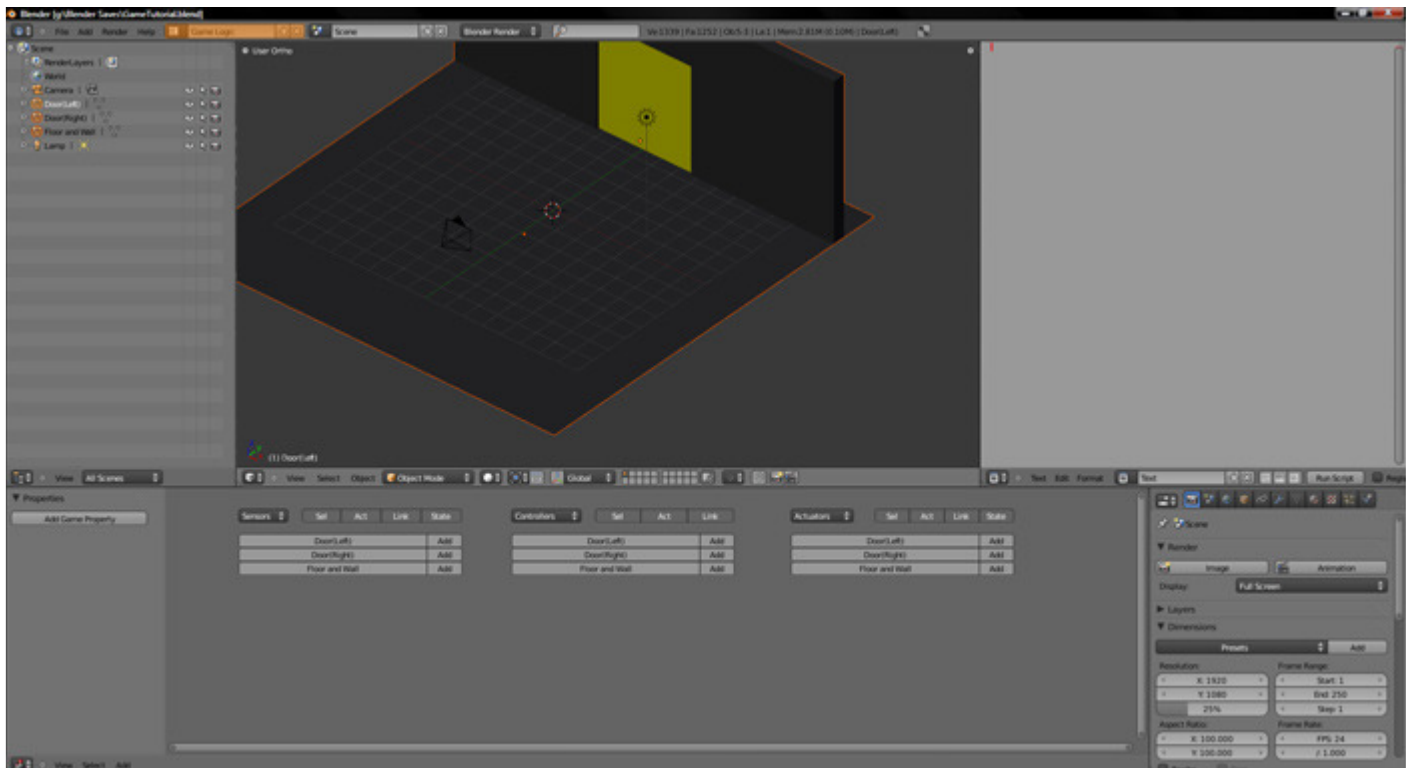


Game Engine - Motion

Dark Scarab Tutorials -- Blender 2.5

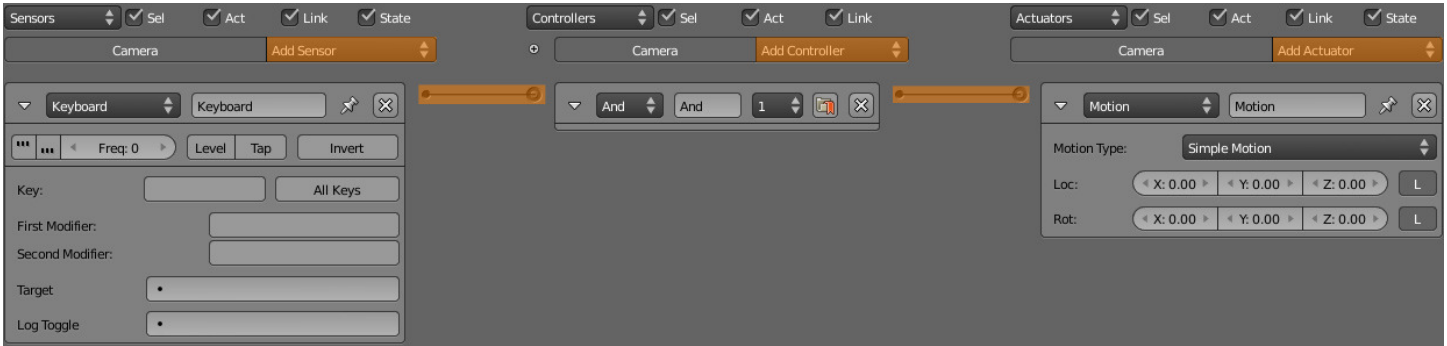
NOTE: In this tutorial, just like nearly all of my tutorials, I have provided what I call keystrokes lines. These are highlighted throughout the tutorial and are meant to allow you to see the actual keystrokes that I went through in order to get the results I get in the tutorial. More advanced users should be able to go through a tutorial without the keystrokes lines assuming I have explained myself sufficiently.

I have included a download for this tutorial, so if you are going to be following along as I do it then you will want to get that. If you are just trying to get the idea for movement in the Game Engine, then you'll probably be fine without it. In any case, I will continue with what is in that file. It should look something like what is shown in the image below. If it isn't already in the Game Logic layout, we need to do that now. You can find this in the dropdown list directly to the right of the help menu. Once you have done that, you should be able to find one of the windows looks like the one below.



We've currently got everything set up, so let's start getting to work. To simulate movement in the first person we need to move the camera as we push down some buttons on the keyboard. Select the camera and you should see its name in the bottom panel with three columns: Sensors, Controllers, and Actuators. For each column there is a pop out menu labeled *Add Sensor*, *Add Controller*, or *Add Actuator*. For the *Add Sensor* pop out, choose *Keyboard*. For the *Add Controller* pop out, choose *And*. For the *Add Actuator* pop out, chose *Motion*. After you have done that connect them together. If you don't understand, you can see it highlighted in the image below.

Select the camera, Add a *Keyboard Sensor*, Add an *And Controller*, and Add a *Motion Actuator*, Connect the dots



Before we get any further I want to make it clear what these columns represent. As an analogy, let's say we have motion detecting lights and the lights turn on when it sees movement. Sensors are something like the motion detector. When the motion sensor is triggered it will activate, which will cause the actuator to activate. The actuator is the actual action that results when a sensor is triggered. Continuing with the motion detector analogy, the actuator would be actual event of the light turning on. So, if we were to set something like this up in the game engine, we would have the motion detector as the sensor and the 'lights on' event as the actuator. For now, we will not worry about the controller.

Let's start by working with the Keyboard sensor we added earlier. All we need to do is choose a key we want to use as our forward button. To set the key, click on the field labeled Key and then press the button on the keyboard you are using to move forward. I used the up arrow.

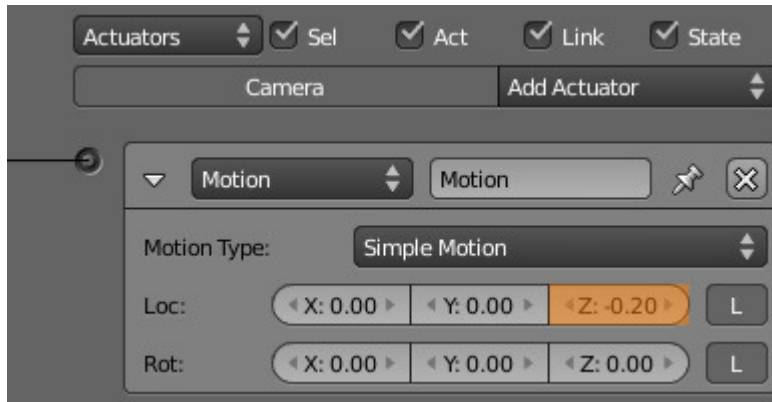
Select the camera, Click on the field labeled Key, Press your key for moving forward



Now we are going to move on to the actuators column. Since we want our camera to move when we press a key, we have chosen the *Motion* actuator. All motion is by default set to local, which means the axis directions are relative to the camera and the motion will not use the global axis directions. If you are not sure what the local direction is at some point, you can grab the camera like you are going to move it and then click on the X, Y, or Z button twice. The first time you hit it, it is in Global movement, the second is Local. If you do

that, you will find that forward motion for the camera is along the Z axis. So, back in the actuator settings, the third column for the Loc setting is for the Z axis. Change that to -0.20. You can figure out whether it should be positive or negative by actually moving the camera when you grab it and looking at the value shown in the menu bar of the 3D viewing window.

Change Z value for Loc to -0.20

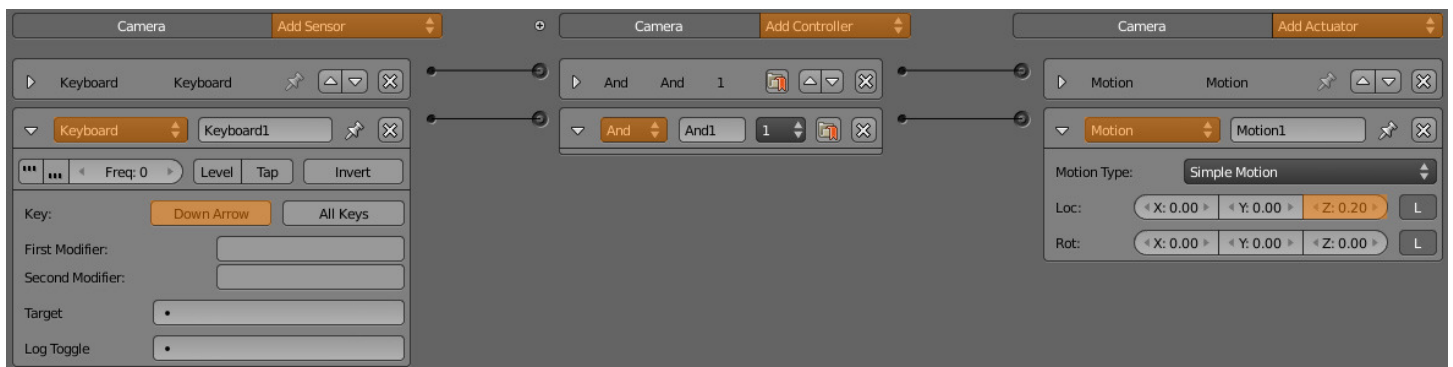


At this point we need to test it out in the Game Engine to make sure that our first logic bricks work correctly. First, go into camera view by pressing NumPad 0 and hover your mouse over the 3D view window. To start the game engine, press P. After it starts press the button you assigned as the forward button and you should see the camera start moving towards the wall. To get out of Game Mode, hit the escape key.

Camera View (0), P, Test forward key, Esc

Alright, we have gotten one movement done. Next, you will want to move backwards. For this I will keep this tutorial short: To do move backwards we do the exact same thing we just did to move forward. The only differences are the key you choose and the Z-axis motion value will become positive rather than negative. Make sure you use a different key to move backwards that you chose to move forwards.

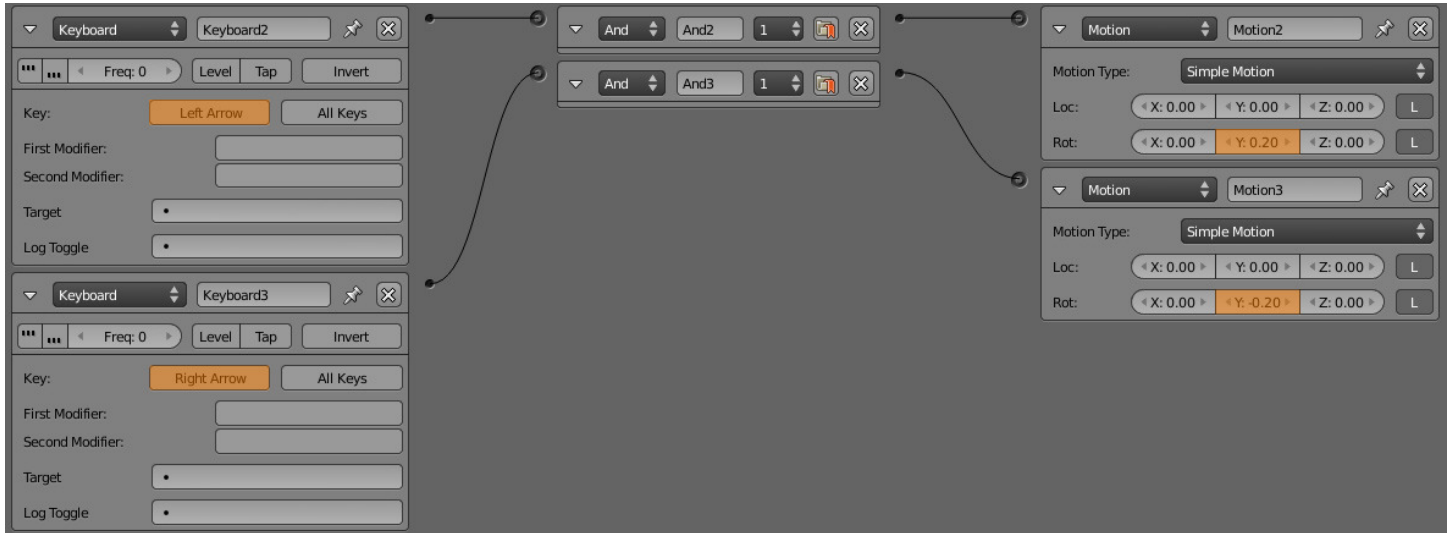
Add backwards movement



Ok, now we want to turn. Again, both left and right are almost exactly like forwards and backwards. The differences are that the key will be different, and instead of changing the *Loc* setting for the motion actuator, we are going to be using the *Rot* setting, which is right beneath *Loc*. Once again, these settings are for local motion. If you test this out like we did for moving backwards and forwards, you will see that we need to

rotate around the Y axis. So in the Y field for *Rot*, change 0.00 to 0.20. After you have done that, go through the same process for rotating to the right, only make sure that the rotation is negative. Below are my settings for turning left and right.

Add left and right sensor, controller, and actuator logic bricks



As soon as you have finished adding all of the logic bricks and changing the settings, you should be able to start the game engine (by pressing P) and begin moving around with the keys you chose to activate the motion. While all this should get you started, if you find the movement is too fast or too slow or if you want to change the keys, you can easily go back change the values we set for any of the settings to accommodate your preferences. Have fun!