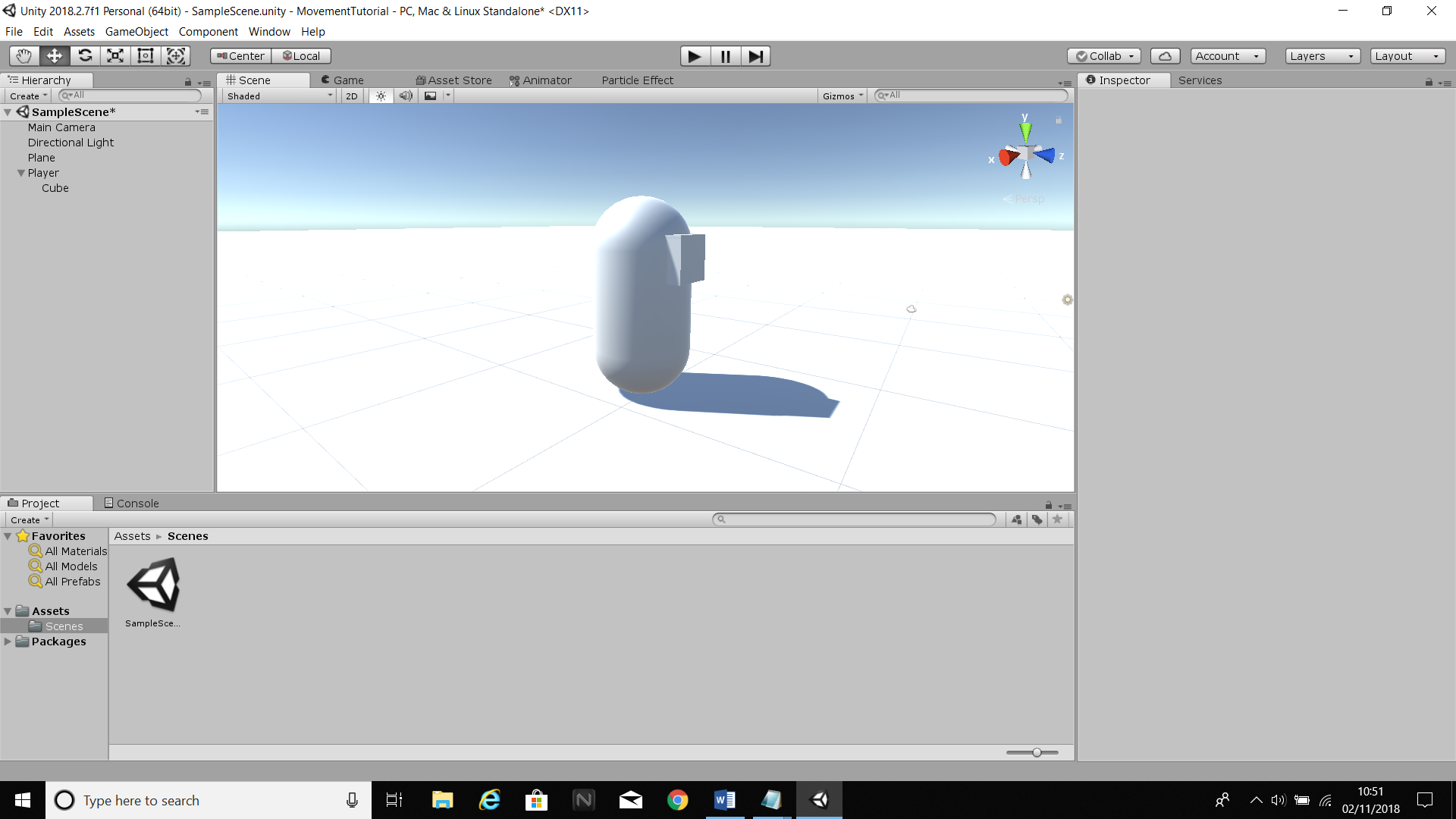
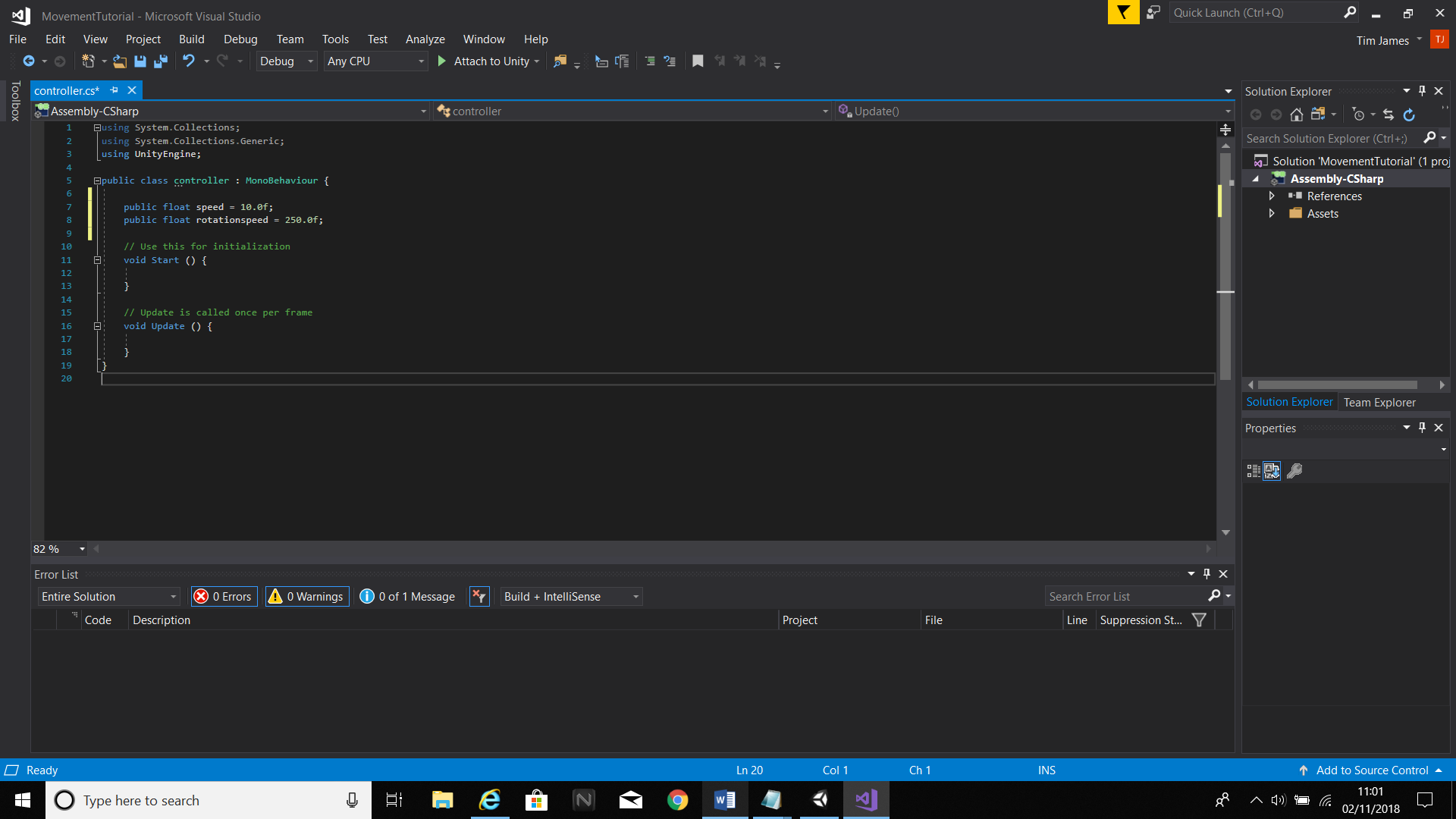
Programming Character Movement

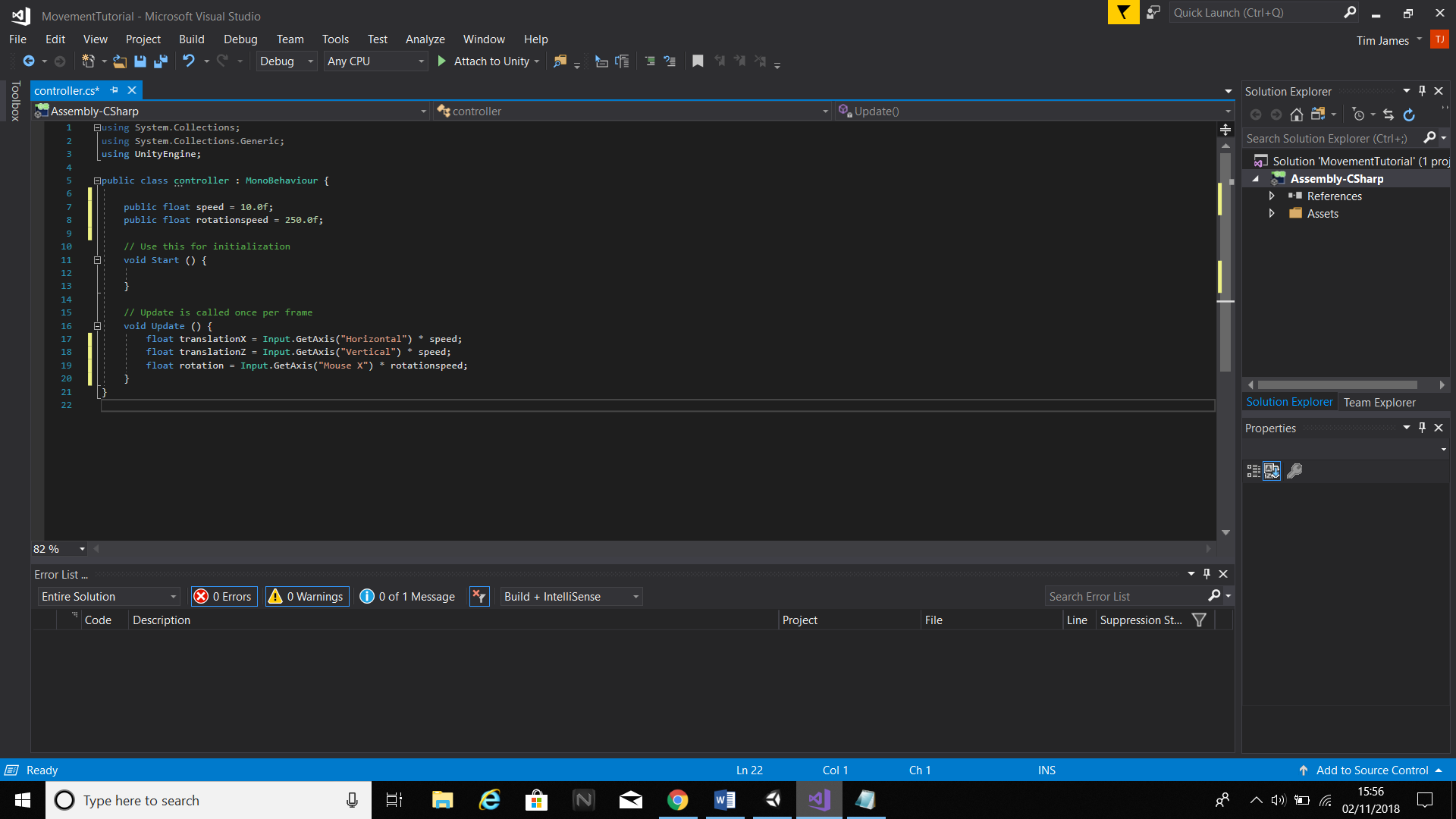
For the purpose of the tutorial, create a capsule in the scene which will serve as the player model. Parent a cube to the capsule so that it is easy to tell where the front of the player is.



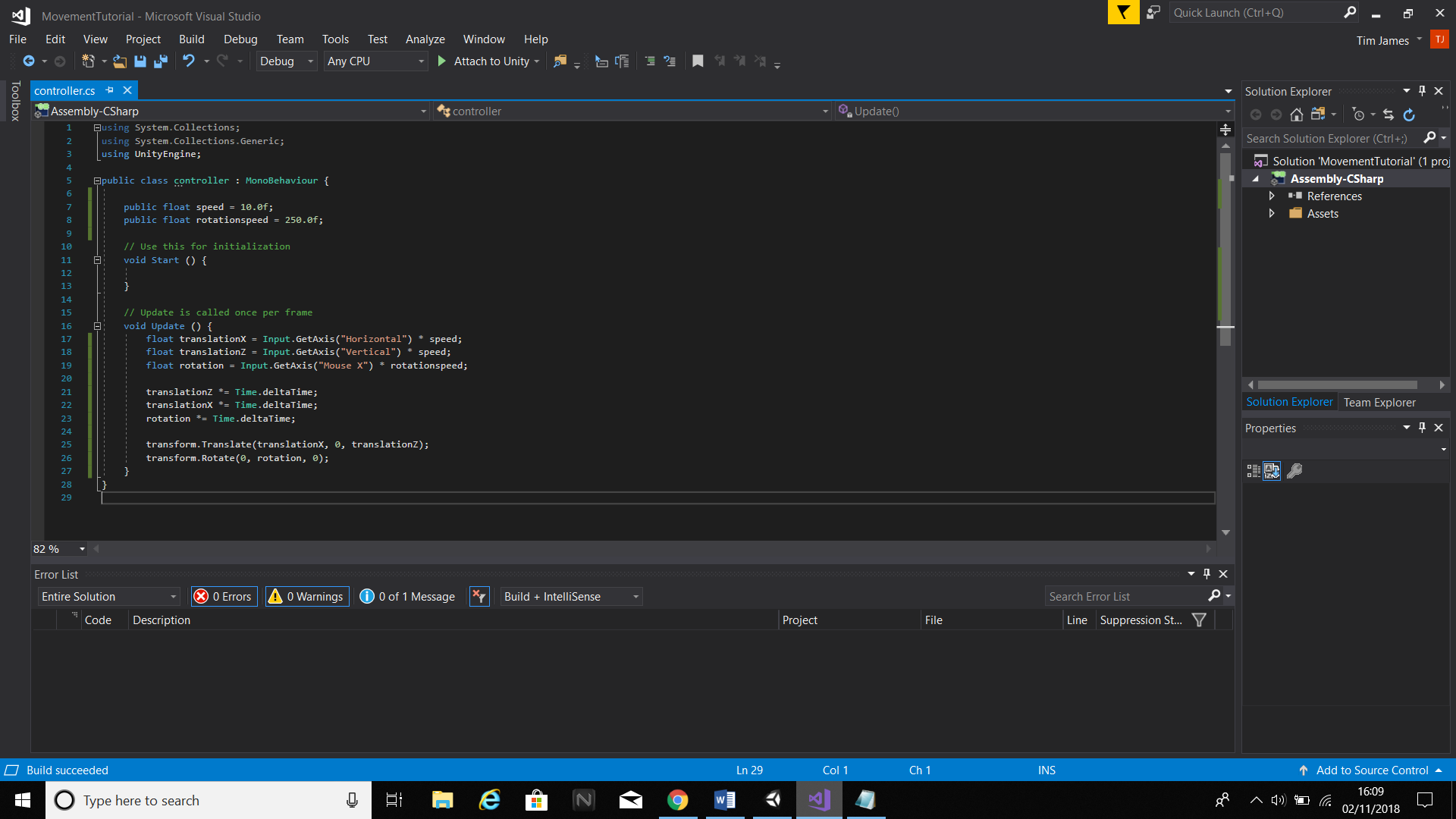
Create a script called “controller, and attach it to the player. In this script, create two public floats. Call the first one “speed” and set it to 10, and call the other “rotationspeed” and set it to 250.



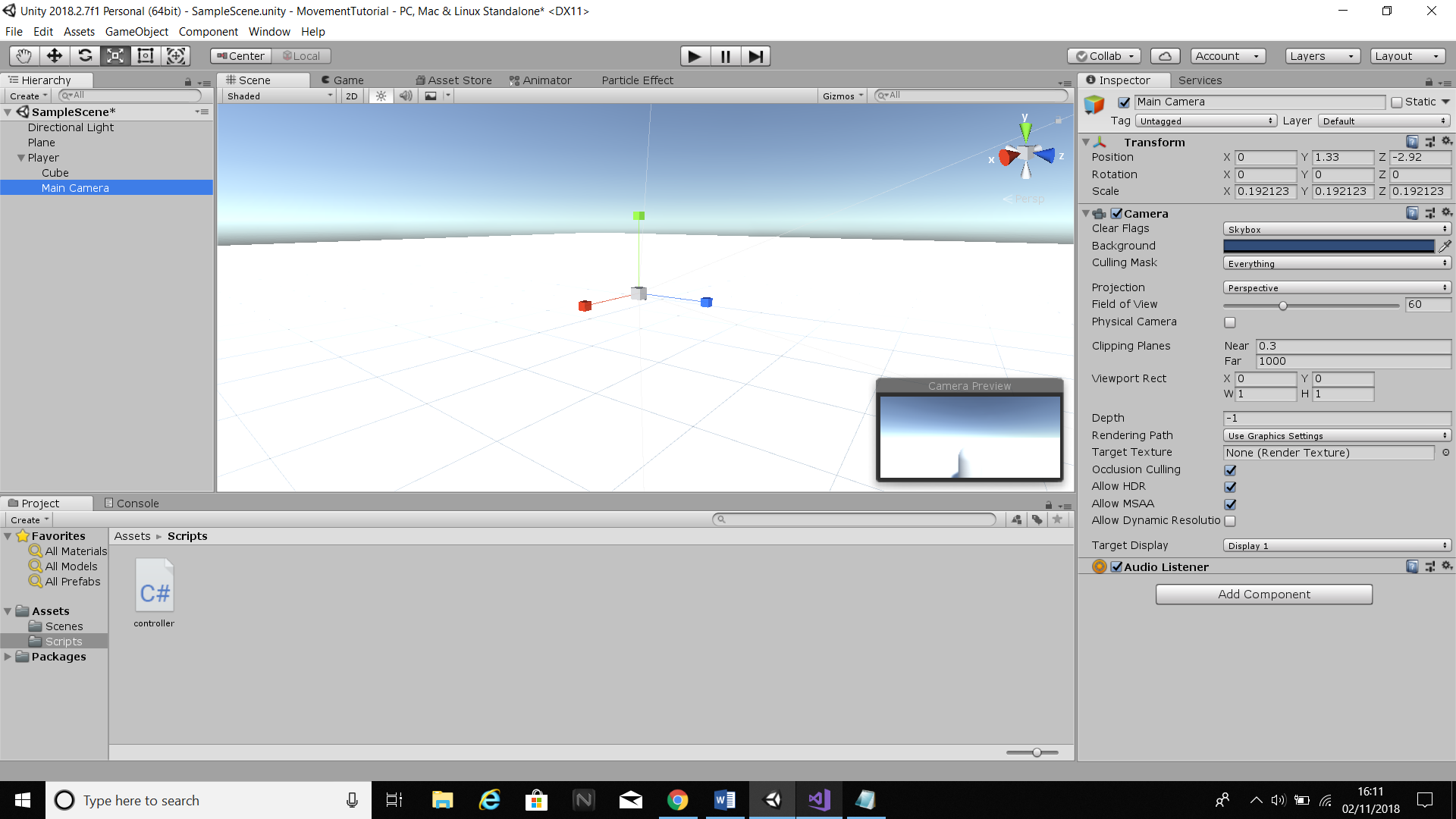
Create 3 floats in void Update, called translationX, translationZ and rotation. For each, use the Input.GetAxis command to call the keys used for movement. For translationX, set the axis to “Horizontal”. For translationZ, use “Vertical”; for rotation, “Mouse X”. Remember to use quotation marks or it will not work. Multiply translationX and translationZ by speed, and rotation by rotationspeed.



Now ensure that the transformations play in real time by multiplying all transformation floats by delta time. After this is done, all that’s left to do in terms of character movement is apply the transformations. Translate in the X and Z axes and rotate in the Y axis using the previously stated floats in Update. The finished code should look something like this.

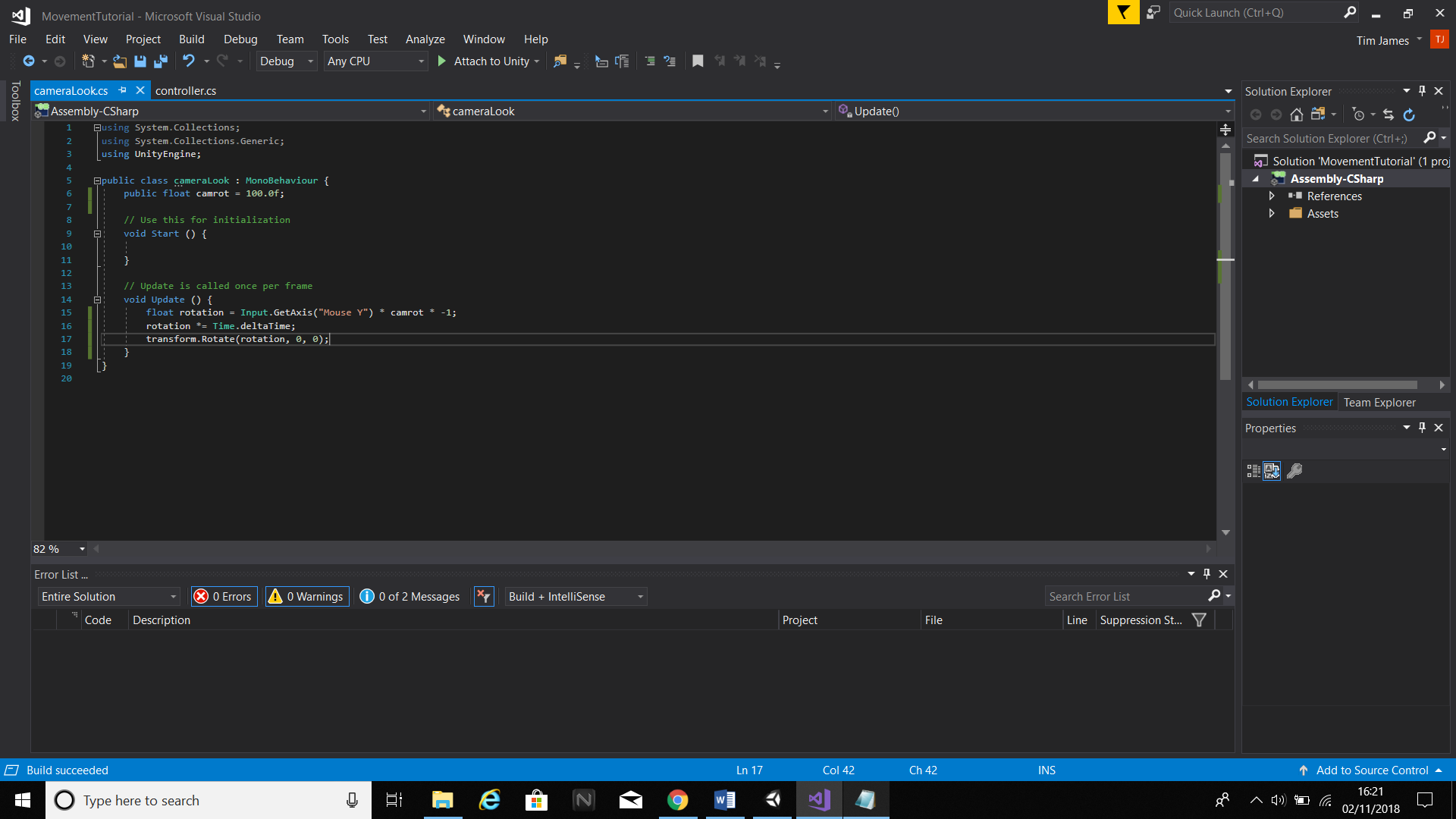


If we play the scene, the player will move but the camera will not follow. To fix this, parent the camera to the player then adjust it to the centre of the object (0 in the X axis).



The basics of character movement are now completed. However, the camera can only move sideways and not up or down. To make it do this, create a new script called “cameraLook” and attach it to the camera. The functions needed in this script are all very similar to those in the controller script.

Create a public float called “camrot” and set it to 100. Now, in Update, create a float called “rotation and use Input.GetAxis to attach the function to Mouse Y. Multiply by camrot and then optionally by -1 if you want the camera to move in accordance with the mouse, otherwise the look will be inverted. Multiply rotation by delta time and rotate along the X axis. The finished camera script should look like this:



Now if you press play you should notice the camera rotates along both axes, in addition to the player being able to move.