Rewrite Development Journal

DavidMelville - 1742561

4/2/19

I started my project by setting up a repository on github for version control which I will manage with github desktop.

For the game itself I started off by making a copy of the Maze escape game we made in semester one as it has a lot of the basic functionality that I will need and is a good foundation to start from. The most important feature it comes with is the movement system and obstacles that kill on contact. My plan is to add movement to the obstacles and make them into my basic enemies then make derived classes for the individual enemy types. To begin with I removed the collectable objects and exit classes since I know I won’t need them. I decided to leave the moving camera in case I decided to make larger levels which would be very easy since this game loads its levels from a notepad file. I also left in the animation manager because I’m going to need it for my attack animations.

11/2/19

This week I focused on how my project is going to be organised and outlined what I need to do for each class. I added header and cpp files for each class I need to add which are enemy, bully, bat, cat ,sheep (enemy is the parent class) and weapon. I put some pseudo code and TODO lists in each of the classes.

* Enemy is based off the hazards from the maze escape game so they already have the ability to hurt the player. I need to make them get the player’s position and move towards it. I need to do this using sin and cos to make sure that they have a constant speed. They also need to be blocked by walls the same way the player is.
* Bully is just the basic enemy type with no special behaviour so it didn’t need anything added except which sprite to use
* Bat uses the same movement but it can ignore obstacles. I realised as I was writing the pseudocode that I should have used the bat as the parent class since it doesn’t need a collision function and have the other enemies as children. I didn’t change it straight away because the enemy was carried over from maze escape and I had to make sure it wouldn’t affect anything else if I changed it. I prefer to be cautious with c++
* The Cat class needs to get the distance between itself and the player so it knows when to stop for a moment then charge. I assume there is built in functionality to get this using Pythagoras’s theorem but I’ll need to look it up when I get to this point.
* The sheep is similar to the cat but it needs to be invincible until it is frozen mid charge.

18/2/19

I’m focusing on maths this week as my prelim is coming up and I’m my preferred university courses both require a B at higher for my conditional.

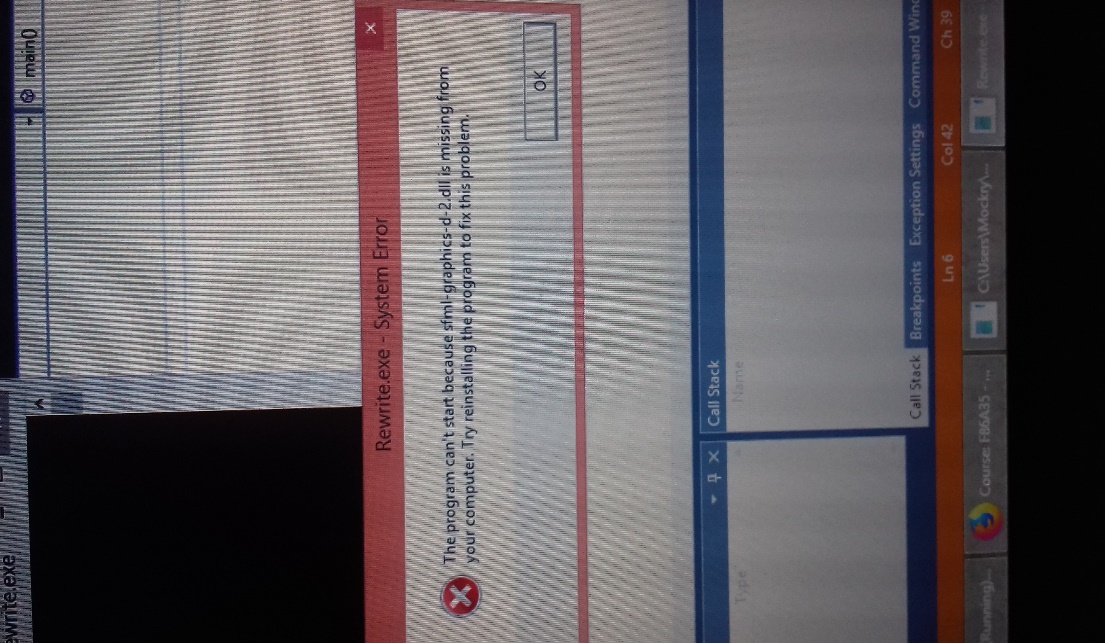
I did look over my classes to see if I could think of anything to add.

I put a stored speed variable into the enemy class so that I had a way to revert it after the time freeze. I put a note into my level manager about how I would handle my enemy spawning. I plan on making a version of the LoadLevel function that only searches the file for spawn points and run it every twenty seconds. In preparation for this I added variables for total waves, remaining waves, spawn delay and time since spawn.

25/2/19

Continued studying maths and doing past papers.

4/3/19

Visual Studio is acting up on me. When I try to

open Rewrite it tells me that I’m missing an sfml

library (pictured). Strangely it can open projects

I started in college or downloaded from git

Just fine. I’m going to consult Sarah about it.

I want to avoid reinstalling visual studio if I can

since that can be problematic by itself.

11/3/19

I happened to see some of my classmates ongoing projects in Unity and I realised I’d much prefer doing my project in Unity as well. I asked Sarah if I was allowed to change from sfml and she said it should be fine but she’d ask Marion just to be sure. I never considered Unity when I was planning because I’ve never used it before and knew nothing about it. Emma explained how adding components and scripts works and It seems a lot easier to work with than c++. On top of that I much prefer working with an editor where you can see what you are doing, how things look and move things around by hand. I found a promising tutorial series on youtube that I can use as a basis for my game. Its actually for a top-down shooter but it’s the movement and aiming I’m interested in. I should be able to change the way attacking works once I’ve got a handle on unity.

<https://www.youtube.com/watch?v=lkDGk3TjsIE&list=PLiyfvmtjWC_XBKJVuCtMXrkNnMDNB16W9>

18/3/19

Started getting used to Unity by going through the built in basic tutorials. Basic is the key word there and it was more a waste of 15 minutes. I started the tutorial series by gamesplusjames which begins with movement and aiming. Movement was easy as Unity has built in functions for detecting input and applying movement along the X and Z axis. Facing the mouse was more complicated. I assume there are several ways of doing it but this tutorial taught me to create a ray from the camera towards the cursor and detect where it intercepts the plane I created for the ground. I save this intercept as a vector 3 and make the player face it but only on the x and z planes. I leave the Y direction as one so the player doesn’t face down towards the ground.

Following that I made an enemy that follows the player around. This was less complicated than the playerObject. The player movement is controlled by a script called PlayerController. My EnemyController script searches the level for any object with the PlayerController script which will only ever be the player. I then simply set it to face the players location and move forward.

This did result in the enemy trying to climb over the player but I figured out I can freeze the Y movement of the rigid body component. I also froze the rotation so that objects only move around when I tell them to. By doings this I gathered than rigid bodies are Unity’s equivalent of rectangles in c# and c++ in that they are the things than are interacted with. Put another way, an object with no rigid body might be visible to me but it doesn’t exist to the program.

25/3/19

The next part of the tutorial was adding a gun and bullets to the player. I thought about skipping this one but I figured even if I don’t need guns for this project it’s a handy thing to know and its extra practice at Unity. It turned out to be worthwhile since this video is the one that introduced prefabs. The gun works by continually spawning bullet prefabs at the gun’s location in theory. I can’t get it working and I don’t want to just watch the video again since I feel like I’m probably missing a small detail.

I made a basic square texture on draw.io to use for my ground before trying to find my mistake with the bullets again. I couldn’t figure out what I was missing though and watched the tutorial again.

Turns out that the script spawns a BulletController type object on the gun but I needed to specify exactly which one. On the plus side I realised that in Unity public variables mean they can be altered directly from the editor without going through visual studio.

1/4/19

Continued the tutorials and added a script for managing enemy health and making them killable. As I was testing it I decided to keep the shooting mechanics rather than change it to melee only as it feels fun to use as it is and I can still maintain the tension by making the bullets despawn quickly and giving the player limited range. I finished the last tutorial and added a player health manager that lets me take damage and die as well as flash white to indicate damage being taken. This was the last tutorial in the series which I suspect is unfinished. Its just as well though because I want to learn how to do this for myself.

8/4/19

Had to work at my job a lot this week so I was mostly planning what I would do next. I figure that I need a script that will track how many enemies are active, the time limit and how many waves are remaining so it can control beating or failing the level.

I also decided how what to do about my enemy models. Since Unity has built in lighting and shadows in its 3d mode using basic 3d models would look better than if I tried to draw 2d sprites. This would be easier on my schedule because I have an adobe subscription up until the 10th of May but I can use 3d max for as long as I want.

15/4/19

Created a UI manager script that tracks the player’s current health, active enemies and time remaining and displays them on screen. My GDD said that the health display would change colour as it decreased so I took the health part out of this script and made a separate health display script.

I implemented my freeze time mechanic by adding a function to my enemyController script. I would rather have the playerController detect the input and then find and freeze each active enemy but I don’t know how to do that and I’m trying to only use tutorials when I have to. Instead I have the enemy controller detect if the right mouse button is down and sets the enemies movement speed to 0 for 3 seconds before returning to its original value. I added a variable for storing the original speed so that it will work on enemies with different movement speeds. The enemyController already had a pointer to the player’s health manager so it could apply damage so the freeze function could also apply the health penalty.

I initially had a problem in that the health penalty would apply once per active enemy but I fixed it by making a Boolean called pendingDamage on the player controller. Freezing time sets this to true and if it is true it will apply the health penalty in the LateUpdate function. This ensures that the penalty can only be applied once per frame.

22/4/19

Took a week off to focus on the upcoming Maths Exam

29/4/19

Got through the exam. I added barriers to my first level in unity and saved them as a prefab just for good measure. I used this week to catch up on my AI and 3d modelling work.

6/5/19

Created a script for spawning enemies. I copied the method used for spawning bullets and changed it to spawn a EnemyController at the spawnpoints position. I made an empty object , added the script and assigned the basic enemy to it. Seemed to be broken at first until I realised Unity defaulted the Y position to 80. It was working fine the enemies were just above the camera.

These enemies were unkillable though and they were being created without a healthManager so I took all the data and functions from it and added it to the enemy controller which fixed the problem. This probably isn’t the proper way to do enemy spawning but I’m still trying to avoid tutorials.

I checked my GDD to see what other enemies I need which reminded me to add regenerating health. This was done with a timer in the player health manager and I also created a function to restore 5 health which is called whenever an enemy dies.

13/5/19

I caved on my no tutorial rule and looked up how to add pathfinding. I created and navMesh for my level and added a navAgent component to my enemy prefab. I changed the enemyController’s movement to use the navAgent by simply setting its target to the player’s position.

I started making my Cat enemy by inheriting from the basic enemy and adding a charge mechanic. Unity has a function for getting distance between to objects so writing the function was easy. I couldn’t get it to work for a while until I realised I had to change the basic enemy’s functions to be virtual. When I figured this out I could set the cat to start charging when it is in range and use base.Update when it isn’t. I was quite pleased with myself when I realised I had made a finite state machine.

The charge didn’t work as intended though as the navAgent has an easing function built into and even if I turned the acceleration up it was a bit sluggish. I made the charge use the old movement system of direction \* speed and switch back to the navAgent after the attack. The pathfinding can act a bit strange if the charge hits an obstacle but the cats are only used on the second level which is fairly open.

The sheep enemy inherits the CatEnemy script and I just changed the charge speed and detection range. I changed its update a little so that it has a shield whenever it isn’t frozen while charging and made the freeze timer countdown without also countdown the chargeTimer. (Its easier to explain when looking at the script).

20/5/19

I made a rough boulder shape in 3ds max and imported it to use as an obstacle. Using it I finished all my level layouts.

I started implementing a reset level function and load next level function. The reset worked fine but I couldn’t get the levels to advance. I asked Ed about it and he told me I had to add all my scenes to the build list. Armed with this knowledge I got my level transitions working.

I wanted to add a flying enemy that would avoid ground based obstacles like the pond I made in level 2. The pond has a collision box to stop the player walking over it and Ed assured me it was possible to make certain enemies ignore it. I don’t know what you would call this process though so I couldn’t find any tutorials on it. I got around this by having the BatEnemy fly straight at the player and ignore the navMesh. I slightly raised the spawnPoints with bats assigned so they were high enough to fly over the pond but low enough to be shot. I’m not proud of it but player’s wont know the difference.

27/5/19

All my functionality is down by this point so I focused on decoration.

I made a capsule and fancy hat in 3ds max, unwrapped and textured them for my player character. I did the same for the basic enemy but they get a baseball cap. My animals are basic blobby shapes but I’m not pretending to be an artist. I imported my outdoor scene from 3d modelling and added a sign saying Rewrite Inc. I put one of each model in a line in front of the building and turned this into my title screen. I added a text box showing the controls and made a basic script to load level 1 on left click.

I had some trouble adding scripts to my new models and rather than figure out why I went to my current prefabs and turned off their rendering meshes and put the my imported models on top of them.

I looked up how to do animation in unity and used two key frames to make the satellite dishes on the building rotate to cover the animation requirement for the project. They behaved fairly strangely at first but I realised that they were using interpolation by default and found out how to make their movement linear so they have a smooth constant rotation.

Lastly I downloaded some sound effects from freesound and found some good free music to use for my levels. The songs and some sound effects used a creative commons attribution license so I made a quick credits page showing where they came from. I added sound effects to my prefabs but ran into a bit of trouble. The basic way of playing sound is use GetComponent<AudioSource> then play that audiosource (clip) but this only works when there is only one audio source on an object. I also couldn’t get the enemies to play their death sounds as killing them also destroyed the audio source.

<https://www.youtube.com/watch?v=VrOTsJWIk4s>

This tutorial taught me about serialised fields which lets me declare several audiosources in a script and assign the clips to the variables in the editor like with public variables. I put all the enemy death clips on the player, declared 4 audio sources and assigned the enemy deathsounds. I also made a public string called enemyType and function to play the clips that passed in enemyType and played the appropriate sound using if statements. I probably could have done it as a switch statement but this was the last thing I had to do so I went with what I knew.