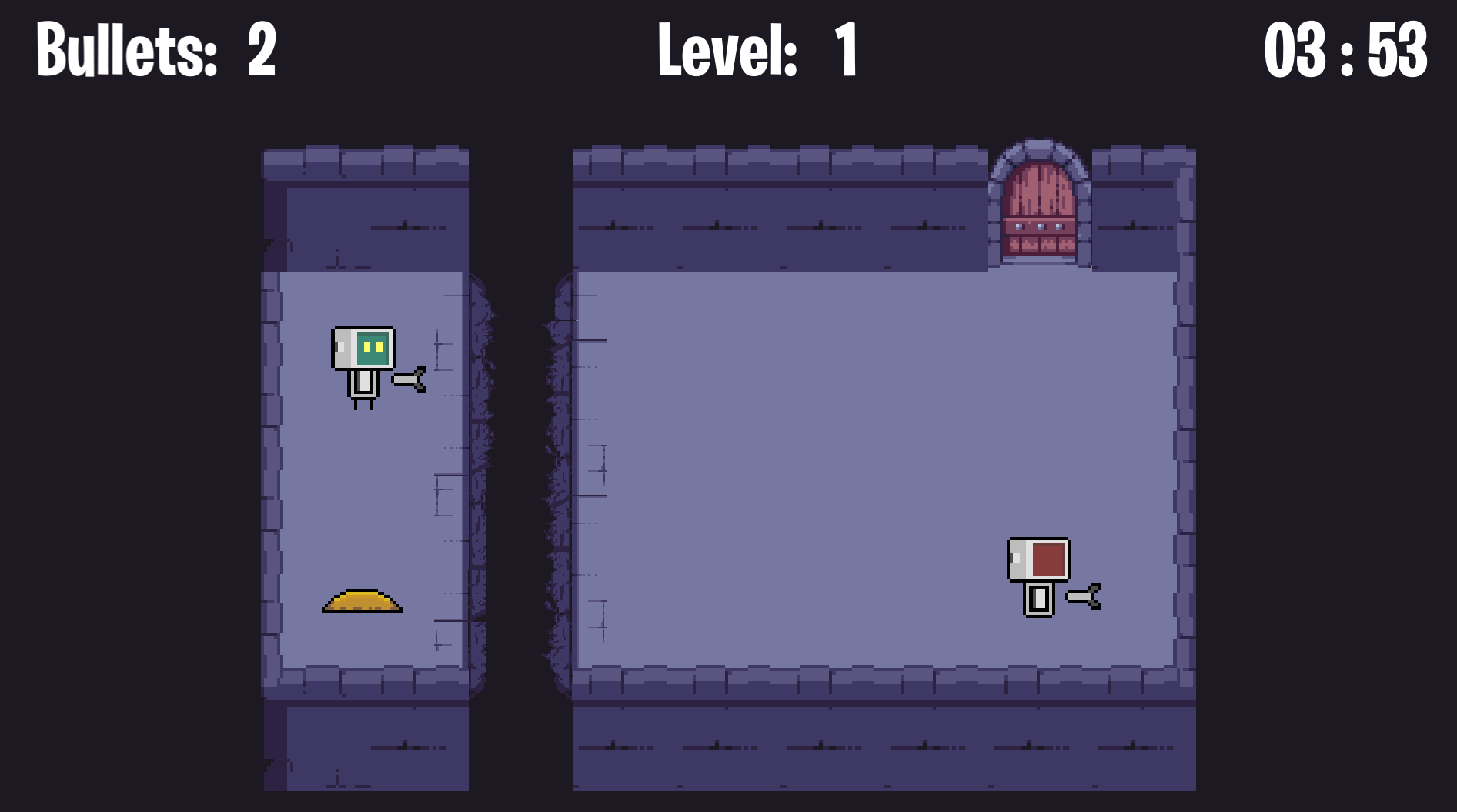
### Ethan Horrigan

# Developer Diary

### One Life



### Enemy Pathing

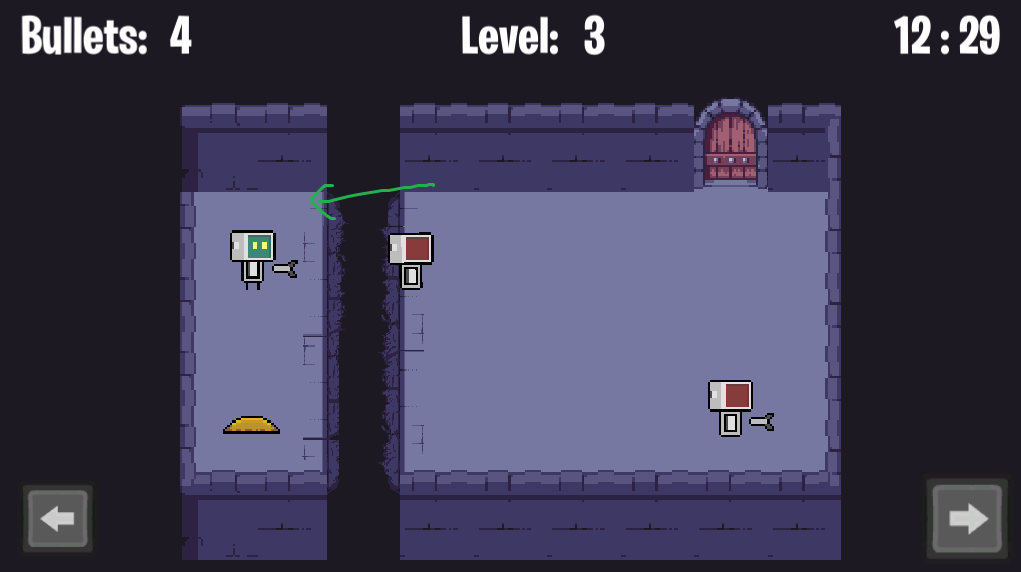
Adapted from: <https://www.youtube.com/watch?v=rhoQd6IAtDo&t=20s>

Enemy collision does not work when enemy pathing is enabled, to fix I need to rewrite the enemy pathing functionality to work with collision and collision variants.

|  |  |  |
| --- | --- | --- |
| COLLISION | COLLIDER | FUNCTION |
| Enemy | Player | Kill Player & Reset |
| Enemy | Wall | Nothing |
| Enemy | Enemy | Nothing |

Enemy pathing now only works if the player is “active”

Enemy pathing can also switch focus if another player has been activated.

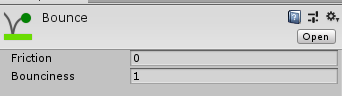


### Shooting

Players can only shoot when Active.

If a player shoots an inactive robot, the inactive robot now becomes the active player and the original robot is now inactive, but the original robot can be reactivated and vice versa.

Originally, I was using Vector3.Reflect and transform. Euler Angles to ricochet bullets off walls, but this was proven to be buggy and non-responsive, I opted for a simpler method instead, which was adding a Physic Material to the bullet.



This suited perfectly in this context due to the nature of the game.

Gun rotation was implemented to make the player feel less static and it also allowed the player to line up shots in order to make use of the ricochet mechanic.

By getting the mouse position of where the player wanted to shoot (Target)  
From the targets X and Y and from the players position X and Y, I could get the angle at which the target was at and rotate the gun to this given angle.

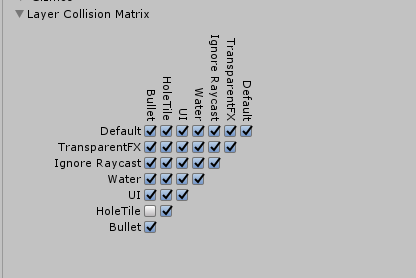
Bullets were handled in a different script because the game effectively has “two” players, I could decrement or increment the total bullets independently from the players. I did this to avoid players having double the number of bullets and ensure both players used the total amount of bullets available given the current level.

In conversation with the designer he said I could implement the ricochet mechanic if I had time, I’m glad I got it implemented as it feels like one of the most important mechanics in the game.

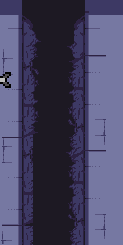
### Ignoring Collison

Ignoring Collison is an important mechanic within the game to ensure fluidity throughout.

Initially, I used Ignore Collision from the Physics2D class, but this quickly got hard to manage. I found a better way to achieve this by using Layer Based Collision (<https://docs.unity3d.com/Manual/LayerBasedCollision.html>)

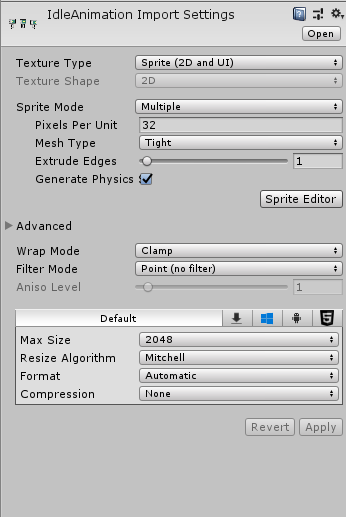


This was mainly used for the mechanic of allowing players to shoot over a certain object but could not walk over. E.g. the “holes” in the ground.



### Animations

Animations were made by multiple frames and then exporting each frame into a tile set. I imported the tile set into Unity & changed the Import Settings to fit a multiple sprite image.

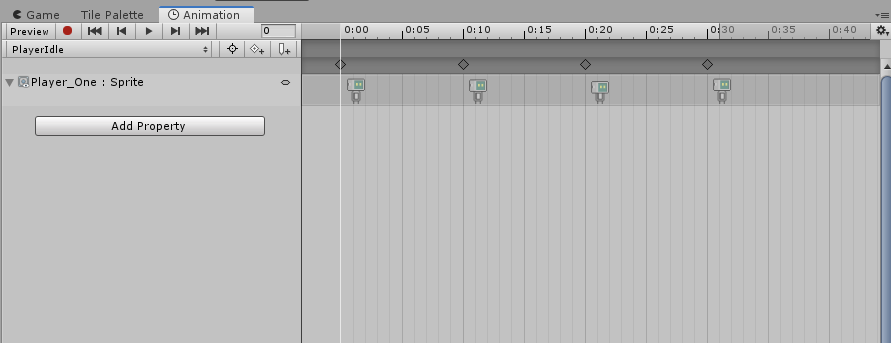


Setting the sprite mode to Multiple and pixels per unit = 32. I used the sprite editor to split each frame into its own sprite.

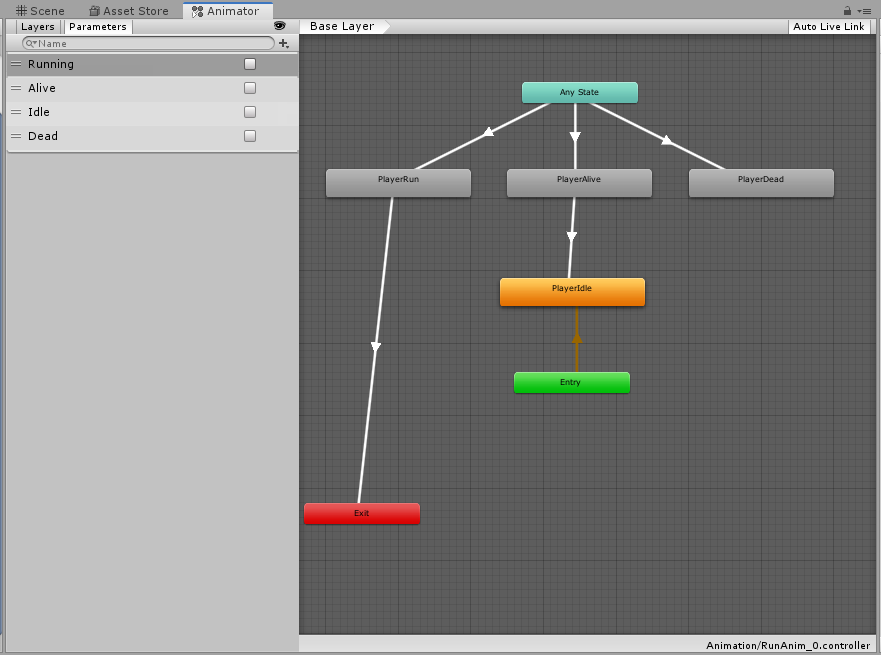
Setting the filter mode to Point (No Filter) and No Compression allowed me to scale up the sprite without any loss of quality at all.

Note: this only works for pixel styled art.

Using Animation in unity to create animations from the sprite sheet.



I used Animator Controller to allow for transitioning between animations.

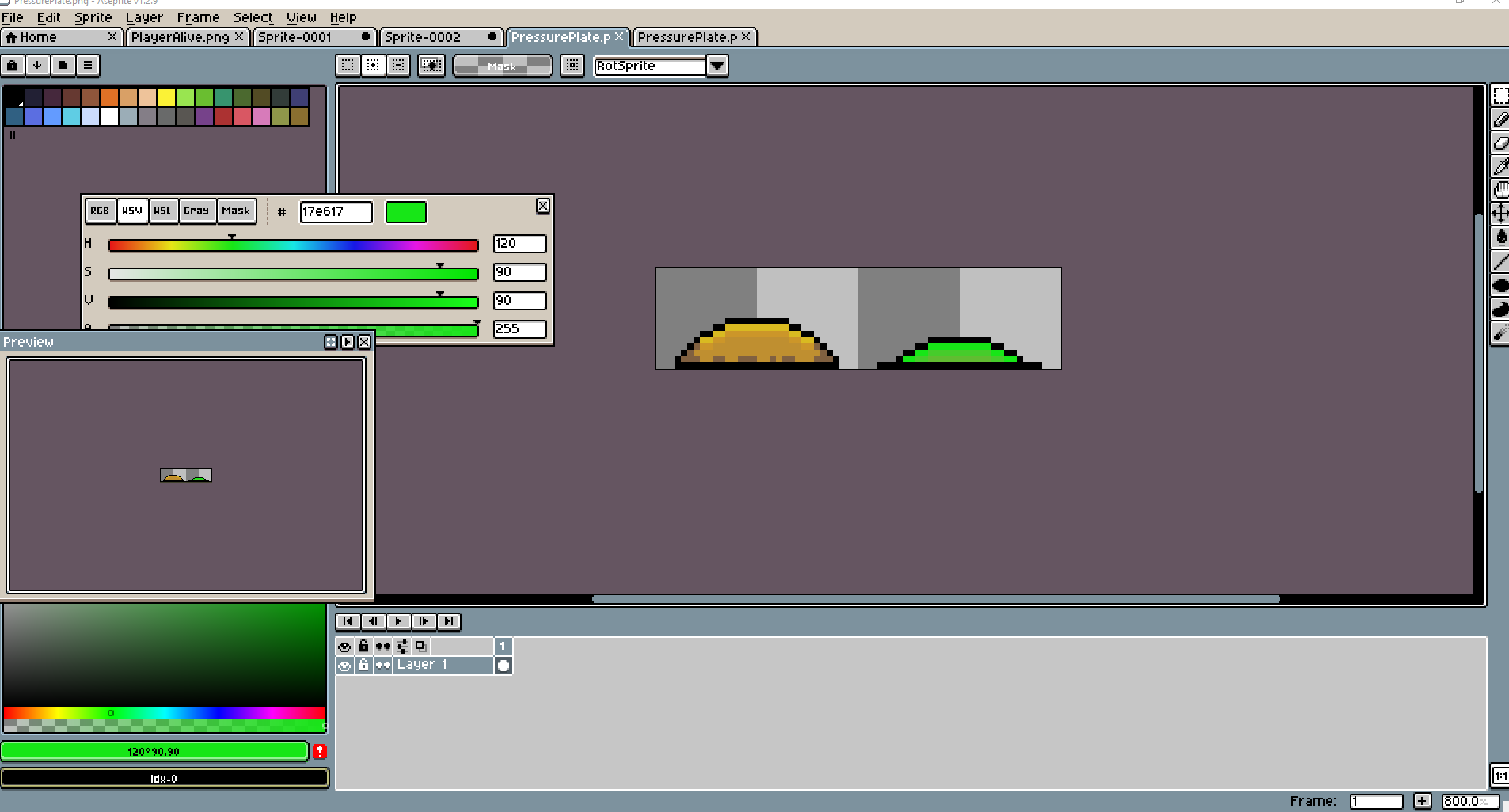


The list of parameters are just different Booleans which can be toggled on or off.

E.g. from any state if I move the player, the Running Boolean is toggled on, allowing the Player Run animation to be played.

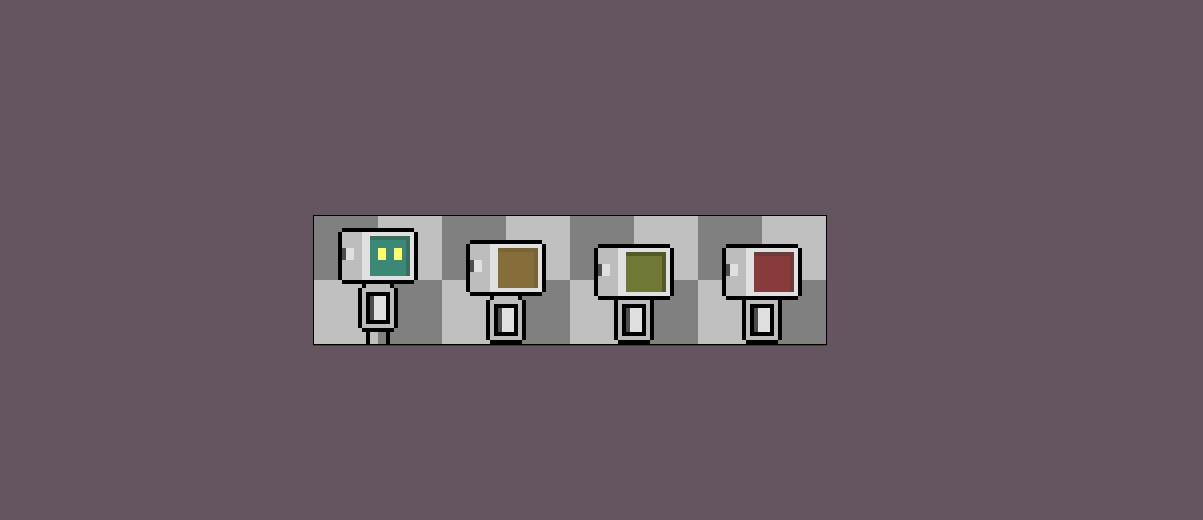
Program: Asperite

To create Sprites, Tile sets & Animations.

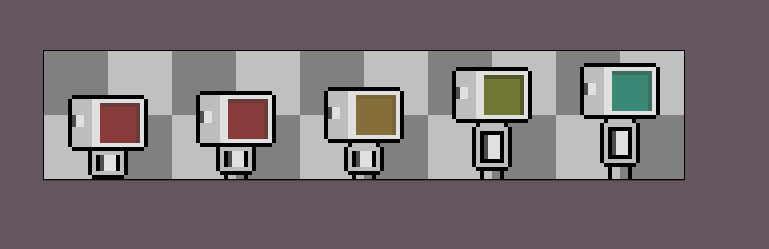


Animation examples:

Deactivate Animation:



Activate Animation:



### Sound Design

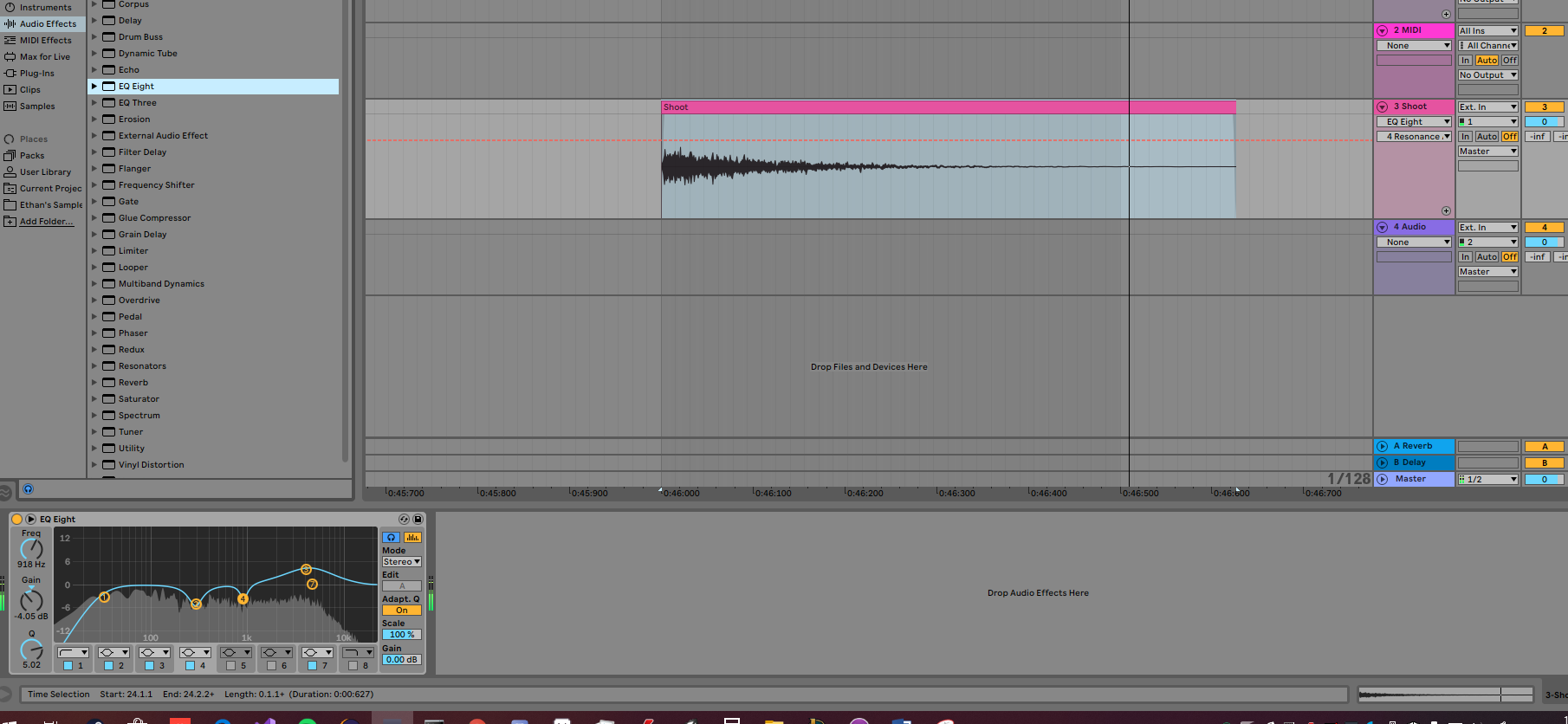
I used <https://freesound.org/> to obtain sound effects and then I imported them into Ableton for further manipulation and processing.

I did this because some of the sound’s quality were not great and I could also process them in any way with the use of equalizers, compression, reverb etc.

I compressed each sound effect with similar settings so the range of volume between sound effects were relatively similar.

This allowed for coherency between sound effects within the game.

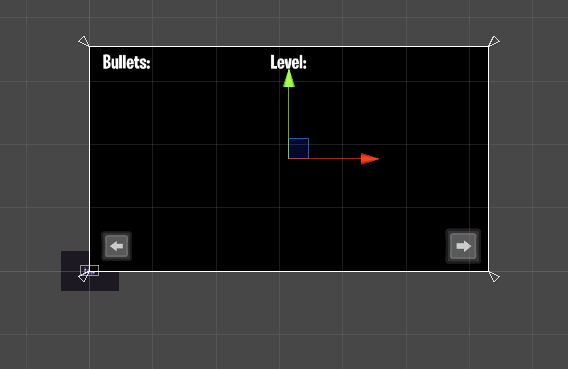
Program: Ableton



### Transitions

This was a simple effect I implemented to make the transitions between levels feel smoother.

To achieve this, I used a black sprite with the opacity at full (100) and created an animation which brought the opacity down to 0 to give a nice fade in effect.



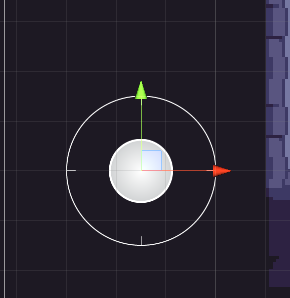


The fade out effect was the opposite, bringing the opacity from 0 – 100 in an animation.

### Joystick

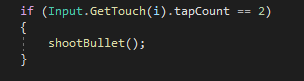
I used Unity’s input.touch feature to implement the joystick.

<https://docs.unity3d.com/ScriptReference/Input.html>



Initially I had the sprite above to move the player, but since you could move the player from anywhere on the screen, I felt the sprite was redundant and took up space, so I removed it.

Double Tap is used to shoot



### Pressure Plates

### 

Pressure plates to allow the player to open doors to progress through levels.

Only active players can trigger these pressure plates and doors only open while these are triggered.

### Constants

### 

I used a constants class to store to unique variables and to avoid unique values with unexplained meaning.

### References

Enemy Pathing [<https://www.youtube.com/watch?v=rhoQd6IAtDo&t=20s>]

Face Mouse [<https://www.youtube.com/watch?v=_XdqA3xbP2A>]

[Layer Based Collision] <https://docs.unity3d.com/Manual/LayerBasedCollision.html>

[Touch] <https://docs.unity3d.com/ScriptReference/Input.GetTouch.html>

[SoundFX] <https://freesound.org/>

[Font] <https://www.fontsmarket.com/font-download/burbank-big-condensed-bold>

[SceneManagment] <https://docs.unity3d.com/ScriptReference/SceneManagement.SceneManager.LoadScene.html>

[Music Playing Throughout] <https://answers.unity.com/questions/1260393/make-music-continue-playing-through-scenes.html>

[TypeWriter Effect]

<https://unitycoder.com/blog/2015/12/03/ui-text-typewriter-effect-script/>