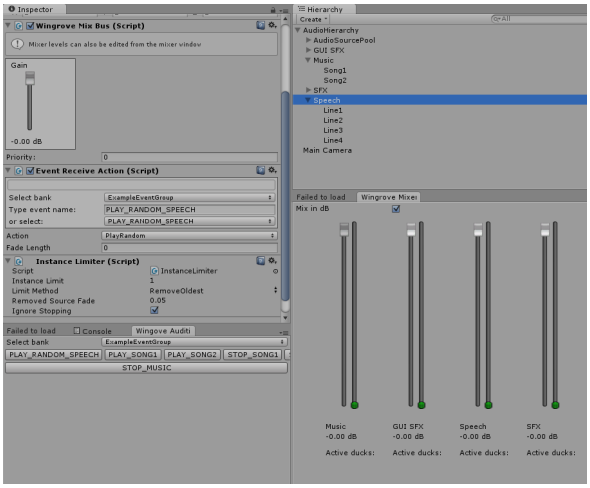
Wingrove Audio 1.08



Wingrove Audio is a complete audio toolkit for Unity, with a powerful set of features to make setting up your game's audio easy. It separates the logic of audio from the logic of the game, meaning your game **code simply has to fire events when the appropriate actions occur**, and your WingroveAudio hierarchy can take care of the rest!

Features:

- **Automatic fades**
- Hierarchy based mixing
- Event driven audio triggering
- Live mixer window
- Quick event testing panel window
- **Audio source pooling, virtualized sounds**
- **Automatic ducking and event based ducking**
- **Instance limiting**
- Load dynamically from Resources
- Easy hierarchy based pause/resume
- Automatic placement of 3D sounds
- Lifetime event triggers (trigger audio with no code)
- Object linked events
- **NGUI event triggers and PlayMaker compatible actions**
- **Multiple listener (e.g. split screen) support!**

In WingroveAudio, you create a hierarchy which holds all the audio in your game. Events are sent from your game to the hierarchy to trigger actions in your hierarchy (e.g. play sound, stop sound, fade, etc...etc...). How an event is handled is determined by the components in the hierarchy.

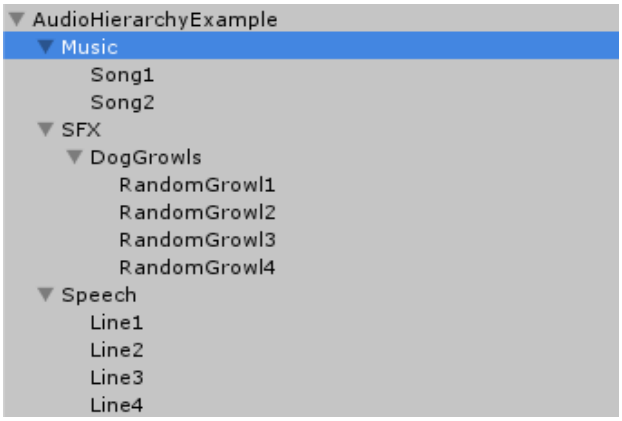
For more advanced tutorials, visit http://www.wickedworx.net/wingroveaudio

For support – e-mail: jon.wingrove@wickedworx.net

Changes in 1.08:

- **New custom roll-off support**
- **DSPTime used instead of delta time (can be turned off, option is on WingroveRoot)**
- **NGUI Hover action selection fix**
- **Adding an optional delay on Event Receivers**

Example hierarchy:



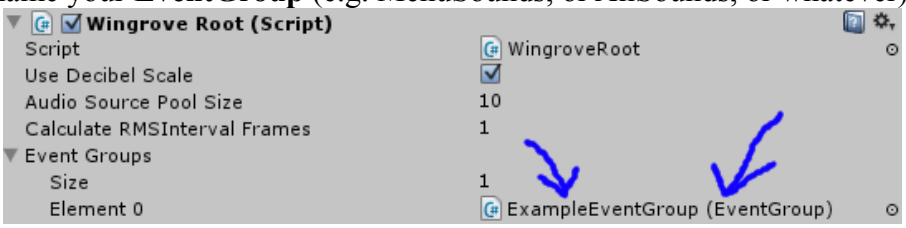
Example code:

```
WingroveAudio.WingoveRoot.Instance.PostEvent("DOG_GROWL"); // send a DOG_GROWL event
```

Getting started:

Creating a hierarchy

- ***Remove any unity AudioListeners from cameras – WingroveAudio uses its own AudioListener. You can add a WingroveListener to cameras/players for 3d sound! If no WingroveListener is added, it is assumed to be at the origin!***
- **Create a new game object in your scene**
  - Name it appropriately (e.g. MyGameAudioRoot)
  - Add a "**WingroveRoot**" component to it.
- Create a new EventGroup **(Assets -> Create -> WingroveAudio -> Event Group)**
  - **EventGroups** contain lists of Events which can be sent to the audio hierarchy
  - Events are used to start, stop or pause sounds
  - Rename your **EventGroup** (e.g. MenuSounds, or AllSounds, or whatever)



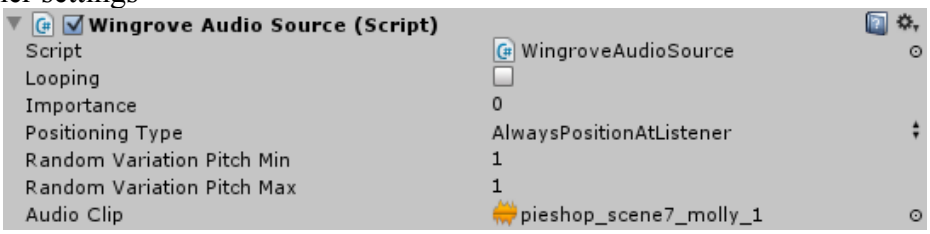
- On your audio root (**WingroveRoot**), add the **EventGroup** to the list of **EventGroups** (see screenshot)

Adding a group

- **Add an empty GameObject underneath WingroveRoot.**
- Name it as you wish (e.g. Music, SFX, etc..) – This is now a group!
- If you wish to give this group a mix level, add a **WingroveMixBus** component to this group object

Adding a sound

- **Add another empty GameObject underneath your new group (note: sounds do not have to be under a group, they can be directly under your root).**
- Name it appropriately (e.g. DoorbellSound, or whatever)
- Add a **WingroveAudioSource** to this new GameObject.
- Drag on your audio clip to the slot on the **WingroveAudioSource** (see picture)
- Set other settings



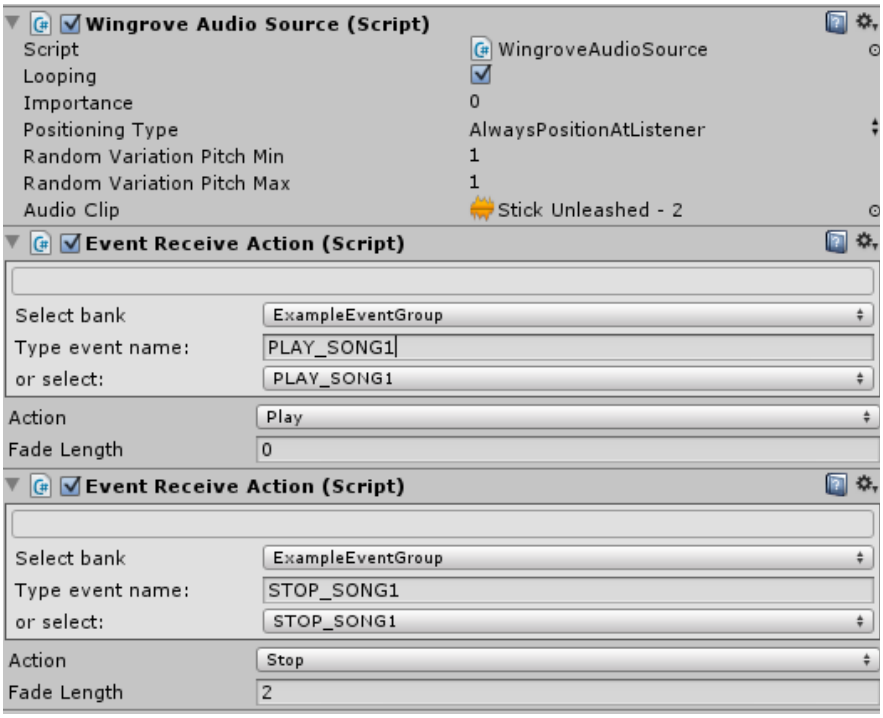
Adding event receivers to your hierarchy

- Add an **EventReceiveAction** component to your game object with your **WingroveAudioSource** on it
- Type in the name of the event that should cause this event to play (e.g. "DOORBELL_PLAY")
- You will see a warning that this event name doesn't exist in the event group - press the button to add it to the event group!
- If your event already exists in an event group - select it using the drop down options - prevent typos!
- For "Action" choose "Play"

In code:

```
WingroveAudio.WingroveRoot.Instance.PostEvent("DOORBELL_PLAY");
```

– your sound will play!!



For more detailed tutorials, please visit http://www.wickedworx.net/wingroveaudio