Audio Manager Documentation

Contributors	Valid for Version	Last Updated
Jonathan Carter	2.5.4 & Higher	30/10/2021

Package Information

The package has the following folders & files, all listed with an asterisk and coloured green are required files for the asset to work. Those without are in red are not needed for the functionality but are required for some cosmetic features.

▼ **Carter Games/Audio Manager/Editor**

- ***AudioManagerEditor.cs**
- ***MusicPlayerEditor.cs**
- ***AudioPlayerEditor.cs**
- **AudioManagerFileEditor.cs**
- ***AudioManagerSettings.cs**
- ***AudioManagerSettingsData.cs**
- ***AudioManagerSettingsDataEditor.cs**

▼ **Carter Games/Assets/Audio Manager/Gizmos**

- AudioManagerFile Icon.png

▼ **Carter Games/Audio Manager/Prefabs**

- ***AudioPrefab.prefab**

▼ **Carter Games/Audio Manager/Recourses**

- LogoAM.png
- Play.png
- Stop.png

▼ **Carter Games/Audio Manager/Demo**

▼ /Audio

> ⚠️ Both these tracks are our own and we own the right to them. Due to this you are not permitted to use these in your own work. They are only there for the example.

- Cloistered Vinyl.wav
- Time Goes By.wav

▼ /Scenes

- AudioManagerExampleScene.unity

▼ /Scripts

- DemoSceneScript.cs

▼ **Carter Games/Audio Manager/Scripts**

- ***AudioManagerScript.cs**
- ***AudioManagerFile.cs**
- ***AudioRemoval.cs**
- ***AudioHelper.cs**
- ***MusicPlayer.cs**
- ***AudioPlayer.cs**
- **AudioEvents.cs**
- ***AudioLibrary.cs**
- ***TransitionType.cs**

Change Log: Shows the changes from previous versions of the asset.

Docs: Text file that links to here and provides an offline copy of this page.

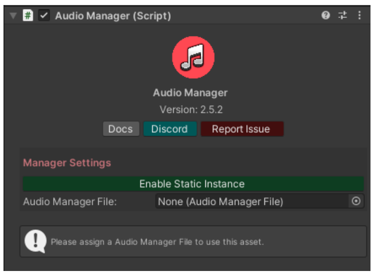
Contents

Getting Started

✨ Firstly, thank you for deciding to use our asset for your project. If you like our asset, feel free to leave us a review! If you find that our asset is not up to scratch or find and issue please do let us know either via our email: **support@carter.games** and we will do our best to help you with the issues you are facing.

Import all files from the Audio Manager package into your project. The files are all contained in their own file structure, you can just import it as is, or merge the induvial folders into your own project. Once this is done you'll need to run the setup to start using the asset. If you plan to store your audio clips out side of "Assets/Audio" then you will need to change the base scanning directory from the Audio Manager Global Settings, found under **Tools/Audio Manager | CG/Global Settings.** If you just want the default directory you can just move on to the setup.

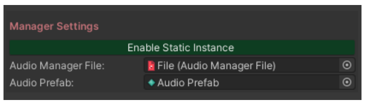
Setup & Usage of Scripts



Audio Manager File: None (Audio Manager File)

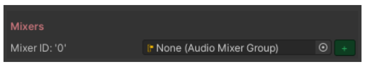
Please assign a Audio Manager File to use this asset.

First add the audio manager component to any gameobject you like. We recommend putting the script on a empty gameobject. Once added the inspector will show up asking for an **Audio Manager File** before moving forward. If this is your first time adding an Audio Manager script to an object, you'll find that the field may be populated for you by the scripts automatic setup which generates the file structure for the asset and makes a default Audio Manager File for you to use.



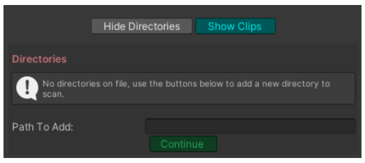
Audio Manager File: File (Audio Manager File)
Audio Prefab: Audio Prefab

Next you'll need to assign the prefab that is to play the audio when you request it. We provide a correctly setup prefab in the package which we recommend you use with the asset. If you have lost it or are upgrading from an older version you may find your prefab is outdated. See the **Audio Prefab** section for more on how the prefab should be setup.



Mixer ID: '0' None (Audio Mixer Group)

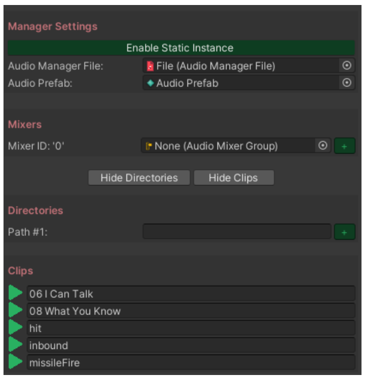
You can also assign audio mixers should you wish, though these are optional and are not required to use the manager to function. When you add mixer groups here you can use the ID listed next to them to use them when calling for a clip to be played without needed a direct reference to the mixer in the script you are calling from.



No directories on file, use the buttons below to add a new directory to scan.

Path To Add:

From here you are almost ready to use the Audio Manager. The inspector should now show the options for directories & clips. Open the directories tab by pressing the show directories button if it is not already open. Here you'll see an empty field and a button to continue. If you audio clips are going to be stored in the base scan directory you can just press the continue button and the manager will scan and add any clips it finds. However if you want to use a folder structure you'll need to define the subdirectories here. See the **How Scanning Works** section for examples of how to write these. You can add or remove elements with the + & - buttons along each row. When all the directories are valid the manager will scan, you may notice a little editor lag as this runs.



Audio Manager File: File (Audio Manager File)
Audio Prefab: Audio Prefab

Mixer ID: '0' None (Audio Mixer Group)

Path #1:

06 I Can Talk
08 What You Know
hit
inbound
missileFire

From here all of the clips in the directories selected should be in the clips section, press the show clips button to reveal the clips if needed, with both the directories and clip sections you can hide them to reduce the space the inspector takes up when not needed. Assuming you see the clips all listed you will be ready to use the Audio Manager.

Usage

The Audio Manager has a variety of methods you can use to play audio clips in your scene. There are two main ways of passing in params, either but individual params or with a hashtable. We provide a version of a hashtable called **AudioArgs** which will make sure the params you pass in are setup correctly, such as fixing formatting issues you may accidently make. To play any method you will need a reference to the script, this can either be the static instance provided or a direct reference made by yourself. Below is a few examples of how to play a clip called MySound:

```
// Reference to script
public AudioManager audioManager;

// This will play MySound at the default volume of 1 and default pitch of 1.
audioManager.Play("MySound");

// This will play MySound at the volume of 0.5 with the default pitch of 1. You get the idea…..
audioManager.Play("MySound", .5f);

// Using the static instance
AudioManager.instance.Play("MySound");
AudioManager.instance.Play("MySound", .5f);
```

Audio Args & Hashtables

As of 2.5.0 we have changes how the methods in the main audio manager script work. You can now use the AudioArgs static method or a Hashtable to define all the params for the method. With this change we have also added a few additional params, such as priority and reverb & spatial blend that are not in our standard methods which allows for full customisability. This is also where our play at position and on gameobject methods have been moved to. In 2.5.1 we have moved the AudioArgs to the Audio Helper static class so it can be used without the Audio Manager being static**.** To use the arguments you can do any of the following, note that inn 2.5.0 you need to pass string values with PascalCase, from **2.5.1 or later** string values should be in lowercase:

```
// Using the static Audio Helper class
Play("myClip", AudioHelper.AudioArgs("position", Vector3.one, "time", 1f));

// Using a Hashtable
Play("myClip", new Hashtable("position", Vector3.one, "time", 1f));
```

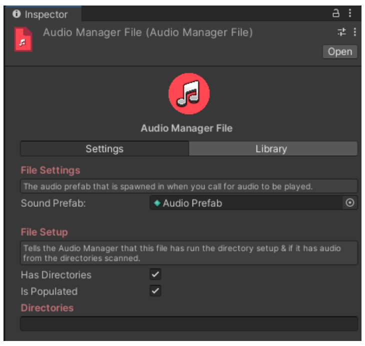
You can also cache a reference for better performance and easy editing like so:

```
// With Hashtables
// Set in start/awake
var table = new Hashtable();
table.Add("Position", Vector3.one);
table.Add("Time", 1f);

// When you want to play a clip...
Play("myClip", table);
```

Inspector Breakdowns

Audio Manager File



Inspector

Audio Manager File (Audio Manager File)

Open

Audio Manager File

Settings | Library

File Settings

The audio prefab that is spawned in when you call for audio to be played.

Sound Prefab: Audio Prefab

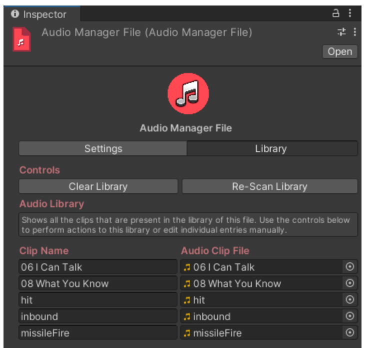
File Setup

Tells the Audio Manager that this file has run the directory setup & if it has audio from the directories scanned.

Has Directories ✓

Is Populated ✓

Directories



Inspector

Audio Manager File (Audio Manager File)

Open

Audio Manager File

Settings | Library

Controls

Clear Library | Re-Scan Library

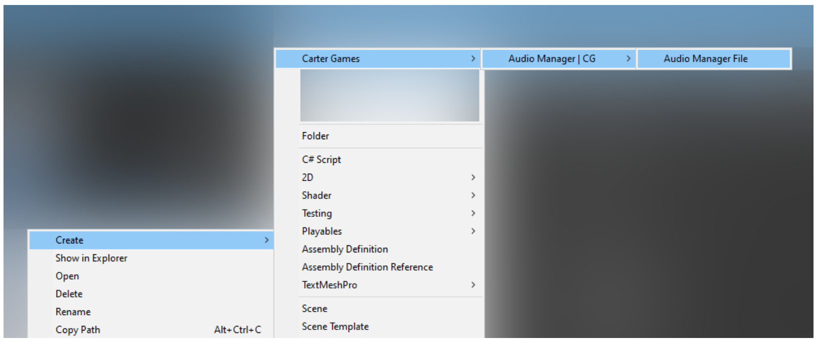
Audio Library

Shows all the clips that are present in the library of this file. Use the controls below to perform actions to this library or edit individual entries manually.

Clip Name	Audio Clip File
06 I Can Talk	06 I Can Talk
08 What You Know	08 What You Know
hit	hit
inbound	inbound
missileFire	missileFire

Summary

The Audio Manager File Scriptable Object stores all the audio that has been scanned by the manager instance that is using said file. This file is required for the **Audio Manager** and **Audio Player** scripts to function. You can have multiple files and multiple instances of the manager running at once should you need to as each instance will run independently.



You can create an Audio Manager File by using the create asset menu following the path shown above. One of these is created for you when you run the setup mentioned before. Most projects will only really need one file. However the option is there to create more.

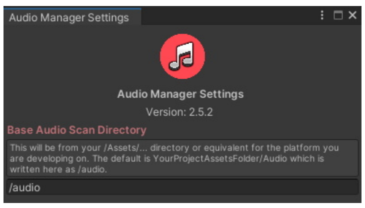
Breakdown

The Audio Manager File (AMF) has a custom inspector which better display the information it holds for the user as well as explains what each field does.

⚠️ If you haven't imported the AudioManagerFileEditor.cs script into your project you will not see this inspector. Instead you will see a normal scriptable object inspector.

There are two tabs in the inspector. The first tab is the settings that this AMF is setup with, all of these values are handled by the inspector in the Audio Manager script and should be left as is in the file, editing this here can cause errors in the inspector of the manager. The second tab shows all the clips that the AMF holds, this is what is used when you call for a sound the be played. You have the option here to clear and re-scan this library from the AMF should you need it.

Audio Manager Settings



Audio Manager Settings

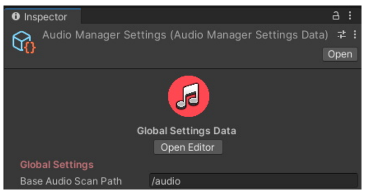
Audio Manager Settings

Version: 2.5.2

Base Audio Scan Directory

This will be from your /Assets/... directory or equivalent for the platform you are developing on. The default is YourProjectAssetsFolder/Audio which is written here as /audio.

/audio



Inspector

Audio Manager Settings (Audio Manager Settings Data)

Open

Global Settings Data

Open Editor

Global Settings

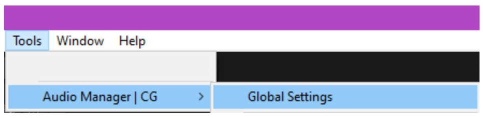
Base Audio Scan Path /audio

Summary

The audio manager settings are a set of global options that effect the asset as a whole. You will only have one of these per project and don't have the option to make your own. When you run the setup for the asset it will make one of these for you if there isn't one in the project and it will sit in the same location as the **AudioManagerEditor.cs** class is.

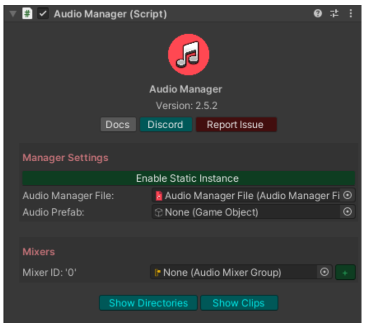
Breakdown

You can edit the options either form the scriptable object or via the editor window. The editor window provided more context on each option and what they do so it is recommend you use that over the scriptable object. The editor window can be accessed either by the custom inspector on the settings scriptable object or via the navigation bar under **Tools/Audio Manager | CG/Global Settings.**



Base Audio Scan Path - This controls the base directory that is used for all audio in the project. By default it is set to /audio in the assets folder of your project. You can change this to any directory in the assets folder of your project, though we recommend you only change this if necessary and before actually using the asset to avoid issues with existing audio manager files.

Audio Manager



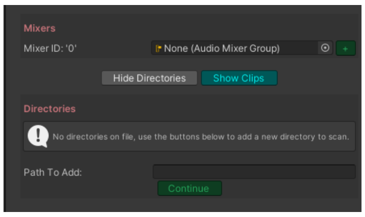
Audio Manager
Version: 2.5.2

Manager Settings

Audio Manager File: Audio Manager File (Audio Manager Fi
Audio Prefab: None (Game Object)

Mixers

Mixer ID: '0' None (Audio Mixer Group)



Mixers

Mixer ID: '0' None (Audio Mixer Group)

Directories

No directories on file, use the buttons below to add a new directory to scan.

Path To Add:

Summary

The Audio Manager script is the main script for the asset. it handles the scanning and setup for audio in your project. When you first add the component to a GameObject you'll be greeted by a display similar to what you can see on the left here. This script requires an Audio Manager File to work, it will generate one when you first add the component.

Setup Details

We go through this in the **Setup** section, but here is a more detailed rundown about what is what.

When you first add the script, it run the initial setup for the asset by creating the paths for the Audio Manager Files in the defined base scan path, which is just **/audio** in your projects assets folder by default. If it is your first time adding the component, it will also make an Audio Manager File for you and place it under the base scan path folder generated which by default is under **/audio/files** and assign it in the inspector for you ready for use.

The audio manager has a few options that need setting up before it is ready to use. These steps are only needed on the first run, once you've done this you won't need to repeat the steps unless you change the Audio Manager File the manager uses with a fresh one, in which case you may need to repeat the setup.

- Firstly you can choose to have the Audio Manager be static or not. If you wish to use the static instance then just press the button and wait for the editor to refresh, once done the button will change to a red disable button. Note that you may need to restart Unity and/or your IDE of choice to get the instance via your IDE auto complete.
- Next you will need to assign the audio prefab, we provide one with the asset package, see the Audio Prefab Setup for more information. This is used to play the sounds you request.
- Next you can setup and Mixers you wish to use with the manager. You can add additional mixers with the **+ Button** next to the any mixer field. Each mixer is assigned an ID, which works like a collection index, starting at 0 and going up the more you add. You use this ID to use mixers when calling sounds to be played through the manager.
- Next you'll need to press the show directories button to reveal the scan directories window. Here you can set all the directories from the base directory you wish to scan. If you wish to get audio from the base directory then you can just press continue right away. Once again you can add more via the **+ Button.**
- Once you have pressed the continue button the Audio Manager will have scanned all the clips in the defined the directories and display them for you. If it runs into any errors, the asset will show a message explaining what has gone wrong most of the time.

You are now set the use the Audio Manager and the rest of asset scripts that require the Audio Manager File to operate.

Breakdown

Manager Settings

- **Enable/Disable Static Instancing** - toggles the static instance on the audio manager script, it is off by default but can be enabled if you want to use a static instance for referencing. When toggling, the editor will take a little time to reload. Note that if you use the instance, you can only have 1 audio manager script in the game. Any extra version will auto-destroy on enable.
- **Audio Manager File -** The audio manager file to read/write from.
- **Audio Prefab -** The prefab to spawn when you call for a clip to play through the manager.

Mixers

- **Mixers -** The audio mixer(s) to use when playing the clips through.

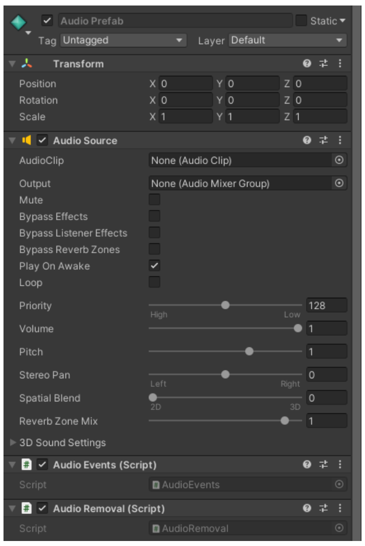
Directories

- **Directories -** A collection of directories that are to be scanned, a blank field just scanned the base location. You can't have duplicates and the manager wil warn you if you do.

Clips

- **Clips -** Each field will show a play button in green and a clip name which is the name you need to call to play the clip. You can copy directory from this field. Pressing the green button will preview the clip.

Audio Prefab



```
Audio Prefab                         Static
Tag Untagged           Layer Default
Transform
Position     X 0    Y 0    Z 0
Rotation     X 0    Y 0    Z 0
Scale        X 1    Y 1    Z 1
Audio Source
AudioClip                 None (Audio Clip)
Output                    None (Audio Mixer Group)
Mute
Bypass Effects
Bypass Listener Effects
Bypass Reverb Zones
Play On Awake             ✓
Loop
Priority                  High        Low   128
Volume                                      1
Pitch                                       1
Stereo Pan                Left      Right   0
Spatial Blend             2D           3D   0
Reverb Zone Mix                             1
3D Sound Settings
Audio Events (Script)
Script                    AudioEvents
Audio Removal (Script)
Script                    AudioRemoval
```

Summary

The audio prefab is used to play the clips that you request from the **Audio Manager** & **Audio Player**. These objects spawn in when you call for a clip to be played and when finished the object with disable for re-use.

Breakdown

We provide a fully setup prefab in the asset package, though should you wish to create your own or may alternative versions you will need the following components attached:

Audio Source

The source that plays the clip, this is all setup when you call for a clip to be played.

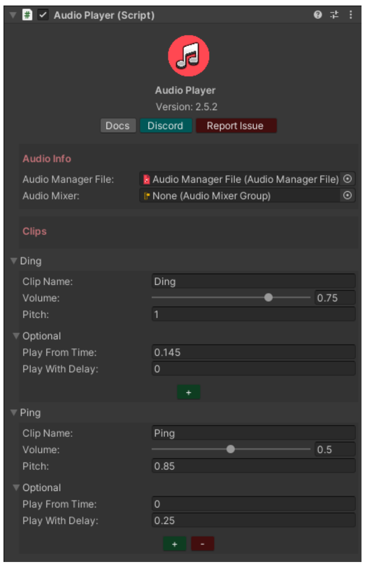
Audio Removal

This script just handles the clean-up of the object once the clip has finished playing ready for re-use. This is a required component as the audio manager can't function properly without it.

Audio Events

If you wish to use the **OnClipEnded** Action then you will also need this script. If you don't use the event they you will be fine to not add this script.

Audio Player



```
Audio Player (Script)
Audio Player
Version: 2.5.2
Docs   Discord   Report Issue
Audio Info
Audio Manager File:    Audio Manager File (Audio Manager File)
Audio Mixer:           None (Audio Mixer Group)
Clips
Ding
Clip Name:             Ding
Volume:                0.75
Pitch:                 1
Optional
Play From Time:        0.145
Play With Delay:       0
+
Ping
Clip Name:             Ping
Volume:                0.5
Pitch:                 0.85
Optional
Play From Time:        0
Play With Delay:       0.25
+ -
```

Summary

The audio player script lets you play clips from a Audio Manager File library from the inspector. This is designed for use with UI elements and UnityEvents. This is designed to just be a basic stopgap as you'll more commonly want to write your own implementation for playing specific audio.

Breakdown

The inspector for this script should be setup for how you want your audio to play when you call the method to play the sounds on it.

- **Audio Manager File** - the library to read from for the clips.
- **Audio Mixer -** the mixer group to play the clips through.

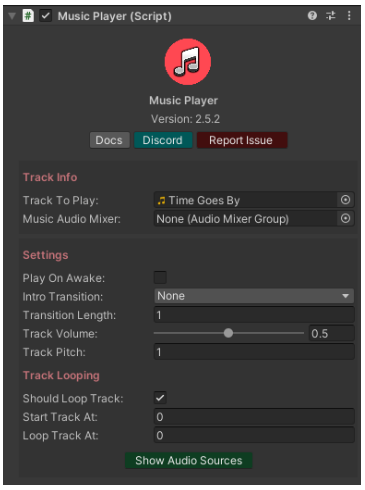
The next section defines all the clips that are to be played, you can add or remove elements with the **+** & **-** symbols. Each clip has several fields, below is what each of them are:

- **Clip Name** - The name of the clip you want to play, this has to be a clip that is currently in the audio manager file entered at the top of the player for it to work.
- **Volume -** The volume the clip should play as, this is limited between 0-1, use the mixer group, or increase in the audio clip file to change it further if needed.
- **Pitch -** The pitch for the clip to play at.

The rest of the settings are optionall for each clip but allow them to be played using the from time and with delay setups like in the main audio manager script.

- **Play From Time -** The start time clip should play from.
- **Play With Delay -** The delay there sould be before the clip should play.

Music Player



Track Info

Track To Play:	Time Goes By
Music Audio Mixer:	None (Audio Mixer Group)

Settings

Play On Awake:	☐
Intro Transition:	None
Transition Length:	1
Track Volume:	0.5
Track Pitch:	1

Track Looping

Should Loop Track:	✓
Start Track At:	0
Loop Track At:	0

Summary

The music player is a script that allow you to manage some basic background music within your games. The script lets you change the track via code with a few transitions to choose from. This script has a **static instance** by default and is not affected by the main Audio Manager static instance toggle.

Breakdown

The inspector for this script should be setup with the first track you wish to play, the rest of the tracks should be stored in your own scripting to be inserted into the player when needed.

- **Track To Play -** the track that will play when will script is first called
- **Music Audio Mixer -** the audio mixer to play the clip from

The settings for the player allow you to change how the player plays your tracks, You can edit all the values when through code.

- **Play On Awake** - should the click play on the Awake() method
- **Intro Transition -** the transition the player uses when the track first plays
- **Transition Length -** the length of the transition for changing tracks
- **Track Volume -** edits the volume of the clip between 0-1
- **Track Pitch -** edits the pitch of the track

The looping of the track you select is optional, should you want to use it you'll need to set it up for each track you want to use. When calling to play a track in code you can set the start and end times.

- **Should Loop Track -** should the track loop once it reaches the end of the audio clip
- **Start Track At -** the time in seconds where the clip should start from
- **Loop Track At -** the point in seconds where the track will loop from

Methods, Properties & Actions

Audio Manager

Properties

▼ Click to expand

CanPlayAudio

Toggles whether or not the manager can play audio. This doesn't effect other Audio Manager instances or the other player scripts.

Returns: Bool

HasClip(string request)

Checks to see if a clip exists in the audio manager library.

Returns: Bool

GetNumberOfClips

Gets the number of clips currently in this instance of the Audio Manager.

Returns: Int

GetRandomSound

Picks a random sound from the current AMF and returns it.

Returns: AudioClip

GetAudioManagerFile

Gets the current AMF in use.

Returns: AudioManagerFile

IsClipPlaying(string clip)

Checks to see if the clip in question is playing.

Returns: Bool

Methods

▼ Click to expand

Play

Plays a clip that you request as is.

▼ Click to expand for further details

Play(string request, float volume = 1f, float pitch = 1f)
Play(string request, AudioMixerGroup mixer, float Volume = 1f, float pitch = 1f)
Play(string request, int mixerID, float volume = 1f, float pitch = 1f)
Play(string request, AudioArgs args)

Plays the requested clip with optional values for volume and pitch. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Name	Data Type	Usage Type	Description
Request	String	Required	The clip name you wish to play. If it can't find the clip the manager will warn you the clip requested was not found.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayAndGetSource

Plays a clip that you request as is, but also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayAndGetSource(string request, float volume = 1, float pitch = 1)
PlayAndGetSource(string request, AudioMixerGroup mixer, float volume = 1, float pitch = 1)
PlayAndGetSource(string request, int mixerID, float volume = 1, float pitch = 1)
PlayAndGetSource(string request, AudioArgs args)

Play a sound that is scanned into the audio manager and return the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSource

Parameters Explained

Name	Data Type	Usage Type	Description
Request	String	Required	The clip name you wish to play. If it can't find the clip the manager will warn you the clip requested was not found.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayFromTime

Plays a clip that you request, but starting at the time you enter.

▼ Click to expand for further details

PlayFromTime(string request, float time, float volume = 1f, float pitch = 1f)
PlayFromTime(string request, float time, AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayFromTime(string request, float time, int mixerID, float volume = 1f, float pitch = 1f)
PlayFromTime(string request, float time, AudioArgs args)

Plays the requested clip with optional values for volume and pitch from the set time, handy if the clip doesn't start right away. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Name	Data Type	Usage Type	Description
Request	String	Required	The clip name you wish to play. If it can't find the clip the manager will warn you the clip requested was not found.
Time	Float	Required	The time in seconds where the clip should start from. Ideal if your clip doesn't play sound right away.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayFromTimeAndGetSource

Plays a clip that you request, but starting at the time you enter. But also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayFromTimeAndGetSource(string request, float time, float volume = 1, float pitch = 1)
PlayFromTimeAndGetSource(string request, AudioMixerGroup mixer, float time, float volume = 1, float pitch = 1)
PlayFromTimeAndGetSource(string request, int mixerID, float time, float volume = 1, float pitch = 1)
PlayFromTimeAndGetSource(string request, float time, AudioArgs args)

Play a sound from a particular time code on the audio clip audioManagerFile and returns the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSource

Parameters Explained

Name	Data Type	Usage Type	Description
Request	String	Required	The clip name you wish to play. If it can't find the clip the manager will warn you the clip requested was not found.
Time	Float	Required	The time in seconds where the clip should start from. Ideal if your clip doesn't play sound right away.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayWithDelay

Plays a clip that you request, but starting after the defined delay.

▼ Click to expand for further details

PlayWithDelay(string request, float delay, float volume = 1f, float pitch = 1f)
PlayWithDelay(string request, float delay, AudioMixerGroup mixer, float Volume = 1f, float pitch = 1f)
PlayWithDelay(string request, float delay, int mixerID, float volume = 1f, float pitch = 1f)
PlayWithDelay(string request, float delay, AudioArgs args)

Plays the requested clip with optional values for volume and pitch after the entered delay. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Name	Data Type	Usage Type	Description
Request	String	Required	The clip name you wish to play. If it can't find the clip the manager will warn you the clip requested was not found.
Delay	Float	Required	The time in seconds for how long the clip should wait before playing.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayWithDelayAndGetSource

Plays a clip that you request, but starting after the defined delay. But also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayWithDelayAndGetSource(string request, float delay, float volume = 1f, float pitch = 1f)
PlayWithDelayAndGetSource(string request, AudioMixerGroup mixer,float delay, float volume = 1f, float pitch = 1f)
PlayWithDelayAndGetSource(string request, int mixerID, float delay, float volume = 1f, float pitch = 1f)
PlayWithDelayAndGetSource(string request, float delay, AudioArgs args)

Plays the requested clip with optional values for volume and pitch after the entered delay. Also returns the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSource

Parameters Explained

Name	Data Type	Usage Type	Description
Request	String	Required	The clip name you wish to play. If it can't find the clip the manager will warn you the clip requested was not found.
Delay	Float	Required	The time in seconds for how long the clip should wait before playing.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayFromRange

Plays a random clip that you request from a collection of strings as is.

▼ Click to expand for further details

PlayFromRange(List<string> or string[] request, float volume = 1f, float pitch = 1f)
PlayFromRange(List<string> or string[] request, AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayFromRange(List<string> or string[] request, int mixerID, float volume = 1f, float pitch = 1f)
PlayFromRange(List<string> or string[] request, AudioArgs args)

Plays a random clip from the entered strings with optional values for volume and pitch. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Name	Data Type	Usage Type	Description
Request	List<String> String[]	Required	The clip names you wish to select from to play. If it can't find the clip the manager will warn you the clip requested was not found.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayFromRangeAndGetSource

Plays a random clip that you request from a collection of strings as is. But also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayAndGetSource(List<string> or string[] request, float volume = 1, float pitch = 1)
PlayAndGetSource(List<string> or string[] request, AudioMixerGroup mixer, float volume = 1, float pitch = 1)
PlayAndGetSource(List<string> or string[] request, int mixerID, float volume = 1, float pitch = 1)
PlayAndGetSource(List<string> or string[] request, AudioArgs args)

Plays a random clip from the entered strings and return the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSource

Parameters Explained

Name	Data Type	Usage Type	Description
Request	List<String> String[]	Required	The clip names you wish to select from to play. If it can't find the clip the manager will warn you the clip requested was not found.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayFromRangeFromTime

Plays a random clip that you request from a collection of strings as is.

▼ Click to expand for further details

PlayFromRangeFromTime(List<string> or string[] request, float volume = 1f, float pitch = 1f)
PlayFromRangeFromTime(List<string> or string[] request, AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayFromRangeFromTime(List<string> or string[] request, int mixerID, float volume = 1f, float pitch = 1f)
PlayFromRangeFromTime(List<string> or string[] request, AudioArgs args)

Plays a random clip from the entered strings with optional values for volume and pitch. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Aa Name	Data Type	Usage Type	Description
Request	List<String> String[]	Required	The clip names you wish to select from to play. If it can't find the clip the manager will warn you the clip requested was not found.
Time	Float	Required	The time in seconds where the clip should start from. Ideal if your clip doesn't play sound right away.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayFromRangeFromTimeAndGetSource

Plays a random clip that you request from a collection of strings as is. But also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayAndGetSource(List<string> or string[] request, float time, float volume = 1, float pitch = 1)
PlayAndGetSource(List<string> or string[] request, float time, AudioMixerGroup mixer, float volume = 1, float pitch = 1)
PlayAndGetSource(List<string> or string[] request, float time, int mixerID, float volume = 1, float pitch = 1)
PlayAndGetSource(List<string> or string[] request, float time, AudioArgs args)

Plays a random clip from the entered strings and return the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSource

Parameters Explained

Aa Name	Data Type	Usage Type	Description
Request	List<String> String[]	Required	The clip names you wish to select from to play. If it can't find the clip the manager will warn you the clip requested was not found.
Time	Float	Required	The time in seconds where the clip should start from. Ideal if your clip doesn't play sound right away.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayFromRangeWithDelay

Plays a random clip that you request from a collection of strings as is.

▼ Click to expand for further details

PlayFromRangeWithDelay(List<string> or string[] request, float delay, float volume = 1, float pitch = 1)
PlayFromRangeWithDelay(List<string> or string[] request, float delay, AudioMixerGroup mixer, float volume = 1, float pitch = 1)
PlayFromRangeWithDelay(List<string> or string[] request, float delay, int mixerID, float volume = 1, float pitch = 1)
PlayFromRangeWithDelay(List<string> or string[] request, float delay, AudioArgs args)

Plays a random clip from the entered strings with the requested delay before playing the clip. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Name	Data Type	Usage Type	Description
Request	List<String> String[]	Required	The clip names you wish to select from to play. If it can't find the clip the manager will warn you the clip requested was not found.
Delay	Float	Required	The time in seconds for how long the clip should wait before playing.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayFromRangeWithDelayAndGetSource

Plays a random clip that you request from a collection of strings as is. But also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayFromRangeWithDelayAndGetSource(List<string> or string[] request, float delay, float volume = 1, float pitch = 1)
PlayFromRangeWithDelayAndGetSource(List<string> or string[] request, float delay, AudioMixerGroup mixer, float volume = 1, float pitch = 1)
PlayFromRangeWithDelayAndGetSource(List<string> or string[] request, float delay, int mixerID, float volume = 1, float pitch = 1)
PlayFromRangeWithDelayAndGetSource(List<string> or string[] request, float delay, AudioArgs args)

Plays a random clip from the entered strings with the requested delay before playing as well as returning the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSource

Parameters Explained

Name	Data Type	Usage Type	Description
Request	List<String> String[]	Required	The clip names you wish to select from to play. If it can't find the clip the manager will warn you the clip requested was not found.
Delay	Float	Required	The time in seconds for how long the clip should wait before playing.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayRandom

Plays a random clip from the library of clips the audio manager has scanned.

▼ Click to expand for further details

PlayRandom(float volume = 1f, float pitch = 1f)
PlayRandom(AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayRandom(int mixerID, float volume = 1f, float pitch = 1f)
PlayRandom(AudioArgs args)

Plays a random clip from the audio manager library with optional values for volume and pitch. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Name	Data Type	Usage Type	Description
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayRandomAndGetSource

Plays a random clip from the library of clips the audio manager has scanned. But also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayRandomAndGetSource(float volume = 1f, float pitch = 1f)
PlayRandomAndGetSource(AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayRandomAndGetSource(int mixerID, float volume = 1f, float pitch = 1f)
PlayRandomAndGetSource(AudioArgs args)

Plays a random clip from the audio manager library with optional values for volume and pitch. Also returns the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSoruce

Parameters Explained

Name	Data Type	Usage Type	Description
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayRandomFromTime

Plays a random clip from the library of clips the audio manager has scanned. With the clip starting at the entered time.

▼ Click to expand for further details

PlayRandomFromTime(float time, float volume = 1f, float pitch = 1f)
PlayRandomFromTime(float time, AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayRandomFromTime(float time, int mixerID, float volume = 1f, float pitch = 1f)
PlayRandomFromTime(float time, AudioArgs args)

Plays a random clip from the audio manager library with optional values for volume and pitch from the set time. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Name	Data Type	Usage Type	Description
Time	Float	Required	The time in seconds where the clip should start from. Ideal if your clip doesn't play sound right away.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayRandomFromTimeAndGetSource

Plays a random clip from the library of clips the audio manager has scanned. With the clip starting at the entered time. But also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayRandomFromTimeAndGetSource(float time, float volume = 1f, float pitch = 1f)
PlayRandomFromTimeAndGetSource(float time, AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayRandomFromTimeAndGetSource(float time, int mixerID, float volume = 1f, float pitch = 1f)
PlayRandomFromTimeAndGetSource(float time, AudioArgs args)

Plays a random clip from the audio manager library with optional values for volume and pitch from the set time. Also returns the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSource

Parameters Explained

Name	Data Type	Usage Type	Description
Time	Float	Required	The time in seconds where the clip should start from. Ideal if your clip doesn't play sound right away.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayRandomWithDelay

Plays a random clip from the library of clips the audio manager has scanned. With the clip playing after the defined delay.

▼ Click to expand for further details

PlayRandomWithDelay(float delay, float volume = 1f, float pitch = 1f)
PlayRandomWithDelay(float delay, AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayRandomWithDelay(float delay, int mixerID, float volume = 1f, float pitch = 1f)
PlayRandomWithDelay(float delay, AudioArgs args)

Plays a random clip from the audio manager library with optional values for volume and pitch after the entered delay. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Parameters Explained

Name	Data Type	Usage Type	Description
Delay	Float	Required	The time in seconds for how long the clip should wait before playing.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

PlayRandomWithDelayAndGetSource

Plays a random clip from the library of clips the audio manager has scanned. With the clip playing after the defined delay. But also returns the AudioSource component that is used for that clip.

▼ Click to expand for further details

PlayRandomWithDelayAndGetSource(float delay, float volume = 1f, float pitch = 1f)
PlayRandomWithDelayAndGetSource(float delay, AudioMixerGroup mixer, float volume = 1f, float pitch = 1f)
PlayRandomWithDelayAndGetSource(float delay, int mixerID, float volume = 1f, float pitch = 1f)
PlayRandomWithDelayAndGetSource(float delay, AudioArgs args)

Plays a random clip from the audio manager library with optional values for volume and pitch after the entered delay. Also returns the audio source the clip is on for you to check / use as needed. There are several overload methods are also available for audio mixers, either by manual assignment or via the audio manager mixer ID number.

Returns: AudioSource

Parameters Explained

Name	Data Type	Usage Type	Description
Delay	Float	Required	The time in seconds for how long the clip should wait before playing.
Volume	Float	Optional	Changes the volume used for the clip, note this limited between 0-1. This value has a default value of 1 which is used if not altered.
Pitch	Float	Optional	Changes the pitch of the clip to the entered value. This value has a default value of 1 which is used if not altered.
AudioMixerGroup	AudioMixerGroup	Required On Overload	The audio mixer to play the clip with.
MixerID	Int	Required On Overload	The ID of the mixer you wish to use. The ID's are listed in the inspector of the audio manager inspector.
Args	AudioArgs Hashtable	Required On Overload	The Hashtable to use to set the parameters. The Audio Manager provides a class called AudioArgs that can be used here, or if you wish you can use a normal Hashtable. Here you can edit almost all values on the a normal Audio Source component. See **AudioArgs** for more information on usage.

Audio Player

Methods

▼ Click to expand

Play

Plays all the clips setup in the inspector of the component using the settings in the inspector.

▼ Click to expand for further details

Play()

Plays all the clips assigned in the inspector. This uses the settings from the inspector for the clip to set the values for it. You can edit the volume, pitch, start time & delay from the custom inspector for each clip in the player.

Music Player

Actions

▼ Click to expand

All actions are public and static, they can be accessed like so: MusicPlayer.OnTrackStarted

OnTrackStarted

Triggered whenever a new track is started on a track loops around to the start again.

OnTrackEnded

Triggered whenever the track has ended.

OnTrackLooped

Triggered whenever a track loops around back to the start.

OnTrackChanged

Triggered whenever the user changes the active track.

OnTrackTransitionComplete

Triggered whenever the active track transition has completed

Properties

▼ Click to expand

IsTrackPlaying

Gets whether or not there is a track currently playing?

Returns: Bool

GetActiveTrack

Gets the track that is currently being player by the music player script.

Returns: AudioClip

GetTrackPosition

Gets the current time in seconds that the track is at.

Returns: Float

GetActiveSource

Gets the audio source component that is playing the active track.

Returns: AudioSource

ShouldLoop

Gets/Sets whether or not the active track should loop.

Methods

▼ Click to expand

SetVolume

SetVolume(float value)

Sets the volume of the music player to the entered value. Range is between 0-1.

Stop

Stop()

Stops the active track from playing without any transition.

PlayTrack

PlayTrack(AudioClip track)
PlayTrack(AudioClip track, TransitionType transitionType)
PlayTrack(AudioClip track, TransitionType transitionType, float transitionDuration)
PlayTrack(AudioClip track, float startTime)
PlayTrack(AudioClip track, float startTime, TransitionType transitionType)
PlayTrack(AudioClip track, float startTime, TransitionType transitionType, float transitionDuration)
PlayTrack(AudioClip track, float startTime, float endTime)
PlayTrack(AudioClip track, float startTime, float endTime, TransitionType transitionType)
PlayTrack(AudioClip track, float startTime, float endTime, TransitionType transitionType, float transitionDuration)

Plays the track requested on the music player script. There are several overloads for this method that allow for additional options. If just using the audioclip param the transition will stay as the last one used in the inspector, the start time will be set to 0 and the end time will be set to the track length.

Parameters

Name	Data Type	Description
track	AudioClip	The track that you wish to play. Use null if you want to just end the current track without changing to a new one.
transitionType	TransitionType	The type of transition the music player should use when changing the track. All types below: - None - FadeIn - FadeOut - FadeInOut - CrossFade
startTime	Float	The time when the track should start playing from. Time is in seconds.
endTime	Float	The time when the track should end/loop from. Time is in seconds.
transitionDuration	Float	The amount of time the transition should take to complete. Default is 1 second.

Audio Events

Actions

▼ Click to expand

OnClipEnd

Triggered when any clip has finished playing.

F.A.Q

How scanning works

By default the script scans the "/audio" directory if you have left a directory field left blank, you can change the path via the **Audio Manager Global Settings** which we recommend you do before scanning for the first time. to go to any sub-directory within the "/audio" folder, or add additional directories using the inspector display. This field is not case sensitive as shown below:

- **Example 1:** mygame – "/audio/mygame" will be scanned
- **Example 2:** MyGame – "/audio/mygame" will be scanned (capitals are ignored)
- **Example 3:** MyGame/SFX – "/audio/mygame/sfx" will be scanned

The script will automatically update with the new sounds once you have entered a valid directory. Once scanned the script will list all the audioclips it has found. This will update on the fly when you add new files into your scanned directory with the audio manager selected.

Warning and error codes

The scripts have a selection of error messages in the form of console warnings, all errors from this script come with the prefix "Audio Manager |" so you will know it is from this package. Most errors shouldn't come up, but they should explain what you've done wrong and how to fix it.

- **Warning Code 1:** Make sure you have a sound prefab assigned to the AMF you are using, this is caused when there is not prefab found.
- **Warning Code 2:** Make sure you have spelt the clip you want correctly, this warning shows up if the audio could not be found in the audio manager when called.
- **Warning Code 3:** Could not find audio mixer, Please ensure the mixer is in the inspector and you have the right ID.
- **Warning Code 4:** No AudioSource Component found on the Sound Prefab. Please ensure a AudioSource Component is attached to your prefab.

However, if you run into a problem or get an error and are unsure, feel free to drop us an email or send us a message via our discord and we'll do our best to help you out.

A file isn't scanning

The audio manager will scan all valid audio file types that work with unity. There are a few reasons why a clip won't scan. Try some of the steps below if you are having this issue.

- Make sure the file type is accepted by Unity.
- Make sure the clip name is not the same as another in the project.
- If updating a clip with a new one with the same name, delete the old one and let the audio manager rescan, then add the new one.

A clip won't play

There can be a few reason why a clip won't play:

- Check that you have spelt the clip name correctly.
- Check that the clip is present in the audio manager file you are using.

I called a function and the clip plays a million times!

This is due to you having the call in an update() or similar, if you have the call in update you need to have either a Boolean or a coroutine to stop it been called more than once.

The Inspector hasn't loaded correctly when I added the script

If this has happened, please sent me screenshots and ways to replicate the problem so I can fix it. email: support@carter.games

Got an issue that isn't listed?

Please get in touch with us so we can do our best to help you out. Email: support@carter.games