“Hi Tarik,

I have done a bit of work with Unity3D. Essentially, you'll set up Unity to act as an "external / non-ChucK synth" -- i.e., a process that listens for a stream of control parameters output by Wekinator. You'll listen on port 12000 by default, for an OSC message with the message name "/OSCSynth/params," with each control parameter attached as a float.

See<http://wiki.cs.princeton.edu/index.php/ChucK/Wekinator/Instructions#Implementing_your_own_synthesis_class_outside_ChucK> for the general instructions on this.

Unity-specific instructions:

I'm attaching a sample Unity scene, "Hello3.unity", that can be controlled by Wekinator. This scene has an invisible object in it, called OSCEmpty, which has a set of scripts attached to it (including Osc.cs and UDPPacketIO.cs), which are necessary for OSC to run with unity. This object also has a custom OSC script attached to it, called "OscReceiver.js." This script listens for the /OSCSynth/params message string from Wekinator, then unpacks the message and uses the 2 control parameters to change the behavior of other objects in the scene (i.e., the rotations of Cube1, Sphere1).

In order to get this example running with Wekinator, you need to tell Wekinator to use an OSC synth with 2 continuous (non-discrete) parameters. Make sure you can get the Wekinator to work with one of the "quick and easy walkthroughs" on the Wekinator wiki first, so you'll know how the software basically works. Then hook it up to unity and try creating examples (using whatever inputs/features you want to use to control Unity) with control parameter values in the range of -20 to 20 for both parameter 1 (cube rotation) and 2 (sphere rotation).

Good luck!

Rebecca

“