

Sound [and Fury!]

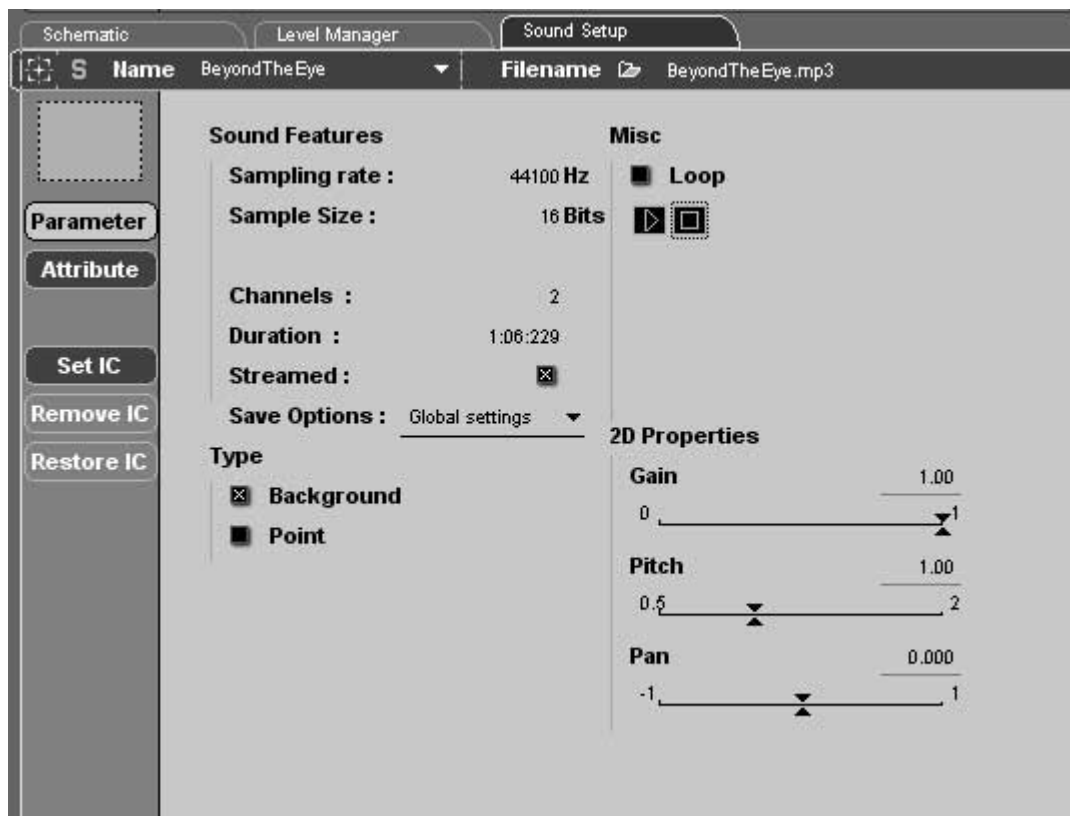
So far in the class we've focused on creating delightful visuals for your interactive masterpieces. Now as you might imagine, it's time to direct our attention to one of our other important senses: Sound. Good sound is essential to creating compelling experiences, and luckily Virtools provides a number of ways to make this happen.

The Basics

Perhaps the most helpful thing you can do when you start experimenting with sound in Virtools is to read the Virtools Sound Whitepaper [the first entry when you search for Sound in Virtools help], which explains the Virtools sound model and related functionality.

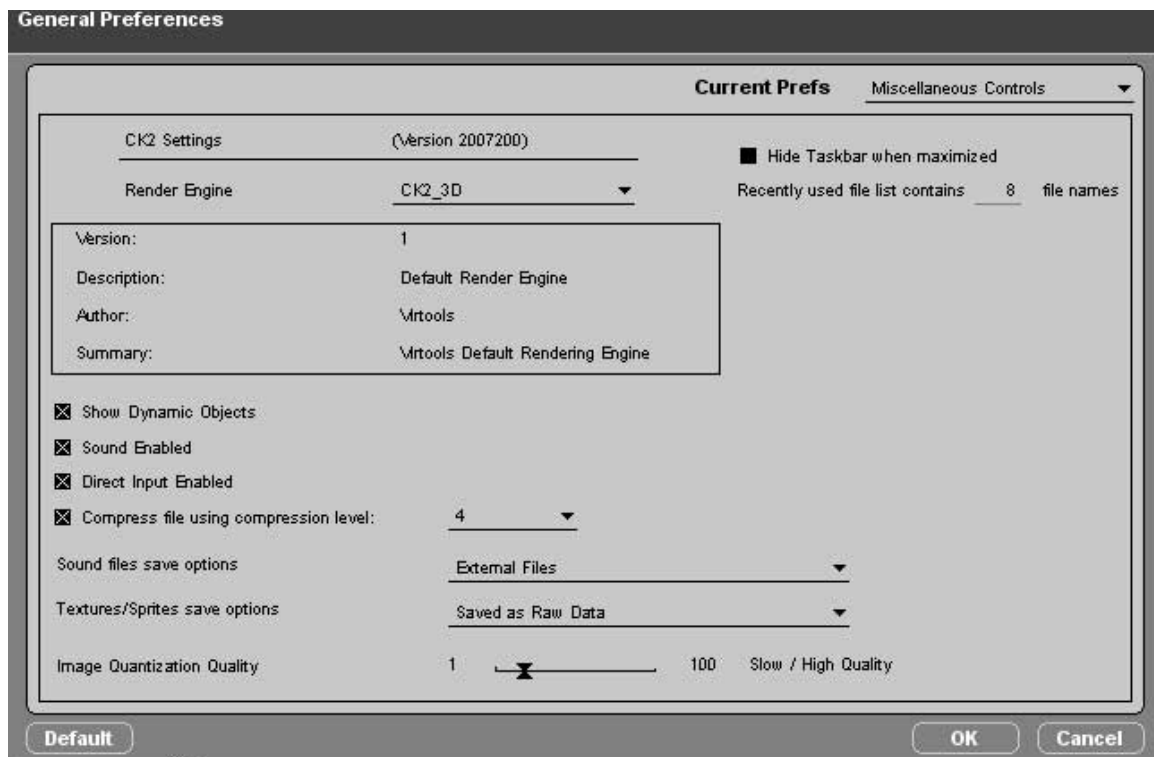
The two most important BB's you will use to get sound to play are Sounds-> Basic -> Play Sound Instance and Wave Player. The essential difference between these two is that Play Sound Instance is for short non-streamed sounds that can be played over each other on multiple channels, for example event-based sound effects. The Wave Player can play large streamed files such as a long background MP3 for your scene.

So the first thing you might want to do to get started with sound would be to open the sound section of the Virtools Resources and under Music drag in their funky BeyondTheEye.mp3. You should see the sound's setup open, and it will look like this:



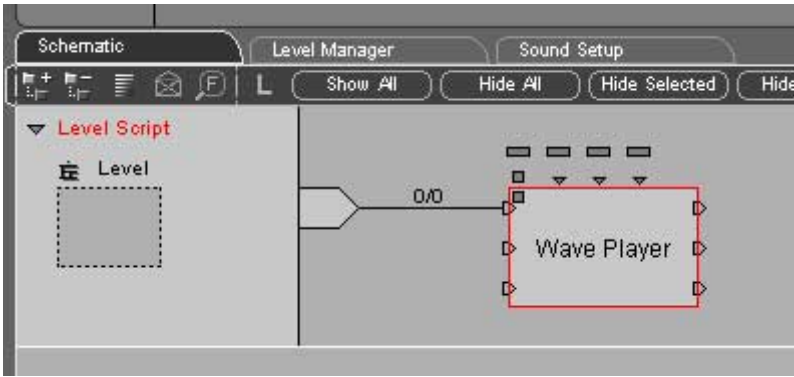
90% of sound problems can be solved by clicking the little play icon, to make sure you can hear the sound. If you can't, it probably means that Virtools has lost the reference to the file. In this case you can either use the "Filename" button to browse to the file, or shut your .cmo, open the correct resource file, and re-open the cmo.

The other thing you can do to help keep track of your sounds is to save them in the .cmo itself. To do this, from the file menu, hit Options ->General Preferences and navigate to the Miscellaneous Controls tab and change the Sound Files Save Option from External Files to "included inside .cmo file." This will keep all sounds inside a give .cmo, so you don't necessarily need to worry about having access to the same file path anymore.



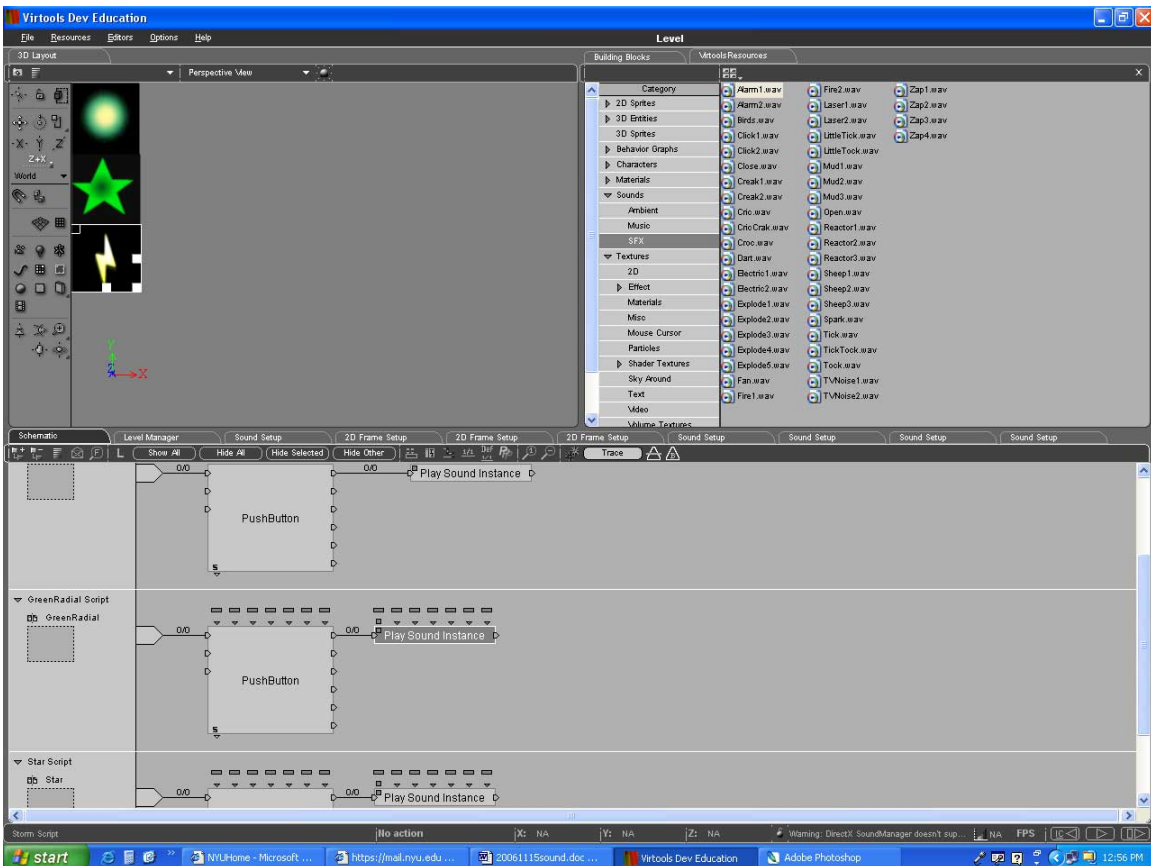
Naturally, this will create extremely large files, and so you should really only do this one time before the show.

Now that you have a sound in your .cmo you can create a level script and drop the Sounds-> Basic -> Wave Player BB onto it, and hook it up like so:



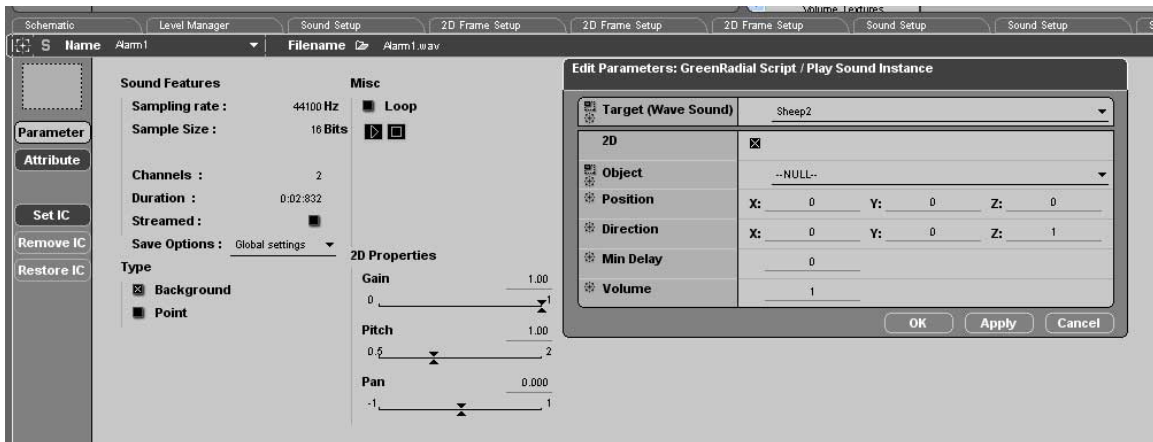
When you hit play, you will hear the fresh stylings of "Beyond the Eye.mp3"

Let's say we wanted to intermix various delightful sound effects with this rocking soundtrack. Drag in several more sounds from the Virtools Resources sound library. Then drag in a couple sprites and add the Interface -> Controls -> Push Button building block to them as we did in the previous lecture. Hook the pressed Bout to the Sounds -> Basic -> Play Sound Instance BB and then hit play. You're scripts should look roughly like this:



You'll probably immediately notice that with the sounds playing you get another little bell sound that indicates sound sort of Virtools error, and that that Output Console is complaining about DirectX9 types changes or something equally abstruse. To fix

these kinds of problems, you need to make sure that the Sound Setup Parameters correspond to the parameters you're using with the Play sound BB's. For example, the first thing you'll want to check is that if you're playing a sound with "Play Sound Instance," you should make sure that sound's "Streamed" check box is **NOT CHECKED**. Also, if you're playing the sound as a 2D sound (as specified in the BB's parameters), then the sound's "Type" checkboxes should be set to **Background**. If you want to take advantage of Virtools 3D sound model, then you have to check, Point in the Sound Setup, and uncheck "2D" in the BB's parameters. The other parameters then give you the ability to specify information about the 3D location of the sound. Below is an example of a sound's setup that is synced with its Play Sound Instance BB parameters and will not generate any errors.

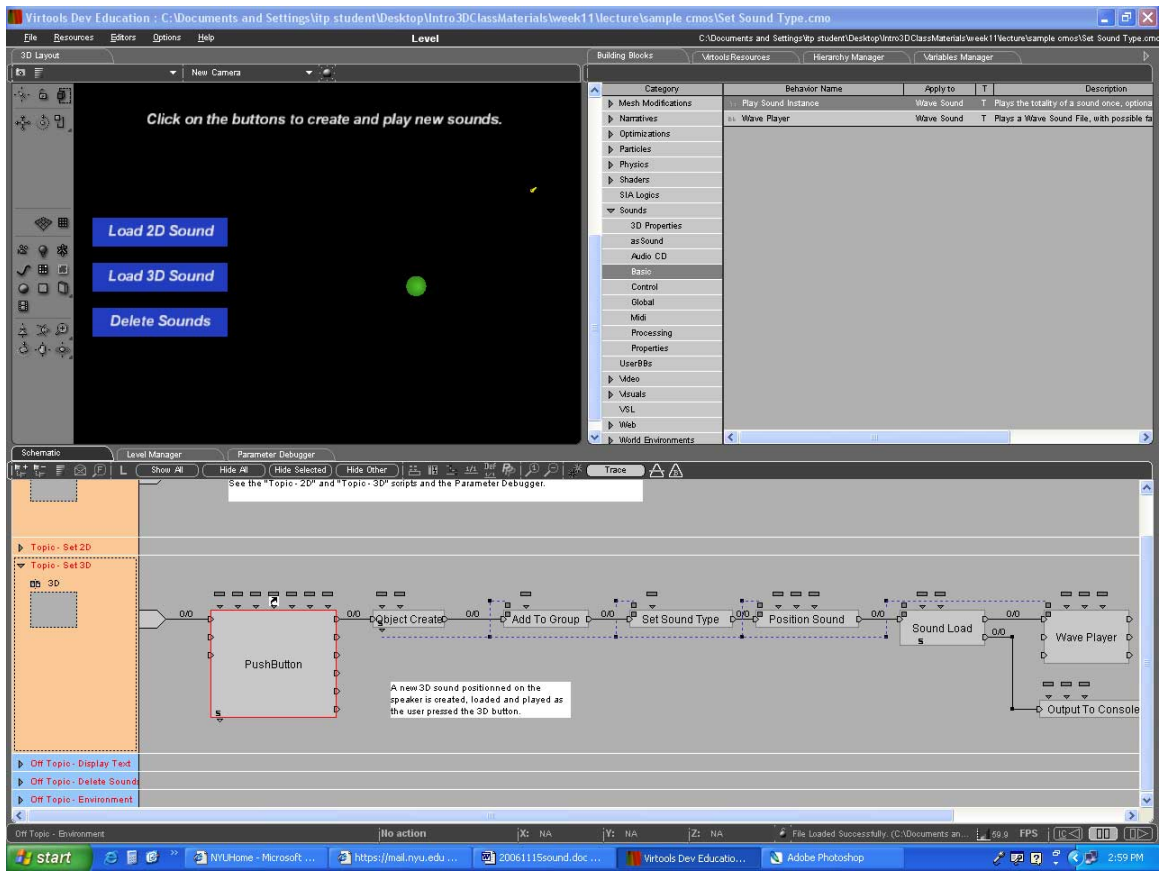


Once you get all the sound parameters organized, play your scene again, and notice that as you press the buttons the sounds both blend together, and also that multiple instances of a sound can be played on top of each other.

Set Sound Type

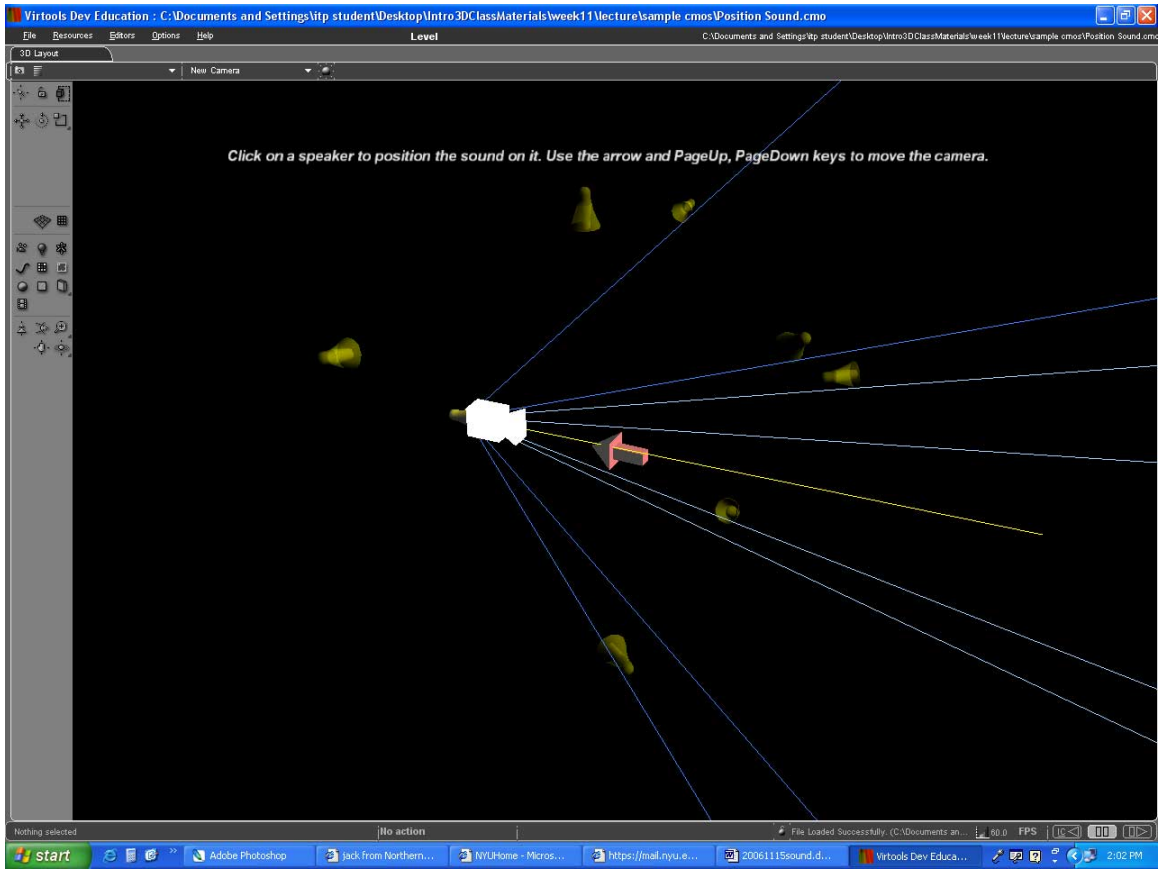
Now that you've set up 2D sounds let's start further exploring Virtools 3D sound model. A good reference .cmo to look at in order to hear the difference between 2D sounds and 3D sounds is the Set_Sound_Type BB demo. In this we see a speaker rotating around a "listener." When you play it and hit Load 3D sound you'll hear the goat sound panning in your speakers depending on the position of the speaker icon. If you have a good surround sound system, you should be able to aurally detect the speaker's position in 3D space.

Now if you hit Load 2D sound, you'll see that the wind sound is independent if the position of the speaker.



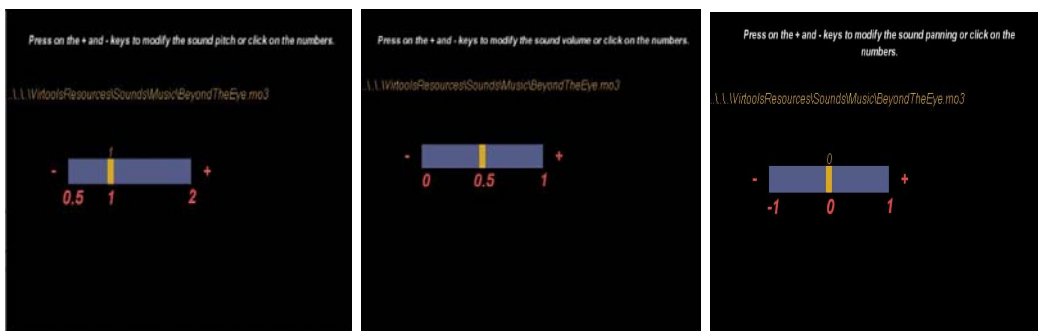
Position Sound

Virtools Position Sound demo is another good way to help familiarize oneself with the program's 3D sound model. Pictured below we see a number of "speakers" positioned in various ways relative to the red arrow symbolizing the listener. Here you can click on any of the various speakers to hear how the sound changes based on the speaker to which it's assigned. Changing the speaker is of course implemented with the Sounds -> 3D Properties -> Position Sound BB, which allows you to Virtools helpfully pops a camera frustum onto the speaker to help you visualize the various sound cone properties. Note that you can further affect the sounds by rotating or translating the various speakers.



Sound Properties

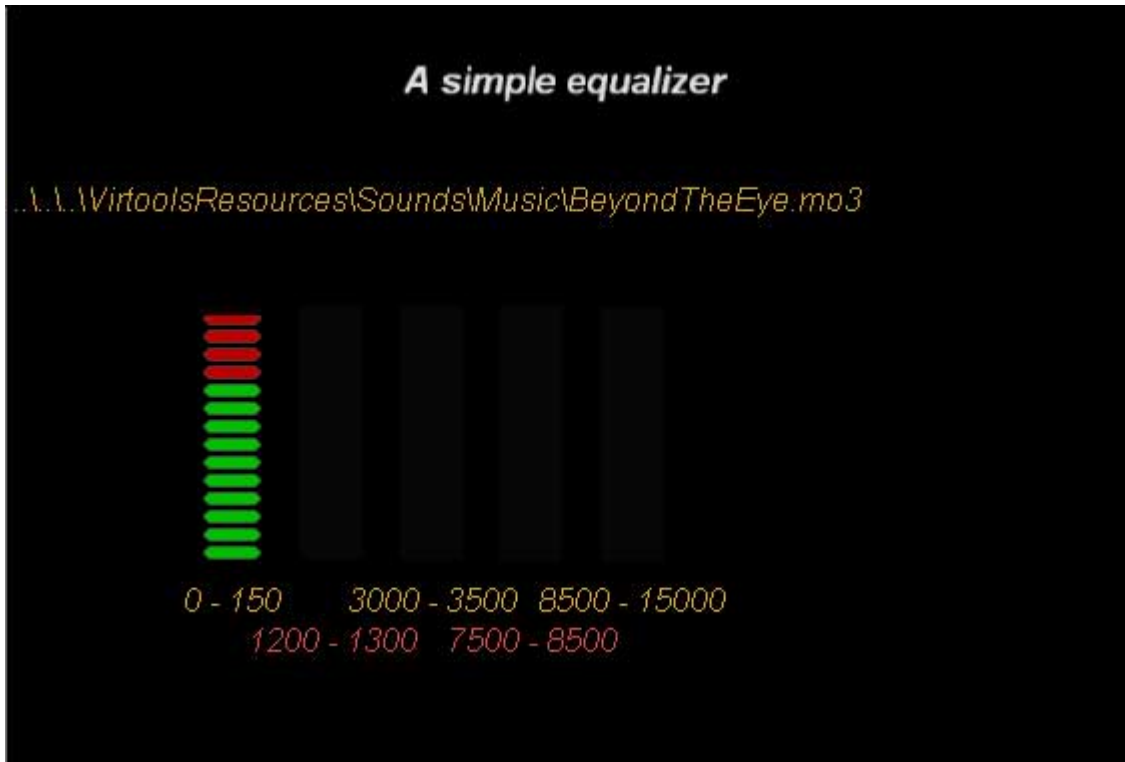
Virtools also provides Building blocks that allow you to access a couple fundamental properties of sounds including, as shown below, pitch, volume, and left-right panning. Note that altering the pitch will make the sound play twice as fast.



Spectrum Analyzer

Finally, another building block that's potentially useful if you want your scene to respond to sound would be the Sounds -> Processing -> Get Sound Spectrum. This component asks you to specify a frequency range and then returns the percentage of

the full sound signal that occupies that range. It can be used as below to create a little equalizer read out, and given that, you could use it to detect beats or other sound events.



Now, armed with these tools, you can create lush soundscapes to accompany your stunning visuals. It's time to end the silence.