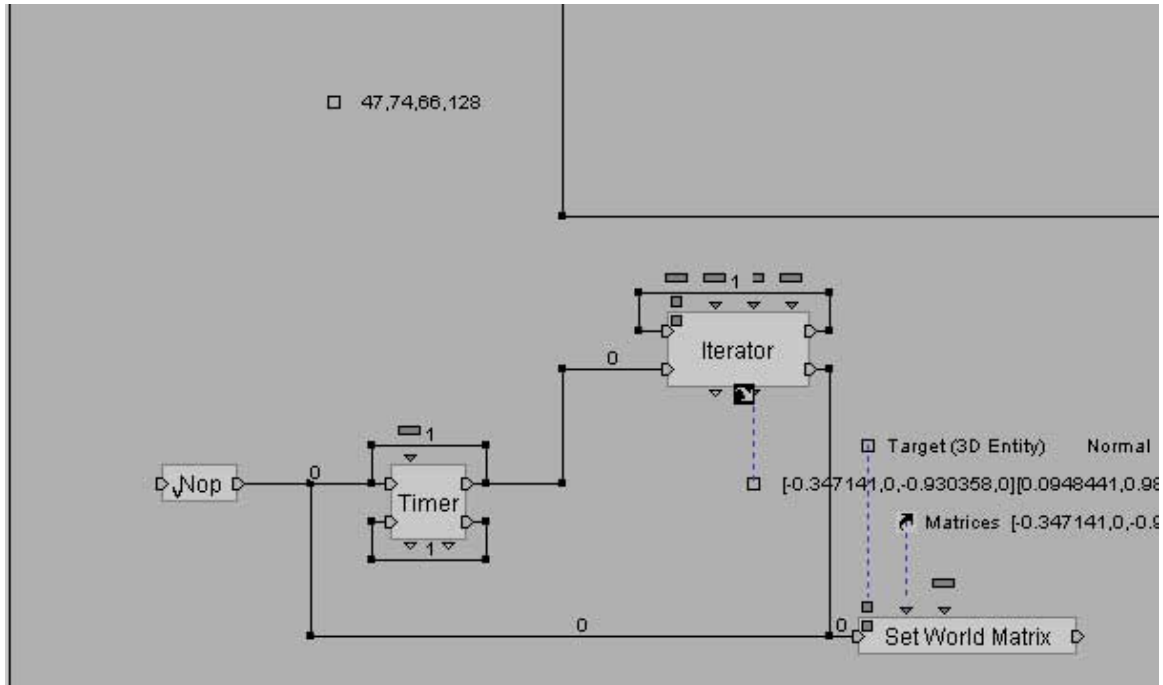


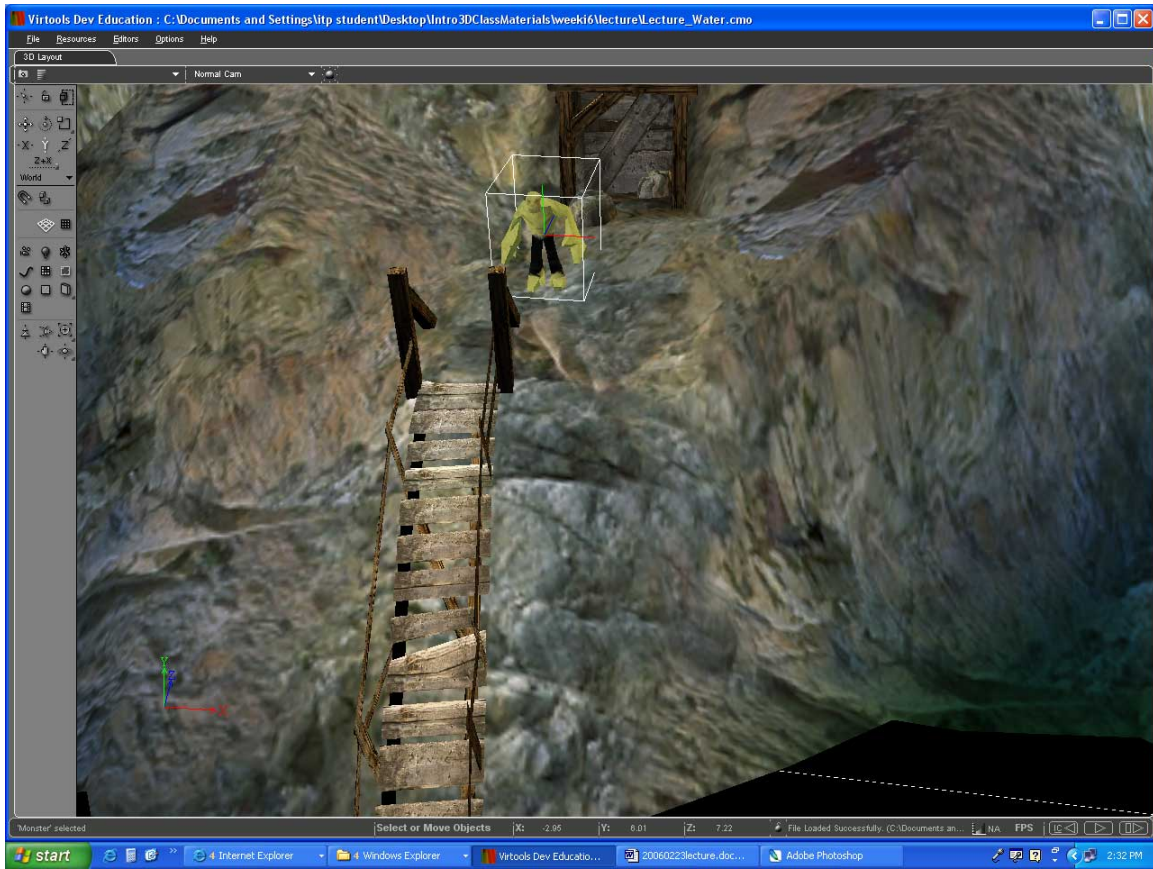
Pulling the Strings

Now that we know how to set up, animate, and control a character, let's look at some issues bearing on creating compelling interactive environments for them.

Let's begin by loading a scene and populating it with our lovely character. My favorite virtual place right now is Virtools' "water demo." You can find this in \\3d\3d\Olson\week6\lecture\Lecture_water.cmo. I begin by just disconnecting the camera changing script so that we have a consistent viewpoint.



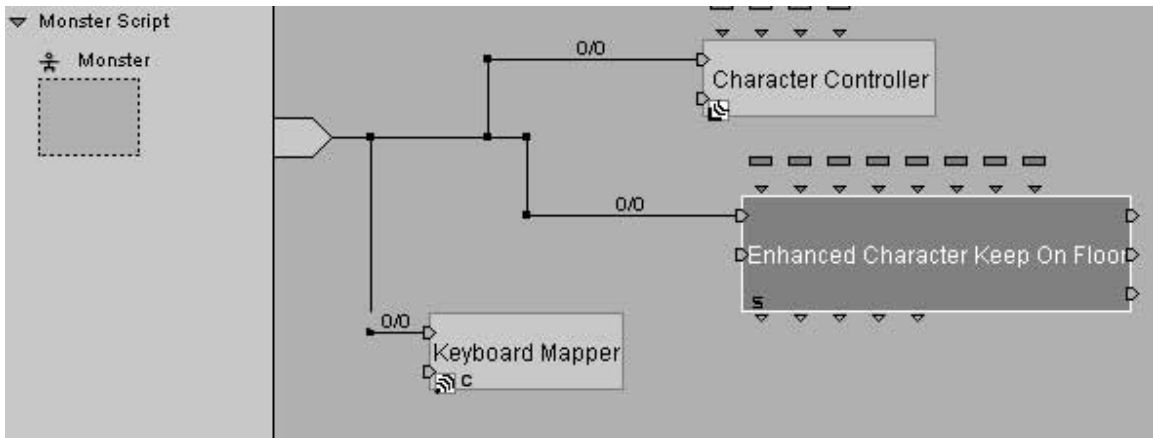
Now let's drag in a character from the Virtools Resources. I like the "monster" character from Characters -> Skin Character. Make all the other objects in the scene invisible from the level manager, and find your monster. Move him to the stone platform at the right of the bridge, and set his IC.



He will come in with basic “stand” and walk animations, so add a script to him including a Character Controller and a Keyboard Mapper as we did in the previous lecture to allow him to move around.

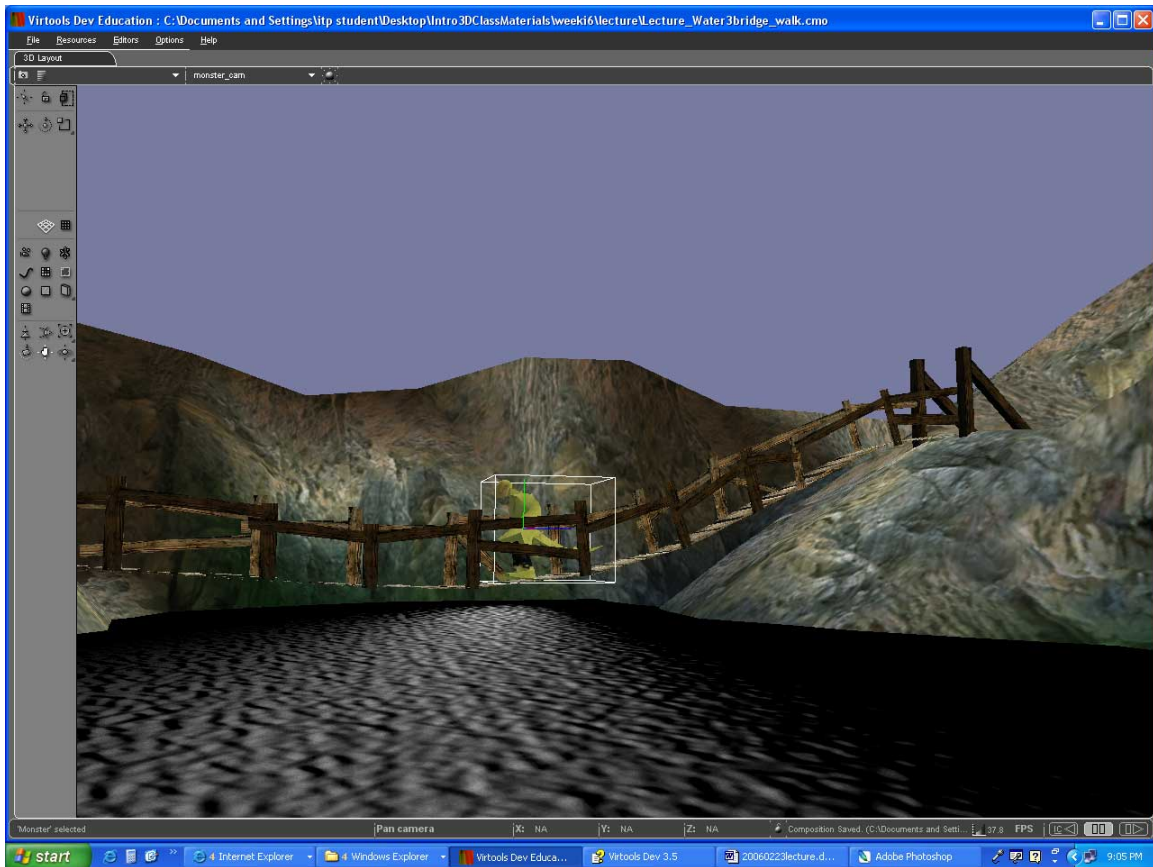
Keep on Floor Part Deux

You’ll notice that he’ll walk out over the water, so again, as before, we’ll add the “floor” attribute to the landscape and sol_pont (bridge) elements. So your Character script should look about like this:



Now if you walk him around, you'll see that he sticks to the rocks pretty well, but we can't seem to get him up on the bridge. This is because he's already on a "floor" and doesn't "see" the bridge as it starts. To ameliorate this condition, check the keep on floor BB's parameters. Change the Detection Offset parameter to something greater than 0 [I used 1]. Virtools defines this parameter to be: "Detection Offset: distance from the bottom of the object after which a character will take a floor above, rather than a floor below. It is also the maximum height, in the world's Y axis, that a character can climb."

Now you should be able to drive him out on the bridge [and drive him off if you're feeling vindictive].



Obstacles

Given your users' proclivities for careening drunkenly around your delicate environments, it's lucky that your precious characters are endowed with the superhuman ability to pass right through putatively solid matter. However, this phenomenon may begin to act at odds with the unimpeachable realism you're trying to create. This brings us to the need for obstacles. They operate in a pretty similar way to the floor manager.

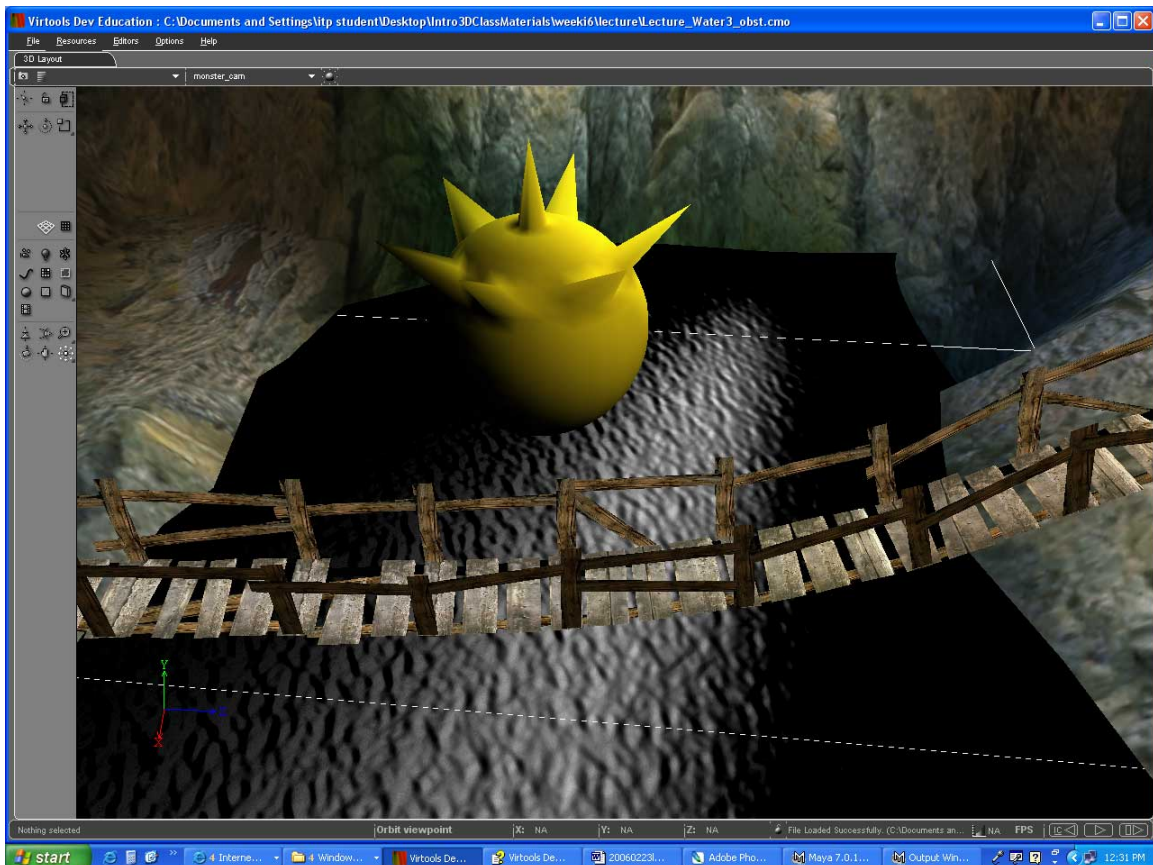
First off, create a new group called "obstacles." Then drag into it: UpontD, UpontG, and pont_barriers1 & pont_barriers2. Now you can add the "object slider" building

block to your character. Select "obstacles" as the group parameter. Now if you try to enter the bridge, you'll see that you are denied. This is because the sphere around your character is too large to fit between the gate. Try decreasing the radius parameter to 0.5. Now you'll see that you can still walk across the bridge, but will bounce along the side. Note the object slider is not always perfect, and can take some careful arrangement and tuning. For example, you'll see that our monster will occasional jump over the side of the bridge now. This has to do with the fact that the side planks are pretty narrow, and your monster's spherical "object slider" radius will sometimes miss them. To fix this, it would probably be best to model solid borders to the bridge, import them into Virtools, and just make them invisible.

Proximity / Collisions

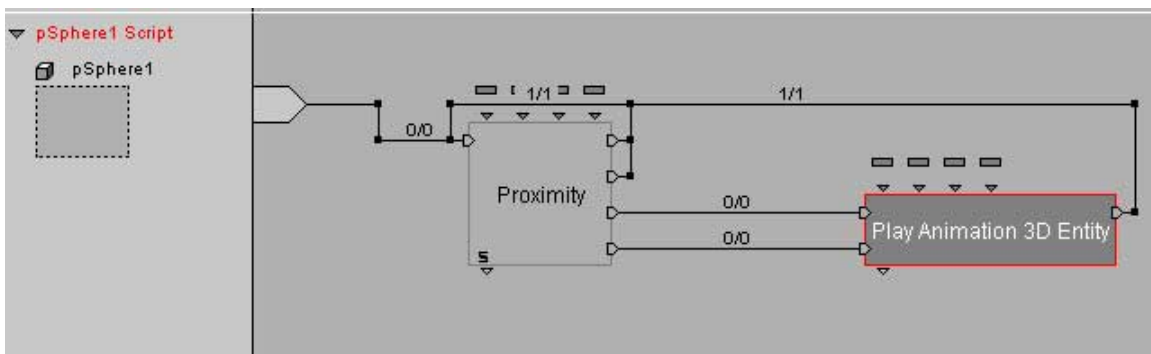
Frequently, in the service of making your world interactive, you'll want to time certain events based on your character reaching certain positions in the world. For example, let's take one of our morphing animations from the tutorial last week. Perhaps we'll put a glow on our spiky sphere, making it into the Orb of Ultimate Power!

So import it into Virtools, set IC for both the Object and its Mesh, and add a script. You might want to grab the 3D Transformations -> Animation -> Play Animation 3D Entity and have it play at the beginning of the scene, just to make sure the animation plays correctly. Viola:

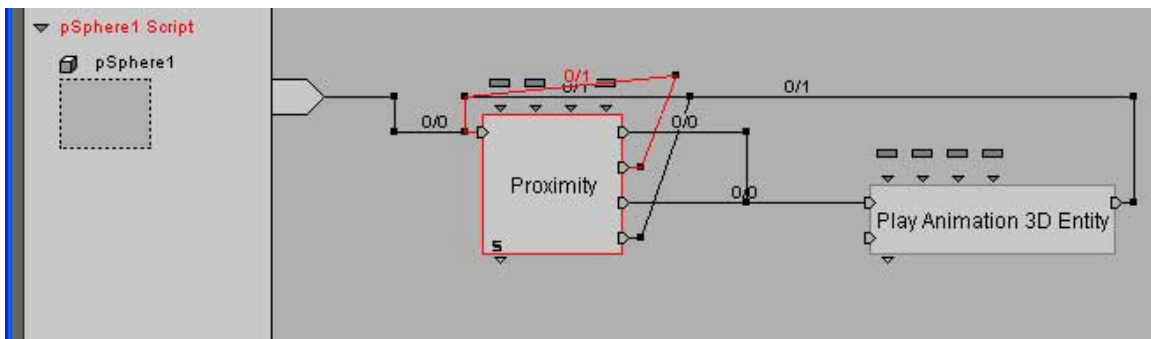


When it does, disconnect the BB, and let's make it so that the orb only pulsates when our monster gets close. There are lots of ways to do this we'll look at both pseudo physics collision detection and physics based collisions later in the semester, for now, a simple and computationally cheap way of simple collision detection is to use a Proximity Sensor.

Proximity sensors [oddly categorized under Logics -> Test], can be slightly confusing. The main principles to keep in mind when dealing with them are 1) they must be externally looped *for all cases*, if you want them to function continuously, and 2) pay careful attention to what you're doing w/respect to the various output transition states. So in this case we're going to simply loop it back on itself in all the cases except "enter" and "exit" range, which will initiate and stop the animation respectively. Note: after the animation operation you must then LOOP BACK to the proximity BB, or else it will stop working.



I found 8 to be a good value for the "distance" parameter. You might also want to note how the sphere's behavior changes if you uncheck the "loop animation" parameter in the Play Animation BB, and reconfigure the Prox like so:

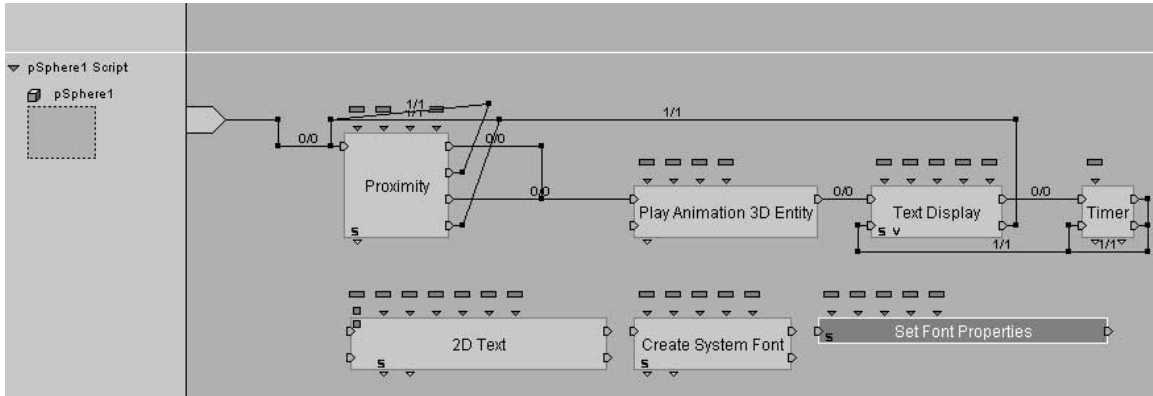


Now you won't ever leave the sphere in its spiky state when you leave the proximity radius.

Text Display

From time to time you'll want to incorporate text into your scenes as part of the interface, or as an expressive element. There are lots of ways to do this, including making sprites w/ jpg textures appear and disappear, but Virtools also has a built in text display system. Let's make an expression pop up when we encounter the Orb of Power. To do this, drag into your sphere script the Interface ->Text -> Display Text

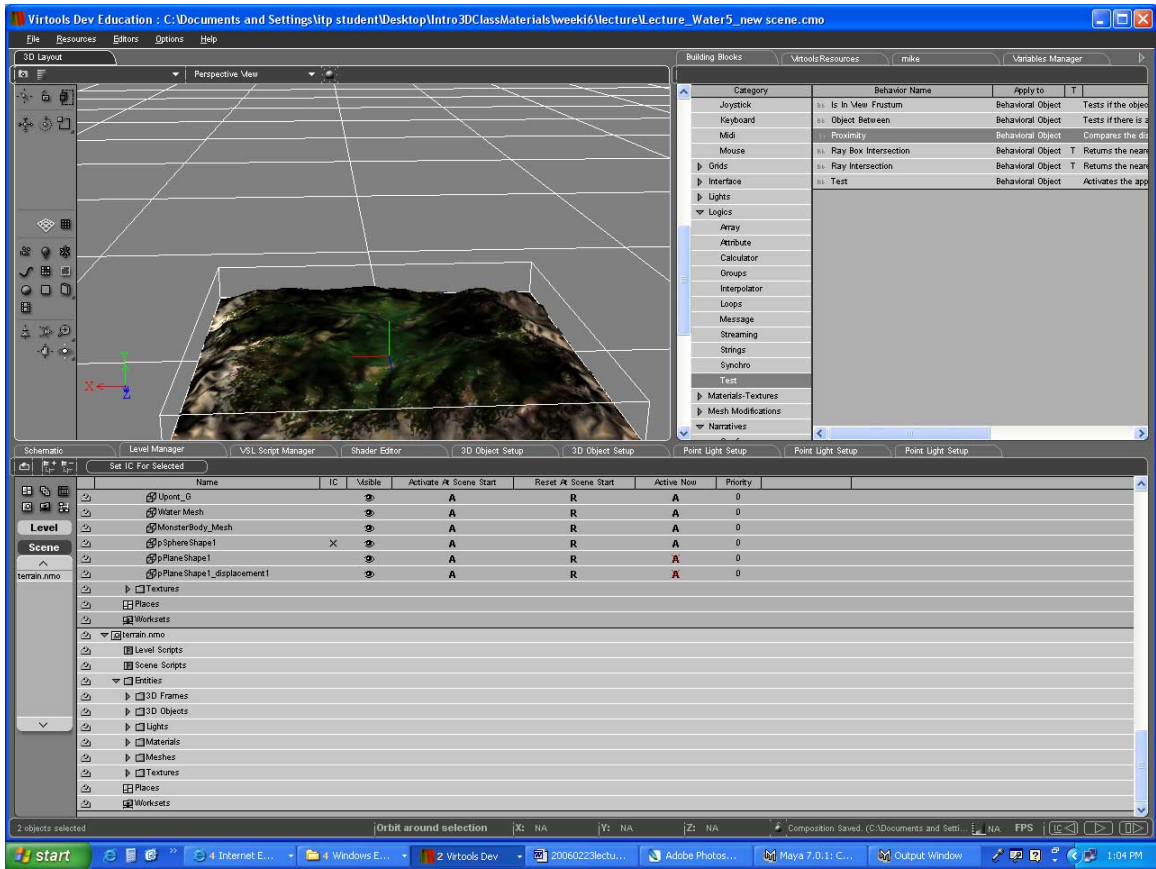
BB (and a timer if you want the text to only appear for a short time). Set the parameters according to your desires, and watch your message pop up when you come near the sphere. Your script will end up looking something like this.



I also included in this picture a couple other helpful BB's for dealing with text.

Merging scenes

Now let's say our little morph animation doesn't seem quite impressive enough for the Orb of Ultimate Power. Perhaps we want it to transport our little monster to a COMPLETELY DIFFERENT WORLD! Naturally maintaining all the objects for 2 whole universes in your composition and having to individually show and hide them would be quite tedious. Virtools has a couple helpful techniques for managing such complex multidimensional worlds. We'll cover Places a little bit later, but for now, let's talk about Scene Management. Scenes are primarily a mechanism for managing discrete time-based events in your world (such as transporting from one distinct area to another). The easiest way to set up scenes is to import your .nmo world files as Scenes. For example let's grab a terrain we created earlier and import it as a scene.



It's subtle, but you should notice two things happen in your level manager: 1) there will be a little scene icon down at the bottom of your list, that if you expand, you should see all the elements of your new scene. There also appears an entry in the scene list to the far left of the level manager. If you click on it, the items in your original scene will disappear, and the new ones take their place.

Scene Management

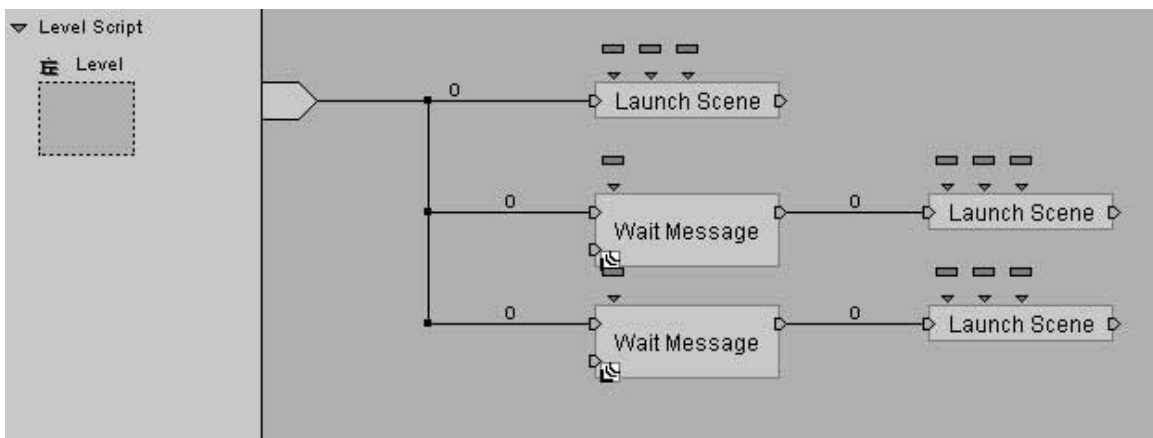
In order to switch from scene to scene at run time, you need to have all the stuff in your original .cmo categorized as a scene as well. So create a new scene in the lower level manager create panel. Call it bridge. Usually when you decide that you're going to create a scene based experience, you'll import all your assets as scenes from the beginning, so you won't have to do this next step. Basically, what we have to do now is drag all or our original composition assets into the new bridge scene. You then will have to 1) make sure the scene still looks ok 2) reset IC for the objects that need it. Now you should be able to toggle between one scene and the other from the scene list. Let's also copy our monster character so that he appears in the terrain scene as well.

That's all well and good, you might say, but how do we switch scenes at run time? To do this we'll do two things 1) make sure we're in bridge world upon start up and 2) switch to our terrain scene after the Orb or power gets triggered. To do the first part, just create a Level script at the top Level (outside of both of your scenes). Grab the Narratives -> Scene Management -> Launch Scene building block, and have it

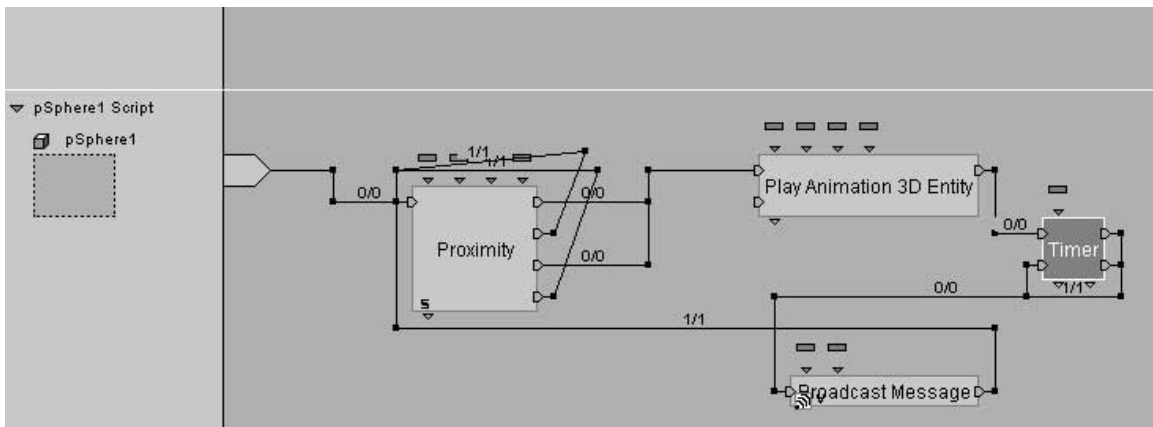
launch the bridge scene on startup. The second part is a little trickier and involves the magic of Messages.

Messaging

Messages at their simplest are simple strings that you can broadcast around Virtools. At their simplest they can consist of simply the name of a message, but you can make them have a name, value, and data if you want. For the time being we'll use the simplest form. We'll do our scene management from the top level script, so grab 2 Logics -> Message -> Wait Message building blocks and drag them into the level script. Make their message parameters "launch_bridge" and "launch_terrain" respectively. Then add two more Launch Scene BB's triggered by the wait message BB's , having them launch the appropriate scenes.



Now all we need to do is go back to our sphere script and add a timer (so we can watch the animation for a second) and a broadcast message BB after the proximity sensor. Have it broadcast the launch_terrain message.



Now if you test your .cmo, you should see the scene change. Lamentably your monster will fall to his death unless you set up a floor in the scene, and make sure that when you change the scene, you position him on it. You also might want to include some way to get back to the bridge scene. I'll leave that as an exercise for the reader.

Debugging

As our scripts are becoming more complex, I just want to include a quick note on debugging them.

- 1) I trust you're all familiar with the trace functionality by now, however, even when it's on, it can occasionally be difficult to see when various links are getting executed. If you're having this difficulty, it can be helpful to increase the link delay on the connection in question (RMB the link & select "Edit Link Delay" from the context menu). Then when the link gets executed, you'll see the little countdown happen over a longer period of time.
- 2) Now that we know how to display text to the screen, it can be very helpful to send your variables there so that you can watch them as your composition unfolds. The Logics -> Strings -> Output to Console BB can also be helpful in recording what's going on in your scene.