Audio Visualizer Tool

Dog Eat Dog Games

Support and more stuff at: [www.dogeatdoggames.com](http://www.dogeatdoggames.com)

Like us: [www.facebook.com/dogeatdoggames](http://www.facebook.com/dogeatdoggames)

Follow us: [www.twitter.com/dogeatdoggames](http://www.twitter.com/dogeatdoggames)

Tutorials/Videos: <https://www.youtube.com/watch?v=toKinLCuOhU&list=PLc2O4sFLm5sTMFejOz20XMo0nvrHTMjbJ>

Contents

What you can do

Quick start

Scenes

Script References

References

License info

**Thank you for your purchase!**

**What you can do**

Add audio visualization effects to your game/application.

* Use the beat to:
  + Call any custom event of your own!
  + Move objects back and forth.
  + Scale/shrink objects.
  + Fade between two materials.
  + Fade between two colors.
* Show audio waveforms
  + Use Unity’s new UI system to display waveforms on panels.
  + Line Renderer waveforms.
  + Pad waveforms.
  + Circular waveforms.
  + Spherical waveforms.
  + Object position waveforms.
  + Object scale waveforms.

**Quick Start**

1. Open AudioWaveforms > Scenes. Run each scene to see the different things you can do!
2. Replace music in each scene with your own audio clips.
   1. In the heirarchy window you should see a gamobject called “AudioSamples” Replace the audioclip on the AudioSource component with your own music.
3. Adjust AudioListener > AudioEventListener parameters until you’re happy with the results (see script references under AudioEventListener for more details)

**Scenes**

* AudioPanel
  + A Unity 4.6 UI style canvas, with waveforms on it. Affected by volume.
* Beat Detection
  + Examples of how to detect those beats!
* Circle
  + A circular waveform with multiple effects reacting to the music in different ways.
* City
  + A city that comes to live with the music.
* DiscoBall
  + Multiple waveform examples that react to the music in different ways.
* Rainbow
  + A combination of pad waveforms and line waveforms.
* Sidescroller
  + A scrolling waveform that could be used as a background.
* Sphere
  + A spherical group of waveforms that react to the music in different ways.

**Script References – Core**

These are the main scripts used to create audio waveforms.

* AudioEventListener
  + Listens to the beat, calls public method in the public OnBeat event.
  + parameters
    - AudioSource - if you have audio sources (Sphere scene). This selects which track you’re listening to for OnBeat events, referenced in the AudioSampler..
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - Sample Buffer Size – buffer this many audio samples, used for beat detection.
    - Beat Threshold - adjusted per song. Lower if you’re not receiving events, raise if you’re receiving too many events.
    - Automatic - automatically adjust beat threshold by tracking audio from the last “samplBuffer” frames.
    - OnBeat() - public UnityEvents can get added here, and are called when a beat is detected.
    - OnFrequencyChanged
      * OnChange – hook in public dynamic float variables here. These values will be changed according to the audio frequency.
      * Min/Max value. Every float hooked in to the OnChange listener, will be changed between these min/max values according to the audio frequency.
* AudioSampler
  + A singleton instance that samples the audio.
  + parameters
    - instance - public static instance of the Audio Sampler
    - Audio Sources - list of audio sources that you want to sample.
      * By default this will grab an AudioSource attached to the same GameObject. This allows easier setup if you just have one audio source you want to sample.
      * If you want multiple audio sources just add them to the list here.
    - Debug - if true, shows audio data being sampled.
  + methods
    - GetAudioSamples(int audioSourceIndex)
      * AudioSourceIndex - which audio source on the AudioSampler are you getting samples from.
      * returns a float[] of the samples taken (multiplied by the audio volume)
    - GetAudioSamples(int audioSourceIndex, int numBins, bool absoluteVal)
      * Like the above method, but returns an array of size ‘numBins’, and potentially takes the absolute value of each sample.
    - GetAvg(int audioSourceIndex, int numSamples, float sensitivity, bool abs)
      * AudioSourceIndex - see above
      * NumSamples - see above
      * Sensitivity - multiplied by the average
      * abs - use absolute value of samples or not (decibal levels samples can be positive or negative).
    - GetRMS() - root means squared
    - GetInstantEnergy() - square and sum audio samples.
    - GetFrequencyVol() - get current volume, within a given frequency range.
    - GetFrequencyData() - return the raw spectrum data in the given frequency range.
    - GetFreqForRange() - return the frequency range values to listen for, with the passed in enum.
* CircleWaveform - move objects in a circle, and in and out using the music.
  + Moves objects in a circle, and up and down with the music.
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting samples from.
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - sensitivity - how sensitive is this script to the audio.
    - objects - The objects you’re going to move around in a circle. Objects should exist in the scene. Typically these are objects with trail renderers and particle systems.
    - rotationSpeed - how fast should the objects rotate, value can be negative.
    - radius - radius of the circle.
    - lerpSpeed - lerp speed related to movement around the circle.
    - useWaveform - move up and down relative to the waveform of the music.
  + methods
    - Boost(mulitplier) - for .1 seconds, boost the rotationSpeed by the passed in multiplier
    - Bump(bool switchSign) - Get the avg decibal level of the audio, and move the radius to equal startRadius\*avg. If ‘switchSign’ is true, the sign of the radius we bump to, will switch between + and -.
* ColorChange - change a material’s colors based on the music.
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting samples from.
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - lowColor - when music decibal level is low, material is this color.
    - highColor - when music decibal level is high, material is this color.
    - sensitivity  - how sensitive is this script to the audio.
    - lerpSpeed - rate of color change.
* CurveWaveform – Child of LineWaveform: display an audio waveform using a line renderer, and an input curve.
* LineWaveform - display the waveform using a line renderer.
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting samples from.
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - points - draw a line between each of these points in order.
    - lineAtt - lineRenderer attributes, like color, width, material, etc.
    - amplitude - height of the waveform.
    - Gizmos size – how big is the gizmos sphere drawn in the Scene view around each point.
    - abs - take the abosulte value of audio samples.
    - OrientPoints() – make each point look at the next point in the list.
    - RenamePoints – rename and number all the points in our points list.
* MaterialChange - lerp between two materials, using the music. (BlendTex shader required)
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting samples from.
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - sensitivity  - how sensitive is this script to the audio.
    - Note: you don’t need lowMat/highMat if the gameobject has a \_Blend attribute in it’s material.
      * lowMat - when music decibal level is low, use this material.
      * highMat - when music decibal level is high, use this material.
    - lerpSpeed - rate of material change.
* Object Position Waveform - move objects up and down, to create a waveform.
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting samples from.
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - objects - objects to move up and down.
    - positionAxis - move the objects along this axis.
    - maxHeight - move objects to this max height.
    - sensitivity  - how sensitive is this script to the audio.
    - lerpSpeed - rate of movement.
    - absoluteVal - take the abosulte value of audio samples.
* Object Scale Waveform - scale objects to create a waveform.
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting samples from.
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - sensitivity  - how sensitive is this script to the audio.
    - objects - objects to scale.
    - scaleAxis - scalethe objects along this axis.
    - maxHeight - move objects to this max height.
    - lerpSpeed - rate of scaling.
    - absoluteVal - take the abosulte value of audio samples.
* Pad Waveform - a 3D waveform made of line-renderers in concentric rings.
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting.
    - numLines - number of lines/rings on the pad.
    - radius - radius of the pad.
    - maxHeight - max height of the pad effects, either ripples or bounces.
    - updateRate - how often the pad effects are updated. Once every ‘updateRate’ frames.
    - rippleColor - color of the ripple waves.
    - rippleWidth – how many lines are in each ripple. Typically 3-5.
    - lineAttributes - lineRenderer attributes, like color, width, material, etc.
    - padType
      * Ripple – animate the inner ring. This state is typically paired with SendRipple() method, which can be called from an AudioEventListener.
      * DampWave - wave played across pad, damped by distance.
      * Wave - wave across the pad.
      * Bounce - bounce rings up and down.
  + Methods
    - SendRipple(float propegationTime) – send a ripple down the pad, that takes “propegationTime” to reach the end of the pad. The ripple height will be determined by “maxHeight” and the current audio frequency.
* Panel Waveform - display a waveform using sprites on a UI panel.
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting samples from.
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - sensitivity  - how sensitive is this script to the audio.
    - sprite - sprite to use for each cell in the waveform.
    - numColumns - columns of the waveform.
    - numRows - rows of the waveform.
    - spacingX - spacing between columns.
    - spacingY - spacing between rows.
    - bottomColor - color of sprites at the bottom, when audio levels are low.
    - topColor - color of sprites at the top, when audio levels are high.
    - updateRate - how often then waveform updates. Once every ‘updateRate’ frames.
* Sphere Waveform - similar to circle waveform, but with a sphere! Move objects around a sphere.
  + parameters
    - audioSource - which audio source on the AudioSampler are you getting samples from.
    - FrequencyRange - the frequency range of the audio we’re sampling.
    - sensitivity  - how sensitive is this script to the audio.
    - objects - objects to move around a sphere.
    - rotationSpeed - speed at which objects are rotated around the sphere.
    - rotationAxis - axis of rotation.
    - radius - radius of sphere
    - lerpSpeed - rate of scaling
    - useWaveform - move the radius of this object up and down relative to the music.
    - rotationType
      * Uniform - rotate around rotation axis
      * Rand - rotate around a random axis
      * Cross - use a cross product of this objects position to center, cross the rotation axis.
  + methods
    - Boost(mulitplier) - for .1 seconds, boost the rotationSpeed by the passed in multiplier
    - Bump(bool switchSign) - Get the avg decibal level of the audio, and move the radius to equal startRadius\*avg. If ‘switchSign’ is true, the sign of the radius we bump to, will switch between + and -.

**Script References – Miscellaneous**

These are small scripts used in the demo scenes.

* CameraCircle - rotate the camera around a target
  + parameters
    - target - transform we rotate around
    - rotationSpeed - speed of rotaion
    - rotaitonAxis - axis of rotation
* CameraMovement - Moves the camera right in the Sidescroller scene.
  + parameters
    - speed - movement speed
    - lerpSpeed - lerp between current and desired position at this rate
* Object Circle- place objects evenly in a circle’s radius.
  + parameters
    - objectsToplace - objects to move around a sphere, typically particles or objects with trail renderers.
    - radius - radius of the sphere.
* Object Sphere - place objects evenly in a sphere’s radius.
  + parameters
    - objectsToplace - objects to move around a sphere, typically particles or objects with trail renderers.
    - radius - radius of the sphere.
* Particle Controller - call particle system.play at a given rate
  + parameters
    - particleSystems - the particle systems we want to use.
    - updateRate - how often effects are played. Once every ‘updateRate’ frames.
* Rotate - rotate this object out of it’s up axis.
  + parameters
    - speed - rotation rate.

**Credits**

Programming and Effects: Kurt Hollowell

Audio: Austin Williams, Devin Williams

BlendTexture shaders:<http://wiki.unity3d.com/index.php?title=Blend_2_Textures>

City Model: <http://www.turbosquid.com/3d-models/cartoony-buildings-max-free/730644>

**License Info**

Subject to Unity’s EULA: <http://unity3d.com/legal/as_terms>

All music composed, arranged, and produced by Austin Williams. © 2015 Austin Williams, and Devin Williams