Endless Enemies

A Post Mortem

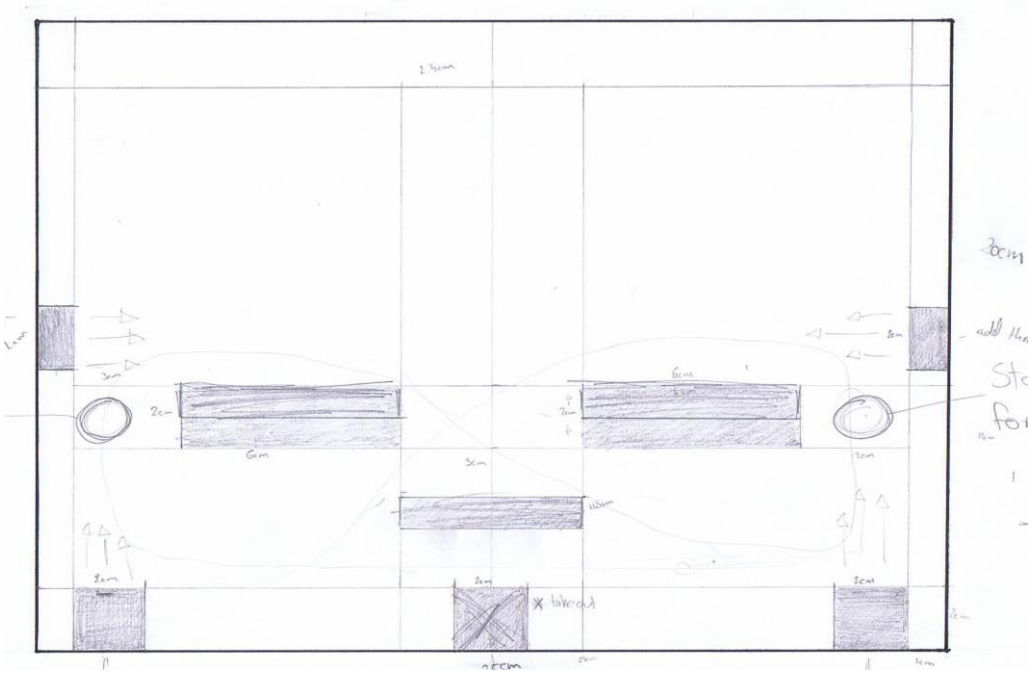
By Dylan Schoffer

I feel that I have accomplished most of what I set out to do with this project, although the art side is lacking as I only used glorious primitives, but the mechanics I did implement work well and the game runs great.

Since week 3, I have been designating a week at a time to completing a different aspect of the games make up. The first week was entirely on getting the basic mechanics into the game, second week was adding in the AI spawner, third week was focused on the audio and UI and the last week was left for polish and final tweaks. During each week I'd be making slight tweaks to the mechanics and assets used in the final product.

The third week when I was working on the audio and the UI was also my mechanical cut off period, I did want to add more stuff in like multiple projectiles from the player or health packs when a wave count is met but if I were to not follow my cut off I do not think the same level of polish in the project would still be there.

First week was grey boxing the level and basic movement mechanics. Last trimester I did a physics game that was a very basic version of what I made for this assessment, so I was able to get movement for the player pretty quickly, ame goes for the AI as it did not take me long to get a very basic AI together. The first major problem in my design was working out how to get the low level AI to perform a particular movement manoeuvre that created an infinite loop (See diagram below for inception of solution). I used particle systems as a visual aid and placed them inside of a trigger. I made the player and the enemy object into rigidbodies as well as gave them unique Tags, I then added a script to the trigger to add force to rigidbodies on contact.



When the AI would touch the trigger they would be thrusted skyward, but once they'd hit the wall they would not turn around. In the diagram you can see a few different shapes, the rectangles are the platforms and the squares were the triggers (acceleration pads) and the circles were another trigger to change the state of the AI.

So the first solution was to use more acceleration pads to push the AI away from the wall when they reached the height of the platforms, but then they would just walk back towards the direction they came from, so I thought about adding another trigger just below the upper pads that would change their movement direction.

I didn't end up doing this at all, instead I made the direction of movement based on a raycast facing the AI's current movement direction and when this short raycast touched a wall it would change its movement and raycast to the opposite direction. Now when the AI is thrusted upward it would come close enough to the wall to switch its direction, including the force from the acceleration pad it was enough to allow it to land on the platform and continue on its way.

Another major hurdle I had to overcome was that the AI would collide with one another and cause 'traffic jams' around the acceleration pads since one of the AI was not affected by them. I was initially trying to add Physics.ignoreCollision's to the AI but it was ineffective. After a bit of research I was able to work out that if I added them to their own Layer I could turn the colliders off for that layer, which stopped them from colliding with one another.

After this I begun working on the for loops that spawn the AI, I made the roof of the map the object that has the spawn script. The only complications I had with the for loops was that I wanted to make the movement direction that they spawned moving in was randomized. Initially they were using enums and states to control their movement but I had to change this to a bool so that I would be able to change it from another script.

After I got this working I came across a small bug in my code that I am still unsure on why it does what it is doing, but it does not give me errors so I digress. The enemy scripts have set damage values that are dealt to the player, however whatever values where put into the script were doubled when they were applied to the player.

For the "meta-game" I used an empty gameobject in the game scene that does not get destroyed on load, it takes variables from the spawn manager in the game scene and returns them as high scores on the game over page. However the gameobject that its referencing to grab these values does not exist anywhere else so it gives a reference error. This does not crash or stop the game from running so I digress.

Overall, I am sufficiently pleased with the outcome of this project. Admittedly, it is similar to the physics game I submitted last trimester but the AI mechanics, particles and polish added to this project is far greater than the previous project.

References

American Purpose. [Font]. Retrieved from http://www.dafont.com/american-purpose-stripe-1.font