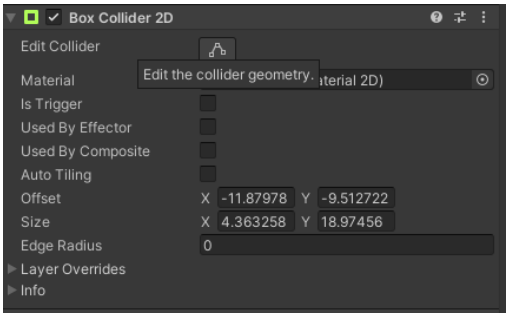
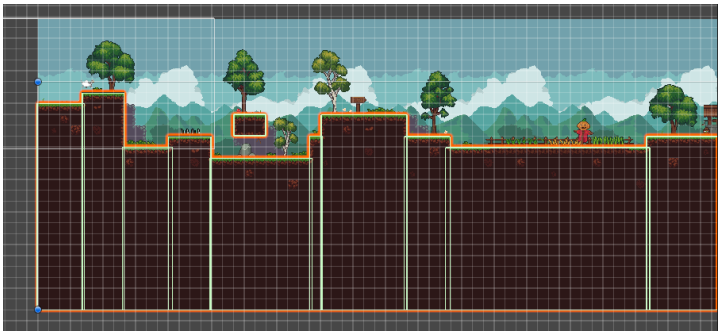
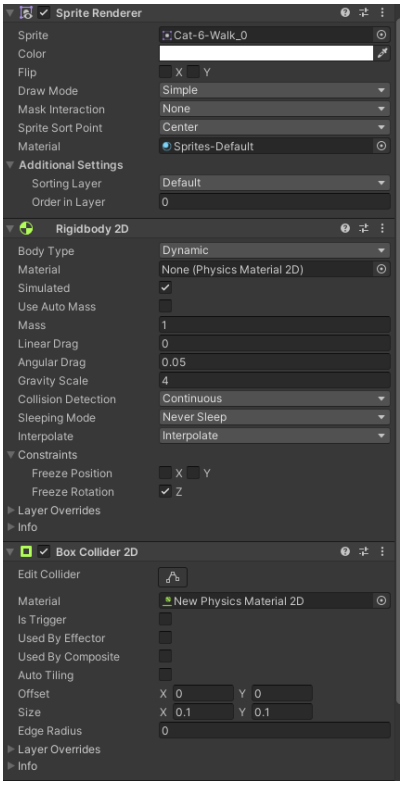
Player Movement Tutorial - Jagoda Kopec

I wasn't feeling too comfortable with starting on my own, so I went to youtube and found a video titled "How to make a 2D Game in Unity" (https://www.youtube.com/watch?v=on9nwbZngyw)
It showed me how to open a new 2D project, which I already knew how to do and then the next step was to quickly grab some 2D Free Assets from the Unity Store. I quickly grabbed some that I liked the look of and continued.
This is not needed, however, I just thought that it might make my experience feel a bit more fun and friendly.
After I found some assets, I imported them in and started to experiment with them. First thing I wanted to figure out how to do is to make the ground have collision, so as to make my future character be able to walk on it without falling through. I started doing that by selecting the ground in my demo and clicking on "Add Component" and then "2D physics" to "Box collider 2D".
Continuing that, I'm just now clicking "Edit Collider" and roughly matching it to the ground tilesets I have. I'm having to add multiple of those as I don't think you can add more points to edit the collider as I have hoped, unless I just haven't figured it out yet, but I did try to google it.



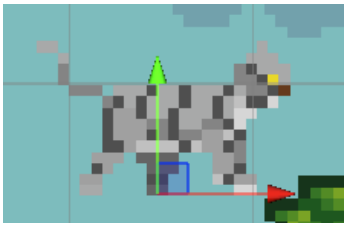


Next was to figure out how to make a player character and make it move and jump. First I want to figure out just movement, so walking forward and backwards. I also found a youtube video for this called "2D Player Movement in Unity" (https://www.youtube.com/watch?app=desktop&v=K1xZ-rycYY8&t=0).
It told me to right click in the Hierarchy and "create empty", then I have named it "player" as that's what the new object will be. I have also added a sprite to this object by clicking on the "Add Component" button on the right hand side (where the transform window is) and selecting "Sprite Renderer", then I just simply pick a sprite so that my character is more than just an invisible object. Then I was advised to also add two more components called "Rigid Body 2D" and "Box Collider", this is so our player can be affected by Unity's build and physics engine, which in my head means that it lets our character interact with other assets and "rules" in unity.

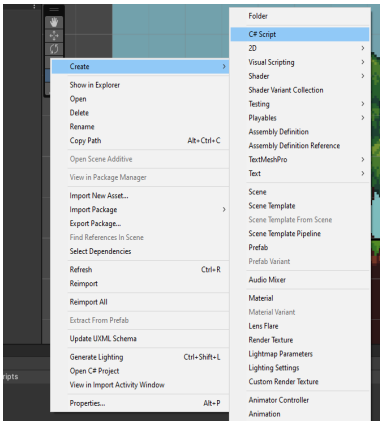


As I was watching the video, I was told to change the “Collision Detection” to Continuous, “Sleeping Mode” to Never sleep and “Interpolate” to Interpolate, and lastly to Check the “Freeze Rotation” box in the “Rigid Body 2D” Component. Sadly, it wasn’t well explained as to why do these things, but I could understand “Collision Detection” being put to Continuous as I think that means that this object will constantly be looking for things that want to collide with it, meaning that for example if I walk on the ground I won’t randomly fall through.

The next step is to create a “Child” for my “player” object. This basically just means that the object I create is attached to the player. The child object is going to be called “groundCheck” and it has to be positioned right at the player’s feet which I can do by selecting the child object and then pressing the button **W** as a shortcut to select a tool that will let me move around the object I have just created.



Then, I went to the “scripts” folder, created a new C# script by right clicking anywhere on the scripts folder and then hovering over “Create” and then “C# Script”. After that I named it “PlayerMovement” and clicked and dragged it onto the player in the hierarchy and then it was time for me to try and code.



As I open the script in Microsoft Visual Studio I have been advised to create four **Private** variables.

private float horizontal; - I’ve come to figure out that this might mean whether the character will move left to right or up or down.

private float speed = 8f; - This will determine how fast my character moves

private float jumpingPower = 16f; - This will determine how high my character is able to jump

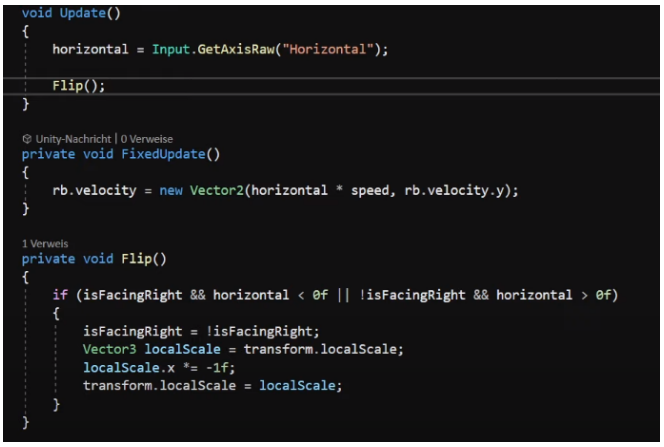
private bool isFacingRight = true; - This determines what side my character is facing when walking.

Next I’ll be adding Serialized Fields which I’ve come to figure out are there to let my script recognise the components I’ve added to my character. (The ground layer will be added later in unity.)

```
[SerializeField] private Rigidbody2D rb;
[SerializeField] private Transform groundCheck;
[SerializeField] private LayerMask groundLayer;
```

Next I have added all these lines of code.
I have come to learn that if I put

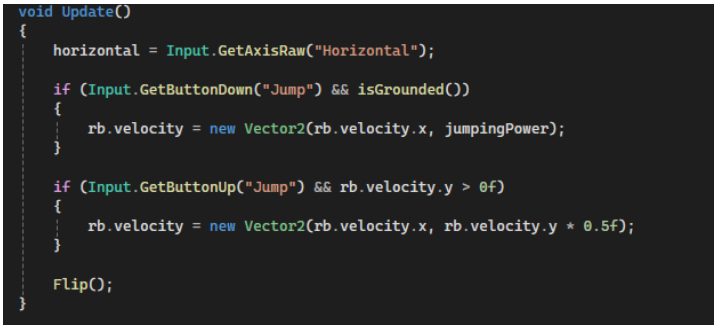
```
{
    rb.velocity = new Vector2(horizontal * speed, rb.velocity.y);
}
```



In “private void FixedUpdate()” this seems to take care of my character moving left and right. And if I were to type
“Private void Flip()” (which also seems to create a new command as flip doesn’t really seem to be a command integrated in Unity)
And under it write if (double tap tab) and put

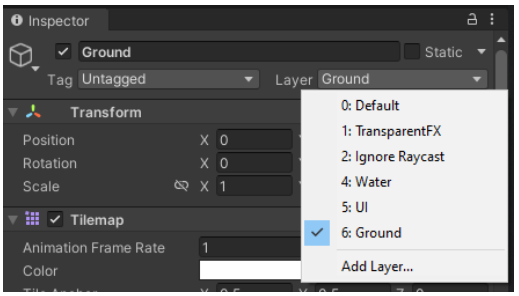
```
if (isFacingRight && horizontal < 0f || !isFacingRight && horizontal > 0f)
{
    isFacingRight = !isFacingRight;
    Vector3 localScale = transform.localScale;
    localScale.x *= -1f;
    transform.localScale = localScale;
}
```

Then it should take care of my character sprite turning to face the direction it’s supposed to be headed in.
I actually somehow had trouble figuring this one out as I’m not sure whether I just have somehow typed it in the wrong place or maybe I messed with the lines in the program, but my “if” code bit was somehow not under the private void Flip() line when I ran the script to see if it was working. Which meant that not only did my character sprite not change when facing a different side, it also confused unity and visual studio and it was begging me to explain what “Flip” was, which was kind of confusing to me. However, in the end I figured out that I had just somehow misplaced the code under a different private void and after I put it in the right place it was working perfectly fine.
(I have also had to put the Flip(); method in the void update window after I have created it in the private void.)

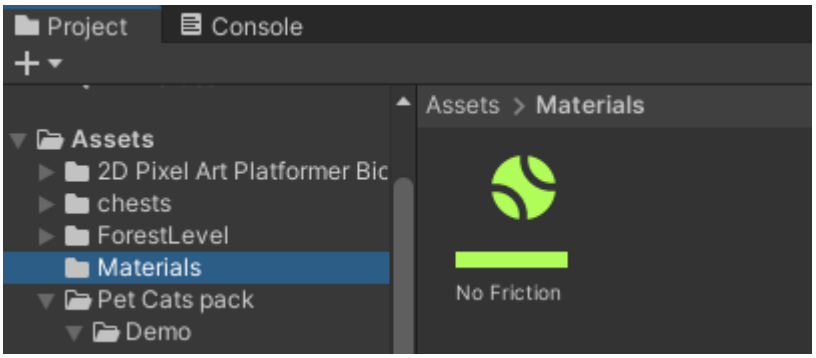


The last step was to make my character jump. I have put this in my Void Update window as advised. As I was listening to the tutorial I learned that the code has taken care of assigning the jump button, that groundCheck tells the character that it’s on the ground (the character is grounded) and that when it’s in that state it’s allowed to jump. It also takes care of how fast and far the character can jump.

Lastly, I had to assign a new layer called “ground” to every piece of asset I have thought of as being the ground. This was easily done by selecting all my ground tiles and adding the layer in the inspector.



Lastly, I was told to add a No Friction Material to my player so that it doesn't stick to walls, and to add the rigidbody 2D component and groundcheck to the script in the inspector window.



At the end I have also added a camera to my player as a child object and set the camera's transform to (0, 0, -1.5) . I did this so that when I play the game the camera follows the player! I have remembered doing this from a programming lesson from my last year.