

Let's Get Physical!

Virtools provides the amazing ability to endow your worlds with a reasonably good simulation of real world physics with the addition of a few building blocks. As we will see, this system becomes a bit of a black box, but I hope you'll find that it works reasonably well in many circumstances.

The Physicalize BB

Let's begin with the Virtools Getting Started Physics demo. C:\Program Files\Virtools\Virtools Dev 3.5\Documentation\Physics\Physics Database\GettingStarted.cmo.

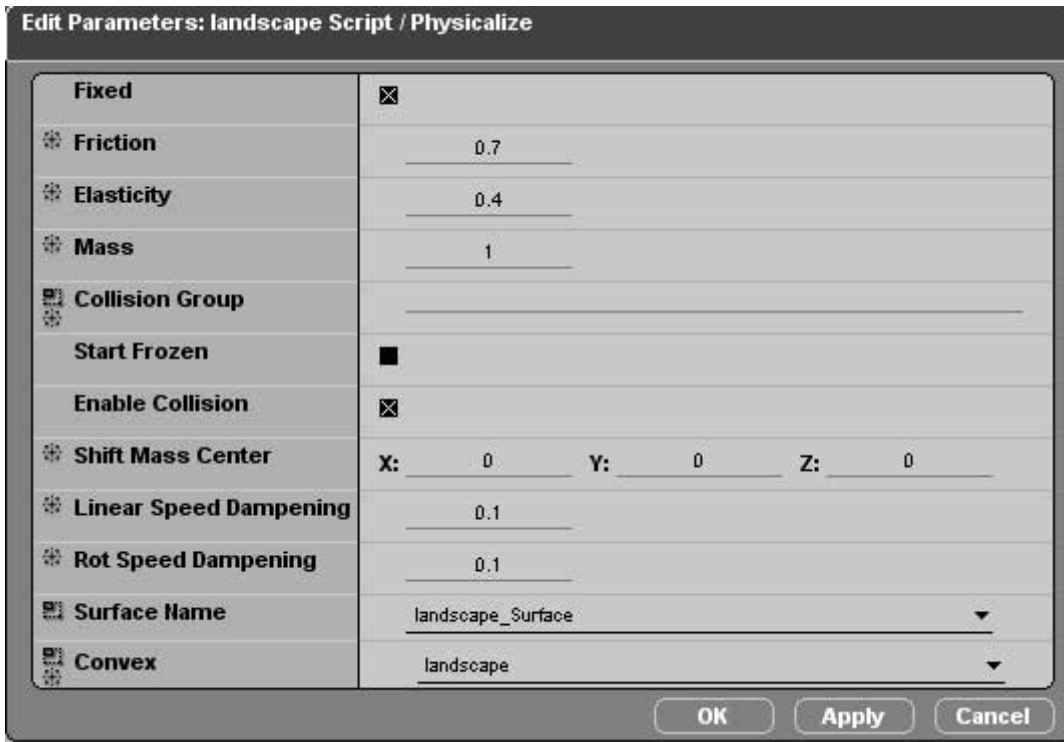
To begin experimenting with physics, all you need to do is add the Physics -> Creation -> Physicalize BB to the two main elements in the scene. In this case the floor and the box.

Be sure to set IC on anything you physicalize before playing the scene.

Gravity

One thing that happens automatically in with the Physics system is that you get Gravity.

If you play the scene at this point, you'll notice that everything that you physicalized falls out of the frame. This introduces us to the first major parameter in the Physics BB: "Fixed"



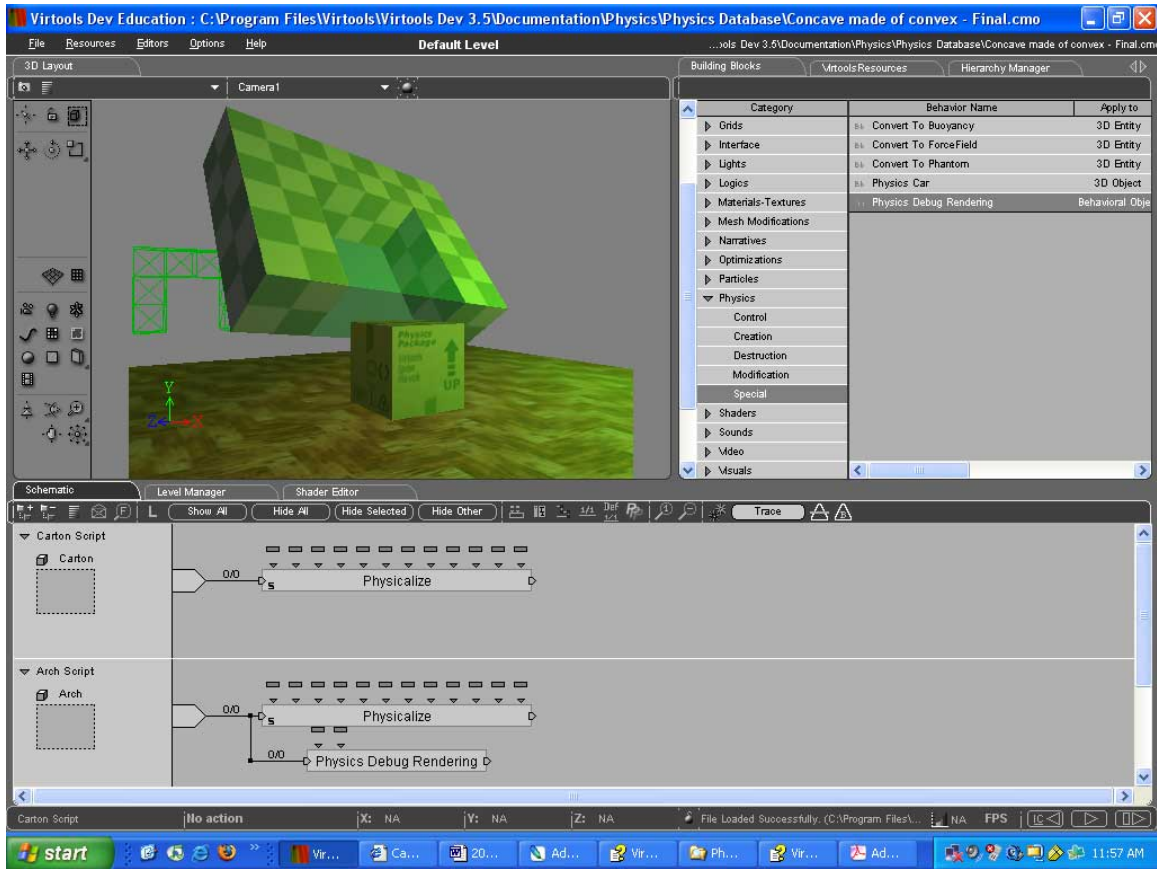
Check this box for the objects in the scene that will remain stationary such as the floor. Now you should see the box bounce on the floor. Feel free to experiment with the various physicalize parameters to see how they affect the behavior of the box on the floor.

An important thing to note right now is how Virtools actually handles collisions between objects. Luckily Virtools has a handy tutorial explaining this:

C:\Program Files\Virtools\Virtools Dev 3.5\Documentation\Physics\Physics User's Guide

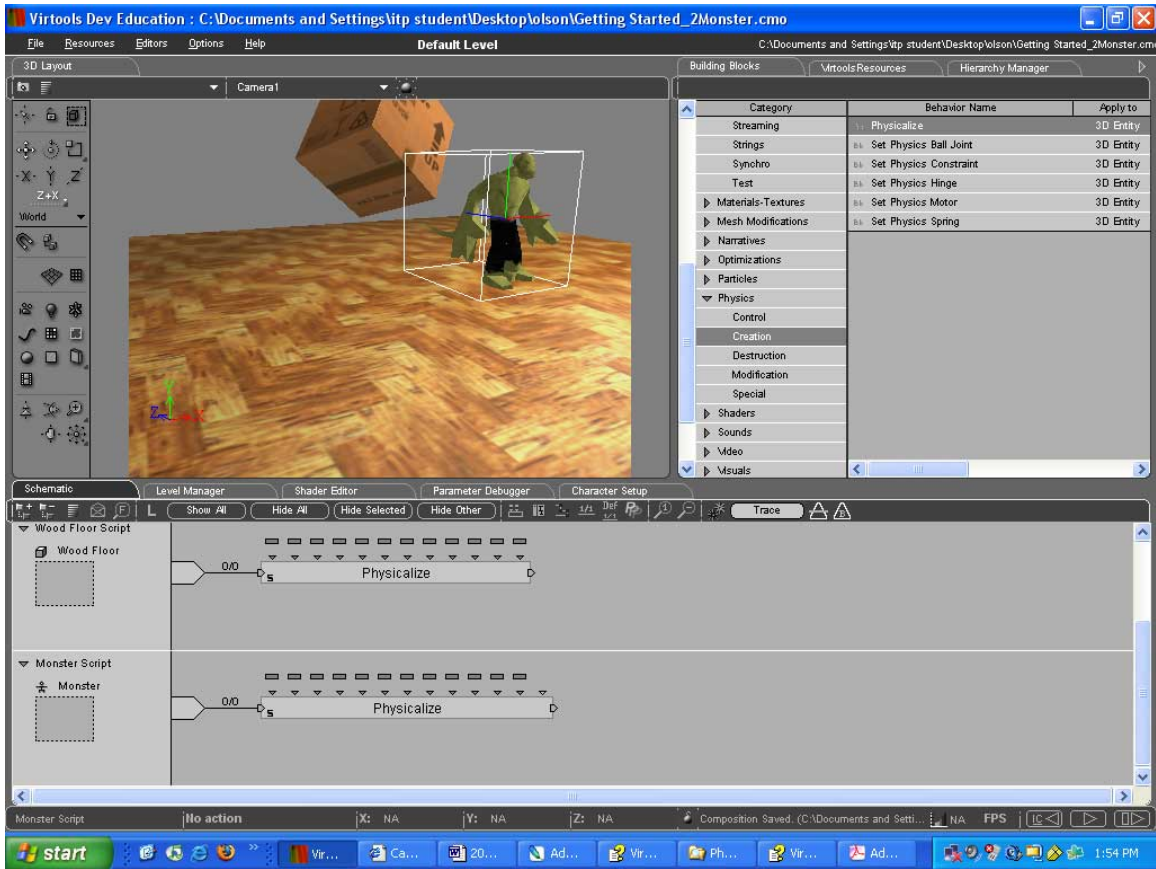
And related demo files. In this case, let's look at:

C:\Program Files\Virtools\Virtools Dev 3.5\Documentation\Physics\Physics Database\Concave made of convex.cmo.



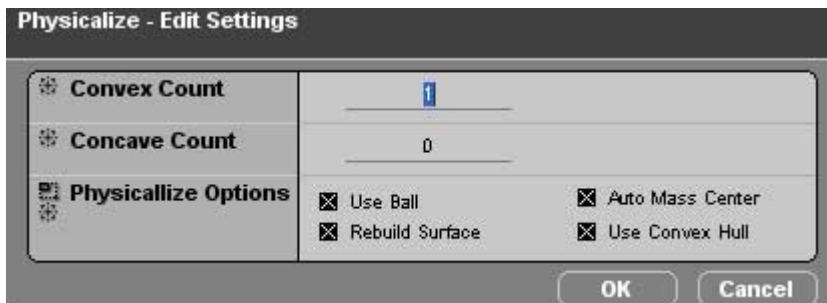
Above is illustrated with the use of the Physics Debug Rendering BB, why the arch seems to bounce off the crate when it seems that it should fall over it. Virtools is using by default a “convex hull” around the arch as an important way to optimize physics collision calculations. The actual shape of the arch is “concave” and this type of geometry is one significantly more complex on which to perform collision calculations than the convex type. The tutorial goes on to explain how you can use invisible meshes to compose concave shapes out of convex components. However, for the sake of simplicity, let’s not worry about that for now.

You may be wondering at this point, “Can I physicalize a character?” the answer is yes, but with a [large] number of provisos. Let’s delve in by dragging into our scene our familiar monster character. Position him over the floor, set IC, and physicalize it.



Play the composition, and see what happens.

Now it might occur to you, "If I've just set up a rocking physics simulation, why is my @\$! Troll still falling through the floor." The answer lies in the issue we explored above. Given it's complexity, the mesh for a character typically causes problems for Virtools' physics engine. To fix this you can either substitute a simpler convex mesh, or even simpler, use Virtool's handy "ball" option in the monster's physicalize settings.



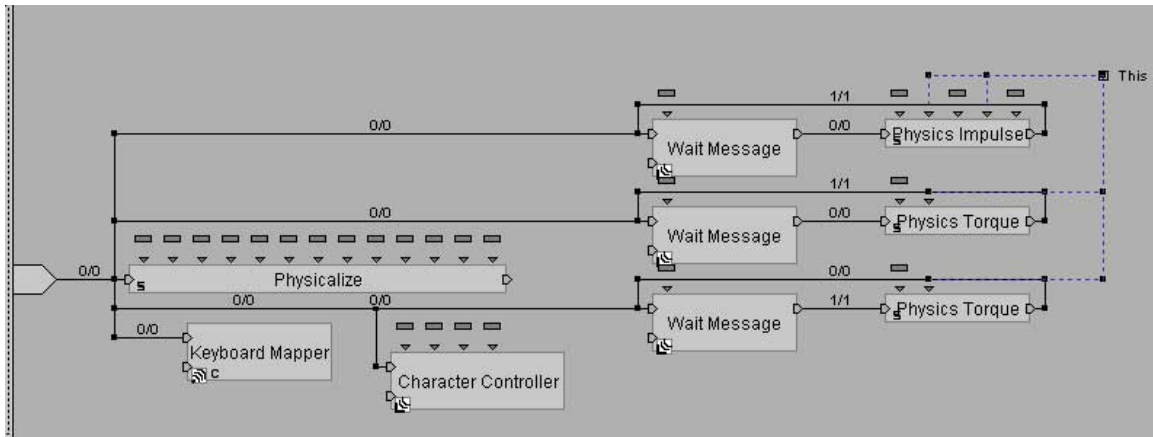
Edit the settings to look like those above. Now you should notice the monster no longer falls through the floor.

However, there still appears to be something wrong with his movement in that his character controller no longer seems to move him around. This is one of the more frustrating elements of the physics system. Once you physicalize something, many

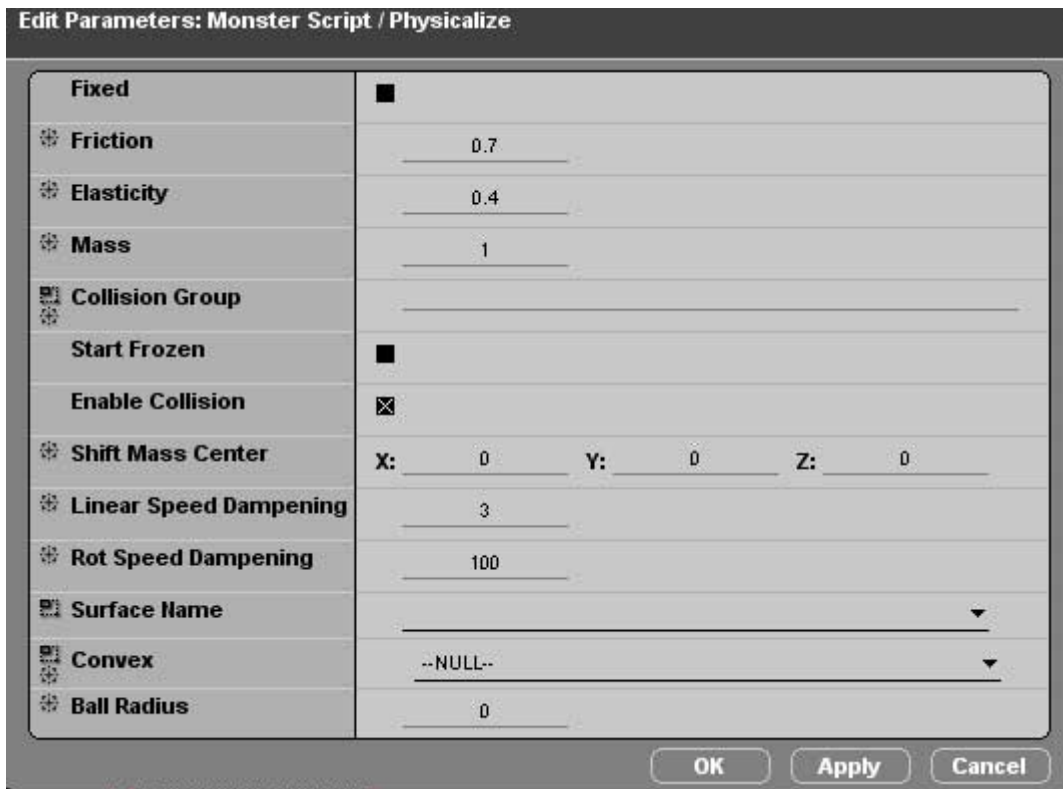
normal building blocks implementing basic 3D transformations WILL NO LONGER WORK. You must begin moving your elements around within the physics system. To do this, we use the magic of impulses.

Impulses

The way to move things around in the physicalized world is to apply physics forces to objects. In order to duplicate our old forward, rotate movement scheme, we need to replace the rotate / translates with physics torques and impulses respectively. So drag in 3 message waiters hooked to the joystick messages and connect them to the above physics BB's as follows:

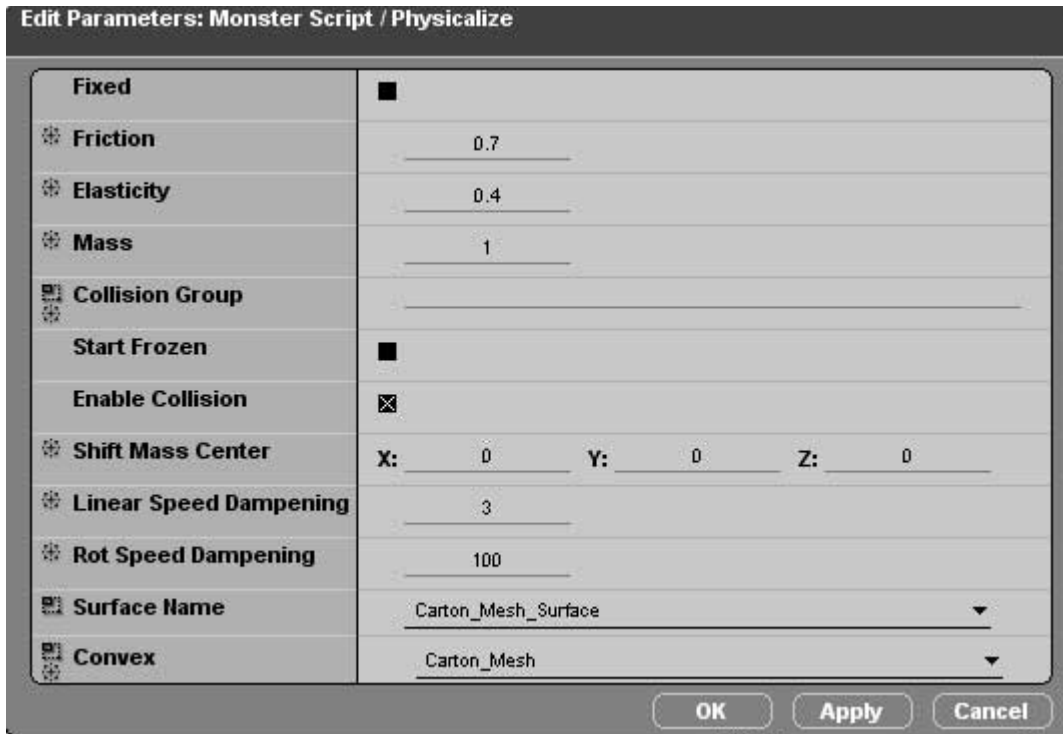


Now you should see the monster move around crazily as you press the various move keys. At this point you might want to edit the monster's physicalize parameters to look like this:



Once you get the various parameters tuned, you should be able to move your monster around the scene more or less normally. However, you'll see that he tends to lose his vertical orientation after a while. This is due to the fact that we're using the "ball" property which doesn't maintain a specific orientation with respect to the ground.

To fix this, we can disable the ball setting, and change the mesh we're using for the monster's collisions to one with a square shape, which will automatically tend to sit on the floor. Uncheck the ball setting checkbox, and reconfigure your monster's physicalize parameters like so:



Now you should see your monster maintain a more consistent attitude, and be able to push the crate around with impunity.

Hopefully, this simple little demo might give you some idea of the complexities involved in dealing with a physicalized character, especially on uneven terrain. You may find that you need to implement a complex system designed to "right" your character should she take a tumble, because the default motions may well not work depending on the situation.