any play methods that have delay in range.
- float maxDelay

  The maximum delay for any play methods that have delay in range.
- float delay

  The delay for any play methods with an exact delay.

6.8.1 Detailed Description

Contains information on sound connections, meant for background music.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 SoundConnection.SoundConnection (string *lvl*, params AudioClip[] *audioList*)

Initializes a new instance of the SoundConnection class.

Ties the level name to a list of AudioClips. It defaults to continuous play through with no delay between clips.

Parameters

lvl	Level name.

audioList	Audio list.

6.8.2.2 SoundConnection.SoundConnection (string *lvl,* SoundManager.PlayMethod *method,* params AudioClip[] *audioList*)

Initializes a new instance of the SoundConnection class.

Parameters

lvl	Level name.
method	PlayMethod.
audioList	Audio list.

6.8.2.3 SoundConnection.SoundConnection (string *lvl,* SoundManager.PlayMethod *method,* float *delayPlay,* params AudioClip[] *audioList*)

Initializes a new instance of the SoundConnection class.

Sets the exact delay for appropriate play methods.

Parameters

lvl	Level name.
method	PlayMethod.
delayPlay	Delay.
audioList	Audio list.

6.8.2.4 SoundConnection.SoundConnection (string *lvl,* SoundManager.PlayMethod *method,* float *minDelayPlay,* float *maxDelayPlay,* params AudioClip[] *audioList*)

Initializes a new instance of the SoundConnection class.

Sets the min and max delay for appropriate play methods.

Parameters

lvl	Level name.
method	PlayMethod.
minDelayPlay	Minimum delay.
maxDelayPlay	Maximum delay.
audioList	Audio list.

6.8.3 Member Function Documentation

6.8.3.1 void SoundConnection.SetToCustom ( )

Sets this SoundConnection to a Custom SoundConnection.

Is not tied to a scene. Must be called with SoundManager.CustomEvent Be careful not to use a level name or SoundManager will get confused. Call this after initializing a SoundConnection.

6.8.4 Member Data Documentation

6.8.4.1 List<float> SoundConnection.baseVolumes = new List<float>()

The base volumes for each clip in the SoundConnection.

6.8.4.2 float SoundConnection.delay

The delay for any play methods with an exact delay.

6.8.4.3 bool SoundConnection.isCustomLevel

Whether it is a custom SoundConnection.

6.8.4.4 string SoundConnection.level

Name of the scene the SoundConnection is attached to, or the name of the custom SoundConnection.

6.8.4.5 float SoundConnection.maxDelay

The maximum delay for any play methods that have delay in range.

6.8.4.6 float SoundConnection.minDelay

The minimum delay for any play methods that have delay in range.

6.8.4.7 SoundManager.PlayMethod SoundConnection.playMethod

The play method.

6.8.4.8 List<AudioClip> SoundConnection.soundsToPlay

The clips in the SoundConnection.

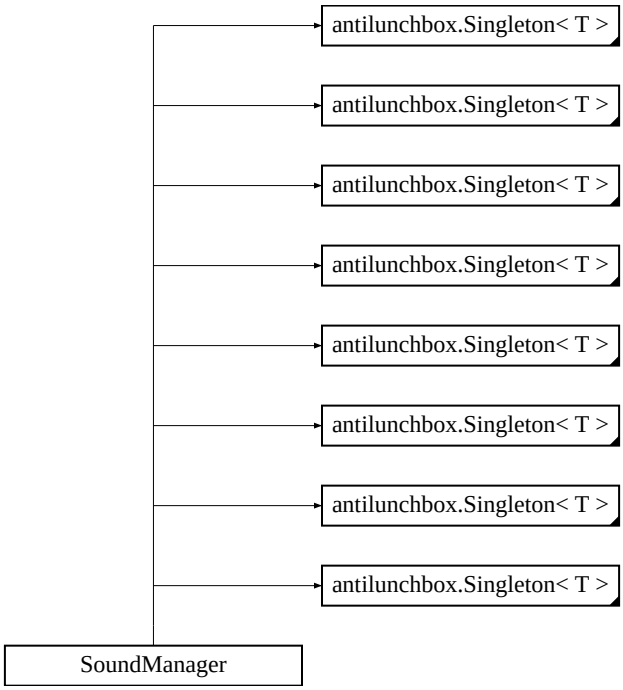
The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/SoundConnection.cs

6.9 SoundManager Class Reference

SoundManager is where most functions will be called from.

Inheritance diagram for SoundManager:



Public Types

- enum PlayMethod { PlayMethod.ContinuousPlayThrough, PlayMethod.ContinuousPlayThroughWithDelay, PlayMethod.↩ ContinuousPlayThroughWithRandomDelayInRange, PlayMethod.OncePlayThrough, PlayMethod.OncePlayThroughWithDelay, PlayMethod.OncePlayThroughWithRandomDelayInRange, Play↩ Method.ShufflePlayThrough, PlayMethod.ShufflePlayThroughWithDelay, PlayMethod.ShufflePlayThroughWithRandomDelayInRange }

  Enum representing what method to play songs

Public Member Functions

- void OnLevelWasLoaded (int level)

  Raises the level was loaded event, which handles the level loading behavior of SMP.
- delegate void SongCallBack ()

  Song callback delegate.

Static Public Member Functions

- static void AddSoundConnection (SoundConnection sc)

  Adds the SoundConnection to the manager.
- static void RemoveSoundConnectionForLevel (string lvl)

  Removes the SoundConnection for level.
- static void ReplaceSoundConnection (SoundConnection sc)

  Replaces the SoundConnection at that level.
- static int SoundConnectionsContainsThisLevel (string lvl)

  Checks if a level has a SoundConnection.
- static SoundConnection GetSoundConnectionForThisLevel (string lvl)

  Gets the sound connection for this level.
- static bool MuteMusic ()

Mutes/Unmutes all sounds.

- static bool MuteMusic (bool toggle)

  Sets Mute to all sounds
- static bool IsMusicMuted ()

  Determines whether this instance is music muted.
- static bool Mute ()

  Mutes/Unmutes all sounds.
- static bool Mute (bool toggle)

  Sets Mute to all sounds
- static bool IsMuted ()

  Determines whether this instance is muted.
- static void SetVolumeMusic (float setVolume)

  Sets the maximum volume of music in the game relative to the global volume.
- static float GetVolumeMusic ()

  Gets the music volume.
- static void SetPitchMusic (float setPitch)

  Sets the pitch of music in the game.
- static float GetPitchMusic ()

  Gets the music pitch.
- static void SetVolume (float setVolume)

  Sets the maximum volume of all sound in the game.
- static float GetVolume ()

  Gets the volume.
- static void SetPitch (float setPitch)

  Sets the pitch of all sound in the game.
- static float GetPitch ()

  Gets the pitch.
- static float GetCrossDuration ()

  Gets the duration of the bgm crossfade.
- static void SetCrossDuration (float duration)

  Sets the duration of the bgm crossfade.
- static string GetDefaultResourcesPath ()

  Gets the default resources path.
- static void SetDefaultResourcesPath (string path)

  Sets the default resources path.
- static SoundConnection GetCurrentSoundConnection ()

  Gets the current SoundConnection
- static void SetCurrentSoundConnection (SoundConnection connection)

  Sets the current SoundConnection
- static void SetDisableBGM (bool disabled)

  Sets whether the background music is disabled
- static void SetDisableSFX (bool disabled)

  Sets whether the sfx is disabled
- static void PlayConnection (SoundConnection sc, bool syncPlaybackTime=false, int trackNumber=0)

  Plays the SoundConnection right then, regardless of what you put at the level parameter of the SoundConnection.
- static void PlayConnection (string levelName, bool syncPlaybackTime=false, int trackNumber=0)

  Plays a SoundConnection on SoundManager that matches the level name.
- static SoundConnection CreateSoundConnection (string lvl, params AudioClip[] audioList)

  Creates a SoundConnection.
- static SoundConnection CreateSoundConnection (string lvl, SoundManager.PlayMethod method, params AudioClip[] audioList)

Creates a SoundConnection.

- static SoundConnection CreateSoundConnection (string lvl, SoundManager.PlayMethod method, float delayPlay, params AudioClip[] audioList)

  Creates a SoundConnection.
- static SoundConnection CreateSoundConnection (string lvl, SoundManager.PlayMethod method, float min↩ DelayPlay, float maxDelayPlay, params AudioClip[] audioList)

  Creates a SoundConnection.
- static void PlayImmediately (AudioClip clip2play, bool loop=false, SongCallBack runOnEndFunction=null)

  Plays the clip immediately regardless of a playing SoundConnection, with an option to loop.
- static void Play (AudioClip clip2play, bool loop=false, SongCallBack runOnEndFunction=null)

  Plays the clip by crossing out what's currently playing regardless of a playing SoundConnection, with an option to loop.
- static void StopMusicImmediately ()

  Stops all music immediately.
- static void StopMusic ()

  Crosses out all AudioSources.
- static void Stop ()

  Stops all sound, music and sfx, immediately.
- static void Pause ()

  Pauses all music and SFX.
- static void UnPause ()

  Un-Pauses all music and SFX.
- static void PauseToggle ()

  Toggles sound paused and unpaused.
- static bool IsPaused ()

  Determines if the sound is paused.
- static void SetIgnoreLevelLoad (bool ignore)

  Sets ignore level load, which will decide whether or not to use SoundManagerPro's level load AI.
- static List< AudioClip > GetCurrentSongList ()

  Gets the current song list.
- static AudioClip GetCurrentSong ()

  Gets the current song.
- static void Next ()

  Goes to the Next song in the song list queue.
- static void Prev ()

  Goes to the Previous song in the song list queue.
- static bool ClipNameIsValid (string clipName)

  Checks if the clip name is valid.
- static bool GroupNameIsValid (string groupName)

  Checks if the group name is valid.
- static AudioSource GetCurrentAudioSource ()

  Gets the current audio source playing BGM.
- static int GetTrackNumber (AudioClip clip)

  Gets the track number in the playlist.
- static void SetSFXCap (int cap)

  Sets the SFX cap.
- static AudioSource PlaySFX (AudioClip clip, bool looping=false, float delay=0f, float volume=float.Max↩ Value, float pitch=float.MaxValue, Vector3 location=default(Vector3), SongCallBack runOnEndFunction=null, SoundDuckingSetting duckingSetting=SoundDuckingSetting.DoNotDuck, float duckVolume=0f, float duck↩ Pitch=1f)

  Plays the SFX on an owned & pooled object, will default the location to (0,0,0), pitch to SoundManager.Instance.↩ pitchSFX, volume to SoundManager.Instance.volumeSFX

- static AudioSource PlaySFX (string clipName, bool looping=false, float delay=0f, float volume=float.Max↩ Value, float pitch=float.MaxValue, Vector3 location=default(Vector3), SongCallBack runOnEndFunction=null, SoundDuckingSetting duckingSetting=SoundDuckingSetting.DoNotDuck, float duckVolume=0f, float duck↩ Pitch=1f)

  Plays the SFX on an owned & pooled object by clipname reference on the SoundManager, will default the location to (0,0,0), pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX
- static AudioSource PlayCappedSFX (AudioClip clip, string cappedID, float volume=float.MaxValue, float pitch=float.MaxValue, Vector3 location=default(Vector3))

  Plays the SFX IFF other SFX with the same cappedID are not over the cap limit.
- static AudioSource PlayCappedSFX (string clipName, string cappedID, float volume=float.MaxValue, float pitch=float.MaxValue, Vector3 location=default(Vector3))

  Plays the SFX IFF other SFX with the same cappedID are not over the cap limit.
- static AudioSource PlayCappedSFX (AudioSource aS, AudioClip clip, string cappedID, float volume=float.↩ MaxValue, float pitch=float.MaxValue)

  Plays the SFX IFF other SFX with the same cappedID are not over the cap limit.
- static AudioSource PlayCappedSFX (AudioSource aS, string clipName, string cappedID, float volume=float.↩ MaxValue, float pitch=float.MaxValue)

  Plays the SFX IFF other SFX with the same cappedID are not over the cap limit.
- static AudioSource PlaySFX (AudioSource aS, AudioClip clip, bool looping=false, float delay=0f, float volume=float.MaxValue, float pitch=float.MaxValue, SongCallBack runOnEndFunction=null, SoundDucking↩ Setting duckingSetting=SoundDuckingSetting.DoNotDuck, float duckVolume=0f, float duckPitch=1f)

  Plays the SFX another audiosource of your choice, will default the looping to false, pitch to SoundManager.Instance.↩ pitchSFX, volume to SoundManager.Instance.volumeSFX
- static AudioSource PlaySFX (AudioSource aS, string clipName, bool looping=false, float delay=0f, float volume=float.MaxValue, float pitch=float.MaxValue, SongCallBack runOnEndFunction=null, SoundDucking↩ Setting duckingSetting=SoundDuckingSetting.DoNotDuck, float duckVolume=0f, float duckPitch=1f)

  Plays the SFX another audiosource of your choice, will default the looping to false, pitch to SoundManager.Instance.↩ pitchSFX, volume to SoundManager.Instance.volumeSFX
- static void StopSFXObject (AudioSource aS)

  Stops the SFX on another audiosource
- static AudioSource PlaySFX (GameObject gO, AudioClip clip, bool looping=false, float delay=0f, float volume=float.MaxValue, float pitch=float.MaxValue, SongCallBack runOnEndFunction=null, SoundDucking↩ Setting duckingSetting=SoundDuckingSetting.DoNotDuck, float duckVolume=0f, float duckPitch=1f)

  Plays the SFX another gameObject of your choice, will default the looping to false, pitch to SoundManager.Instance.↩ pitchSFX, volume to SoundManager.Instance.volumeSFX
- static AudioSource PlaySFX (GameObject gO, string clipName, bool looping=false, float delay=0f, float volume=float.MaxValue, float pitch=float.MaxValue, SongCallBack runOnEndFunction=null, SoundDucking↩ Setting duckingSetting=SoundDuckingSetting.DoNotDuck, float duckVolume=0f, float duckPitch=1f)

  Plays the SFX another gameObject of your choice, will default the looping to false, pitch to SoundManager.Instance.↩ pitchSFX, volume to SoundManager.Instance.volumeSFX
- static void StopSFXObject (GameObject gO)

  Stops the SFX on another gameObject
- static void StopSFX ()

  Stops all SFX.
- static AudioSource PlaySFXLoop (AudioSource aS, AudioClip clip, bool tillDestroy=true, float volume=float.↩ MaxValue, float pitch=float.MaxValue, float maxDuration=0f, SongCallBack runOnEndFunction=null, Sound↩ DuckingSetting duckingSetting=SoundDuckingSetting.DoNotDuck, float duckVolume=0f, float duckPitch=1f)

  Plays the SFX in a loop on another audiosource of your choice.
- static AudioSource PlaySFXLoop (AudioSource aS, string clipName, bool tillDestroy=true, float volume=float.MaxValue, float pitch=float.MaxValue, float maxDuration=0f, SongCallBack runOnEnd↩ Function=null, SoundDuckingSetting duckingSetting=SoundDuckingSetting.DoNotDuck, float duck↩ Volume=0f, float duckPitch=1f)

  Plays the SFX in a loop on another audiosource of your choice.

- static AudioSource PlaySFXLoop (GameObject gO, AudioClip clip, bool tillDestroy=true, float volume=float.↩ MaxValue, float pitch=float.MaxValue, float maxDuration=0f, SongCallBack runOnEndFunction=null, Sound↩ DuckingSetting duckingSetting=SoundDuckingSetting.DoNotDuck, float duckVolume=0f, float duckPitch=1f)

  Plays the SFX in a loop on another gameObject of your choice.
- static AudioSource PlaySFXLoop (GameObject gO, string clipName, bool tillDestroy=true, float volume=float.MaxValue, float pitch=float.MaxValue, float maxDuration=0f, SongCallBack runOnEnd↩ Function=null, SoundDuckingSetting duckingSetting=SoundDuckingSetting.DoNotDuck, float duck↩ Volume=0f, float duckPitch=1f)

  Plays the SFX in a loop on another gameObject of your choice.
- static bool MuteSFX (bool toggle)

  Sets mute on all the SFX to 'toggle' value.
- static bool MuteSFX ()

  Toggles mute on SFX.
- static bool IsSFXMuted ()

  Determines whether this instance is SFX muted.
- static void SetVolumeSFX (float setVolume)

  Sets the maximum volume of SFX in the game relative to the global volume.
- static void SetVolumeSFX (float setVolume, bool ignoreMaxSFXVolume, params AudioSource[] audio↩ Sources)

  Sets the volume of a certain group of AudioSources.
- static void SetVolumeSFX (float setVolume, bool ignoreMaxSFXVolume, params GameObject[] sfxObjects)

  Sets the volume of a certain group of SFX Objects.
- static float GetVolumeSFX ()

  Gets the SFX volume.
- static void SetPitchSFX (float setPitch)

  Sets the pitch of SFX in the game.
- static void SetPitchSFX (float setPitch, params AudioSource[] audioSources)

  Sets the pitch of a certain group of AudioSources.
- static void SetPitchSFX (float setPitch, params GameObject[] sfxObjects)

  Sets the pitch of a certain group of SFX Objects.
- static float GetPitchSFX ()

  Gets the SFX pitch.
- static void SaveSFX (AudioClip clip, string grpName)

  Saves the SFX to the SoundManager prefab for easy access for frequently used SFX.
- static void SaveSFX (AudioClip clip, SFXGroup grp)

  Saves the SFX to the SoundManager prefab for easy access for frequently used SFX.
- static void SaveSFX (params AudioClip[] clips)

  Saves the SFX to the SoundManager prefab for easy access for frequently used SFX.
- static void DeleteSFX ()

  Deletes all SFX from the SoundManager.
- static void DeleteSFX (params AudioClip[] clips)

  Deletes certain SFX from the SoundManager.
- static void DeleteSFX (params string[] clipNames)

  Deletes certain SFX from the SoundManager referenced by name.
- static void ApplySFXAttributes (AudioClip clip, int prepool, float baseVolume, float volumeVariation, float pitchVariation)

  Applies the attributes available in the editor to SFX.
- static void ApplySFXAttributes (string clipName, int prepool, float baseVolume, float volumeVariation, float pitchVariation)

  Applies the attributes available in the editor to SFX, referenced by clip name.
- static SFXGroup CreateSFXGroup (string grpName, int capAmount)

  Creates the SFX group and adds it to SoundManager.

- static SFXGroup CreateSFXGroup (string grpName)

  Creates the SFX group and adds it to SoundManager.
- static void MoveToSFXGroup (string clipName, string newGroupName)

  Moves a clip to the specified SFXGroup.
- static void RemoveFromSFXGroup (string clipName)

  Removes a clip from a SFXGroup.
- static AudioClip LoadFromGroup (string grpName)

  Loads a random SFX from a specified SFXGroup.
- static AudioClip[] LoadAllFromGroup (string grpName)

  Loads all SFX from a specified SFXGroup.
- static AudioClip Load (string clipname, string customPath)

  Load the specified clipname, at a custom path if you do not want to use resourcesPath.
- static AudioClip Load (string clipname)

  Load the specified clipname from the stored SFXs.
- static void ResetSFXObject (GameObject sfxObj)

  Resets the SFX object to default values.
- static void Crossfade (float duration, AudioSource fromSource, AudioSource toSource, SongCallBack runOnEndFunction=null)

  Crossfade between two AudioSources.
- static void Crossfade (float duration, GameObject fromSFXObject, GameObject toSFXObject, SongCallBack runOnEndFunction=null)

  Crossfade between two SFX Objects.
- static void CrossIn (float duration, AudioSource source, SongCallBack runOnEndFunction=null)

  Cross in an AudioSource.
- static void CrossIn (float duration, GameObject sfxObject, SongCallBack runOnEndFunction=null)

  Cross in a SFX Object
- static void CrossOut (float duration, AudioSource source, SongCallBack runOnEndFunction=null)

  Cross out an AudioSource.
- static void CrossOut (float duration, GameObject sfxObject, SongCallBack runOnEndFunction=null)

  Cross out a SFX Object.

Public Attributes

- const string VIEW = "view"

  Editor variable – IGNORE AND DO NOT MODIFY
- const string EDIT = "edit"

  Editor variable – IGNORE AND DO NOT MODIFY
- const string HIDE = "hide"

  Editor variable – IGNORE AND DO NOT MODIFY
- Hashtable songStatus = new Hashtable()

  Editor variable – IGNORE AND DO NOT MODIFY
- bool helpOn = false

  Editor variable – IGNORE AND DO NOT MODIFY
- bool showInfo = true

  Editor variable – IGNORE AND DO NOT MODIFY
- bool showDev = true

  Editor variable – IGNORE AND DO NOT MODIFY
- bool showList = true

  Editor variable – IGNORE AND DO NOT MODIFY
- bool showAdd = true

  Editor variable – IGNORE AND DO NOT MODIFY

- bool showSFX = true

  Editor variable – IGNORE AND DO NOT MODIFY
- List< bool > showSFXDetails = new List<bool>()

  Editor variable – IGNORE AND DO NOT MODIFY
- int groupAddIndex = 0

  Editor variable – IGNORE AND DO NOT MODIFY
- int autoPrepoolAmount = 0

  Editor variable – IGNORE AND DO NOT MODIFY
- float autoBaseVolume = 1f

  Editor variable – IGNORE AND DO NOT MODIFY
- float autoVolumeVariation = 0f

  Editor variable – IGNORE AND DO NOT MODIFY
- float autoPitchVariation = 0f

  Editor variable – IGNORE AND DO NOT MODIFY
- bool showAsGrouped = false

  Editor variable – IGNORE AND DO NOT MODIFY
- List< SoundConnection > soundConnections = new List<SoundConnection>()

  The sound connections.
- AudioSource[] audios

  The 2 tracks for background music.
- string currentLevel

  The current level.
- SoundConnection currentSoundConnection

  The current SoundConnection.
- float crossDuration = 5f

  The crossfade duration.
- bool showDebug = true

  Editor variable – IGNORE AND DO NOT MODIFY
- bool offTheBGM = false

  Turn off the background music.
- bool isPaused = false

  Whether sound is paused.
- bool movingOnFromSong = false

  Whether the background music is being forced to move on to the next song.
- const int SOUNDMANAGER_FALSE = -1
- SongCallBack OnSongEnd

  Called when a song ends, AFTER crossfade out ends as well.
- SongCallBack OnSongBegin

  Called when a song begins, AFTER crossfade in ends.
- SongCallBack OnCrossOutBegin

  Called when crossfade out begins.
- SongCallBack OnCrossInBegin

  Called when crossfade in begins.
- bool ignoreLevelLoad = false

  Set this if you wish to ignore the level loading functionality.
- string resourcesPath = "Sounds/SFX"

  Path to folder where SFX are held in resources
- List< AudioClip > storedSFXs = new List<AudioClip>()

  List of local AudioClip SFXs added in inspector or through SaveSFX()
- List< GameObject > unOwnedSFXObjects = new List<GameObject>()

  List of other unowned gameobjects with SFX attached.

- Dictionary< int, string > cappedSFXObjects = new Dictionary<int, string>()

  Dictionary of instance ID to cappedID to keep track of capped SFX
- Dictionary< AudioSource, float > delayedAudioSources = new Dictionary<AudioSource, float>()

  Dictionary of delayed AudioSources
- Dictionary< AudioSource, SongCallBack > runOnEndFunctions = new Dictionary<AudioSource, SongCallBack>()

  Dictionary of sfx with runonendfunctions
- List< SFXGroup > sfxGroups = new List<SFXGroup>()

  List of SFXGroups.
- List< string > clipToGroupKeys = new List<string>()

  Editor variable – IGNORE AND DO NOT MODIFY
- List< string > clipToGroupValues = new List<string>()

  Editor variable – IGNORE AND DO NOT MODIFY
- bool offTheSFX = false

  Turn off the SFX.
- int capAmount = 3

  The default cap amount.
- List< int > sfxPrePoolAmounts = new List<int>()

  The sfx pre pool amounts.
- List< float > sfxBaseVolumes = new List<float>()

  The sfx base volumes.
- List< float > sfxVolumeVariations = new List<float>()

  The sfx volume variations.
- List< float > sfxPitchVariations = new List<float>()

  The sfx pitch variations.
- float SFXObjectLifetime = 10f

  The SFX object lifetime for objects outside of the prepool amount.
- List< string > currentPockets = new List<string>() { "Default" }

  The current SoundPocket s by name.

Static Public Attributes

- static float duckStartSpeed = .1f

  The start speed of the ducking effect.
- static float duckEndSpeed = .5f

  The end speed of the ducking effect.

Properties

- bool viewAll `[get, set]`

  Editor variable – IGNORE AND DO NOT MODIFY
- static SoundManager Instance `[get, set]`

  Gets or sets the instance.
- float volume1 `[get, set]`

  Gets or sets the volume of BGM track 1.
- float volume2 `[get, set]`

  Gets or sets the volume of BGM track 2.
- float maxMusicVolume `[get, set]`

  Gets or sets the max music volume.
- float maxVolume `[get, set]`

Gets or sets the max volume.

- bool mutedMusic [get, set]

  Gets or sets a value indicating whether this SoundManager muted music.

- bool muted [get, set]

  Gets or sets a value indicating whether this SoundManager is muted.

- float volumeSFX [get, set]

  Gets or sets the SFX volume.

- float pitchSFX [get, set]

  Gets or sets the SFX pitch.

- float maxSFXVolume [get, set]

  Gets or sets the max SFX volume.

- bool mutedSFX [get, set]

  Gets or sets a value indicating whether this SoundManager muted SFX.

6.9.1 Detailed Description

SoundManager is where most functions will be called from.

This is the main controller of sound activity in your app.

6.9.2 Member Enumeration Documentation

6.9.2.1 enum SoundManager.PlayMethod

Enum representing what method to play songs

Enumerator

ContinuousPlayThrough

ContinuousPlayThroughWithDelay

ContinuousPlayThroughWithRandomDelayInRange

OncePlayThrough

OncePlayThroughWithDelay

OncePlayThroughWithRandomDelayInRange

ShufflePlayThrough

ShufflePlayThroughWithDelay

ShufflePlayThroughWithRandomDelayInRange

6.9.3 Member Function Documentation

6.9.3.1 static void SoundManager.AddSoundConnection (SoundConnection *sc*) [static]

Adds the SoundConnection to the manager.

It will not add multiple SoundConnections to one level.

Parameters

sc	The SoundConnection.

6.9.3.2 static void SoundManager.ApplySFXAttributes (AudioClip *clip,* int *prepool,* float *baseVolume,* float *volumeVariation,* float *pitchVariation*) [static]

Applies the attributes available in the editor to SFX.

Attributes like prepool amount, base volume, volume variation, and pitch variation.

Parameters

clip	Clip.
prepool	Prepool amount.
baseVolume	Base volume.
volumeVariation	Volume variation.
pitchVariation	Pitch variation.

6.9.3.3 static void SoundManager.ApplySFXAttributes (string *clipName,* int *prepool,* float *baseVolume,* float *volumeVariation,* float *pitchVariation*) [static]

Applies the attributes available in the editor to SFX, referenced by clip name.

Attributes like prepool amount, base volume, volume variation, and pitch variation.

Parameters

clipName	Clip name.
prepool	Prepool amount.
baseVolume	Base volume.
volumeVariation	Volume variation.
pitchVariation	Pitch variation.

6.9.3.4 static bool SoundManager.ClipNameIsValid (string *clipName*) [static]

Checks if the clip name is valid.

Returns

If the name is valid.

Parameters

clipName	The clip name to check.

6.9.3.5 static SFXGroup SoundManager.CreateSFXGroup (string *grpName,* int *capAmount*) [static]

Creates the SFX group and adds it to SoundManager.

Returns

The SFX group.

Parameters

grpName	Group name.
capAmount	Cap amount.

6.9.3.6 static SFXGroup SoundManager.CreateSFXGroup (string *grpName*) [static]

Creates the SFX group and adds it to SoundManager.

Returns

The SFX group.

Parameters

grpName	Group name.

6.9.3.7 static SoundConnection SoundManager.CreateSoundConnection (string *lvl*, params AudioClip[] *audioList*) [static]

Creates a SoundConnection.

You can use a scene name or a custom name. Will default to ContinousPlayThrough.

Returns

The SoundConnection.

Parameters

lvl	The level name of the SoundConnection.
audioList	Audio list.

6.9.3.8 static SoundConnection SoundManager.CreateSoundConnection (string *lvl*, SoundManager.PlayMethod *method*, params AudioClip[] *audioList*) [static]

Creates a SoundConnection.

You can use a scene name or a custom name.

Returns

The SoundConnection.

Parameters

lvl	The level name of the SoundConnection.
method	The PlayMethod.
audioList	Audio list.

6.9.3.9 static SoundConnection SoundManager.CreateSoundConnection (string *lvl*, SoundManager.PlayMethod *method*, float *delayPlay*, params AudioClip[] *audioList*) [static]

Creates a SoundConnection.

You can use a scene name or a custom name. This overload is used for PlayMethods that use Delay.

Returns

The SoundConnection.

Parameters

lvl	The level name of the SoundConnection.
method	The PlayMethod.
delayPlay	The exact delay.

audioList	Audio list.

6.9.3.10 static SoundConnection SoundManager.CreateSoundConnection (string *lvl*, SoundManager.PlayMethod *method*, float *minDelayPlay*, float *maxDelayPlay*, params AudioClip[] *audioList*) [static]

Creates a SoundConnection.

You can use a scene name or a custom name. This overload is used for PlayMethods that use Random Delay In Range.

Returns

The SoundConnection.

Parameters

lvl	The level name of the SoundConnection.
method	The PlayMethod.
minDelayPlay	The minimum delay.
maxDelayPlay	The maximum delay.
audioList	Audio list.

6.9.3.11 static void SoundManager.Crossfade (float *duration*, AudioSource *fromSource*, AudioSource *toSource*, SongCallBack *runOnEndFunction* = null) [static]

Crossfade between two AudioSources.

Parameters

duration	Duration.
fromSource	From source.
toSource	To source.
runOnEnd↩ Function	Run on end function.

6.9.3.12 static void SoundManager.Crossfade (float *duration*, GameObject *fromSFXObject*, GameObject *toSFXObject*, SongCallBack *runOnEndFunction* = null) [static]

Crossfade between two SFX Objects.

Parameters

duration	Duration.
fromSFXObject	From SFX object.
toSFXObject	To SFX object.
runOnEnd↩ Function	Run on end function.

6.9.3.13 static void SoundManager.CrossIn (float *duration*, AudioSource *source*, SongCallBack *runOnEndFunction* = null) [static]

Cross in an AudioSource.

Parameters

duration	Duration.
source	Source.
runOnEnd↩ Function	Run on end function.

6.9.3.14 static void SoundManager.CrossIn (float *duration,* GameObject *sfxObject,* SongCallBack *runOnEndFunction =* `null`) [static]

Cross in a SFX Object

Parameters

duration	Duration.
sfxObject	Sfx object.
runOnEnd↩ Function	Run on end function.

6.9.3.15 static void SoundManager.CrossOut (float *duration,* AudioSource *source,* SongCallBack *runOnEndFunction =* `null`) [static]

Cross out an AudioSource.

Parameters

duration	Duration.
source	Source.
runOnEnd↩ Function	Run on end function.

6.9.3.16 static void SoundManager.CrossOut (float *duration,* GameObject *sfxObject,* SongCallBack *runOnEndFunction =* `null`) [static]

Cross out a SFX Object.

Parameters

duration	Duration.
sfxObject	Sfx object.
runOnEnd↩ Function	Run on end function.

6.9.3.17 static void SoundManager.DeleteSFX ( ) [static]

Deletes all SFX from the SoundManager.

6.9.3.18 static void SoundManager.DeleteSFX (params AudioClip[] *clips*) [static]

Deletes certain SFX from the SoundManager.

Parameters

clips	Clips.

6.9.3.19 static void SoundManager.DeleteSFX (params string[] *clipNames*) [static]

Deletes certain SFX from the SoundManager referenced by name.

Parameters

clipNames	Clip names.

6.9.3.20 static float SoundManager.GetCrossDuration ( ) [static]

Gets the duration of the bgm crossfade.

Returns

The cross duration.

6.9.3.21 static AudioSource SoundManager.GetCurrentAudioSource ( ) [static]

Gets the current audio source playing BGM.

Will return null if not playing anything at all.

Returns

The current audio source.

6.9.3.22 static AudioClip SoundManager.GetCurrentSong ( ) [static]

Gets the current song.

Returns

The current song.

6.9.3.23 static List<AudioClip> SoundManager.GetCurrentSongList ( ) [static]

Gets the current song list.

Returns

The current song list in the current SoundConnection.

6.9.3.24 static SoundConnection SoundManager.GetCurrentSoundConnection ( ) [static]

Gets the current SoundConnection

Returns

The current sound connection.

6.9.3.25 static string SoundManager.GetDefaultResourcesPath ( ) [static]

Gets the default resources path.

Used in SoundManager.Load

Returns

The default resources path.

6.9.3.26 static float SoundManager.GetPitch ( ) [static]

Gets the pitch.

Prioritizes music.

Returns

The pitch.

6.9.3.27 static float SoundManager.GetPitchMusic ( ) [static]

Gets the music pitch.

Prioritizes music.

Returns

The music pitch.

6.9.3.28 static float SoundManager.GetPitchSFX ( ) [static]

Gets the SFX pitch.

Returns

The SFX pitch.

6.9.3.29 static SoundConnection SoundManager.GetSoundConnectionForThisLevel (string *lvl*) [static]

Gets the sound connection for this level.

Returns

The sound connection for this level.

Parameters

lvl	The level name of the SoundConnection.

6.9.3.30 static int SoundManager.GetTrackNumber (AudioClip *clip*) [static]

Gets the track number in the playlist.

Otherwise returns -1 if not in playlist.

Returns

The track number.

Parameters

clip	Clip.

6.9.3.31 static float SoundManager.GetVolume ( ) [static]

Gets the volume.

Returns

The volume.

6.9.3.32 static float SoundManager.GetVolumeMusic ( ) [static]

Gets the music volume.

Returns

The music volume.

6.9.3.33 static float SoundManager.GetVolumeSFX ( ) [static]

Gets the SFX volume.

Returns

The SFX volume.

6.9.3.34 static bool SoundManager.GroupNameIsValid (string *groupName*) [static]

Checks if the group name is valid.

Returns

If the group name is valid.

Parameters

groupName	The SFXGroup name to check.

6.9.3.35 static bool SoundManager.IsMusicMuted ( ) [static]

Determines whether this instance is music muted.

Returns

`true` if this instance is music muted; otherwise, `false`.

6.9.3.36 static bool SoundManager.IsMuted ( ) [static]

Determines whether this instance is muted.

Returns

If all sounds are muted or not.

6.9.3.37 static bool SoundManager.IsPaused ( ) [static]

Determines if the sound is paused.

Returns

`true` if is paused; otherwise, `false`.

6.9.3.38 static bool SoundManager.IsSFXMuted ( ) [static]

Determines whether this instance is SFX muted.

Returns

`true` if this instance is SFX muted; otherwise, `false`.

6.9.3.39 static AudioClip SoundManager.Load (string *clipname,* string *customPath*) [static]

Load the specified clipname, at a custom path if you do not want to use resourcesPath.

If custompath fails or is empty/null, it will query the stored SFXs. If that fails, it'll query the default resourcesPath. If all else fails, it'll return null.

Returns

The clip.

Parameters

clipname	Clip name.
customPath	Custom path.

6.9.3.40 static AudioClip SoundManager.Load (string *clipname*) [static]

Load the specified clipname from the stored SFXs.

If that fails, it'll query the default resourcesPath. If all else fails, it'll return null.

Returns

The clip.

Parameters

clipname	Clipname.

6.9.3.41 static AudioClip [] SoundManager.LoadAllFromGroup (string *grpName*) [static]

Loads all SFX from a specified SFXGroup.

Returns

The all clips from the group.

Parameters

grpName	Group name.

6.9.3.42 static AudioClip SoundManager.LoadFromGroup (string *grpName*) [static]

Loads a random SFX from a specified SFXGroup.

Returns

The random clip.

Parameters

grpName	Group name.

6.9.3.43 static void SoundManager.MoveToSFXGroup (string *clipName*, string *newGroupName*) [static]

Moves a clip to the specified SFXGroup.

If the group doesn't exist, it will make the group.

Parameters

clipName	Clip name.
newGroupName	New group name.

6.9.3.44 static bool SoundManager.Mute ( ) [static]

Mutes/Unmutes all sounds.

Returns a bool if it's muted or not, priority is given to the music mute

Returns

If all sounds are NOW muted or not.

6.9.3.45 static bool SoundManager.Mute (bool *toggle*) [static]

Sets Mute to all sounds

Returns

If all sounds are NOW muted or not.

Parameters

toggle	If set to `true` toggle.

6.9.3.46 static bool SoundManager.MuteMusic ( ) [static]

Mutes/Unmutes all sounds.

Returns a bool if it's muted or not.

Returns

If the sound is NOW muted or not.

6.9.3.47 static bool SoundManager.MuteMusic (bool *toggle*) [static]

Sets Mute to all sounds

Returns

If the sound is NOW muted or not.

Parameters

toggle	The mute to set.

6.9.3.48 static bool SoundManager.MuteSFX (bool *toggle*) [static]

Sets mute on all the SFX to 'toggle' value.

Returns the result.

Returns

If SFX is NOW muted or not.

Parameters

toggle	The mute to set.

6.9.3.49 static bool SoundManager.MuteSFX ( ) [static]

Toggles mute on SFX.

Returns the result.

Returns

If the SFX is NOW muted or not.

6.9.3.50 static void SoundManager.Next ( ) [static]

Goes to the Next song in the song list queue.

Becareful using this in OncePlayThrough methods.

6.9.3.51 void SoundManager.OnLevelWasLoaded (int *level*)

Raises the level was loaded event, which handles the level loading behavior of SMP.

Parameters

level	Level.

6.9.3.52 static void SoundManager.Pause ( ) [static]

Pauses all music and SFX.

Call UnPause to unpause

6.9.3.53 static void SoundManager.PauseToggle ( ) [static]

Toggles sound paused and unpaused.

6.9.3.54 static void SoundManager.Play (AudioClip *clip2play,* bool *loop* = false, SongCallBack *runOnEndFunction* = null) [static]

Plays the clip by crossing out what's currently playing regardless of a playing SoundConnection, with an option to loop.

Calls an event once the clip is done. You can resume a SoundConnection afterwards if you so choose, using Instance.currentSoundConnection. However, it will not resume on it's own. Callbacks will only fire once.

Parameters

clip2play	The clip to play.
runOnEnd↩ Function	Function to run once the clip is done. Is added to OnSongEnd
loop	Whether the clip should loop

6.9.3.55 static AudioSource SoundManager.PlayCappedSFX (AudioClip *clip,* string *cappedID,* float *volume* = float.MaxValue, float *pitch* = float.MaxValue, Vector3 *location* = default(Vector3)) [static]

Plays the SFX IFF other SFX with the same cappedID are not over the cap limit.

Will default the location to (0,0,0), pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.↩ volumeSFX

Returns

The resulting AudioSource.

Parameters

clip	Clip.
cappedID	Capped ID.
volume	Volume.
pitch	Pitch.
location	Location.

6.9.3.56 static AudioSource SoundManager.PlayCappedSFX (string *clipName,* string *cappedID,* float *volume* = float.MaxValue, float *pitch* = float.MaxValue, Vector3 *location* = default(Vector3)) [static]

Plays the SFX IFF other SFX with the same cappedID are not over the cap limit.

Will default the location to (0,0,0), pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.↩ volumeSFX

Returns

The resulting AudioSource.

Parameters

clipName	Clip name.
cappedID	Capped ID.
volume	Volume.
pitch	Pitch.
location	Location.

6.9.3.57 static AudioSource SoundManager.PlayCappedSFX (AudioSource *aS*, AudioClip *clip*, string *cappedID*, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`) [static]

Plays the SFX IFF other SFX with the same cappedID are not over the cap limit.

Will default the pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX

Returns

The resulting `AudioSource`.

Parameters

aS	`AudioSource` to play on.
clip	Clip.
cappedID	Capped ID.
volume	Volume.
pitch	Pitch.

6.9.3.58 static AudioSource SoundManager.PlayCappedSFX (AudioSource *aS*, string *clipName*, string *cappedID*, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`) [static]

Plays the SFX IFF other SFX with the same cappedID are not over the cap limit.

Will default the pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX

Returns

The resulting `AudioSource`.

Parameters

aS	`AudioSource` to play on.
clipName	Clip name.
cappedID	Capped ID.
volume	Volume.
pitch	Pitch.

6.9.3.59 static void SoundManager.PlayConnection (SoundConnection *sc*, bool *syncPlaybackTime* = `false`, int *trackNumber* = `0`) [static]

Plays the SoundConnection right then, regardless of what you put at the level parameter of the SoundConnection.

Parameters

sc	The SoundConnection.
syncPlayback↩ Time	Whether the playback times should be synced.
trackNumber	Track number to skip to, starting at 0.

6.9.3.60 static void SoundManager.PlayConnection (string *levelName*, bool *syncPlaybackTime* = `false`, int *trackNumber* = `0`) `[static]`

Plays a SoundConnection on SoundManager that matches the level name.

Parameters

levelName	The levelName of the SoundConnection.
syncPlayback↩ Time	Whether the playback times should be synced.
trackNumber	Track number to skip to, starting at 0.

6.9.3.61 static void SoundManager.PlayImmediately (AudioClip *clip2play*, bool *loop* = `false`, SongCallBack *runOnEndFunction* = `null`) `[static]`

Plays the clip immediately regardless of a playing SoundConnection, with an option to loop.

Calls an event once the clip is done. You can resume a SoundConnection afterwards if you so choose, using Instance.currentSoundConnection. However, it will not resume on it's own. Callbacks will only fire once.

Parameters

clip2play	The clip to play.
runOnEnd↩ Function	Function to run once the clip is done. Is added to OnSongEnd
loop	Whether the clip should loop

6.9.3.62 static AudioSource SoundManager.PlaySFX (AudioClip *clip*, bool *looping* = `false`, float *delay* = `0f`, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`, Vector3 *location* = `default(Vector3)`, SongCallBack *runOnEndFunction* = `null`, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = `0f`, float *duckPitch* = `1f`) `[static]`

Plays the SFX on an owned & pooled object, will default the location to (0,0,0), pitch to SoundManager.Instance.↩ pitchSFX, volume to SoundManager.Instance.volumeSFX

Returns

The resulting `AudioSource`.

Parameters

clip	Clip.
looping	Whether it is looping.
delay	Delay.
volume	Volume. If set to float.MaxValue, it will become the default volume currently set.
pitch	Pitch. If set to float.MaxValue, it will become the default pitch currently set.

location	Location.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.63 static AudioSource SoundManager.PlaySFX (string *clipName*, bool *looping* = `false`, float *delay* = `0f`, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`, Vector3 *location* = `default(Vector3)`, SongCallBack *runOnEndFunction* = `null`, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = `0f`, float *duckPitch* = `1f`) `[static]`

Plays the SFX on an owned & pooled object by clipname reference on the SoundManager, will default the location to (0,0,0), pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX

Returns

The resulting `AudioSource`.

Parameters

clipName	Name of the clip on the SoundManager.
looping	Whether it is looping.
delay	Delay.
volume	Volume. If set to float.MaxValue, it will become the default volume currently set.
pitch	Pitch. If set to float.MaxValue, it will become the default pitch currently set.
location	Location.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.64 static AudioSource SoundManager.PlaySFX (AudioSource *aS*, AudioClip *clip*, bool *looping* = `false`, float *delay* = `0f`, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`, SongCallBack *runOnEndFunction* = `null`, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = `0f`, float *duckPitch* = `1f`) `[static]`

Plays the SFX another audiosource of your choice, will default the looping to false, pitch to SoundManager.↩ Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX

Returns

The resulting `AudioSource`.

Parameters

aS	`AudioSource` to play on.
clip	Clip.
looping	Looping.
delay	Delay.

volume	Volume.
pitch	Pitch.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.65 static AudioSource SoundManager.PlaySFX (AudioSource *aS*, string *clipName*, bool *looping* = `false`, float *delay* = `0f`, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`, SongCallBack *runOnEndFunction* = `null`, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = `0f`, float *duckPitch* = `1f`) [static]

Plays the SFX another audiosource of your choice, will default the looping to false, pitch to SoundManager.↩ Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX

Returns

The resulting `AudioSource`.

Parameters

aS	`AudioSource` to play on.
clipName	Clip name.
looping	Looping.
delay	Delay.
volume	Volume.
pitch	Pitch.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.66 static AudioSource SoundManager.PlaySFX (GameObject *gO*, AudioClip *clip*, bool *looping* = `false`, float *delay* = `0f`, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`, SongCallBack *runOnEndFunction* = `null`, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = `0f`, float *duckPitch* = `1f`) [static]

Plays the SFX another gameObject of your choice, will default the looping to false, pitch to SoundManager.↩ Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX

Returns

The resulting `AudioSource`.

Parameters

gO	GameObject to play on.
clip	Clip.
looping	Looping.

delay	Delay.
volume	Volume.
pitch	Pitch.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.67 static AudioSource SoundManager.PlaySFX (GameObject *gO*, string *clipName*, bool *looping* = `false`, float *delay* = `0f`, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`, SongCallBack *runOnEndFunction* = `null`, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = `0f`, float *duckPitch* = `1f`) [static]

Plays the SFX another gameObject of your choice, will default the looping to false, pitch to SoundManager.↩ Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX

Returns

The resulting `AudioSource`.

Parameters

gO	GameObject to play on.
clipName	Clip name.
looping	Looping.
delay	Delay.
volume	Volume.
pitch	Pitch.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.68 static AudioSource SoundManager.PlaySFXLoop (AudioSource *aS*, AudioClip *clip*, bool *tillDestroy* = `true`, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`, float *maxDuration* = `0f`, SongCallBack *runOnEndFunction* = `null`, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = `0f`, float *duckPitch* = `1f`) [static]

Plays the SFX in a loop on another audiosource of your choice.

This function is cattered more towards customizing a loop. You can set the loop to end when the object dies or a maximum duration, whichever comes first. tillDestroy defaults to true, pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX, maxDuration to 0f

Returns

The resulting `AudioSource`.

Parameters

aS	AudioSource to play on.
clip	Clip.
tillDestroy	Till destroyed?
volume	Volume.
pitch	Pitch.
maxDuration	Max duration.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.69 static AudioSource SoundManager.PlaySFXLoop (AudioSource *aS*, string *clipName*, bool *tillDestroy* = true, float *volume* = float.MaxValue, float *pitch* = float.MaxValue, float *maxDuration* = 0f, SongCallBack *runOnEndFunction* = null, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = 0f, float *duckPitch* = 1f) [static]

Plays the SFX in a loop on another audiosource of your choice.

This function is cattered more towards customizing a loop. You can set the loop to end when the object dies or a maximum duration, whichever comes first. tillDestroy defaults to true, pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX, maxDuration to 0f

Returns

The resulting AudioSource.

Parameters

aS	AudioSource to play on.
clipName	Clip name.
tillDestroy	Till destroyed?
volume	Volume.
pitch	Pitch.
maxDuration	Max duration.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.70 static AudioSource SoundManager.PlaySFXLoop (GameObject *gO*, AudioClip *clip*, bool *tillDestroy* = true, float *volume* = float.MaxValue, float *pitch* = float.MaxValue, float *maxDuration* = 0f, SongCallBack *runOnEndFunction* = null, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = 0f, float *duckPitch* = 1f) [static]

Plays the SFX in a loop on another gameObject of your choice.

This function is cattered more towards customizing a loop. You can set the loop to end when the object dies or a maximum duration, whichever comes first. tillDestroy defaults to true, pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX, maxDuration to 0f

Returns

The resulting AudioSource.

Parameters

gO	GameObject to play on.
clip	Clip.
tillDestroy	Till destroyed?
volume	Volume.
pitch	Pitch.
maxDuration	Max duration.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.71 static AudioSource SoundManager.PlaySFXLoop (GameObject *gO*, string *clipName*, bool *tillDestroy* = `true`, float *volume* = `float.MaxValue`, float *pitch* = `float.MaxValue`, float *maxDuration* = `0f`, SongCallBack *runOnEndFunction* = `null`, SoundDuckingSetting *duckingSetting* = SoundDuckingSetting.DoNotDuck, float *duckVolume* = `0f`, float *duckPitch* = `1f`) [static]

Plays the SFX in a loop on another gameObject of your choice.

This function is cattered more towards customizing a loop. You can set the loop to end when the object dies or a maximum duration, whichever comes first. tillDestroy defaults to true, pitch to SoundManager.Instance.pitchSFX, volume to SoundManager.Instance.volumeSFX, maxDuration to 0f

Returns

The resulting `AudioSource`.

Parameters

gO	GameObject to play on.
clipName	Clip name.
tillDestroy	Till destroyed?
volume	Volume.
pitch	Pitch.
maxDuration	Max duration.
runOnEnd↩ Function	Run on end function.
duckingSetting	Ducking setting.
duckVolume	Duck volume.
duckPitch	Duck pitch.

6.9.3.72 static void SoundManager.Prev ( ) [static]

Goes to the Previous song in the song list queue.

Becareful using this in OncePlayThrough methods.

6.9.3.73 static void SoundManager.RemoveFromSFXGroup (string *clipName*) [static]

Removes a clip from a SFXGroup.

Parameters

clipName	Clip name.

6.9.3.74 static void SoundManager.RemoveSoundConnectionForLevel (string *lvl*) [static]

Removes the SoundConnection for level.

It will not remove anything if that level does not exist.

Parameters

lvl	Level name of the SoundConnection.

6.9.3.75 static void SoundManager.ReplaceSoundConnection (SoundConnection *sc*) [static]

Replaces the SoundConnection at that level.

If a SoundConnection doesn't exist, it will just add it.

Parameters

sc	The SoundConnection.

6.9.3.76 static void SoundManager.ResetSFXObject (GameObject *sfxObj*) [static]

Resets the SFX object to default values.

Parameters

sfxObj	SFX object.

6.9.3.77 static void SoundManager.SaveSFX (AudioClip *clip,* string *grpName*) [static]

Saves the SFX to the SoundManager prefab for easy access for frequently used SFX.

Will register the SFX to the group.

Parameters

clip	Clip.
grpName	Group name.

6.9.3.78 static void SoundManager.SaveSFX (AudioClip *clip,* SFXGroup *grp*) [static]

Saves the SFX to the SoundManager prefab for easy access for frequently used SFX.

Will register the SFX to the group specified. If the group doesn't exist, it will be added to SoundManager.

Parameters

clip	Clip.
grp	Group.

6.9.3.79 static void SoundManager.SaveSFX (params AudioClip[] *clips*) [static]

Saves the SFX to the SoundManager prefab for easy access for frequently used SFX.

Parameters

clips	Clips.

6.9.3.80 static void SoundManager.SetCrossDuration (float *duration*) [static]

Sets the duration of the bgm crossfade.

Parameters

duration	Cross duration.

6.9.3.81 static void SoundManager.SetCurrentSoundConnection (SoundConnection *connection*) [static]

Sets the current SoundConnection

Parameters

connection	SoundConnection.

6.9.3.82 static void SoundManager.SetDefaultResourcesPath (string *path*) [static]

Sets the default resources path.

Used in SoundManager.Load

Parameters

path	Path.

6.9.3.83 static void SoundManager.SetDisableBGM (bool *disabled*) [static]

Sets whether the background music is disabled

Parameters

disabled	Disabled.

6.9.3.84 static void SoundManager.SetDisableSFX (bool *disabled*) [static]

Sets whether the sfx is disabled

Parameters

disabled	Disabled.

6.9.3.85 static void SoundManager.SetIgnoreLevelLoad (bool *ignore*) [static]

Sets ignore level load, which will decide whether or not to use SoundManagerPro's level load AI.

Parameters

ignore	Ignore.

6.9.3.86 static void SoundManager.SetPitch (float *setPitch*) [static]

Sets the pitch of all sound in the game.

Parameters

setPitch	Set pitch.

6.9.3.87 static void SoundManager.SetPitchMusic (float *setPitch*) [static]

Sets the pitch of music in the game.

Parameters

setPitch	Set pitch.

6.9.3.88 static void SoundManager.SetPitchSFX (float *setPitch*) [static]

Sets the pitch of SFX in the game.

Parameters

setPitch	Set pitch.

6.9.3.89 static void SoundManager.SetPitchSFX (float *setPitch,* params AudioSource[] *audioSources*) [static]

Sets the pitch of a certain group of AudioSources.

Parameters

setPitch	Set pitch.
audioSources	Audio sources.

6.9.3.90 static void SoundManager.SetPitchSFX (float *setPitch,* params GameObject[] *sfxObjects*) [static]

Sets the pitch of a certain group of SFX Objects.

Parameters

setPitch	Set pitch.
sfxObjects	Sfx objects.

6.9.3.91 static void SoundManager.SetSFXCap (int *cap*) [static]

Sets the SFX cap.

Parameters

cap	Cap.

6.9.3.92 static void SoundManager.SetVolume (float *setVolume*) [static]

Sets the maximum volume of all sound in the game.

Parameters

setVolume	Set volume.

6.9.3.93 static void SoundManager.SetVolumeMusic (float *setVolume*) [static]

Sets the maximum volume of music in the game relative to the global volume.

Parameters

setVolume	Set volume.

6.9.3.94 static void SoundManager.SetVolumeSFX (float *setVolume*) [static]

Sets the maximum volume of SFX in the game relative to the global volume.

Parameters

setVolume	Set volume.

6.9.3.95 static void SoundManager.SetVolumeSFX (float *setVolume,* bool *ignoreMaxSFXVolume,* params AudioSource[] *audioSources*) [static]

Sets the volume of a certain group of AudioSources.

Can set to ignore the max SFX volume.

Parameters

setVolume	Set volume.
ignoreMaxSFX↩ Volume	Ignore max SFX volume?
audioSources	Audio sources.

6.9.3.96 static void SoundManager.SetVolumeSFX (float *setVolume,* bool *ignoreMaxSFXVolume,* params GameObject[] *sfxObjects*) [static]

Sets the volume of a certain group of SFX Objects.

Can set to ignore the max SFX volume.

Parameters

setVolume	Set volume.
ignoreMaxSFX↩ Volume	Ignore max SFX volume.

sfxObjects	SFX objects.

6.9.3.97 delegate void SoundManager.SongCallBack ( )

Song callback delegate.

6.9.3.98 static int SoundManager.SoundConnectionsContainsThisLevel (string *lvl*) [static]

Checks if a level has a SoundConnection.

If it does, it returns the index. If it doesn't it returns constant SOUNDMANAGER_FALSE(-1)

Returns

Index of the SoundConnection or SOUNDMANAGER_FALSE(-1)

Parameters

lvl	The level name of the SoundConnection.

6.9.3.99 static void SoundManager.Stop ( ) [static]

Stops all sound, music and sfx, immediately.

6.9.3.100 static void SoundManager.StopMusic ( ) [static]

Crosses out all AudioSources.

6.9.3.101 static void SoundManager.StopMusicImmediately ( ) [static]

Stops all music immediately.

6.9.3.102 static void SoundManager.StopSFX ( ) [static]

Stops all SFX.

6.9.3.103 static void SoundManager.StopSFXObject (AudioSource *aS*) [static]

Stops the SFX on another audiosource

Parameters

aS	AudioSource to stop.

6.9.3.104 static void SoundManager.StopSFXObject (GameObject *gO*) [static]

Stops the SFX on another gameObject

Parameters

gO	GameObject to stop.

6.9.3.105 static void SoundManager.UnPause ( ) [static]

Un-Pauses all music and SFX.

6.9.4 Member Data Documentation

6.9.4.1 AudioSource [ ] SoundManager.audios

The 2 tracks for background music.

6.9.4.2 float SoundManager.autoBaseVolume = 1f

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.3 float SoundManager.autoPitchVariation = 0f

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.4 int SoundManager.autoPrepoolAmount = 0

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.5 float SoundManager.autoVolumeVariation = 0f

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.6 int SoundManager.capAmount = 3

The default cap amount.

6.9.4.7 Dictionary<int, string> SoundManager.cappedSFXObjects = new Dictionary<int, string>()

Dictionary of instance ID to cappedID to keep track of capped SFX

6.9.4.8 List<string> SoundManager.clipToGroupKeys = new List<string>()

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.9 List<string> SoundManager.clipToGroupValues = new List<string>()

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.10 float SoundManager.crossDuration = 5f

The crossfade duration.

6.9.4.11 string SoundManager.currentLevel

The current level.

6.9.4.12 List<string> SoundManager.currentPockets = new List<string>() { "Default" }

The current SoundPocket s by name.

6.9.4.13 SoundConnection SoundManager.currentSoundConnection

The current SoundConnection.

6.9.4.14 Dictionary<AudioSource, float> SoundManager.delayedAudioSources = new Dictionary<AudioSource, float>()

Dictionary of delayed AudioSources

6.9.4.15 float SoundManager.duckEndSpeed = .5f [static]

The end speed of the ducking effect.

6.9.4.16 float SoundManager.duckStartSpeed = .1f [static]

The start speed of the ducking effect.

6.9.4.17 const string SoundManager.EDIT = "edit"

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.18 int SoundManager.groupAddIndex = 0

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.19 bool SoundManager.helpOn = false

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.20 const string SoundManager.HIDE = "hide"

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.21 bool SoundManager.ignoreLevelLoad = false

Set this if you wish to ignore the level loading functionality.

6.9.4.22 bool SoundManager.isPaused = false

Whether sound is paused.

6.9.4.23 bool SoundManager.movingOnFromSong = false

Whether the background music is being forced to move on to the next song.

It is recommended to not modify this value.

6.9.4.24 bool SoundManager.offTheBGM = false

Turn off the background music.

6.9.4.25 bool SoundManager.offTheSFX = false

Turn off the SFX.

6.9.4.26 SongCallBack SoundManager.OnCrossInBegin

Called when crossfade in begins.

6.9.4.27 SongCallBack SoundManager.OnCrossOutBegin

Called when crossfade out begins.

6.9.4.28 SongCallBack SoundManager.OnSongBegin

Called when a song begins, AFTER crossfade in ends.

6.9.4.29 SongCallBack SoundManager.OnSongEnd

Called when a song ends, AFTER crossfade out ends as well.

6.9.4.30 string SoundManager.resourcesPath = "Sounds/SFX"

Path to folder where SFX are held in resources

6.9.4.31 Dictionary<AudioSource, SongCallBack> SoundManager.runOnEndFunctions = new Dictionary<AudioSource, SongCallBack>()

Dictionary of sfx with runonendfunctions

6.9.4.32 List<float> SoundManager.sfxBaseVolumes = new List<float>()

The sfx base volumes.

At runtime, this is NOT used, so don't modify this.

6.9.4.33 List<SFXGroup> SoundManager.sfxGroups = new List<SFXGroup>()

List of SFXGroups.

At runtime, this is NOT used, so don't modify this.

6.9.4.34 float SoundManager.SFXObjectLifetime = 10f

The SFX object lifetime for objects outside of the prepool amount.

6.9.4.35 List<float> SoundManager.sfxPitchVariations = new List<float>()

The sfx pitch variations.

At runtime, this is NOT used, so don't modify this.

6.9.4.36 List<int> SoundManager.sfxPrePoolAmounts = new List<int>()

The sfx pre pool amounts.

At runtime, this is NOT used, so don't modify this.

6.9.4.37 List<float> SoundManager.sfxVolumeVariations = new List<float>()

The sfx volume variations.

At runtime, this is NOT used, so don't modify this.

6.9.4.38 bool SoundManager.showAdd = true

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.39 bool SoundManager.showAsGrouped = false

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.40 bool SoundManager.showDebug = true

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.41 bool SoundManager.showDev = true

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.42 bool SoundManager.showInfo = true

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.43 bool SoundManager.showList = true

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.44 bool SoundManager.showSFX = true

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.45 List<bool> SoundManager.showSFXDetails = new List<bool>()

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.46 Hashtable SoundManager.songStatus = new Hashtable()

Editor variable – IGNORE AND DO NOT MODIFY

6.9.4.47 List<SoundConnection> SoundManager.soundConnections = new List<SoundConnection>()

The sound connections.

6.9.4.48 const int SoundManager.SOUNDMANAGER_FALSE = -1

6.9.4.49 List<AudioClip> SoundManager.storedSFXs = new List<AudioClip>()

List of local AudioClip SFXs added in inspector or through SaveSFX()

6.9.4.50 List<GameObject> SoundManager.unOwnedSFXObjects = new List<GameObject>()

List of other unowned gameobjects with SFX attached.

6.9.4.51 const string SoundManager.VIEW = "view"

Editor variable – IGNORE AND DO NOT MODIFY

6.9.5 Property Documentation

6.9.5.1 SoundManager SoundManager.Instance [static],[get],[set]

Gets or sets the instance.

The instance.

6.9.5.2 float SoundManager.maxMusicVolume [get],[set]

Gets or sets the max music volume.

The max music volume.

6.9.5.3 float SoundManager.maxSFXVolume [get],[set]

Gets or sets the max SFX volume.

The max SFX volume.

6.9.5.4 float SoundManager.maxVolume [get],[set]

Gets or sets the max volume.

The max volume.

6.9.5.5 bool SoundManager.muted [get], [set]

Gets or sets a value indicating whether this SoundManager is muted.

`true` if muted; otherwise, `false`.

6.9.5.6 bool SoundManager.mutedMusic [get], [set]

Gets or sets a value indicating whether this SoundManager muted music.

`true` if muted music; otherwise, `false`.

6.9.5.7 bool SoundManager.mutedSFX [get], [set]

Gets or sets a value indicating whether this SoundManager muted SFX.

`true` if muted SFX; otherwise, `false`.

6.9.5.8 float SoundManager.pitchSFX [get], [set]

Gets or sets the SFX pitch.

The SFX pitch.

6.9.5.9 bool SoundManager.viewAll [get], [set]

Editor variable – IGNORE AND DO NOT MODIFY

6.9.5.10 float SoundManager.volume1 [get], [set]

Gets or sets the volume of BGM track 1.

The volume.

6.9.5.11 float SoundManager.volume2 [get], [set]

Gets or sets the volume of BGM track 2.

The volume.

6.9.5.12 float SoundManager.volumeSFX [get], [set]

Gets or sets the SFX volume.

The SFX volume.

The documentation for this class was generated from the following files:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager.cs
- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Editor.cs
- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Essentials.cs
- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Variables_Music.cs
- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Variables_SFX.cs

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Internal.cs
- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_SFX.cs
- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_SFX_Internal.cs

6.10 SoundManagerTools Class Reference

Some useful extension functions to use in the SoundManager.

Static Public Member Functions

- static void Shuffle< T > (ref List< T > theList)

  Shuffle the specified list.
- static void ShuffleTwo< T, K > (ref List< T > theList, ref List< K > otherList)

  Shuffles two lists together identically.
- static void make2D (ref AudioSource theAudioSource)

  Make an `AudioSource` *play any clip like it's 2D.*
- static void make3D (ref AudioSource theAudioSource)

  Make an `AudioSource` *play any clip like it's 3D.*
- static float VaryWithRestrictions (this float theFloat, float variance, float minimum=0f, float maximum=1f)

  Vary a float with restrictions.
- static float Vary (this float theFloat, float variance)

  Vary a float.
- static FieldInfo[] GetAllFieldInfos (this Type type)

  Returns all instance fields on an object, including inherited fields `http://stackoverflow.com/a/1155549/154165`

6.10.1 Detailed Description

Some useful extension functions to use in the SoundManager.

6.10.2 Member Function Documentation

6.10.2.1 static FieldInfo [] SoundManagerTools.GetAllFieldInfos (this Type *type*) [static]

Returns all instance fields on an object, including inherited fields `http://stackoverflow.com/a/1155549/154165`

6.10.2.2 static void SoundManagerTools.make2D (ref AudioSource *theAudioSource*) [static]

Make an `AudioSource` play any clip like it's 2D.

Parameters

theAudioSource	The audio source.

6.10.2.3 static void SoundManagerTools.make3D (ref AudioSource *theAudioSource*) [static]

Make an `AudioSource` play any clip like it's 3D.

Parameters

theAudioSource	The audio source.

6.10.2.4 static void SoundManagerTools.Shuffle< T > (ref List< T > *theList*) [static]

Shuffle the specified list.

Parameters

theList	The list.

Template Parameters

T	The 1st type parameter.

6.10.2.5 static void SoundManagerTools.ShuffleTwo< T, K > (ref List< T > *theList,* ref List< K > *otherList*) [static]

Shuffles two lists together identically.

Parameters

theList	The list.
otherList	The second list.

Template Parameters

T	The 1st type parameter.
K	The 2nd type parameter.

6.10.2.6 static float SoundManagerTools.Vary (this float *theFloat,* float *variance*) [static]

Vary a float.

Returns

The varied float.

Parameters

theFloat	The float.
variance	Variance.

6.10.2.7 static float SoundManagerTools.VaryWithRestrictions (this float *theFloat,* float *variance,* float *minimum* = 0f, float *maximum* = 1f) [static]

Vary a float with restrictions.

Returns

The varied float.

Parameters

theFloat	The float.
variance	Variance.
minimum	Minimum value.
maximum	Maximum value.

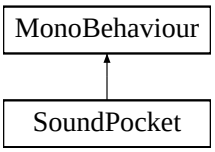
The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Extensions/SoundManagerTools.cs

6.11 SoundPocket Class Reference

Pockets modify the SFX on the SoundManager whenever they are enabled.

Inheritance diagram for SoundPocket:



Public Member Functions

- void Setup ()

  Setup this instance.
- void DestroyMe ()

  Destroys this instance when finished.

Public Attributes

- string pocketName = "Pocket"

  Name of the SoundPocket.
- SoundPocketType pocketType = SoundPocketType.Additive

  Determines how this pocket will be added to the SoundManager once the SoundPocket is loaded.
- List< AudioClip > pocketClips = new List<AudioClip>()

  The audio clips in the SoundPocket.
- List< string > sfxGroups = new List<string>()

  These are possible group names for SFXs to be applied to.
- List< string > clipToGroupKeys = new List<string>()

  Editor variable – IGNORE AND DO NOT MODIFY
- List< string > clipToGroupValues = new List<string>()

  Editor variable – IGNORE AND DO NOT MODIFY
- List< int > sfxPrePoolAmounts = new List<int>()

  The sfx prepool amounts.
- List< float > sfxBaseVolumes = new List<float>()

  The sfx base volumes.
- List< float > sfxVolumeVariations = new List<float>()

  The sfx volume variations.
- List< float > sfxPitchVariations = new List<float>()

The sfx pitch variations.

- bool showAsGrouped = false

  Editor variable – IGNORE AND DO NOT MODIFY
- List< bool > showSFXDetails = new List<bool>()

  Editor variable – IGNORE AND DO NOT MODIFY
- int groupAddIndex = 0

  Editor variable – IGNORE AND DO NOT MODIFY
- int autoPrepoolAmount = 0

  Editor variable – IGNORE AND DO NOT MODIFY
- float autoBaseVolume = 1f

  Editor variable – IGNORE AND DO NOT MODIFY
- float autoVolumeVariation = 0f

  Editor variable – IGNORE AND DO NOT MODIFY
- float autoPitchVariation = 0f

  Editor variable – IGNORE AND DO NOT MODIFY

6.11.1 Detailed Description

Pockets modify the SFX on the SoundManager whenever they are enabled.

They'll automatically destroy themselves afterwards. Add this as a component to any GameObject you want. It is recommended to handle anything here strictly in the editor.

6.11.2 Member Function Documentation

6.11.2.1 void SoundPocket.DestroyMe ( )

Destroys this instance when finished.

Will destroy the GameObject too if it's the only component on it.

6.11.2.2 void SoundPocket.Setup ( )

Setup this instance.

6.11.3 Member Data Documentation

6.11.3.1 float SoundPocket.autoBaseVolume = 1f

Editor variable – IGNORE AND DO NOT MODIFY

6.11.3.2 float SoundPocket.autoPitchVariation = 0f

Editor variable – IGNORE AND DO NOT MODIFY

6.11.3.3 int SoundPocket.autoPrepoolAmount = 0

Editor variable – IGNORE AND DO NOT MODIFY

6.11.3.4 float SoundPocket.autoVolumeVariation = 0f

Editor variable – IGNORE AND DO NOT MODIFY

6.11.3.5 List<string> SoundPocket.clipToGroupKeys = new List<string>()

Editor variable – IGNORE AND DO NOT MODIFY

6.11.3.6 List<string> SoundPocket.clipToGroupValues = new List<string>()

Editor variable – IGNORE AND DO NOT MODIFY

6.11.3.7 int SoundPocket.groupAddIndex = 0

Editor variable – IGNORE AND DO NOT MODIFY

6.11.3.8 List<AudioClip> SoundPocket.pocketClips = new List<AudioClip>()

The audio clips in the SoundPocket.

6.11.3.9 string SoundPocket.pocketName = "Pocket"

Name of the SoundPocket.

If a SoundPocket already exists on the SoundManager, it will not be readded.

6.11.3.10 SoundPocketType SoundPocket.pocketType = SoundPocketType.Additive

Determines how this pocket will be added to the SoundManager once the SoundPocket is loaded.

If additive, these SFX will be added to the SoundManager. If subtractive, the SFX currently on the SoundManager will be removed before these are added.

6.11.3.11 List<float> SoundPocket.sfxBaseVolumes = new List<float>()

The sfx base volumes.

6.11.3.12 List<string> SoundPocket.sfxGroups = new List<string>()

These are possible group names for SFXs to be applied to.

If the group exists on the SoundManager, it'll be added to that group. Otherwise, a new group will be created.

6.11.3.13 List<float> SoundPocket.sfxPitchVariations = new List<float>()

The sfx pitch variations.

6.11.3.14 List<int> SoundPocket.sfxPrePoolAmounts = new List<int>()

The sfx prepool amounts.

6.11.3.15 List<float> SoundPocket.sfxVolumeVariations = new List<float>()

The sfx volume variations.

6.11.3.16 bool SoundPocket.showAsGrouped = false

Editor variable – IGNORE AND DO NOT MODIFY

6.11.3.17 List<bool> SoundPocket.showSFXDetails = new List<bool>()

Editor variable – IGNORE AND DO NOT MODIFY

The documentation for this class was generated from the following file:

- C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/SoundPocket.cs

Chapter 7

File Documentation

7.1 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/AudioSourcePro.cs File Reference

Classes

- class AudioSourcePro

  SoundManagerPro's version of an AudioSource with additional features.

Namespaces

- package antilunchbox

Enumerations

- enum antilunchbox.ClipType { antilunchbox.ClipType.AudioClip, antilunchbox.ClipType.ClipFromSoundManager, antilunchbox.ClipType.ClipFromGroup }

  Specifies how to load `AudioClip`s.

- enum antilunchbox.AudioSourceAction { antilunchbox.AudioSourceAction.None, antilunchbox.AudioSourceAction.Play, antilunchbox.AudioSourceAction.PlayLoop, antilunchbox.AudioSourceAction.PlayCapped, antilunchbox.AudioSourceAction.Stop }

  Specifies what an AudioSubscription should do when an event is fired.

- enum antilunchbox.AudioSourceStandardEvent { antilunchbox.AudioSourceStandardEvent.OnStart, antilunchbox.AudioSourceStandardEvent.OnVisible, antilunchbox.AudioSourceStandardEvent.OnInvisible, antilunchbox.AudioSourceStandardEvent.OnCollisionEnter, antilunchbox.AudioSourceStandardEvent.OnCollisionExit, antilunchbox.AudioSourceStandardEvent.OnTriggerEnter, antilunchbox.AudioSourceStandardEvent.OnTriggerExit, antilunchbox.AudioSourceStandardEvent.OnMouseEnter, antilunchbox.AudioSourceStandardEvent.OnMouseClick, antilunchbox.AudioSourceStandardEvent.OnEnable, antilunchbox.AudioSourceStandardEvent.OnDisable, antilunchbox.AudioSourceStandardEvent.OnCollisionEnter2D, antilunchbox.AudioSourceStandardEvent.OnCollisionExit2D, antilunchbox.AudioSourceStandardEvent.OnTriggerEnter2D, antilunchbox.AudioSourceStandardEvent.OnTriggerExit2D, antilunchbox.AudioSourceStandardEvent.OnParticleCollision }

  Standard events to bind to that are automatically provided by the Unity Engine.

7.2 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/AudioSubscription.cs File Reference

Classes

- class AudioSubscription

  Class that contains all the data needed to bind an AudioSourceAction to an event.

7.3 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/ProxyEventAttribute.cs File Reference

Classes

- class ProxyEventAttribute

  Proxy event attribute for custom event binding.

7.4 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/SFXGroup.cs File Reference

Classes

- class SFXGroup

  Used to group SFX together with certain attributes to share.

7.5 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/SFXPoolInfo.cs File Reference

Classes

- class SFXPoolInfo

  Contains information on SFX Pools.

7.6 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/SoundConnection.cs File Reference

Classes

- class SoundConnection

  Contains information on sound connections, meant for background music.

7.7 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Classes/SoundPocket.cs File Reference

Classes

- class SoundPocket

Pockets modify the SFX on the SoundManager whenever they are enabled.

Enumerations

- enum SoundPocketType { SoundPocketType.Additive, SoundPocketType.Subtractive }

  Determines how this pocket will be added to the SoundManager once the SoundPocket is loaded.

7.7.1 Enumeration Type Documentation

7.7.1.1 enum SoundPocketType

Determines how this pocket will be added to the SoundManager once the SoundPocket is loaded.

If additive, these SFX will be added to the SoundManager. If subtractive, the SFX currently on the SoundManager will be removed before these are added.

Enumerator

Additive

Subtractive

7.8 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Extensions/AudioSourceTools.cs File Reference

Classes

- class AudioSourceTools

  Extending SoundManager SFX functions to regular `AudioSource`s.

7.9 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Extensions/SoundManagerTools.cs File Reference

Classes

- class SoundManagerTools

  Some useful extension functions to use in the SoundManager.

7.10 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/Singleton.cs File Reference

Classes

- class antilunchbox.Singleton< T >

  Singleton base class that will cause any inheriting class to create itself when referenced in any way at all.

Namespaces

- package antilunchbox

7.11 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager.cs File Reference

Classes

- class SoundManager

  SoundManager is where most functions will be called from.

Enumerations

- enum SoundDuckingSetting { SoundDuckingSetting.DoNotDuck, SoundDuckingSetting.OnlyDuckSFX, SoundDuckingSetting.OnlyDuckMusic, SoundDuckingSetting.DuckAll }

  Sound ducking setting.

7.11.1 Enumeration Type Documentation

7.11.1.1 enum SoundDuckingSetting

Sound ducking setting.

Can not duck, duck only SFX, duck only Music, or duck everything.

Enumerator

DoNotDuck

OnlyDuckSFX

OnlyDuckMusic

DuckAll

7.12 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Editor.cs File Reference

Classes

- class SoundManager

  SoundManager is where most functions will be called from.

7.13 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Essentials.cs File Reference

Classes

- class SoundManager

  SoundManager is where most functions will be called from.

7.14 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Internal.cs File Reference

Classes

- class SoundManager

  SoundManager is where most functions will be called from.

7.15 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_SFX.cs File Reference

Classes

- class SoundManager

  SoundManager is where most functions will be called from.

7.16 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_SFX_Internal.cs File Reference

Classes

- class SoundManager

  SoundManager is where most functions will be called from.

7.17 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Variables_Music.cs File Reference

Classes

- class SoundManager

  SoundManager is where most functions will be called from.

7.18 C:/Users/Patrick/Documents/Testing4_5/Assets/SoundManagerPro/Scripts/Managers/SoundManager_Variables_SFX.cs File Reference

Classes

- class SoundManager

  SoundManager is where most functions will be called from.