# MASTER AUDIO PLUGIN V2.7 - By Dark Tonic, Inc. (c) 2013

## Table of Contents

This code was written to be the be-all end-all for video game audio management! We are always open to hearing your ideas for improvements, suggestions and problems. Email us any time at support@darktonic.com

Demo video at: http://bit.ly/17saxEf (under five minutes and moves briskly).

## 1. Solutions! Master Audio solves the following problems

1) Too many instances of the same audio clip playing simultaneously or near-simultaneously. For example, if an enemy has a death scream sound, you may kill 30 of them with a single blow. 30 audio clips playing simultaneously is not only unnecessary but it drags the CPU down, especially

on mobile devices. Master Audio lets you specify the maximum number of each sound that can be playing at a single time.

2) The ability to randomize a certain sound to actually play from a pool of weighted variation sounds. This goes hand in hand with setting up the maximum number of each sound that can be played. All X sounds in a group can be the same, or variations. You have complete control over that.

3) Having to write code to trigger each sound. Master Audio eliminates the need for this in most cases by letting you specify sounds to play when certain events occur. No coding is needed to do this.

4) Not being able to play sounds when the AudioSource is attached to a game object that is being despawned or destroyed. What happens normally is that you hear a brief blip of the sound and then when the game object is gone, the sound abruptly stops. Master Audio keeps all its Audio Sources in a central location separate from your prefabs so that this doesn't happen.

5) Being able to stop all currently playing instances of a certain sound. Since Master Audio knows where all its Audio Sources are at all times, it can trivially do this with one simple line of code.

6) Adjusting the volume of categories of sound effects with a slider. Unity does not have anything natively, we have included the concept of a pro mixer with buses. More on this later!

7) Music ducking. You can configure Master Audio to have the music "duck" (meaning get quieter and ramp back up) for whichever sound(s) you like with no coding needed. The amount of ducking is also configurable.

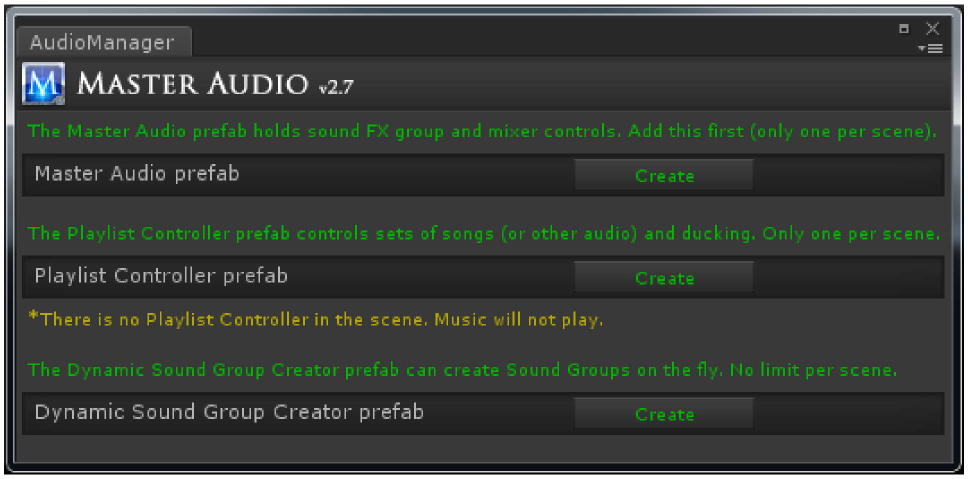
8) Music cross-fading. You can set up multiple music playlists that picks tracks that can shuffle, cross-fade and auto advance.

### *Limitations:*

1) If you are using the free edition of Master Audio Free, it is limited to 6 Sound Groups. Everything thing else is exactly the same as the full Master Audio plugin except that you do not have access to the source code. If you wish to purchase the full Master Audio, buy it here.

## 2. Quick Start: How to set up your scene to utilize Master Audio

1) From the Unity "Window" menu, select "Audio Manager". This is a small window that will help explain and set up what you need for Master Audio. It looks like this:


AudioManager

MASTER AUDIO v2.7

The Master Audio prefab holds sound FX group and mixer controls. Add this first (only one per scene).

Master Audio prefab — Create

The Playlist Controller prefab controls sets of songs (or other audio) and ducking. Only one per scene.

Playlist Controller prefab — Create

*There is no Playlist Controller in the scene. Music will not play.

The Dynamic Sound Group Creator prefab can create Sound Groups on the fly. No limit per scene.

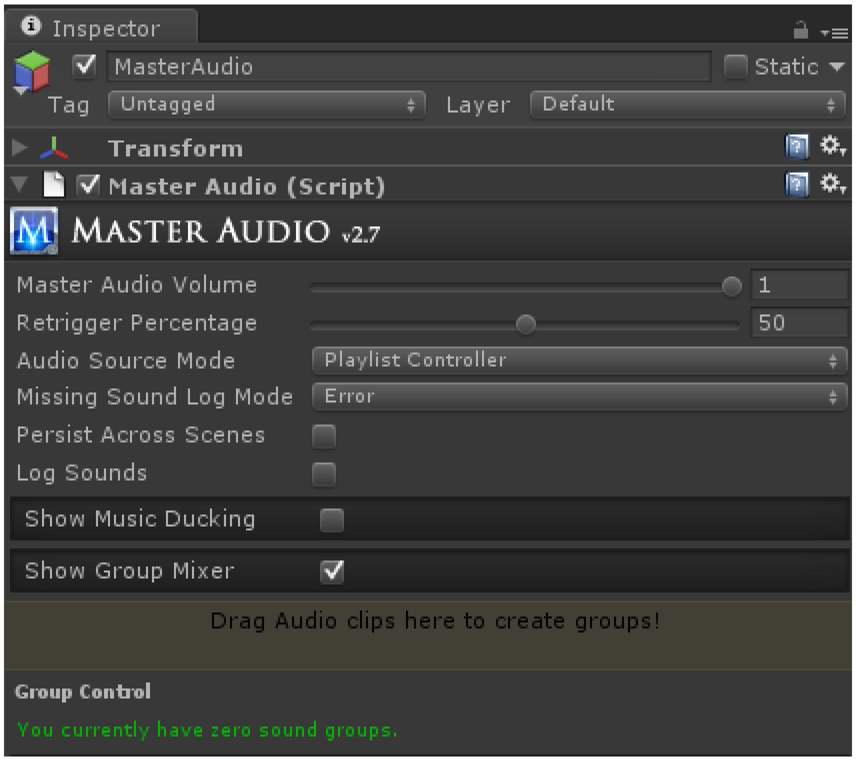
Dynamic Sound Group Creator prefab — Create

2) First, create the Master Audio prefab with the "Create" button. The position of the MasterAudio prefab will be used for all triggered 2D sounds (if you have any), so put it somewhat close to the AudioListener in your scene if you have any 2D sounds planned.

3) To take full advantage of music playlists and cross-fading, you can optionally create the Playlist Controller prefab with its "Create" button as well. This is recommended, but if you have scripts that need to manipulate an existing music AudioSource pitch or volume this may cause conflicts. In that case, you may choose to not use the Playlist Controller.

4) Configuring your first Sound Group.

   a. Click on the MasterAudio prefab in the Hierarchy. Your Inspector should now look something like this:


Inspector

MasterAudio | Static

Tag: Untagged | Layer: Default

Transform

Master Audio (Script)

MASTER AUDIO v2.7

Master Audio Volume: 1

Retrigger Percentage: 50

Audio Source Mode: Playlist Controller

Missing Sound Log Mode: Error

Persist Across Scenes: ☐

Log Sounds: ☐

Show Music Ducking: ☐

Show Group Mixer: ✓

Drag Audio clips here to create groups!

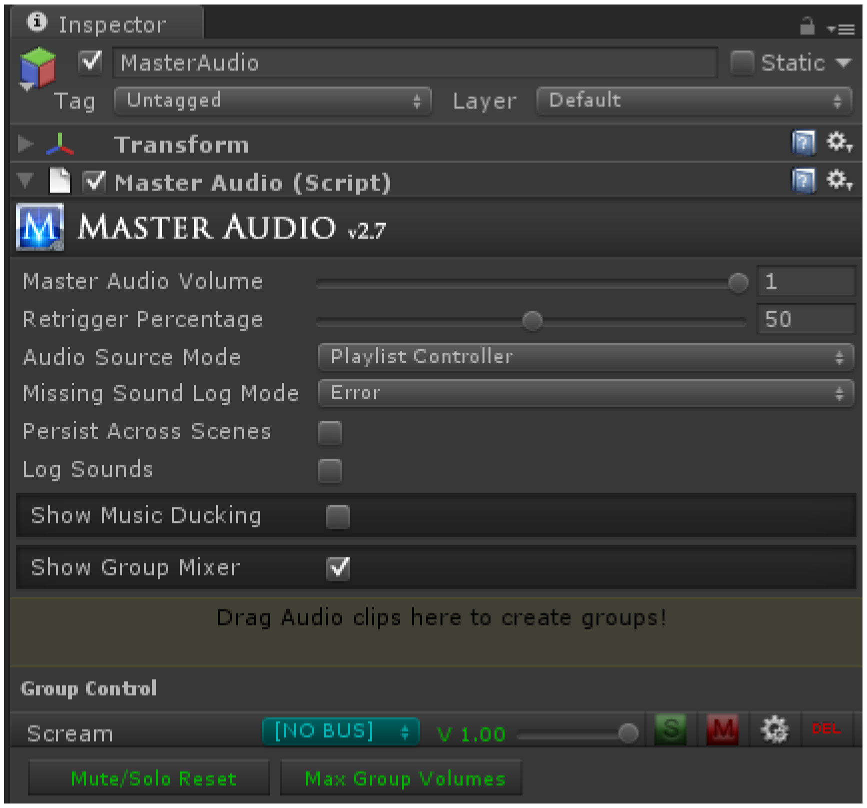
Group Control

You currently have zero sound groups.

b. We're going to use the Group Mixer section to create our first Sound Group.

   i. Drag your Scream audio clip into the colored rectangle area that says "Drag Audio clips...". It will automatically create a Sound Group for you that has controls below that.

c. Now your Inspector should look something like this:



Inspector

MasterAudio — Static

Tag: Untagged — Layer: Default

Transform

Master Audio (Script)

MASTER AUDIO v2.7

Master Audio Volume: 1

Retrigger Percentage: 50

Audio Source Mode: Playlist Controller

Missing Sound Log Mode: Error

Persist Across Scenes: ☐

Log Sounds: ☐

Show Music Ducking: ☐

Show Group Mixer: ✓

Drag Audio clips here to create groups!

Group Control

Scream — [NO BUS] — V 1.00 — S — M — DEL

Mute/Solo Reset — Max Group Volumes

d. Notice that "Scream" now shows up under the last section. If you expand the MasterAudio prefab, you will see it now has a child prefab called Scream, and if you expand that, you will see a child prefab for each variation (only one this time).

e. Each of the variations has an Audio Source component where you can individually tweak the pitch / pan / etc to create different variations. You can also add effects such as Reverb or Distortion to individual variations. Or you can drag entirely different Audio Clips in there for variations as well.

   i. Note: the prefab template for variations can be found in the MasterAudio/Sources/Prefabs folder and is called GroupVariation. If you change any properties on the Audio Source in this prefab, it will carry over into all Master Audio variations created afterward. So change it there first after you get the Doppler settings (and any others) working to your liking.

f. The Sound Groups listed in the "Show Group Mixer" section have a couple buttons for each group.

   i. Go – clicking this will select the Sound Group in the Hierarchy so you can make additional changes.

ii. Delete  icon – clicking this will delete the Sound Group. Sometimes you might want to delete it and recreate it with a different number of children. It's just faster that way.

iii. "S" for Solo. If any Sound Groups are soloed, only the soloed groups will produce audio.

iv. "M" for Mute. This will mute the Sound Group. It will produce no audio while muted.

5) Configuring your additional Sound Groups.

   a. Drag your "blast" sound clip into the Audio Clip field. It will automatically create the "Blast" Sound Group.

   b. Click the gear (settings)icon on the mixer row for Blast. It will now take you to the Group settings.

   c. Change the Weight field to 5. This means that 6 Blast sounds will be able to be played simultaneously. More on this later.

## 3. Triggering the Audio - "no code" methods

Now that you have sounds set up, let's see how to trigger them automatically. There are a few scripts to help out on this:

1. **ButtonClicker.cs** is a script that works with NGUI only. It triggers two MasterAudio Sound Groups. One for when you click the mouse button down, and one for when you release. If you have an NGUI button with a collider, go ahead and attach this script to it. You will notice that the Inspector has two dropdowns in the ButtonClicker section. They each contain the list of Sound Groups in MasterAudio.cs. Go ahead and choose Scream and Blast for the two sounds and try it out by clicking the play button. Now it automatically triggers the sounds for you on these events.

2. **EventSounds.cs** is a script that you can attach to prefabs to trigger MasterAudio sounds for certain MonoBehavior, PoolManager and other events. There are events for:

   a. OnBecameVisible

   b. OnBecameInvisible

   c. OnEnable

d. OnDisable

e. OnCollisionEnter

f. OnTriggerEnter

g. OnMouseEnter

h. OnMouseDown (Mouse Click is what it's called).

i. For PoolManager users, we also have OnSpawned and OnDespawned

**Note:** OnBecameVisible (and OnBecameVisible) will only work inside a prefab that has a Renderer component inside it. In cases where batching will reassign or not use the Renderer (NGUI / 2D Toolkit, etc), you may opt to use the OnEnable / OnDisable events instead (or the Pool Manager events). They don't provide exactly the same functionality but it will work for most purposes.

The Sound part works exactly the same as ButtonClicker's, but there are additional settings as well. You will see a section for each event type in the Inspector if you attach EventSounds to one of your prefabs. Each section has:

i. Sound Group dropdown (as before)

ii. Volume control to make the sound quieter

iii. Delay Sound (seconds) - this allows you to schedule a sound to be played X seconds from now. **This features is only available on Unity 4.X+**

iv. If you have a Shuriken particle system attached to this object, you can emit particles as well with the other two properties there.

v. Additionally, the Trigger and Collision events have layer and tag filters. If you enable these, you can specify which layer(s) and / or tag(s) the object you're colliding with must be to trigger the sound.

At the top are the following Group Controls:

i. Sound Spawn Mode - 3 possible settings.

   a. Master Audio Location: The sound will emanate from MasterAudio's position.

   b. Caller Location: This will trigger the sound in 3D from the prefab's position.

c. Attach To Caller: This will reparent the variation prefab under the prefab that has the Event script. If the caller prefab becomes disabled, the variation will automatically be reparented back into the Master Audio hierarchy.

ii. Disable Sounds: Checking this will disable all event sounds on this prefab.

3. **EventCalcSounds.cs** is a script just like **EventSounds.cs**, with slightly more CPU-intensive operations. It has one event type:

i. AudioSourceEnded - This is only usable when you have an AudioSource component with a sound/music on your prefab. If you do, this can trigger a MasterAudio sound every time the AudioSource finishes playing. If your AudioSource is looped, this will keep happening every time the sound loops again.

## 4. Triggering the Audio - code methods

If you need to trigger any Sound Groups during times other than those provided by the included scripts, you can use the following single lines of code:

1) *MasterAudio.PlaySound(string soundGroupName, string variationName, float delaySoundTime);*

This plays the sound from the position of MasterAudio. 2nd and 3rd parameters are optional. VariationName is optional and lets you play a specific variation (or its clones created from Weight >1) by name. DelaySoundTime lets you schedule a sound to be played X seconds from now (Unity 4.X+ only).

2) *MasterAudio.PlaySound(string soundGroupName,Transform trans, string variationName, bool attachToSource, float delaySoundTime);*

This is the same as the 1 parameter version, but you are passing in the Transform object as well so that the sound will trigger from its position. If you also pass in true for the last parameter, the Sound Group's variation will become a child of the "calling" GameObject so that the sound can "follow" the caller.

-Both of these methods have an alternate PlaySoundVariable method you can use instead if you need to play the sound quieter (you specify the volume percentage).

*3) MasterAudio.PlaySoundVariable(string soundGroupName, float volumePercentage, Transform trans, string variationName, bool attachToSource, float delaySoundTime);*

-This is identical except that you can specify a quieter volume if you choose. All 4 PlaySound methods return a PlaySoundResult object with the following properties:

1. SoundPlayed (boolean)
2. SoundScheduled (boolean) - false unless you scheduled a sound with the delaySoundTime field.

3. ActingVariation (SoundGroupVariation) - this will give you access to the actual variation used, if a sound was played.

You can also use the PlaySoundResult to be notified of when a sound is finished playing like this:

```
        var result = MasterAudio.PlaySound("Scream");
if (result.SoundPlayed) {
        StartCoroutine(result.ActingVariation.DetectSoundFinished(this));
        }
```

Then simply add the following method to the same class to receive the Message Sent:

```
        void SoundFinished(object soundGroupName) {
                // do something, like play an animation!
        }
```

**Note**: You will want to turn on "Never Interrupt Clips" for any Sound Group you plan to use this on to ensure 100% proper function. This saves the code from needing to be needlessly complex.

*5)  MasterAudio.StopAllOfSound(string soundGroupName);*

This will stop all sounds of the given type instantly.

-You can always check the MasterAudio.SoundsReady property through code (returns true or false) if you want to check if MasterAudio has finished initializing. This is only needed in rare startup cases during Awake on objects that are present during Scene load. All the other methods check this anyway to make sure it is true.  Generally try not to trigger sounds during Scene Awake as MasterAudio initializes itself then.

## 5. Controlling the Audio - code methods

New in Master Audio V2.3 are several methods you can call to modify the volume levels and mute/solo switches.

*1) PlaylistController. MasterVolume* - can be read or set. Value between 0 and 1.

*2) MasterAudio.MasterVolumeLevel* - can be read or set. Value between 0 and 1.

*3) MasterAudio.GetGroupVolume(string soundType) - returns a float.*

*4) MasterAudio.SetGroupVolume(string soundType, float volume)*

*5) MasterAudio.MuteGroup(string soundType)*

*6) MasterAudio.UnmuteGroup(string  soundType)*

*7) MasterAudio.SoloGroup(string soundType)* - also unmutes the group.

*8) MasterAudio.UnsoloGroup(string soundType)*

*9) MasterAudio.GrabGroup(string  soundType)* - in case you want to read or manipulate other properties of the group such as limit mode or "neverInterrupt" settings, grab the object here.

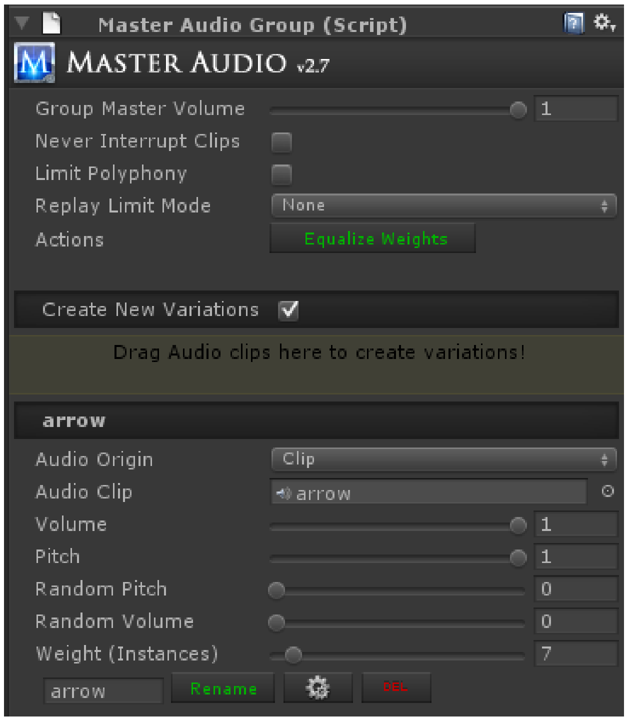
*11) MasterAudio.SetBusVolumeByName(float volume, string busName)* - this can change a bus volume.

*12) MasterAudio.GrabBusByName(string busName)* - you can grab the Bus to read or change its properties.

## 6. Audio Groups: Fine-tuning through the Inspector panel

Additional settings are on each Sound Group (under then MasterAudio prefab). It allows you to quickly change the variation clips and volumes.

1) The Inspector for a Sound Group looks like this:



Group Master Volume 1
Never Interrupt Clips
Limit Polyphony
Replay Limit Mode None
Actions Equalize Weights
Create New Variations ✓
Drag Audio clips here to create variations!
arrow
Audio Origin Clip
Audio Clip arrow
Volume 1
Pitch 1
Random Pitch 0
Random Volume 0
Weight (Instances) 7
arrow Rename DEL

The controls are described below.

a. Group Master Volume. This allows you to make all clips under this Sound Group quieter without having to go into each clip and adjust. Its default setting of 1 is full volume.

b. Never Interrupt Clips. Checking this will make sure that Retrigger Percentage is 100% for this particular sound group so they are never interrupted (see next section for retrigger control explanation).

c. Limit Polyphony - this is an optional setting to limit the number of simultaneous variations that can be played in this Sound Group to a number smaller than the number of variations. If you check this setting, the next setting will appear.

d. Polyphony Voice Limit - this is only visible and active if Limit Polyphony is checked. This limits the number of simultaneously playing variations in the Sound Group. i.e. you can set this to 3 even though you have 10 variations, and only 3 can play at the same time.

e. Replay Limit Mode. This can be used to limit the amount of retriggers of this Sound Group, either by time or frames since the last trigger by MasterAudio. It has 3 modes.

   i. 'None' is the default, which does nothing to limit retriggers.

   ii. Frame Based will let you choose the number of frames to wait before retriggering is allowed.

    iii. Time Based will let you set the amount of time to wait before retriggering is allowed.

f. Equalize Weights button. This will set the Weight of all Variations in this Sound Group to one (equal weight). Weights control how often each variation will be triggered in relation to the other variations. More on this below.

g. Create New Variations – this section is used to create a new variation. You drag 1 or more Audio clips into the colored rectangle to add variations to the Sound Group. This is optional and the variations will be played randomly from a pool.

h. Variation Settings (Clip1 / Clip2 / etc). Here you can quickly fine tune your variations without going into each Variation prefab.

    i. Audio Origin - you choose either Clip (the default) or Resource File. If you choose Resource File, you will type or paste the name of the file in Resources in the Resource Filename field.

    ii. Audio Clip – you can change the Audio Clip of the variation by dragging and dropping here. Only visible for Audio Origin of Clip.

    iii. Resource filename - only visible for Audio Origin of Resource Filename. Do not put the file extension here. i.e. for King.mp3 enter "King".

    iv. Volume - the volume of the clip.

    v. Random Pitch - Here you can specify the max pitch to randomly vary by each time the clip is played, based on the original clip pitch. It will fluctuate up OR down by a number no higher than the value you specify here.

    vi. Random Volume - same as random pitch, for volume instead.

    vii. Weight – you can make each variation trigger more or less often than the other variations by changing this value. For example, if you have 2 variations, and variation A has a weight of 4 and variation B has a weight of 1, then variation A will be triggered 4 times as often. This saves you from having to create more variations than you would need otherwise for duplicates. A weight of zero can be specified to not use the variation but not delete it either.

    viii. Rename – type the new name in the text box and click rename. Remember to not have any duplicate variation names in a single Sound Group.

    ix. Delete button – deletes the variation.

x. Go – takes you to the variation prefab in the Hierarchy, so you can tweak additional settings. The same clip settings can be found on the clip prefab, however if you wish to modify the Audio Source properties itself, this is a shortcut.

i. MasterAudio will automatically play the clips under each Sound Group in random order. It will make sure, taking the variation weights into account, that the "random pool" plays all weighted variations before refilling the random pool. This will mean that you get an even distribution of your weighted sounds over time regardless of application.

**A few words on weights vs. variations.**

MasterAudio will create additional variation children for each weight greater than one once you press Play.

1) if you need the ability to play up to 5 of the SAME sound in a polyphonic manner, use a single variation with a weight of 5.

2) If you need the ability to play different sounds in the same group, use more than one variation. Each variation can have its own weight, which will also have clones created at runtime for polyphonic purposes.

## 7. Master Audio: Advanced Options

Some additional settings on MasterAudio (top-level prefab) are:

1) Master Audio Volume: this will control the volume of all sounds coming out of MasterAudio. The calculation is:

   *clipVolume * groupVolume * busVolume * masterAudioVolume. Buses are explained shortly!*

2) Retrigger Percentage: This is the percentage of the sound that must be played before MasterAudio will allow the child to trigger again from the beginning. i.e. if you leave this at the default of 50, then if all children of a Sound Group are busy playing, but one of them has played at least 50% of its clip, MasterAudio will retrigger the sound from that child.

3) Audio Source Mode: this is automatically set to PlaylistController. If you cannot or do not want to use the PlaylistController because it has two Audio Sources for cross fading, switch this to "Single Audio Source". This will disable the PlaylistController section and functionality, but you can still use the music ducking by adding the MusicDucker.cs script to the GameObject with your single "music" Audio Source. More about this under the Music Ducking section later.

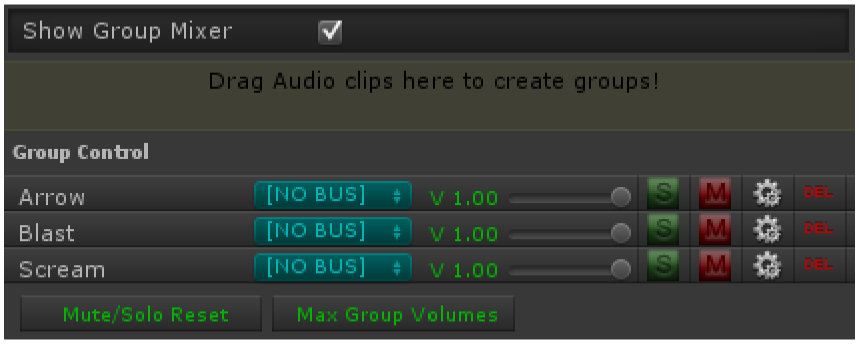
4) Missing Sound Log Mode: Debug or Error, defaults to Error. This controls how Master Audio logs missing sound requests.

5) Persist Across Scenes: Checking this will make it so the Master Audio prefab (and the Playlist Controller prefab if you have one) not be destroyed when loading new scenes. If you are going to use this option, please use a "bootstrapper" scene that only ever occurs once at the beginning. Otherwise you could end up with more than one Master Audio prefab in a scene, which is not allowed.

6) Log Sounds: This will output things to console about which random child has been played, whether there were none available to play, and a lot more. Turn this on for debugging only.

7) To configure the sounds that cause Music ducking, click on MasterAudio in the Hierarchy. There is a section for "Music Ducking Controls". Expand that. To add a sound, click the plus icon, then choose the MasterAudio Sound Groups from the dropdown lists that appear below. That's it! When testing, pay attention to the music getting quieter during those sounds. Note, it may be hard to notice the effect on very quick sounds. Try it on longer sounds. To remove the last sound in this section, click the minus icon.

8) Pro Audio Mixer controls: this section in the MasterAudio Inspector is a mixer for your Sound Groups. It looks about like this:



a. The slider is to control the master volume of that Sound Group.

b. “S” is a solo switch.

c. “M” is a mute switch.

d. The gears icon (settings) will select the Sound Group in the hierarchy so you can tweak variations and additional controls.

e. “Del“ will delete the Sound Group and all variations in it.

f. When "playing" in the Unity editor, the Delete button is replaced by a speaker icon. You can click that to audition the Sound Group whenever you like.

g. Quick buttons for deselecting all mute / solo switches and another for settings all group volumes to 1 (the maximum).

h. For those of you who have not used a mixing board before, here's an explanation of solo and mute switches.

    i. If you have zero soloed groups, all groups will produce sound except the ones that are muted.

    ii. If at least one group is soloed, you will only hear the soloed groups – all non-soloed groups will not be heard.

    iii. Selecting solo will deselect mute, and vice versa.

i. Buses, new in V 2.4.1! The blue dropdown is for assigning Sound Groups to buses. This allows you to control the volume of several Sound Groups at once. In essence, it's a sound router.

    i. By default there are no buses. The text "[NO BUS]" means the Sound Group does not go to a bus.

    ii. To create a bus, select "[NEW BUS]" from the dropdown. A new Bus Control section will show up under the Group Control section. It's still part of the Group mixer section of MasterAudio. You can type into the text field that says "[BUS NAME]" to change the name of the bus.

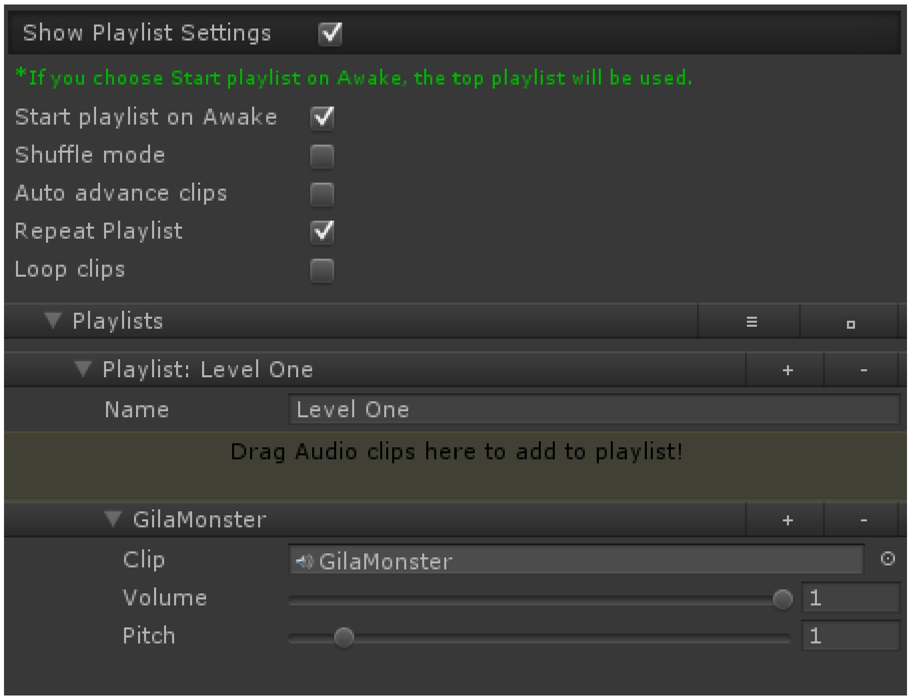
    iii. Each bus has a volume, solo and mute switches, and a delete button. These work as expected, except that the solo and mute switches actually solo or mute all Sound Groups assigned to the bus (to make things less confusing).

    iv. When you have at least one bus created, there will be a checkbox at the top of the Sound Group section to "group by bus". This is very helpful when you have a lot of sounds! It defaults to "on".

    v. Note - all mixer and bus controls now work in real time during Editor play!

## 8. Playlist Controller and Music Playlists

**Music Playlists** - the last section in the MasterAudio prefab's Inspector. Here you can set up multiple playlists of music tracks that MasterAudio uses for your soundtrack.


Show Playlist Settings ✓

*If you choose Start playlist on Awake, the top playlist will be used.

Start playlist on Awake ✓

Shuffle mode

Auto advance clips

Repeat Playlist ✓

Loop clips

Playlists

Playlist: Level One

Name: Level One

Drag Audio clips here to add to playlist!

GilaMonster

Clip: GilaMonster

Volume: 1

Pitch: 1

The settings are as follows (no settings are cross-playlist):

a. Start playlist on Awake - pretty self-explanatory. This will play the first clip in the top playlist as soon as the scene begins. If you have Shuffle mode turned on as well, it will play a random clip instead.

b. Shuffle mode - if you turn this on, tracks in the current playlist will be played in random order. All will be played before the random pool refills. If this is not turned on, the tracks will be played in order top to bottom.

c. Auto advance clips - if you turn this on, as soon as a track ends, another will start playing. It will be the next track if you are not using shuffle, or a random track if you are.

d. Repeat Playlist - if you enable this, the first track will play after the last (only happens on non-shuffle mode). Otherwise, music will stop playing when you run out of tracks.

e. Loop clips - if you enable this, each track you play will loop. Note - **you cannot use looping and auto advance at the same time.**

f. The Music playlist - you can add any number of clips here. They will play from top to bottom if you have not enabled shuffle mode. If you have more than one clip, up and down arrows will appear to change sequence of the clips.

g. To create additional playlists, click the plus icon in Playlist one.

h. The minus icon on a playlist row deletes the playlist.

i. You can move the playlist order (if you have more than one playlist) by clicking the up and down arrow icons.

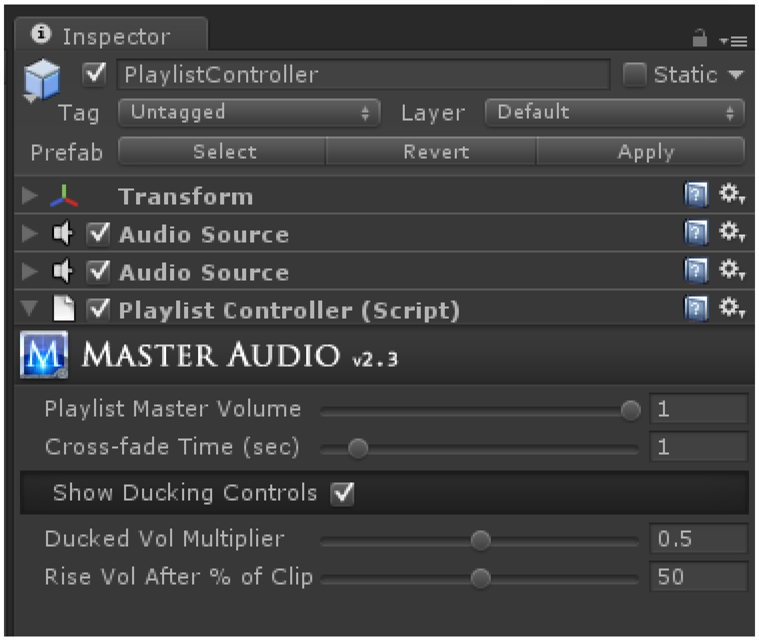
j. You can add songs to the playlist by dragging one or more clips into the colored rectangle.

k. Code options - to control the playlist from code, you have the following options to call:

```
MasterAudio. TriggerNextPlaylistClip();

MasterAudio. TriggerRandomPlaylistClip();

MasterAudio.TriggerPlaylistClip(string clipName);

MasterAudio.ChangePlaylistByName(string playlistName, bool playFirstClip);

MasterAudio.ChangePlaylistByIndex(int playlistIndex, bool playFirstClip);

MasterAudio.NextPlaylist();

MasterAudio.PreviousPlaylist();

MasterAudio.StopPlaylist(); // stops playing the current song and fades out to silence.

MasterAudio.CurrentPlaylistSettings  //this returns a list of the Music Settings.
```

I wouldn't recommend calling the random method if you are not in shuffle mode, and vice versa. The results may be unpredictable.

**Note:** In order to use the playlist and/or ducking features, you will need to put a PlaylistController prefab in your scene. There is a button here to do that named "Create PlaylistController". The Playlist Controller controls are shown below.



Inspector

PlaylistController Static

Tag: Untagged Layer: Default

Prefab: Select Revert Apply

Transform

Audio Source

Audio Source

Playlist Controller (Script)

MASTER AUDIO v2.3

Playlist Master Volume: 1

Cross-fade Time (sec): 1

Show Ducking Controls: ✓

Ducked Vol Multiplier: 0.5

Rise Vol After % of Clip: 50

Playlist Controller settings:

1) Playlist Master Volume - This controls the master volume of all songs in the playlist. The volume of each clip played will be the song's volume (set in the Playlist section in the MasterAudio prefab) * Playlist Master Volume.

2) Cross Fade Time - This controls how long the cross fade will be when changing tracks. If you want no cross fade, set this to zero.

The settings for Music Ducking are:

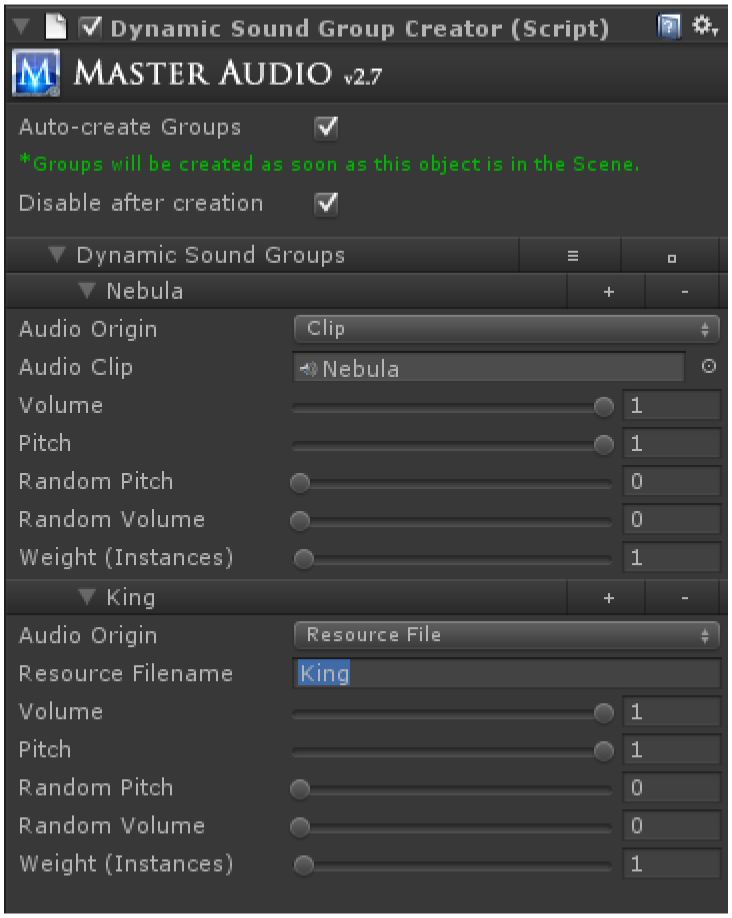
1) Ducked Vol Multiplier - This controls how quiet the music will get when one of your duck-inducing sounds is played by MasterAudio. For instance, if you set it to .5, then your music will go to half volume, then it will ramp back up to normal volume during the second half of the sound being played.

2) Rise Vol After % of Clip – This controls when the music volume ramping back up starts. It defaults to 50. That means that after 50% of the clip that caused the ducking has been played, then volume will start ramping back up over the remaining duration of the clip.

3) If you are not using the PlaylistController and still want music ducking, drag MusicDucker.cs (or add it from the Component -> Dark Tonic menu onto the GameObject with your single "music" AudioSource. The ducking will just magically work.

## 9. Dynamic Creation and Modification of Sound Groups

There is a prefab called DynamicSoundGroupCreator, which can be created from the Audio Manager window. You can attach this to dynamic content that you want to create Sound Groups based on its settings. These Sound Group clips can come from Resource files or other locations.  It looks like this:



Dynamic Sound Group Creator (Script)

MASTER AUDIO v2.7

Auto-create Groups ✓

*Groups will be created as soon as this object is in the Scene.

Disable after creation ✓

Dynamic Sound Groups

Nebula

| | |
|---|---|
| Audio Origin | Clip |
| Audio Clip | Nebula |
| Volume | 1 |
| Pitch | 1 |
| Random Pitch | 0 |
| Random Volume | 0 |
| Weight (Instances) | 1 |

King

| | |
|---|---|
| Audio Origin | Resource File |
| Resource Filename | King |
| Volume | 1 |
| Pitch | 1 |
| Random Pitch | 0 |
| Random Volume | 0 |
| Weight (Instances) | 1 |

The settings are explained here.

1) Auto-create Groups - if you check this, the Sound Groups specified in the lower section will be created in the OnStart method of the script. In other words, as soon as this prefab is Instantiated, it will create the groups. If you do not check this box, you will need to call the CreateGroups method yourself from some script.

2) Disable after creation. If you check this box, the prefab and any subprefabs under it will be disabled after the groups are created.

3) Dynamic Sound Groups section. Here you specify settings for any number of Sound Groups to create. You cannot have any variations per se, but you can make multiple instances of the same clip with the weight property. This section looks and functions exactly the same as the Sound Group

Variation Inspector, except that each item here will create a Sound Group, not a variation of a single Sound Group.

**New in V2.7!** There are now some code methods to modify clips used by variations at runtime:

*1. MasterAudio.ChangeVariationClip(string soundGroupName, bool changeAllVariations,*

*string variationName, AudioClip clip);*

*2. MasterAudio.ChangeVariationClipFromResources(string soundGroupName, bool changeAllVariations,*

*string variationName, string resourceFileName);*

If you use changeAllVariations = true, the variationName is ignored.

There are also new methods to create new Sound Groups on the fly.

*1. MasterAudio.CreateNewSoundGroup (string soundGroupName, AudioClip clip, int variationCount = 1,*

*float pitch = 1f, float volume = 1f, float randomPitch = 0f, float randomVolume = 1f);*

*2. MasterAudio.CreateNewSoundGroupFromResources (string soundGroupName,*

*string resourceFileName, int variationCount = 1, float pitch = 1f, float volume = 1f,*

*float randomPitch = 0f, float randomVolume = 1f);*

All the parameters with values assigned are optional to pass.

## 10. Playmaker integration

I have included the optional "PlaymakerCustomActions" package so that you don't have to write any code to integrate with Playmaker. There are 19 custom actions included, under the Audio category. These should cover every method you would call manually.  This is a list of the custom actions.

1. Master Audio Group Mute
2. Master Audio Group Solo
3. Master Audio Group Toggle Mute

4. Master Audio Group Toggle Solo
5. Master Audio Group Unmute
6. Master Audio Group Unsolo
7. Master Audio Playlist Clip By Name
8. Master Audio Playlist Clip Next
9. Master Audio Playlist Clip Random
10. Master Audio Playlist Next
11. Master Audio Playlist Previous
12. Master Audio Playlist Start By Index
13. Master Audio Playlist Start By Name
14. Master Audio Playlist Stop
15. Master Audio Play Sound
16. Master Audio Set Bus Volume
17. Master Audio Set Group Volume
18. Master Audio Set Master Volume
19. Master Audio Stop All Of Sound

## 11. Final Words

Make sure to check out our other plugins such as Killer Waves at http://www.darktonic.com/p/developer.html. Support is available by emailing support@darktonic.com or in the Unity forums.

Thank you!

-All at Dark Tonic