MULTI LISTENER POOLING AUDIO SYSTEM

ALMENARA GAMES

About

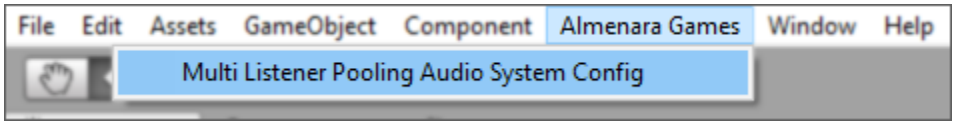
Multi Listener Pooling Audio System (**MLPAS**) is a tool that allows users to implement multiples **Audio Listeners** and manage the audio of their projects in a simple and optimized way using **Channels** and **Audio Objects** easy to use that manage the settings of the different audios that can be played via a pooling system.

Features

- Support multiple **Audio Listeners** with 3D audio positioning using only one voice per **Audio Source**.
- Automatic audio blending between **Audio Listeners** (up to 4 listeners).
- Pooling system using our easy-to-use scriptable object "**Audio Object**" to facilitate the task of creating references to Audio Clips and setting various configurations of Audio Sources such as: Volume, Pitch, Random clips, Looping, Starting Pitch and Volume Alterations, Spatial mode, Mixer Output, and other settings.
- Capacity to Play, Stop, Pause/Unpause, Delay, Mute, Fade In/Out the audios.
- Basic system to occlude the sounds through colliders using raycasting.
- Persistent sounds between scenes.
- Reverb settings per listener.
- Variety of methods for play the sounds according to your needs.

Setup

1. First open the **Multi Listener Pooling Audio System Config** inside the "Almenara Games" menu item**:**



2. Setup the **Multi Listener Pooling Audio System Config** the way you need it.

Sfx Mixer Group	Default SFX Mixer Group Output.
Bgm Mixer Group	Default BGM Mixer Group Output.
Occlude Check	Layer Mask used for check whether or not a collider occludes an occludable sound.
Occlude Multiplier	The higher the value, the less the audio is heard when occluded.
Max Audio Sources	Max pooled Multi Audio Sources.

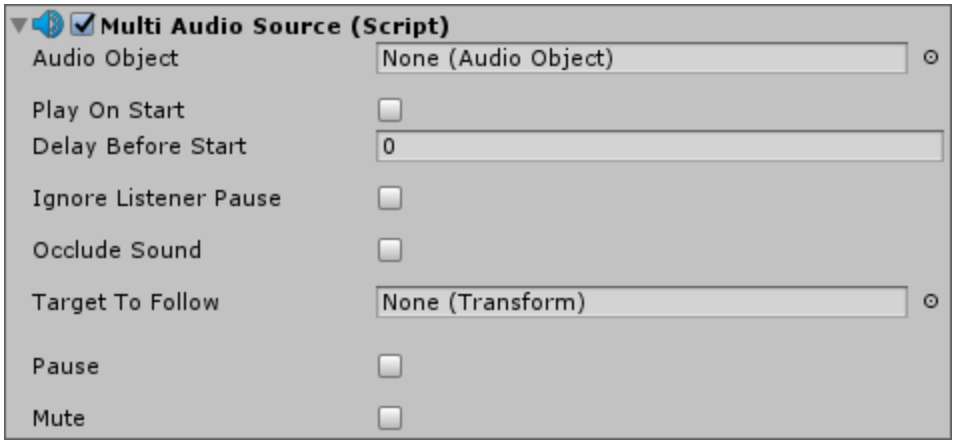
3. You are ready to start using **Multi Listener Pooling Audio System** (**MLPAS**).

Details

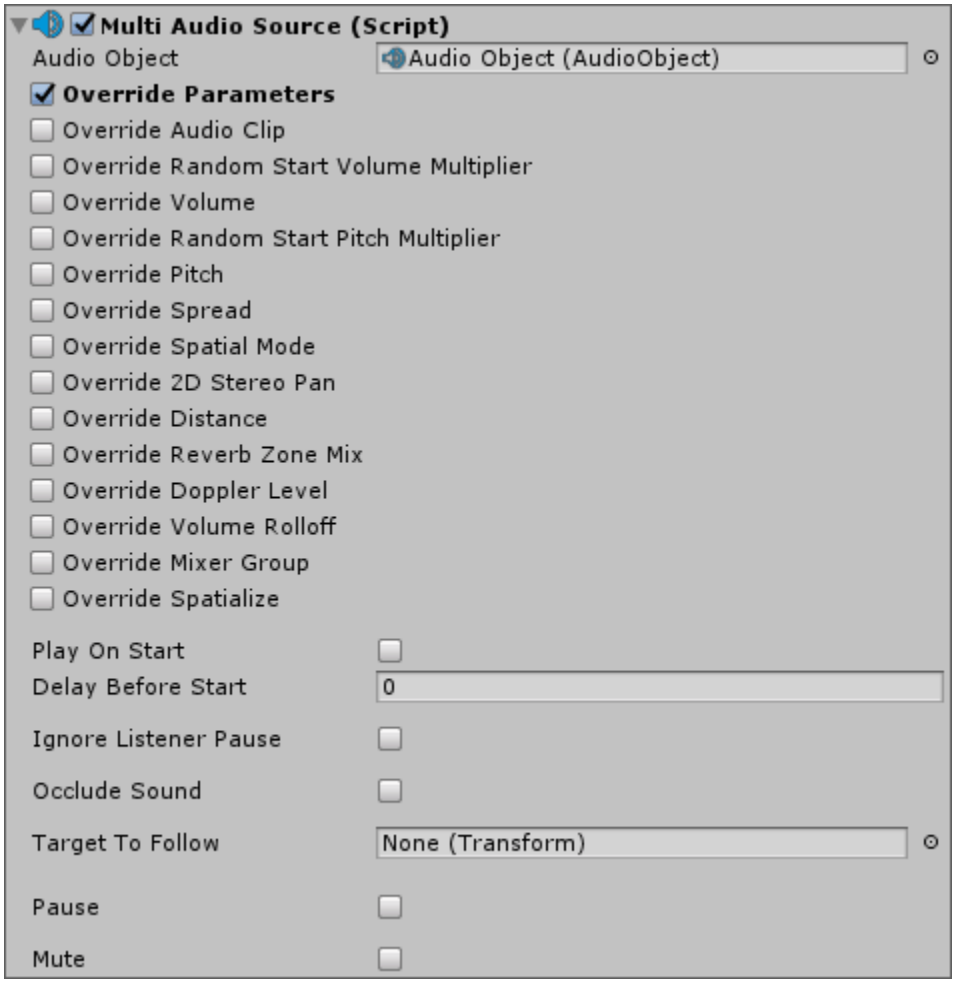
The **Multi Listener Pooling Audio System** (**MLPAS**) works using a custom **Audio Listener** labeled as "**Multi Audio Listener**" and a custom **Audio Source** labeled as "**Multi Audio Source**", these components have some different options compared to the default ones and allows users to manage the audios of their project in a more simple way, additionaly the **MLPAS** makes use of a pooling system based on **AudioObjects** and **Channels** to ensure proper management of the different audios in your project.

Multi Audio Source

The **Multi Audio Source** is a replace of the default **Audio Source**, unlike the default **Audio Source** the **Multi Audio Source** plays **Audio Objects** and only contains properties to override the assigned **Audio Object** properties and some other basic options such as mute, pause or play on start. It was created to be used mainly with the pooling system, however it can also be used individually through the various methods it contains.

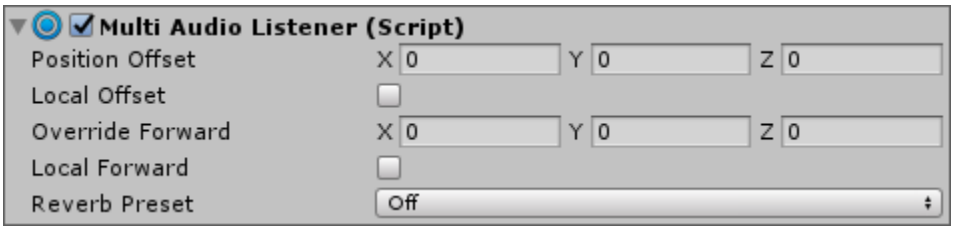


This are the override properties:



Multi Audio Listener

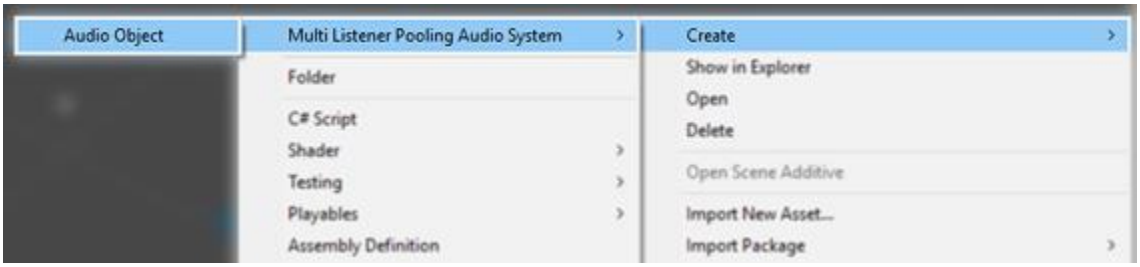
The **Multi Audio Listener** is a replace of the default **Audio Listener** that allows multiple instances of itself in the scene and has some options that makes it more customizable. As you can see below, each **Multi Audio Listener** has its own “Reverb Preset”, this replaces the default reverb zone and reverb filter that includes Unity, so if you want to make for example a cave that have some reverb, you can change the “Reverb Preset” of an individual **Multi Audio Listener** when it enters inside a trigger. (Due to certain limitations, custom presets can’t be used.)



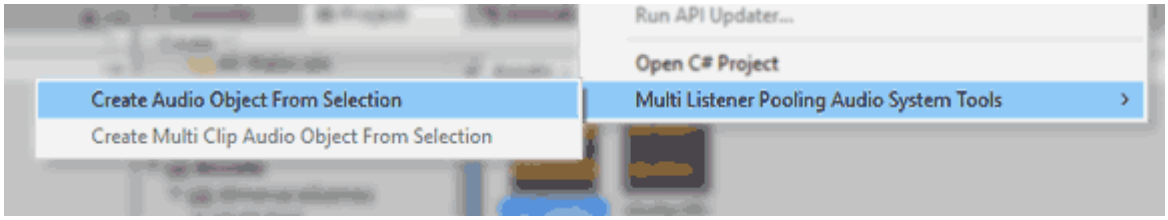
Audio Object

The **Audio Object** is a scriptable object that contains much of the settings of a default **Audio Source** plus some other things like the capacity of assign multiple **Audio Clips** for play a random clip from the list each time is played or applying a customizable random pitch/volume alteration to the audio, the **Audio Object** is like the **Audio Clip** of the **Multi Audio Source** with the difference that all the settings of the source are inside the clip and not in the source itself.

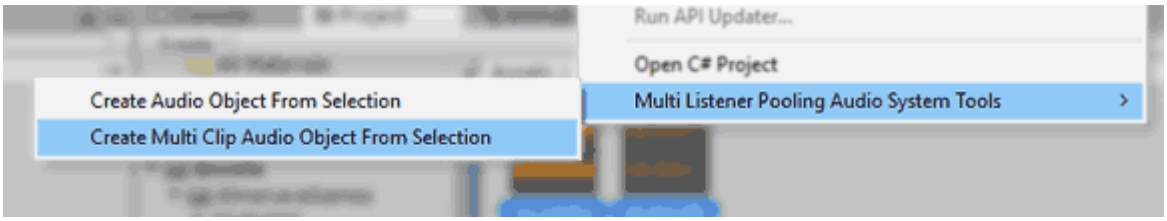
The Audio Objects are created like any other asset in unity.



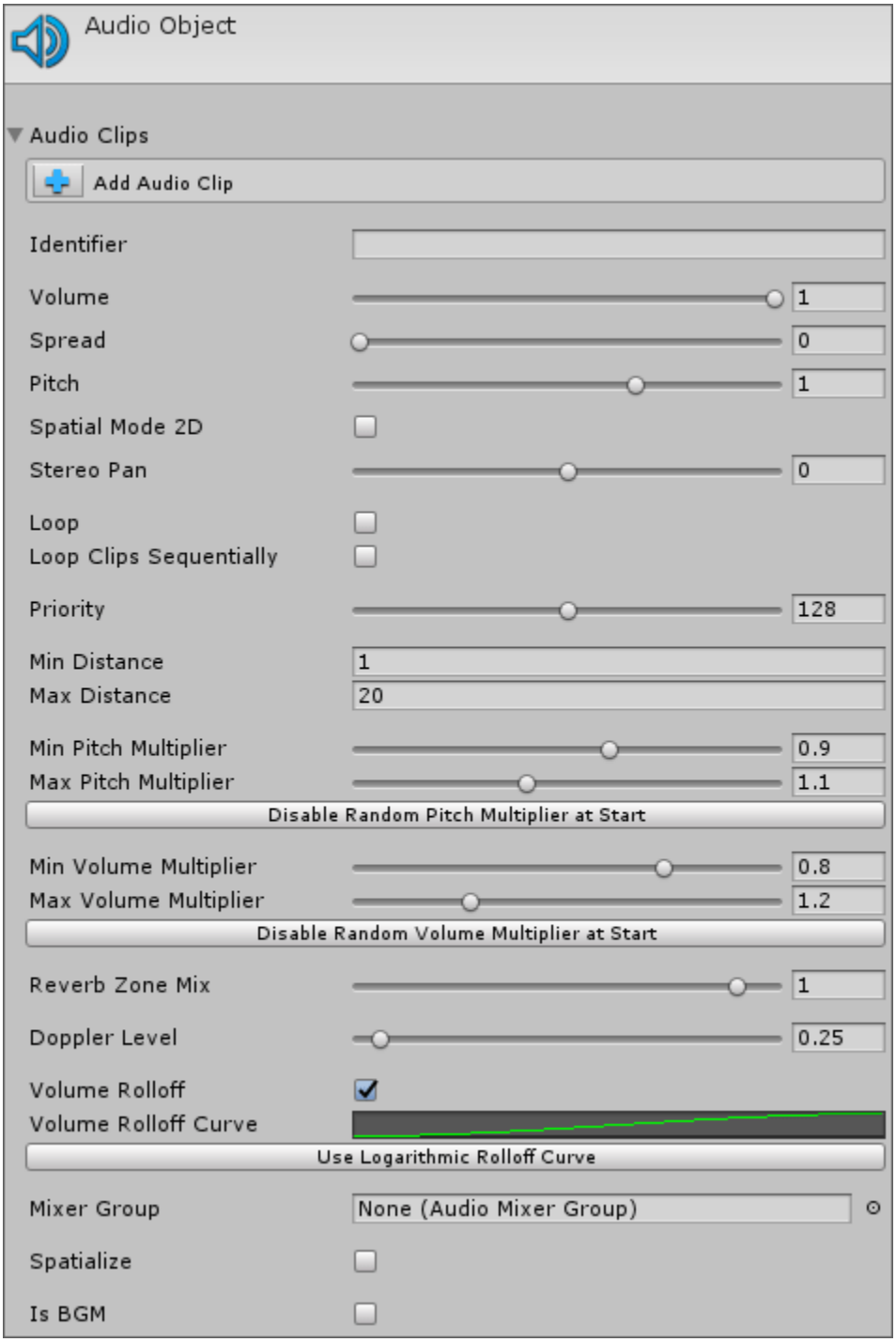
Additionally you can create an **Audio Object** from an **Audio Clip** by right clicking the **Audio Clip** and selecting the option:



Or create a “Multi Clip” **Audio Object** by selecting multiple **Audio Clips** then right clicking one of the **Audio Clips** and selecting the option:



This is how the **Audio Object** looks:



Channels

The channels are used to filter and assign a unique ID to the pooled **Multi Audio Sources**, by this way you can later stop, pause or change some properties of an audio already played. A channel can play only one **Audio Source** at a time, this can be useful if you want to play the voice of a character because this will going to interrupt the last sentence with the new one played avoiding overlapping voices of the same character. All of the audios are played at the channel -1 which is equivalent to not use channels, to use the channels you need to play the audios in a channel between 0 and 2147483647.

Multi Audio Manager

The **Multi Audio Manager** contains a variety of methods for play and manage the sounds according to your needs and is the one that initializes the system using a singleton.

These are some of the methods you will find in the **Multi Audio Manager:**

PlayAudioObject	Plays an **AudioObject**.
StopAudioSource	Stops the specified **MultiAudioSource**.
PauseAudioSource	Pauses/Unpauses the specified **MultiAudioSource**.
FadeInAudioObject	Plays an **AudioObject** with a fade in.

This is how the **Multi Audio Manager** component looks:

Check the DEMO scenes for better understand of the system.

Known Bug:

Sometimes in the Unity Editor the looped audios restarts when the application loses focus, this only happens in the editor because of a bug with the method **OnApplicationFocus()** on the newer Unity versions.

API Details:

For more specific information about the properties and methods of the components of the system, check below:

Index

Audio Object

(Scriptable Object)
AlmenaraGames.AudioObject

Properties

identifier	The identifier to get this **AudioObject**. Remember that the **AudioObject** needs to be inside the "*Resources**Global Audio Objects*" folder in order to be accessed via this identifier.
clips	Gets or sets the clips.
RandomClip	Returns a random clip from the list.
volume	The overall volume of the sound.
spread	The spread of a 3d sound in speaker space.
pitch	The frequency of the sound. Use this to slow down or speed up the sound. A negative pitch value will going to make the sound plays backwards.
spatialMode2D	A 2D sound ignores all types of spatial attenuation, including spatial position and spread.
stereoPan	The 2D Stereo pan. Only for 2D spatial mode.
loop	Is the audio clip looping?
loopClipsSequentially	The source will going to automatically changes the current clip to the next clip on the list at the end of the current loop.
priority	The priority of the sound. Note that a sound with a larger priority value will more likely be stolen by sounds with smaller priority values.
minDistance	Withing the Min Distance, the volume will stay at the loudest possible. Outside of this Min Distance it begins to attenuate.
maxDistance	Max Distance is the distance where the sound is completely inaudible.
minPitchMultiplier	Min random value for multiply the pitch of the sound.
maxPitchMultiplier	Max random value for multiply the pitch of the sound.
minVolumeMultiplier	Min random value for multiply the volume of the sound.

GO BACK TO INDEX

maxVolumeMultiplier	Max random value for multiply the volume of the sound.
reverbZoneMix	The amount by which the signal from the sound will be mixed into the global reverb associated with the Reverb from the listeners. The range from 0 to 1 is linear (like the volume property) while the range from 1 to 1.1 is an extra boost range that allows you to boost the reverberated signal by 10 dB.
dopplerLevel	Specifies how much the pitch is changed based on the relative velocity between the listener and the source.
volumeRolloff	Enables or disables sound attenuation over distance.
volumeRolloffCurve	The volume rolloff curve. Sets how the sound attenuates over distance.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
spatialize	Enables or disables custom spatialization for the source.
isBGM	Is the sound a BGM?

GO BACK TO INDEX

Multi Audio Listener

AlmenaraGames.MultiAudioListener

Properties

LocalForward	Overrides the listener direction in local space.
LocalOffset	Interpret the position offset in local space.
OverrideForward	Overrides the listener direction, useful for fixed camera angle games like Top Downs or 2D Platformers. Use (0, 0, 0) to disable the forward override.
PositionOffset	Sets a position offset to the listener.
RealListener	The listener with the offset and forward applied.
ReverbPreset	This reverb preset will going to affect all nearest **Multi Audio Sources**, useful to activate it using a trigger when entering a cave, etc.

GO BACK TO INDEX

Multi Audio Source

AlmenaraGames.MultiAudioSource

Properties

AudioObject	Gets or sets the **AudioObject**.
NearestListenerBlend	The blend amount of the nearest listener. (Read Only)
AveragePosition	The average position of all the listeners. (Read Only)
Channel	Gets the channel where the **MultiAudioSource** is playing. (Read Only)
TargetToFollow	Gets or sets the the target to follow.
Mute	Mutes the **MultiAudioSource**.
OverrideValues	Use the override values.
RandomVolumeMultiplierOverride	Overrides the use of the random volume multiplier.
RandomPitchMultiplierOverride	Overrides the use of the random pitch multiplier.
AudioClipOverride	Gets or sets the Audio Clip for override the **AudioObject** default clips.
VolumeOverride	Gets or sets the volume for override the **AudioObject** default volume.
PitchOverride	Gets or sets the pitch for override the **AudioObject** default pitch.
SpatialMode2DOverride	Gets or sets the spatial mode for override the **AudioObject** default spatial mode.
StereoPanOverride	Gets or sets the 2D stereo pan for override the **AudioObject** default 2D stereo pan.
SpreadOverride	Gets or sets the spread for override the **AudioObject** default spread.
DopplerLevelOverride	Gets or sets the doppler level for override the **AudioObject** default doppler level.
ReverbZoneMixOverride	Gets or sets the reverb zone mix for override the **AudioObject** default reverb zone mix.
MinDistanceOverride	Gets or sets the min hearable distance for override the **AudioObject** default min hearable distance.
MaxDistanceOverride	Gets or sets the max hearable distance for override the **AudioObject** default max hearable distance.
MixerGroupOverride	Gets or sets the mixer group for override the **AudioObject** default mixer group.

GO BACK TO INDEX

OccludeSound	Occludes the sound if there is a Collider between the source and the listener with one of the **Multi Listener Pooling Audio System Config** *occludeCheck* layers.
Delay	Gets or sets the delay before the **MultiAudioSource** starts playing the **AudioObject** or Audio Clip.
VolumeRolloffOverride	Gets or sets the **MultiAudioSource** volume rolloff for override the **AudioObject** default volume rolloff.
VolumeRolloffCurveOverride	Gets or sets the volume rolloff curve for override the **AudioObject** default volume rolloff curve.
SpatializeOverride	Gets or sets the spatialize value for override the **AudioObject** default spatialize value.
IgnoreListenerPause	Gets or sets a value indicating whether this **MultiAudioSource** ignores the listener pause.
PersistsBetweenScenes	Gets or sets a value indicating whether this **MultiAudioSource** persists between scenes.
LocalPause	Gets or sets a value indicating whether this **MultiAudioSource** is paused.
MaxDistance	Gets the current max hearable distance of this **MultiAudioSource**. (Read Only)
MinDistance	Gets the current min hearable distance of this **MultiAudioSource**. (Read Only)
Loop	Gets a value indicating whether this **MultiAudioSource** is looping. (Read Only)

Methods

Play	Plays the **MultiAudioSource**.
PlayOverride	Plays the **MultiAudioSource** using its **AudioObject** but with another Audio Clip.
PlayFadeIn	Plays the MultiAudioSource with a fade in specified in seconds.
PlayFadeInOverride	Plays the **MultiAudioSource** using its AudioObject but with another Audio Clip and a fade in specified in seconds.
Stop	Stops playing the **MultiAudioSource**.
StopDelayed	Stops playing the **MultiAudioSource** with a delay specified in seconds.
FadeOut	Stops playing the **MultiAudioSource** using a fade out specified in seconds.
FadeOutDelayed	Stops playing the **MultiAudioSource** with a delay specified in seconds using a fade out.

GO BACK TO INDEX

InterruptDelayedStop	Interrupts the delayed stop.

GO BACK TO INDEX

Methods Explained

Play

public **void** Play();

public **void** Play(**int** _channel);

public **void** Play(**Transform** _targetToFollow);

public **void** Play(**int** _channel,**Transform** _targetToFollow);

public **void** Play(**float** _delay);

public **void** Play(**int** _channel,**float** _delay);

public **void** Play(**Transform** _targetToFollow,**float** _delay);

public **void** Play(**int** _channel,**Transform** _targetToFollow,**float** _delay);

Parameters

_channel	The **Channel** the audio will play at.
_delay	The delay specified in seconds to start playing the audio.
_targetToFollow	The target that the **MultiAudioSource** will go to follow.

Example

This example plays the **MultiAudioSource** "audioSfx" at **Channel** 3.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public MultiAudioSource audioSfx;

        void Start ()
        {

        audioSfx.Play(3);

        }
}
```

GO BACK TO THE METHODS

PlayOverride

public **void** PlayOverride(**AudioClip** audioClipOverride);

public **void** PlayOverride(**AudioClip** audioClipOverride,**int** _channel);

public **void** PlayOverride(**AudioClip** audioClipOverride,**Transform** _targetToFollow);

public **void** PlayOverride(**AudioClip** audioClipOverride,**int** _channel,**Transform** _targetToFollow);

public **void** PlayOverride(**AudioClip** audioClipOverride,**float** _delay);

public **void** PlayOverride(**AudioClip** audioClipOverride,**int** _channel,**float** _delay);

public **void** PlayOverride(**AudioClip** audioClipOverride,**Transform** _targetToFollow,**float** _delay);

public **void** PlayOverride(**AudioClip** audioClipOverride,**int** _channel,
Transform _targetToFollow,**float** _delay);

Parameters

audioClipOverride	The AudioClip to override the **AudioObject** clip**.**
_targetToFollow	The target that the **MultiAudioSource** will go to follow.
_delay	The delay specified in seconds to start playing the audio.
_channel	The **Channel** the audio will play at.

Example

This example plays the **MultiAudioSource** “audioSfx” at **Channel** 3 with another Audio Clip “newAudio” and a delay of 2 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public MultiAudioSource audioSfx;
public AudioClip newAudio;

        void Start ()
        {

        audioSfx.PlayOverride(newAudio,3,2f);

        }
}
```

GO BACK TO THE METHODS

PlayFadeIn

public **void** PlayFadeIn(**float** fadeInTime);

public **void** PlayFadeIn(**float** fadeInTime,**int** _channel);

public **void** PlayFadeIn(**float** fadeInTime,**Transform** _targetToFollow);

public **void** PlayFadeIn(**float** fadeInTime,**int** _channel,**Transform** _targetToFollow);

public **void** PlayFadeIn(**float** fadeInTime,**float** _delay);

public **void** PlayFadeIn(**float** fadeInTime,**int** _channel,**float** _delay);

public **void** PlayFadeIn(**float** fadeInTime,**Transform** _targetToFollow,**float** _delay);

public **void** PlayFadeIn(**float** fadeInTime,**int** _channel,**Transform** _targetToFollow,**float** _delay);

Parameters

_channel	The **Channel** the audio will play at.
_delay	The delay specified in seconds to start playing the audio.
fadeInTime	The time in seconds to complete the fade in of the sound.
_targetToFollow	The target that the **MultiAudioSource** will go to follow.

Example

This example plays the **MultiAudioSource** "audioSfx" with a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public MultiAudioSource audioSfx;

        void Start ()
        {

        audioSfx.PlayFadeIn(0.5f);

        }
}
```

GO BACK TO THE METHODS

PlayFadeInOverride

public **void** PlayFadeInOverride(**float** fadeInTime,**AudioClip** audioClipOverride);

public **void** PlayFadeInOverride(**float** fadeInTime,**AudioClip** audioClipOverride,**int** _channel=-1);

public **void** PlayFadeInOverride(**float** fadeInTime,**AudioClip** audioClipOverride,
Transform _targetToFollow=**NULL**);

public **void** PlayFadeInOverride(**float** fadeInTime,**AudioClip** audioClipOverride,
int _channel=-1,**Transform** _targetToFollow=**NULL**);

public **void** PlayFadeInOverride(**float** fadeInTime,**AudioClip** audioClipOverride,**float** _delay);

public **void** PlayFadeInOverride(**float** fadeInTime,**AudioClip** audioClipOverride,
int _channel=-1,float _delay=0f);

public **void** PlayFadeInOverride(**float** fadeInTime,**AudioClip** audioClipOverride,
Transform _targetToFollow=**NULL**,**float** _delay=0f);

public **void** PlayFadeInOverride(**float** fadeInTime,**AudioClip** audioClipOverride,
int _channel=-1,**Transform** _targetToFollow=**NULL**,**float** _delay=0f);

Parameters

audioClipOverride	The AudioClip to override the **AudioObject** clip**.**
fadeInTime	The time in seconds to complete the fade in of the sound.
_targetToFollow	The target that the **MultiAudioSource** will go to follow.
_delay	The delay specified in seconds to start playing the audio.
_channel	The **Channel** the audio will play at.

Example

This example plays the **MultiAudioSource** “audioSfx” with an another Audio Clip “newAudio” with a fade in of 1 second.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public MultiAudioSource audioSfx;
public AudioClip newAudio;

        void Start ()
        {

        audioSfx.PlayFadeInOverride(newAudio,1f);

        }
}
```

GO BACK TO THE METHODS

Stop

public **void** Stop(**bool** disableObject=true);

Parameters

disableObject	Disable the Game Object where the **MultiAudioSource** is attached.

StopDelayed

public **void** StopDelayed(**float** _delay,**bool** disableObject=true);

Parameters

disableObject	Disable the Game Object where the **MultiAudioSource** is attached.
_delay	Delay time specified in seconds for stop the **MultiAudioSource**.

FadeOut

public **void** FadeOut(**float** _fadeOutTime=1f,**bool** disableObject=true);

Parameters

disableObject	Disable the Game Object where the **MultiAudioSource** is attached.
_ fadeOutTime	Fade out time.

FadeOutDelayed

public **void** FadeOutDelayed(**float** _delay,**float** _fadeOutTime=1f,**bool** disableObject=true);

Parameters

disableObject	Disable the Game Object where the **MultiAudioSource** is attached.
_delay	Delay time specified in seconds for stop the **MultiAudioSource**.
_ fadeOutTime	Fade out time.

InterruptDelayedStop

public **void** InterruptDelayedStop();

GO BACK TO THE METHODS

Multi Audio Manager

AlmenaraGames.MultiAudioManager

Properties

Paused	Sets the pause state of all of the listeners.

Methods

PlayAudioObject	Plays an **AudioObject**.
PlayAudioObjectByIdentifier	Plays a Global **AudioObject** by its identifier.
PlayAudioObjectOverride	Plays an **AudioObject** with another Audio Clip.
PlayAudioObjectOverrideByIdentifier	Plays a Global **AudioObject** by its identifier with another Audio Clip.
PlayDelayedAudioObject	Plays an **AudioObject** with a delay specified in seconds.
PlayDelayedAudioObjectByIdentifier	Plays a Global **AudioObject** by its identifier with a delay specified in seconds.
PlayDelayedAudioObjectOverride	Plays an **AudioObject** with another Audio Clip and a delay specified in seconds.
PlayDelayedAudioObjectOverrideByIdentifier	Plays a Global **AudioObject** by its identifier with another Audio Clip and a delay specified in seconds.
FadeInAudioObject	Plays an **AudioObject** with a fade in.
FadeInAudioObjectByIdentifier	Plays **a Global AudioObject** by its identifier with a fade in.
FadeInAudioObjectOverride	Plays an **AudioObject** with a fade in using another Audio Clip.
FadeInAudioObjectOverrideByIdentifier	Plays a Global **AudioObject** by its identifier with a fade in using another Audio Clip.
FadeInDelayedAudioObject	Plays an **AudioObject** with a fade in and a delay specified in seconds.
FadeInDelayedAudioObjectByIdentifier	Plays a Global **AudioObject** by its identifier with a fade in and a delay specified in seconds.
FadeInDelayedAudioObjectOverride	Plays an **AudioObject** with a fade in using another Audio Clip with a delay specified in seconds.
FadeInDelayedAudioObjectOverrideByIdentifier	Plays a Global **AudioObject** by its identifier with a fade in using another Audio Clip with a delay specified in seconds.

GO BACK TO INDEX

StopAllAudioSources	Stops all playing Multi Audio Sources.
StopAllBGMAudios	Stops all Multi Audio Sources marked as BGM audios.
StopAllLoopedAudioSources	Stops all looped Multi Audio Sources.
StopAllNonBGMAudios	Stops all Multi Audio Sources not marked as BGM audios.
StopAllPersistentAudioSources	Stops all persistent Multi Audio Sources.
StopAudioSource	Stops the specified **MultiAudioSource**.
PauseAudioSource	Pauses/Unpauses the specified **MultiAudioSource**.
FadeOutAudioSource	Stops the specified **MultiAudioSource** with a fade out.
GetAudioObjectByIdentifier	Gets the AudioObject with the specific identifier.
GetAudioSourceAtChannel	Gets the **MultiAudioSource** played at the specified Channel (returns **NULL** if there is no MultiAudioSource playing at the specified Channel).

GO BACK TO INDEX

Methods Explained

StopAllAudioSources

public static **void** StopAllAudioSources();

Example

This example stops all **Multi Audio Sources**.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

        void Start ()
        {

        MultiAudioManager.StopAllAudioSources();

        }
}
```

StopAllBGMAudios

public static **void** StopAllBGMAudios();

Example

This example stops all **Multi Audio Sources** marked as BGM.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

        void Start ()
        {

        MultiAudioManager.StopAllBGMAudios();

        }
}
```

GO BACK TO THE METHODS

StopAllNonBGMAudios

public static **void** StopAllNonBGMAudios();

Example

This example stops all **Multi Audio Sources** not marked as BGM.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

        void Start ()
        {

        MultiAudioManager.StopAllNonBGMAudios();

        }
}
```

StopAllLoopedAudioSources

public static **void** StopAllLoopedAudioSources();

Example

This example stops all looped **Multi Audio Sources**.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

        void Start ()
        {

        MultiAudioManager.StopAllLoopedAudioSources();

        }
}
```

GO BACK TO THE METHODS

StopAllPersistentAudioSources

public static **void** StopAllPersistentAudioSources();

Example

This example stops all **Multi Audio Sources** marked as persistent.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

        void Start ()
        {

        MultiAudioManager.StopAllPersistentAudioSources();

        }
}
```

GetAudioSourceAtChannel

public static **MultiAudioSource** GetAudioSourceAtChannel(**int** _channel);

Parameters

_channel	The **Channel** where the **MultiAudioSource** is playing.

Returns

NULL if no **MultiAudioSource** is playing at the specified Channel otherwise returns the **MultiAudioSource** playing at the specified Channel.

GetAudioObjectByIdentifier

public static **AudioObject** GetAudioObjectByIdentifier(**string** identifier);

Parameters

identifier	The Identifier to find the Global **AudioObject**.

Returns

The finded Global **AudioObject**.

GO BACK TO THE METHODS

FadeOutAudioSource

public static **void** FadeOutAudioSource(**int** _channel,**float** fadeOutTime=1f,**bool** disableObject=true);

public static **void** FadeOutAudioSource(**MultiAudioSource** audioSource,**float** fadeOutTime=1f,**bool** disableObject=true);

Parameters

audioSource	The **MultiAudioSource** to fade out.
_channel	The **Channel** to get the **MultiAudioSource** to fade out.
fadeOutTime	The time specified in seconds to fade out the **MultiAudioSource.**
disableObject	Disable the Game Object where the **MultiAudioSource** is attached**.**

Example

This example stops the **MultiAudioSource** played at **Channel** 5 using a fade out of 1 second**.**

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

        void Start ()
        {

        MultiAudioManager.FadeOutAudioSource(5, 1f);

        }
}
```

GO BACK TO THE METHODS

StopAudioSource

public static **void** StopAudioSource(**int** _channel,**bool** disableObject=true);

public static **void** StopAudioSource(**MultiAudioSource** audioSource,**bool** disableObject=true);

Parameters

audioSource	The **MultiAudioSource** to fade out.
_channel	The **Channel** to get the **MultiAudioSource** to fade out.
disableObject	Disable the Game Object where the **MultiAudioSource** is attached**.**

Example

This example stops the **MultiAudioSource** played at **Channel** 2**.**

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

        void Start ()
        {

        MultiAudioManager.StopAudioSource(2);

        }
}
```

GO BACK TO THE METHODS

PauseAudioSource

public static **void** PauseAudioSource(**int** _channel,**bool** pause=true);

public static **void** PauseAudioSource(**MultiAudioSource** audioSource,**bool** pause=true);

Parameters

_channel	The **Channel** to get the **MultiAudioSource** to fade out.
pause	If set to **true** pauses the **MultiAudioSource**.

Example

This example pauses the **MultiAudioSource** played at **Channel** 4**.**

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

        void Start ()
        {

        MultiAudioManager.PauseAudioSource(4);

        }
}
```

GO BACK TO THE METHODS

PlayAudioObject

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**int** channel,**Transform** targetToFollow);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**Transform** targetToFollow);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**Vector3** position);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**int** channel,**Vector3** position);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**int** channel,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObject(**AudioObject** audioObject,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

GO BACK TO THE METHODS

Parameters

audioObject	The **AudioObject** to play.
channel	The channel the audio will play at.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays the **AudioObject** "audioToPlay" at the position "pos".

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public AudioObject audioToPlay;
public Vector3 pos;

        void Start ()
        {

        MultiAudioManager.PlayAudioObject(audioToPlay, pos);

        }
}
```

GO BACK TO THE METHODS

PlayAudioObjectByIdentifier

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**Transform** targetToFollow);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**Vector3** position);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Vector3** position);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

GO BACK TO THE METHODS

Parameters

audioObjectIdentifier	The identifier of the Global **AudioObject**.
channel	The channel the audio will play at.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays **a Global AudioObject** by its identifier “test audio object” at the position “pos” in the **Channel** 2.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public Vector3 pos;

        void Start ()
        {

        MultiAudioManager.PlayAudioObjectByIdentifier(“test audio object”, 2, pos);

        }
}
```

GO BACK TO THE METHODS

PlayAudioObjectOverride

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Transform** targetToFollow);

public static **MultiAudioSource** PlayAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**Transform** targetToFollow);

public static **MultiAudioSource** PlayAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**Vector3** position);

public static **MultiAudioSource** PlayAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**int** channel,**Vector3** position);

public static **MultiAudioSource** PlayAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Vector3** position, **bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

GO BACK TO THE METHODS

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

Parameters

audioClipOverride	**The AudioClip to override the** AudioObject **clip.**
audioObject	**The** AudioObject **to play.**
channel	The channel the audio will play at.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays the **AudioObject** "audioToPlay" with an another **AudioClip** "newAudio" and makes that its pooled **MultiAudioSource** follows the transform "target".

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public AudioObject audioToPlay;
public AudioClip newAudio;
public Transform target;

        void Start ()
        {

        MultiAudioManager.PlayAudioObjectOverride(newAudio, audioToPlay, target);

        }
}
```

GO BACK TO THE METHODS

PlayAudioObjectOverrideByIdentifier

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Transform** targetToFollow);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Vector3** position);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Vector3** position);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

GO BACK TO THE METHODS

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

Parameters

audioClipOverride	**The AudioClip to override the** AudioObject **clip.**
audioObjectIdentifier	**The identifier of the Global** AudioObject**.**
channel	The channel the audio will play at.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays a Global **AudioObject** by its identifier “test audio object” with an another **AudioClip** “newAudio” at the position “pos” in the **Channel** 2.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public Vector3 pos;
public AudioClip newAudio;

        void Start ()
        {

        MultiAudioManager.PlayAudioObjectOverrideByIdentifier(newAudio, “test audio
        object”, 2, pos);

        }
}
```

GO BACK TO THE METHODS

PlayDelayedAudioObject

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Transform** targetToFollow);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Vector3** position);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

GO BACK TO THE METHODS

Parameters

audioObject	The **AudioObject** to play.
channel	The channel the audio will play at.
delay	The delay specified in seconds.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays the **AudioObject** "audioToPlay" at the position "pos" with a delay of 5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public AudioObject audioToPlay;
public Vector3 pos;

        void Start ()
        {

        MultiAudioManager.PlayDelayedAudioObject(audioToPlay, 5, pos);

        }
}
```

GO BACK TO THE METHODS

PlayDelayedAudioObjectByIdentifier

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Vector3** position);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

GO BACK TO THE METHODS

public static **MultiAudioSource** PlayDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

Parameters

audioObjectIdentifier	The identifier of the Global **AudioObject**.
channel	The channel the audio will play at.
delay	The delay specified in seconds.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays **a Global AudioObject** by its identifier “test audio object” at the position “pos” in the **Channel** 2 with a delay of 5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public Vector3 pos;

        void Start ()
        {

        MultiAudioManager.PlayDelayedAudioObjectByIdentifier(“test audio object”, 5,
        2, pos);

        }
}
```

GO BACK TO THE METHODS

PlayDelayedAudioObjectOverride

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**float** delay,**Transform** targetToFollow);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**float** delay,**Vector3** position);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

GO BACK TO THE METHODS

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

Parameters

audioClipOverride	**The AudioClip to override the** AudioObject **clip.**
audioObject	**The** AudioObject **to play.**
channel	The channel the audio will play at.
delay	The delay specified in seconds.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

GO BACK TO THE METHODS

Example

This example plays the **AudioObject** “audioToPlay” with an another **AudioClip** “newAudio” and makes that its pooled **MultiAudioSource** follows the transform “target” with a delay of 5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public AudioObject audioToPlay;
public AudioClip newAudio;
public Transform target;

        void Start ()
        {

        MultiAudioManager.PlayDelayedAudioObjectOverride(newAudio, audioToPlay, 5,
        target);

        }
}
```

GO BACK TO THE METHODS

PlayDelayedAudioObjectOverrideByIdentifier

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Vector3** position);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

GO BACK TO THE METHODS

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

public static **MultiAudioSource** PlayDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound);

Parameters

audioClipOverride	**The AudioClip to override the** AudioObject **clip.**
audioObjectIdentifier	**The identifier of the Global** AudioObject**.**
channel	The channel the audio will play at.
delay	The delay specified in seconds.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

GO BACK TO THE METHODS

Example

This example plays **a Global AudioObject** by its identifier “test audio object” with an another **AudioClip** “newAudio” at the position “pos” in the **Channel** 2 with a delay of 5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public Vector3 pos;
public AudioClip newAudio;

        void Start ()
        {

        MultiAudioManager.PlayDelayedAudioObjectOverrideByIdentifier(newAudio, “test
        audio object”, 5, 2, pos);

        }
}
```

GO BACK TO THE METHODS

FadeInAudioObject

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**int** channel,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**int** channel,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObject(**AudioObject** audioObject,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

GO BACK TO THE METHODS

Parameters

audioObject	The **AudioObject** to play.
channel	The channel the audio will play at.
fadeInTime	The time in seconds to complete the fade in of the sound.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays the **AudioObject** "audioToPlay" at the position "pos" with a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public AudioObject audioToPlay;
public Vector3 pos;

        void Start ()
        {

        MultiAudioManager.FadeInAudioObject(audioToPlay, pos, 0.5f);

        }
}
```

GO BACK TO THE METHODS

FadeInAudioObjectByIdentifier

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

GO BACK TO THE METHODS

public static **MultiAudioSource** FadeInAudioObjectByIdentifier(**string** audioObjectIdentifier,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

Parameters

audioObjectIdentifier	The identifier of the Global **AudioObject**.
channel	The channel the audio will play at.
fadeInTime	The time in seconds to complete the fade in of the sound.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays **a Global AudioObject** by its identifier “test audio object” at the position “pos” in the **Channel** 2 with a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public Vector3 pos;

        void Start ()
        {

        MultiAudioManager.FadeInAudioObjectByIdentifier(“test audio object”, 2, pos);

        }
}
```

GO BACK TO THE METHODS

FadeInAudioObjectOverride

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**int** channel,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

GO BACK TO THE METHODS

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

Parameters

audioClipOverride	The **AudioClip** to override the **AudioObject** clip**.**
audioObject	**The** AudioObject **to play.**
channel	The channel the audio will play at.
fadeInTime	The time in seconds to complete the fade in of the sound.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

GO BACK TO THE METHODS

Example

This example plays the **AudioObject** “audioToPlay” with an another **AudioClip** “newAudio” and makes that its pooled **MultiAudioSource** follows the transform “target” with a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public AudioObject audioToPlay;
public AudioClip newAudio;
public Transform target;

        void Start ()
        {

        MultiAudioManager.FadeInAudioObjectOverride(newAudio, audioToPlay, target,
        0.5f);

        }
}
```

GO BACK TO THE METHODS

FadeInAudioObjectOverrideByIdentifier

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

GO BACK TO THE METHODS

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

Parameters

audioClipOverride	**The AudioClip to override the** AudioObject **clip.**
audioObjectIdentifier	**The identifier of the Global** AudioObject**.**
channel	The channel the audio will play at.
fadeInTime	The time in seconds to complete the fade in of the sound.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

GO BACK TO THE METHODS

Example

This example plays **a Global AudioObject** by its identifier “test audio object” with an another **AudioClip** “newAudio” at the position “pos” in the **Channel** 2 with a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public Vector3 pos;
public AudioClip newAudio;

        void Start ()
        {

        MultiAudioManager.FadeInAudioObjectOverrideByIdentifier(newAudio, “test audio
        object”, 2, pos, 0.5f);

        }
}
```

GO BACK TO THE METHODS

FadeInDelayedAudioObject

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

GO BACK TO THE METHODS

public static **MultiAudioSource** FadeInDelayedAudioObject(**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

Parameters

audioObject	The **AudioObject** to play.
channel	The channel the audio will play at.
delay	The delay specified in seconds.
fadeInTime	The time in seconds to complete the fade in of the sound.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

Example

This example plays the **AudioObject** "audioToPlay" at the position "pos" with a delay of 5 seconds and a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public AudioObject audioToPlay;
public Vector3 pos;

        void Start ()
        {

        MultiAudioManager.FadeInDelayedAudioObject(audioToPlay, 5, pos, 0.5f);

        }
}
```

GO BACK TO THE METHODS

FadeInDelayedAudioObjectByIdentifier

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

GO BACK TO THE METHODS

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectByIdentifier(**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

Parameters

audioObjectIdentifier	The identifier of the Global **AudioObject**.
channel	The channel the audio will play at.
delay	The delay specified in seconds.
fadeInTime	The time in seconds to complete the fade in of the sound.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

GO BACK TO THE METHODS

Example

This example plays **a Global AudioObject** by its identifier “test audio object” at the position “pos” in the **Channel** 2 with a delay of 5 seconds and a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public Vector3 pos;

        void Start ()
        {

        MultiAudioManager.FadeInDelayedAudioObjectByIdentifier(“test audio object”, 5,
        2, pos, 0.5f);

        }
}
```

GO BACK TO THE METHODS

FadeInDelayedAudioObjectOverride

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**float** delay,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(AudioClip audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

GO BACK TO THE METHODS

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverride(**AudioClip** audioClipOverride,**AudioObject** audioObject,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

Parameters

audioClipOverride	**The AudioClip to override the** AudioObject **clip.**
audioObject	**The** AudioObject **to play.**
channel	The channel the audio will play at.
delay	The delay specified in seconds.
fadeInTime	The time in seconds to complete the fade in of the sound.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

GO BACK TO THE METHODS

Example

This example plays the **AudioObject** “audioToPlay” with an another **AudioClip** “newAudio” and makes that its pooled **MultiAudioSource** follows the transform “target” with a delay of 5 seconds and a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public AudioObject audioToPlay;
public AudioClip newAudio;
public Transform target;

        void Start ()
        {

        MultiAudioManager.FadeInDelayedAudioObjectOverride(newAudio, audioToPlay, 5,
        target, 0.5f);

        }
}
```

GO BACK TO THE METHODS

FadeInDelayedAudioObjectOverrideByIdentifier

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**float** fadeInTime=1f);

GO BACK TO THE METHODS

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Transform** targetToFollow,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

public static **MultiAudioSource** FadeInDelayedAudioObjectOverrideByIdentifier(**AudioClip** audioClipOverride,**string** audioObjectIdentifier,**float** delay,**int** channel,**Vector3** position,**AudioMixerGroup** mixerGroup,**bool** occludeSound,**float** fadeInTime=1f);

Parameters

audioClipOverride	**The AudioClip to override the** AudioObject **clip.**
audioObjectIdentifier	**The identifier of the Global** AudioObject**.**
channel	The channel the audio will play at.
delay	The delay specified in seconds.
fadeInTime	The time in seconds to complete the fade in of the sound.
mixerGroup	Set whether the sound should play through an Audio Mixer first or directly to the Listener. Leave **NULL** to use the default SFX/BGM output specified in the **Multi Listener Pooling Audio System Config.**
occludeSound	Occludes the sound if there is a Collider between the source and the listener.
position	The position in which the audio will be played.
targetToFollow	The target that the **MultiAudioSource** will go to follow.

Returns

The pooled **MultiAudioSource**.

GO BACK TO THE METHODS

Example

This example plays **a Global AudioObject** by its identifier “test audio object” with an another **AudioClip** “newAudio” at the position “pos” in the **Channel** 2 with a delay of 5 seconds and a fade in of 0.5 seconds.

```
using UnityEngine;
using AlmenaraGames;

public class TestClass : MonoBehaviour
{

public Vector3 pos;
public AudioClip newAudio;

        void Start ()
        {

        MultiAudioManager.FadeInDelayedAudioObjectOverrideByIdentifier(newAudio, “test
        audio object”, 5, 2, pos, 0.5f);

        }
}
```

GO BACK TO THE METHODS