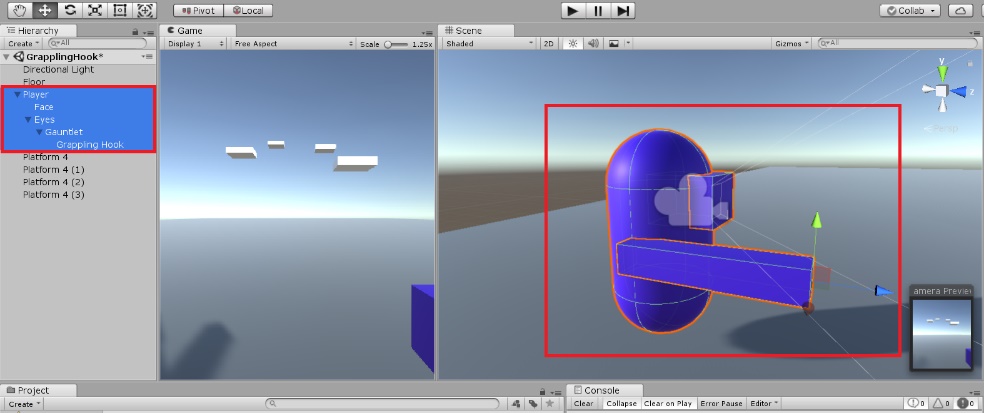
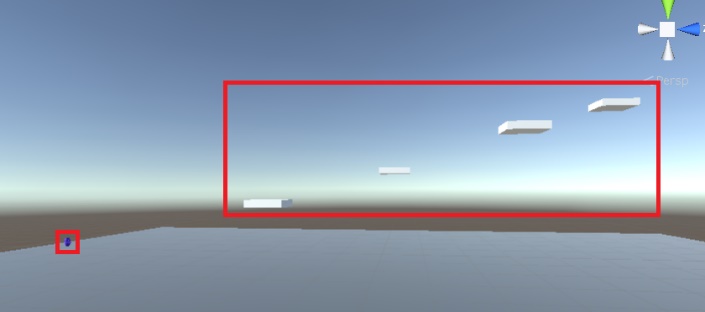
This script will demonstrate how to shoot out a grappling hook when pressing and holding right click

**Step 1 – Creating the scene**

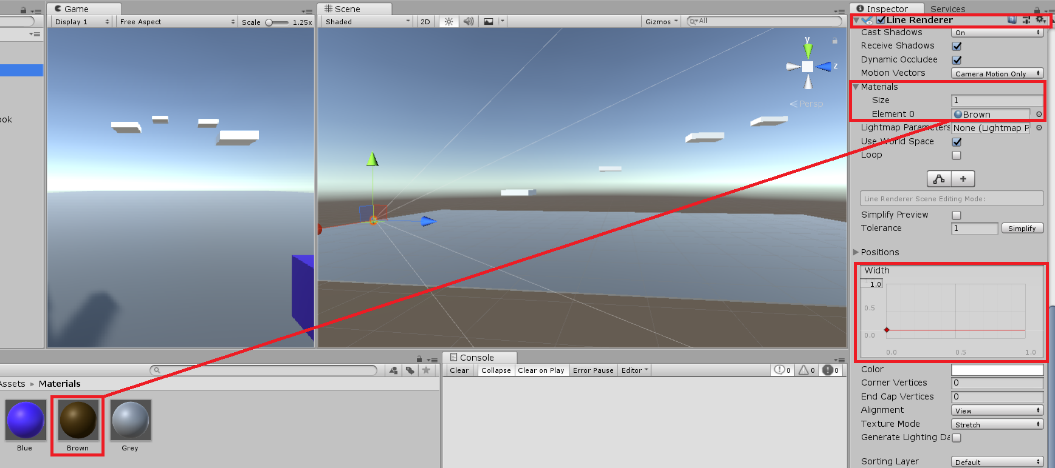
To begin, you must first start out by making everything you need for the scene. You will need a player (preferably a capsule) with a Rigidbody. From this, you will be attaching a smaller cube to the top of the capsule to act as its face. You will then be attaching the eyes (the camera) to the face cube, ensuring that the camera is aligned. After this you will be making a gauntlet which will simply be a cube or cylinder, something long that can act as an arm for the grappling hook to shoot from. This will also require an empty game object to be placed just in front of the makeshift arm/gauntlet, which we will name “grappling hook”. The current set up should look like the following:



You must ensure that the capsule collider has **all** forms of rotations constrained. This can be done under the “constraints” tab within the Rigidbody component on the player inspector. You will also need to create a few floating platforms to use your grappling hook on, feel free to construct the “level” however you wish.

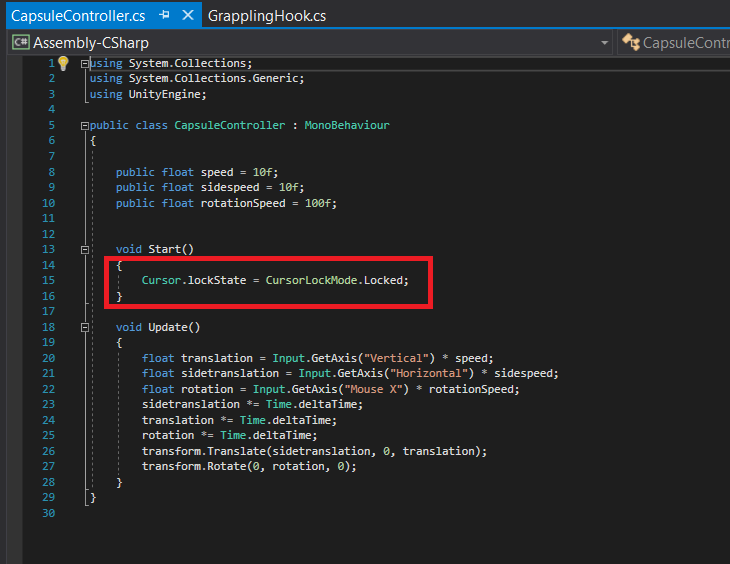


The final step for setting up our scene is adding a Line Render component to our player and making a material which can be placed onto this line render. A material can be made simply by right clicking on the unity editor tab below and selecting the “Material” option within the “Create” drop-down list. From here you can adjust the colour of the material to your liking. It is advised that you decrease the thickness of the line renderer attached to your player to prevent it from clipping through the gauntlet/arm when the grappling hook is shot.



**Step 2 – Making the Controller Movement**

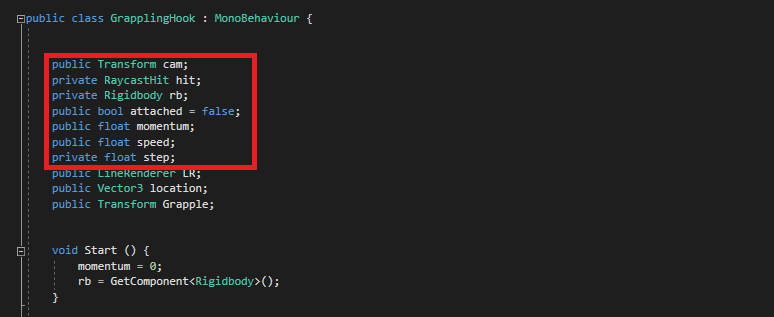
For the sake of convenience, we will be using the exact same controller created in the previous tutorial. However, one key difference will need to be made for the sake of the ease of use of the grappling hook. The following script **must** be added onto your transferred controller script:



This script locks the cursor to the centre point of the screen, it will also make your mouse cursor invisible **unless** you press the “esc” key. This is incredibly convenient as this will allow for easier use of your grappling hook.

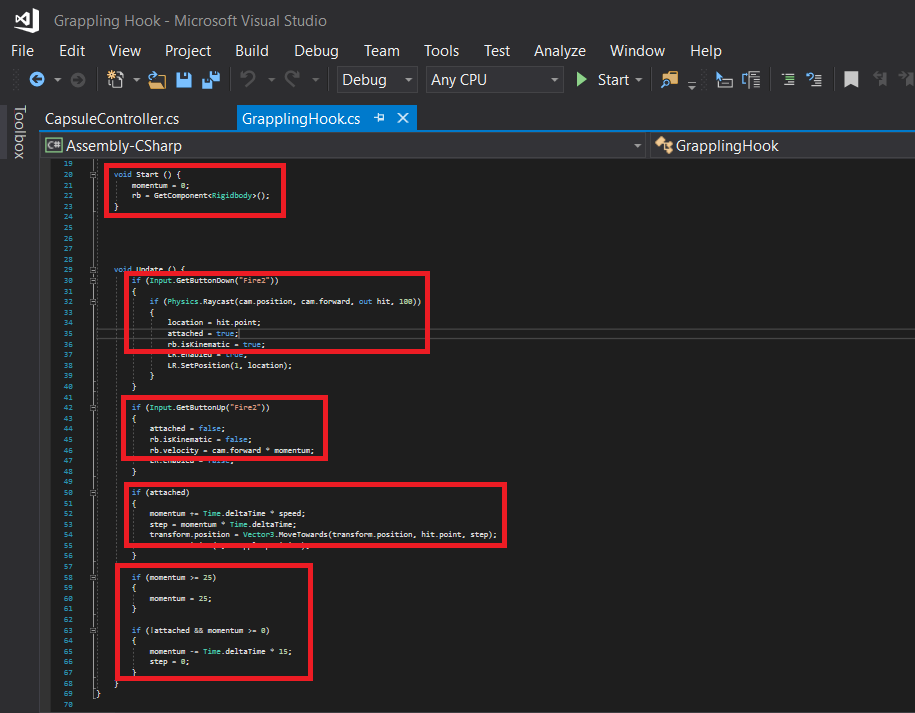
**Step 3 – Making the grappling hook**

The grappling hook we are making will be constructed using a Raycast to identify a point and transform the camera to move towards that point (which in turn will move the rest of the player towards the location identified by this raycast). To begin we will start out by setting the variables:



We will be calling for the Raycast and Rigidbody (named “hit” and “rb” respectively), along with transforming the camera, listed as “cam”. Along with this we will be setting a bool variable to determine whether or not the raycast has or has not hit an object, which will define our player being in a state of “attached” or “not attached” when using the grappling hook. The floats speed, momentum and step are simply used to provide force to the player once attached to an object.

The next part of the script will define momentum and “rb”, along with the construction of the Raycast itself:



The first highlighted code defines the value of momentum to start, which will be set to “0”. We will also define what “rb” is, which will call for the Rigidbody component attached to the object this script is applied to.

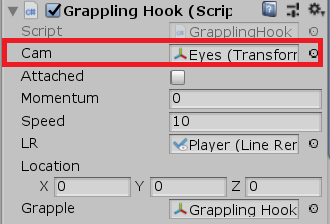
The second and third highlighted code define the conditions of what happens when the “Fire2” (right mouse button) is pressed down and released up. You will be making a raycast which, once in contact with an object, will set the attached bool to “true” and make the rigidbody kinematic. If the button is released, it will make the rigidbody no longer kinematic and the attached bool will be set to false. However, for the sake of fluency with movement the rigidbody will be set to continue moving after release with “cam.forward \* momentum;”.

The fourth and fifth highlighted code defines what happens if the attach bool is/is not set to “true”. If it is true, momentum will gradually increase (along with the ‘step’ value) and the object will transform its position towards the point of contact, referred to as “hit.point”.

The “if (momentum >= 25)” statement is simply to prevent the player from obtaining an infinite amount of momentum. In other words, momentum is capped at a limit of 25 to prevent the player from launching themselves upon releasing the right mouse button after 10+ seconds.

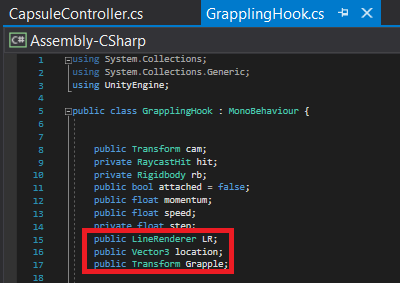
The final if statement defines what happens if the player is **not** attached to an object (and the momentum is greater than 0). In this case, the momentum will gradually decrease overtime, and the “step” will immediately be set to 0 to ensure the player does not have too much speed as to launch the player upon release of the grappling hook.

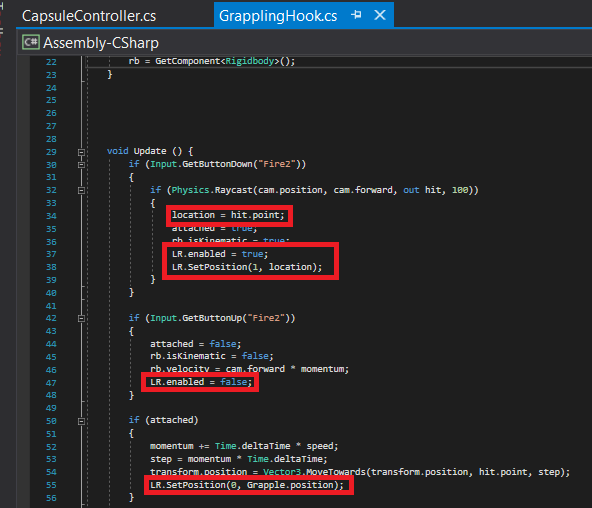
Once this script is applied to the player, you will need to place the objects into their corresponding parameters within the inspectors (such as the camera within the “cam” parameter):



**Step 4 – Giving the grappling hook a line render**

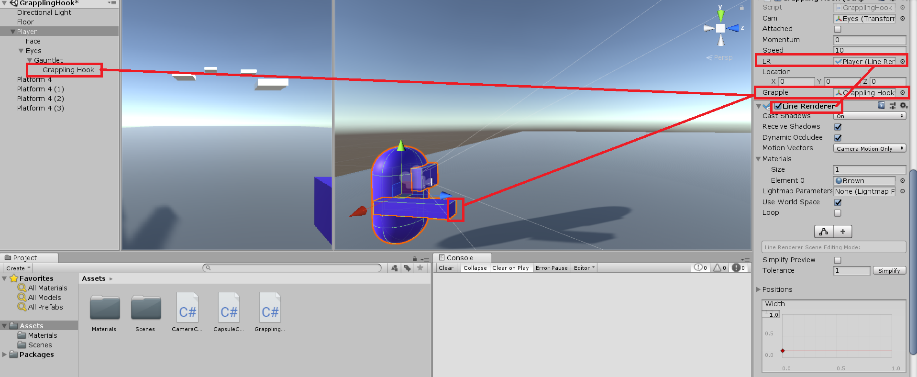
The final step will be incorporating the line render onto the character and the code required to ensure it works with the grappling hook. With our line renderer already being given a colour **and** being applied to our player (from the 1st step), all that is left is the coding, which will look like the following:





The first screenshot simply outlines the additional variables that are related to the Line Renderer. The Line Render (LR) is enabled and disabled accordingly to whether or not the grappling hook is/is not attached to a surface. Not only this, but these additions to the current script also transform the position/origin of the Line Render to the front of the gauntlet, the “Grapple”. The “grapple” object is referring to the Empty GameObject placed just in front of our gauntlet/arm, as it will look rather strange to have the line render originate from within the player.

Once these changes are saved, you must once again apply the components/objects into the correct parameters. For the sake of convenience, it **should** look like the following:



Upon pressing play, you will now notice that not only can you fly through the air by holding right click in the direction of a suspended surface, but you will also emit a long, rope-like line connecting the players’ gauntlet/arm to the contact point of the grappling hook.