Technical Report for Assignment 1

**Practical Assignment - Source Codes**

|  |  |
| --- | --- |
| Name | Low Tong Ee Thom |
| ID | 2202423F |
| Class | P03 |

# Q1: Camera Repositioning due to Object(s) Within Line of Sight

## Video of your solution

[](https://www.youtube.com/embed/7S9DxMf5VLs?feature=oembed)

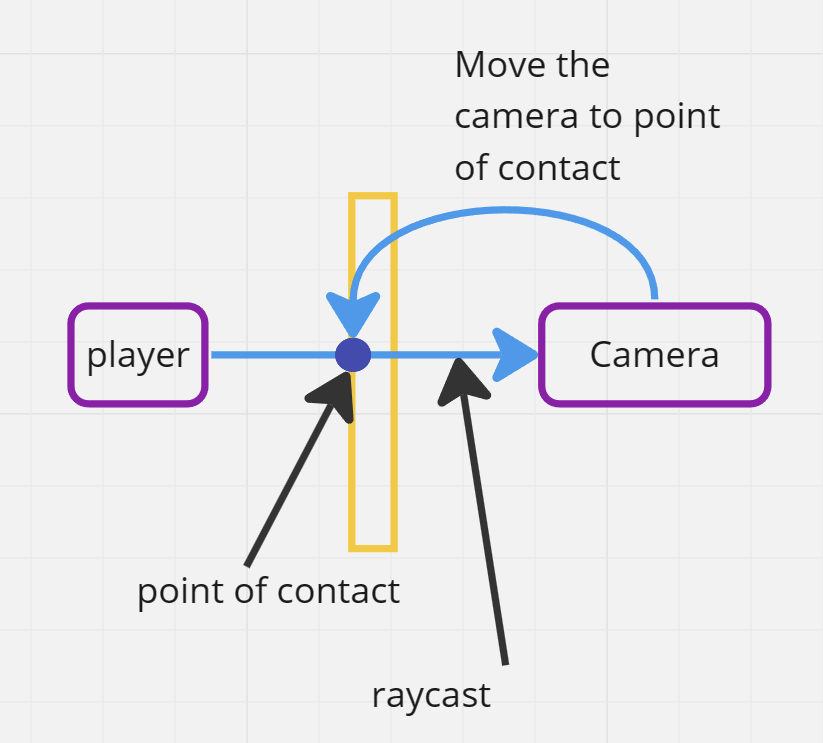
## Describe how you implemented your solution

**Solution:**

Before implementing the solution, the camera would always go through the wall. I realise that in order to make the camera not go through the wall is to **find** the distance between the wall and the camera and then **translate** the camera transform to **offset** that distance in order to show that the camera is infront of the wall.

Ray pointer = new Ray(CameraTransform.transform.position, direction);

So the solution is to send a raycast from the camera to the player. If the raycast hits the wall via collider, get the **closest** point that was hit and shift the camera to that position. That way, the camera would then be in front of the wall instead of behind the wall.



But there can be scenario where the camera has to look at the player through the mirror. So in order to do that, we can set mirror to have the different layermask compared to other material in the game.

Physics.Raycast(pointer , out RaycastHit hitobject , direction.magnitude - padding, LayerMask.GetMask( "Default" )

As a result, it the camera would ignore the collision and would not change position if the raycast hits a glass object.

## Reflect this learning experience

Though the problem was simple. I was quite stump for a while. At first, I though it requires the camera to have some sphere collider in order to prevent the camera from going through the wall. But after a bit of brain stroming (without looking at the hints at first), I though I would have to do some sort of raycasting instead of using colliders. From there, I manage to figure out the solution **without even looking at the hint**. I enjoyed how such a simple problem requires a bit of problem solving to figure out the solution. But I learn quite abit on why camera don’t just phase through the wall. I find it pretty useful to use this if I am trying to make a 3rd person shooter game.

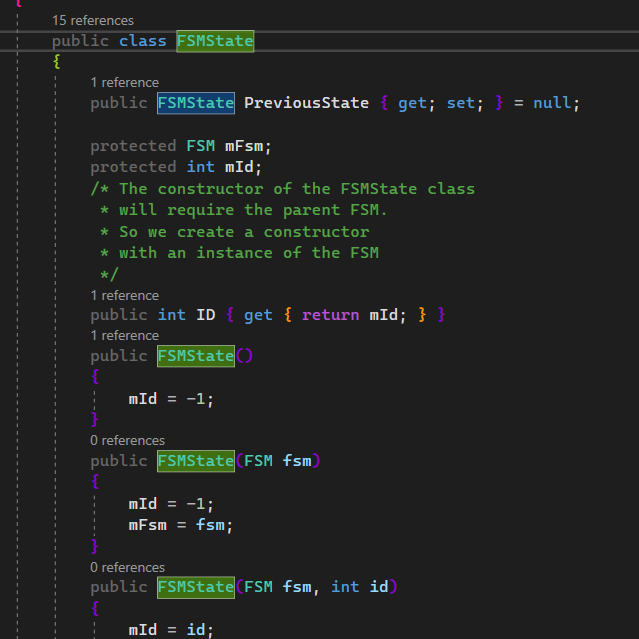
# Q2: Configure a new Character for the Player

*[](https://www.youtube.com/embed/QgxMj3uypFk?feature=oembed)*

## Describe how you implemented your solution

1. **The abstract idea**

Taking reference from the worksheet that utalise the Finite State Machine pattern (FSM), I decided to use FSM for configuring the player.



From the worksheet

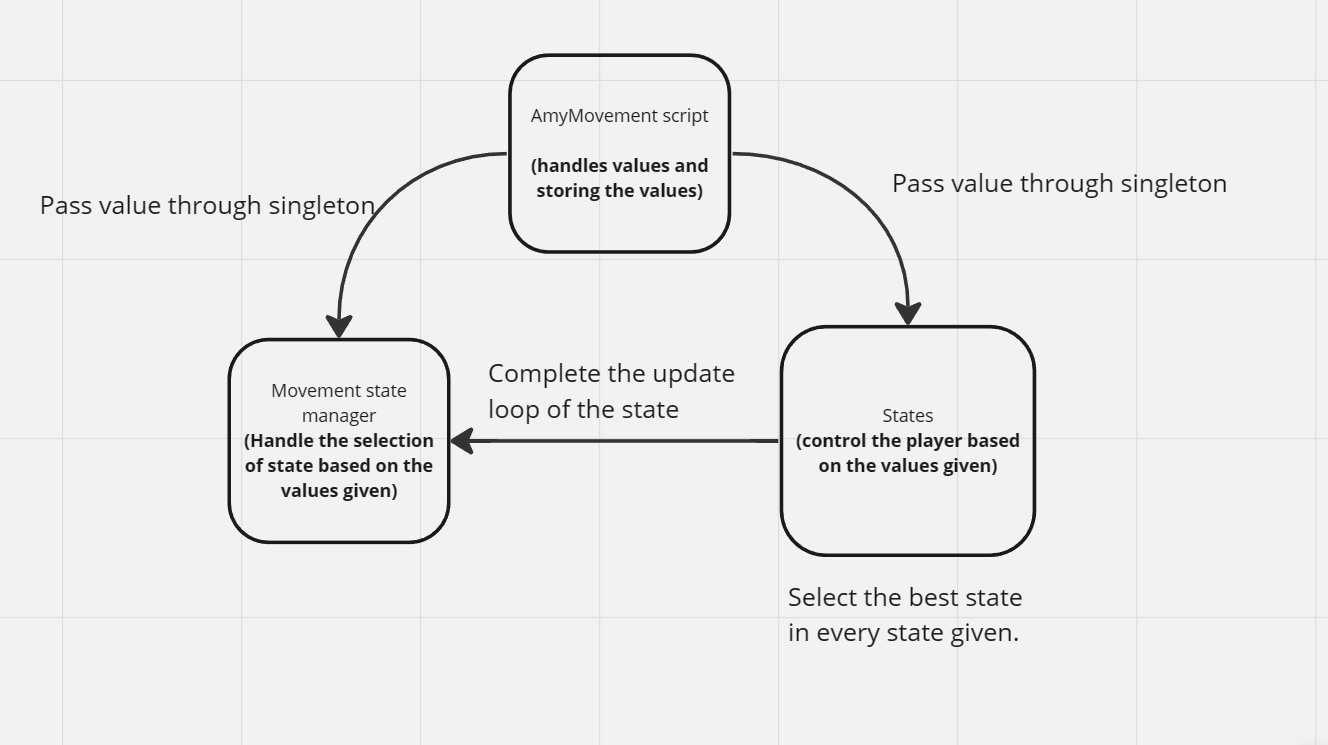
However, I did my own implementation of the FSM using this as reference as I thought I could make one better.

## 

Code snippets of my own FSM.

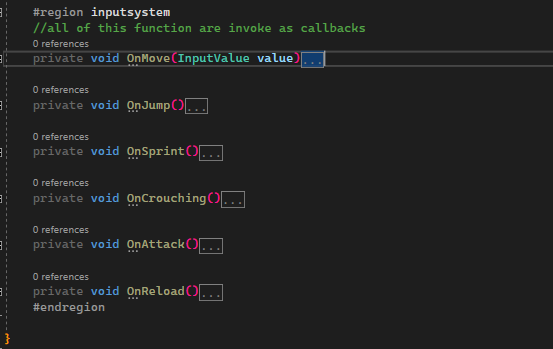
My own implementation of the FSM would separate out the selection of the states and creation of states into different scripts (there are a lot of problems I face which I will go through in the reflection). In each **state**, it would contain the behaviour required for the character to **perform the state**.

I would also use the singleton pattern for my controller so as to keep things organize ( separating the logic of the value retrieval , behaviour of the states into different scripts).



How interaction of the states will plan out

I would take things one step further by using the new input system for any input related system. This can help with controlling the character for different application (mobile or PC)

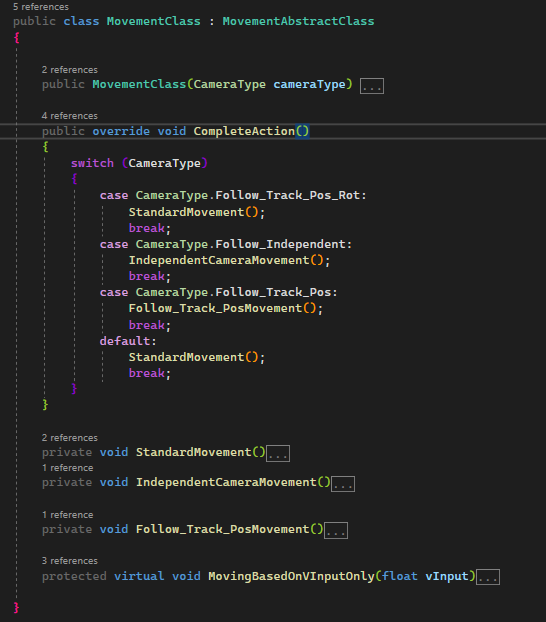


New input system

1. **States**

* *Basic movement:*

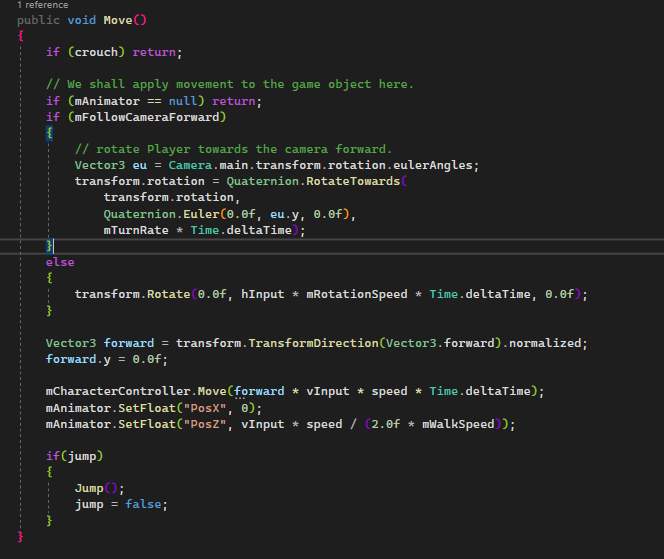
For moving the player around I have a **vector2** to contain the values passed down to the player (when player press the WASD keys or up, down, left right keys). From there, it will be transferred to the **movement state** to handle the movement.



Code snippet of the movement state.

In the state, there will have different implementation of movement depending on the camera (i.e, if the camera follow rotation and position of the player, it will make sure that the player can only move vertically but cant move horizontal. But camera follow position can make the player move vertically, horizontally and diagonally).

The implementation is mostly taken from the worksheet but some minor tweaking to suit the code.

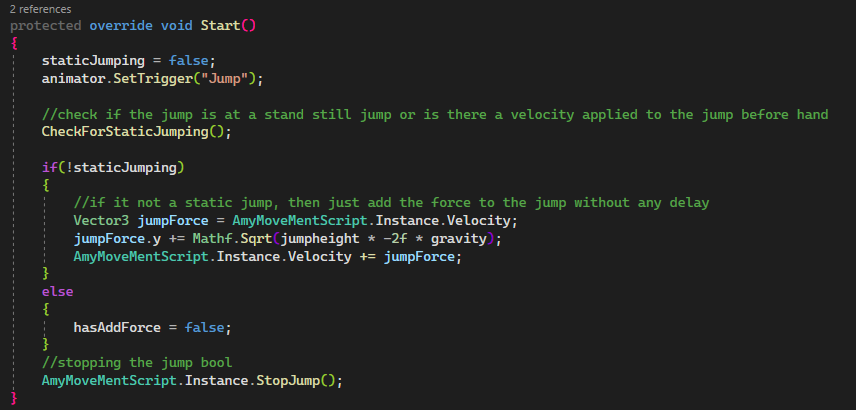


The code that I am referring from which I used to move the player

* *Jumping movement:*

Jumping is triggered when the players press the **space button**. It would then add

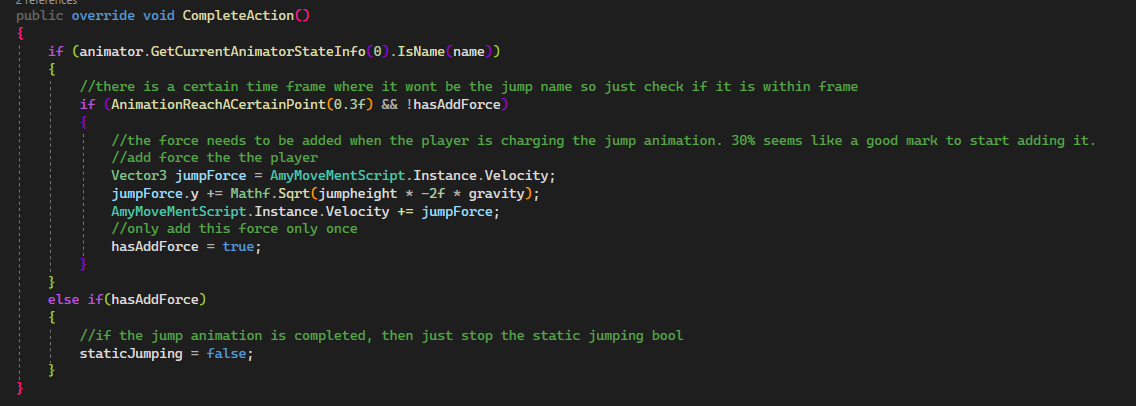
Velocity in the upward direction so it can be called in the **fixupdate call** where it show that the player would jump.



Adding upward movement to the player

The code would also call the **jump animation** to show the player is jumping up.

Another thing to add on is that in **real life people cant move mid air** while jumping at a stand still section (I call it a static jump). Therefore, I also make sure to check if the player has any initial velocity. If it does not, it will complete a static jump where the player cant do anything until the jump state is finish.



This chunk of code is meant to make sure that the player state remains at the jumping state until the character finish jumping.

* *Crouching Movement:*

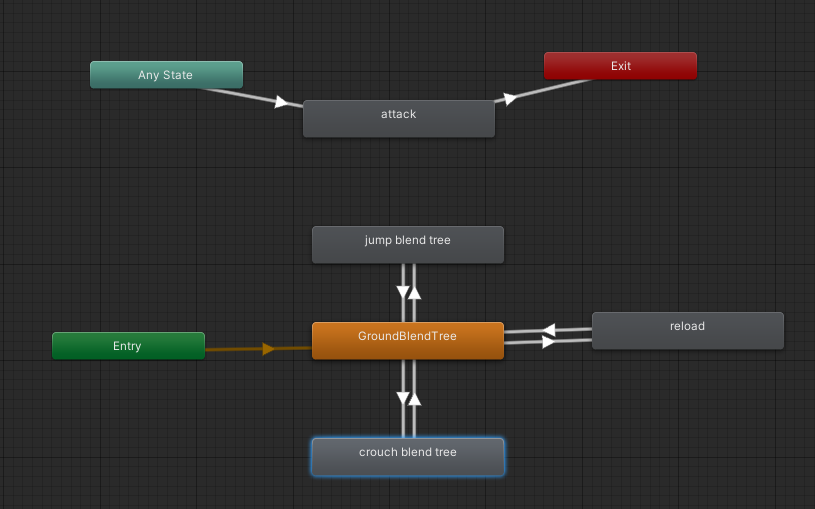
Though the worksheet does not have any crouching movement, it is essentially the same as the basic movement. So the crouching movement would inherit the movement script but with minor tweak to make sure the crouching animation plays.



When the player press the **Tab** button, it would trigger the **Crouch boolen** in the animator.



It will go to the Crouching Blend tree to show the animation when the Crouch boolen is set true.



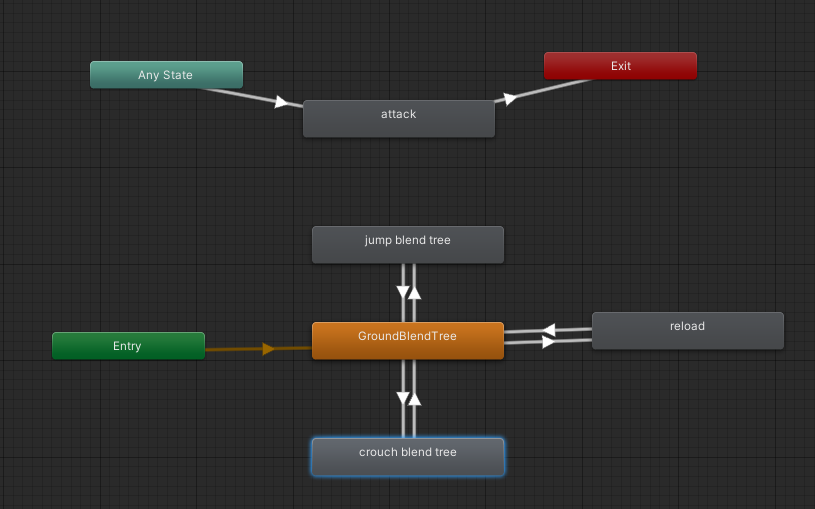
Afterwards, it would set the make sure that the **charactercontroller** and **camera offset y** values is adjusted for the camera.

- *Attack animation*

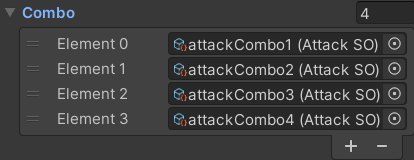
To make the attack animation I would add a trigger called **attack** to call the animation



Then from any state, it can go to the attack animation so that the attack animation can be showed on the player.



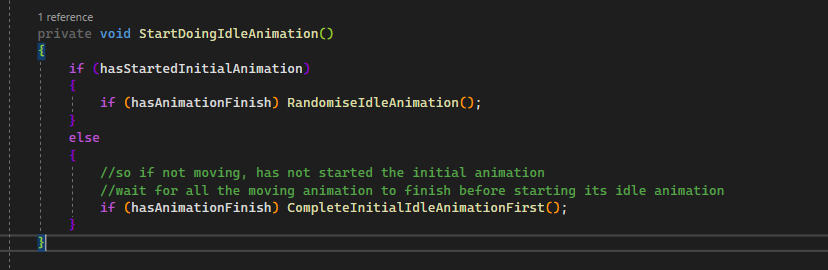
To make it more complex, I decided to use more attack animation. I use **animator override controller** to change the animation when the animation is playing.



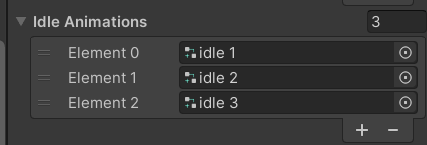
So when the player clicks the mouse button, it would change the attack animation of the player based on the array.

*- Idle animation:*

It is the same thing as the attack animation where I also use **Animator Overrider Controller** to change the idle animation. This time, it will check if the initial idle animation is finish before randomizing the next idle animation



Play a random animation after an idle animation is finish.



Different Idle animation to play in the inspector

## Reflect this learning experience

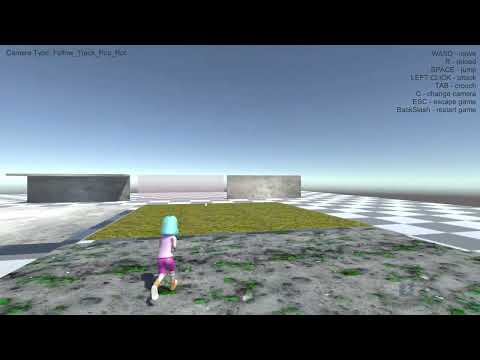
After working on this mechanic.I learn a lot on how to better use **FSM** and know how to use **blend tree** to make the animation. While working on this project I had to do a lot of research on how to do certain task like knowing how to punch while walking or having multiple idle animation. I would learn things like **Animator Layers** and **Animator Override Controllers** which I manage to apply.

I also made a lot of mistakes which I learn while working on this project. For instance, my inital FSM implementation becomes more and more confusing as I continue to implement new feature. There were many mistakes that my FSM could not completely cover and I had to hard code certain values because of that. For instance, my states could not figure out **if it is running continuously, starting or ending**. This cause the logic for certain states like the Idle or fighting animation to be a bit junky and unpredictable. Looking back to the worksheet implementation, I finally understood why it is implemented that way instead of my way of implementation. After making a lot of changes to my FSM, I manage to the code much more cleaner and easier to understand.

This part is one of the harder part of the assignment that I had to complete because there was a lot of features that I really wanted to include in like different crouch animation or idle animation. So figuring out how to do that can be a bit time consuming. But overall, it was a great learning experience for me and I am excited to learn more about **Unity’s animator**.

# Q3: Implement Step Sounds

## Video of your solution

*[](https://www.youtube.com/embed/jG5MvUcIneQ?feature=oembed)*

## Describe how you implemented your solution

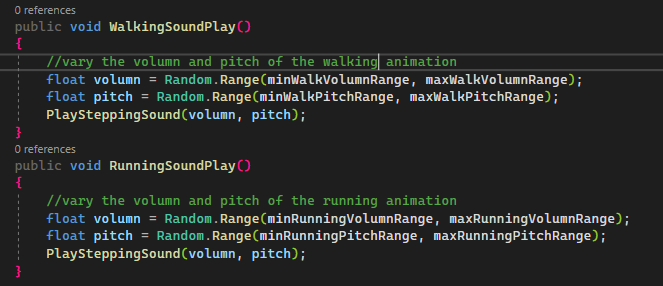
1.**Emulating the stepping sound**

To make each step produce a sound, I had to use the unity animation event.

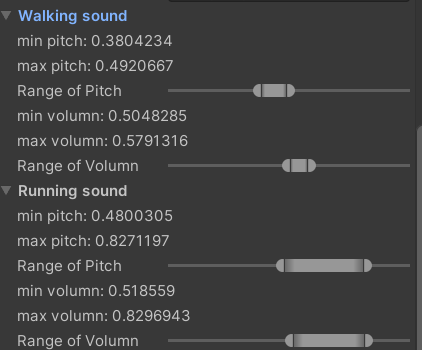
## 

## So in the frame that the character is stepping on the ground, the event will make call out a

function that will produce the the walking / running sound.

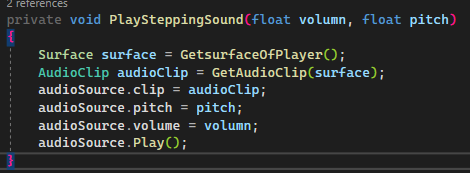


The volume and the pitch will change depending on the min and max value set in the inspector. This value is randomize so that the sound produce by the steps are different every single time.



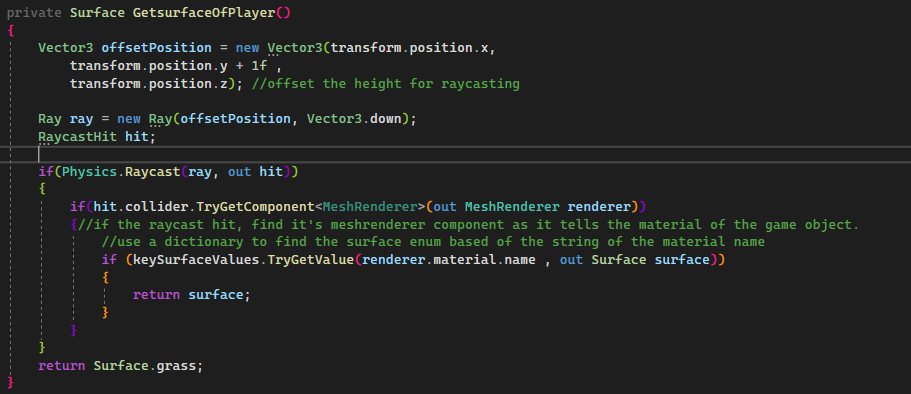
Values are set in the inspector for both Min and max volume and pitch of the walking and running sound.

After the volume and pitch are determined, it determine which surface the player is on before playing the sound

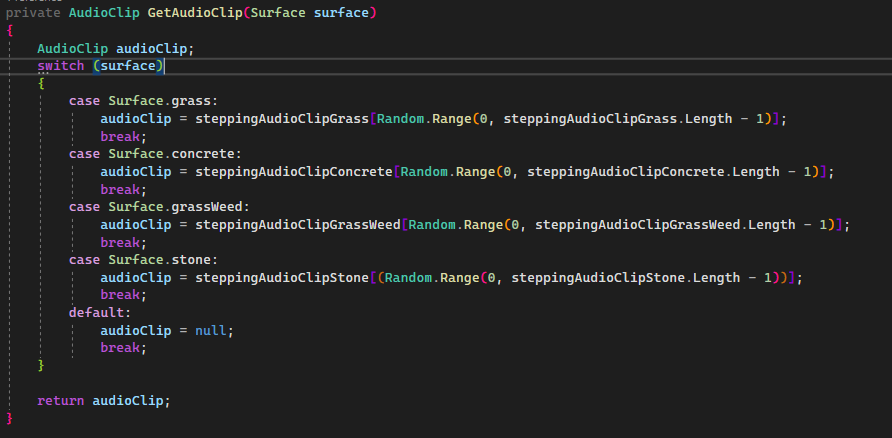


Getting the surface to choose the correct audioclip.

When getting the type of surface the player is on, it would use a ray cast to raycast the material of the gameobject the player is standing on.



There is a **dictionary** that contain the reference of the **material name** and the string associated of it. This makes the look up of the material fast so that it can play the step sound faster.



Lastly, it finds the audioclip associated with the surface and get a random audio clip from the array of audioclips.

## Reflect this learning experience

Though the problem was simple, I did have some difficulty figuring out the implementing this feature.

One of those difficulties was figuring out how to play an **audio clip whenever the player step**. I initially thought I would have to use **colliders in order to trigger the audio clip**. After countless of futile attempts, I decided to research other ways to play the audio clip. That is when I learnt about Unity animation event. After playing around, I figure out how to play the audio clips with ease.

Though I did not have have a lot of difficulties solving this implementation, I do find it a good learning experience because I learn how to use **unity animation event and custom editor**.