Francesca-Chapman-Programming_Learning_Journal_2

Entry 1: 08/02/22

Tutorial Used: "THE ENEMY! | 2D Top Down RPG in Unity #3 | 2D Game Dev Tutorial" by Muddy Wolf - https://www.youtube.com/watch?v=b0O71Wa09nI&t=474s

In this tutorial, I learned how to use triggers to create a radius that detects when the player is in range and causes the enemy to follow them. At first, I couldn't get the trigger to detect the player when it was inside, however I realised I hadn't set it up appropriately, just using a basic sprite with no tags, colliders or rigidbody. Once I made these adjustments the triggers were then able to detect the player and the rest worked as intended.

Tutorial Used: "Enemy Attack & Player Health | 2D Top Down RPG in Unity #4 | 2D Game Dev Tutorial" by Muddy Wolf - https://www.youtube.com/watch?v=VOdYtqV_meo&t=533s

For this tutorial, I added an attack component to the enemy from the previous tutorial so that it deals damage to the player upon contact. It worked initially, until I added the attack counter to limit how often damage could be dealt. I checked my code to make sure it was identical to what was shown in the video, which it was, so instead I made a small change to see if that would make mine work. Originally the CanAttack float wasn't given a value, but when I changed it to equal 1, the code worked like it did in the video.

Entry 2: 15/02/22

Before starting any new tutorials, I went back to my enemy code in order to check it and make any adaptations if necessary. I wanted to make sure it could work alongside my previous top down 2D player component, and since the enemy code uses player health as an example of its attack function, I needed to make sure it would be able to interact with the player health from the player component. I only had to change the reference to player health in order for the code to work, however my player was gaining health rather than losing it. Comparing the two scripts, I realised that the function the enemy attack was referencing takes the damage amount and subtracts it from the total health of the player, and the enemy attack further tells it to subtract the amount, therefore it is subtracting a negative number, making it a positive amount. So, I changed the attack script to add the damage amount which then allowed it to be subtracted from the health properly.

Seeing the enemy targeting working alongside my player, I noticed a few things that required some polish to make it work better. When the player's health reached 0 and it went to the death state, the enemy was still following it and pushing it along, which it shouldn't be able to do. So, I added some conditions to the targeting to only target if the health is above 0, as well as add a new function to stop targeting if it the health reaches 0. This makes it work a lot nicer as the enemy stops detecting the player once they are dead.

The next thing I wanted to try was giving my enemy the example health script from my player. However, when I tried this, the enemy wouldn't take damage when attacked, but the example object with the same scripts would. Upon further observation, I realised that the example object was on a specific layer that the player used to identify objects to be damaged, and so changing the layer of the enemy fixed the issue, though damage was being dealt twice each time due to the extra collider being used to detect the player from a radius. To fix this, I made a separate object for the radius and made it a child of the enemy, but then it would follow the player and not take the enemy with it, so I made the enemy a child of the radius, which then worked. The enemy's damage now went down one at a time, however it created errors as the attack was still trying to deal damage to the range even when the range had no health. I fixed this by changing the layer of the range so that it would no longer be detected by the attack.

Entry 3: 20/02/22

Tutorial Used: "Topdown 2D RPG in Unity - 12 Enemy AI" by Hundred Fires Games - https://www.youtube.com/watch?v=dy8hkDmygRI

I was unhappy with how my current code was working as it was getting too messy and complicated, so I tried a different tutorial. It worked similarly to the other tutorial, ultimately detecting the player within a radius and following them, however this one didn't require another collider, instead the radius was made via code, meaning that OnTriggerEnter isn't needed and therefore won't complicate things when I add back the health system. As well as this, the tutorial also shows how to make the enemy return to its original position when the player is out of range, something that the other tutorial didn't cover. I followed the tutorial mostly fine, however I noticed my code stopped working once I added in the range. After trying a second time, I realised I had set the value of the range far too low due to misreading it in the video, and when this was changed the code worked as intended.

Entry 4: 22/02/22

Like what I was trying to do last week, I wanted to make sure my enemy would be able to interact with my player efficiently. With the new way of targeting, taking the health code from my player and implementing it was a lot easier and didn't take much to adapt unlike before. I know this means that I spent a lot of time last week fixing things only for them to be scrapped, but it was still good to solve those problems and get the experience from them, even if I didn't end up using it.

Everything worked together now, but I wanted to make some adjustments to my enemy code to make it more polished, since killing the player by destroying the object gave errors as the enemy's target was null but it was still trying to use it. I first tried to make a reference to the target in the player's health script, setting it to null when the player's health reached 0, but it gave errors due to how the current health was calculated and even then, the enemy script didn't seem to detect it turning null at all. I decided to change the player's death to an animation, since the actual player component isn't destroyed, so I changed the conditions of stopping the enemy targeting to detecting what animation state the player is in, but again this did nothing. I then thought to make reference to the player's health in the enemy script, and make the enemy check this value when targeting. I initially had it so that any time the player is in range, the enemy only targets if the health is above 0, and then within that statement, another statement will check if the health then reaches 0 whilst the player is in range, and will make the enemy return home if true. This worked to a point, it stopped moving towards the player, however it wouldn't return home until the player was moved out of range, and it also created some strange behaviour where if I moved the dead player to be in the enemy's range again, the enemy would stay still but play its walk animation in the opposite direction that the player was facing. I figured that this was because for the targeting, the health check was an extra condition added on to the range check, with the death check as a separate condition within it to tell it to stop, meaning that it was detecting the player and targeting it, and immediately stopping since the health was at 0. To fix this, I took the health check out of the range check and made a separate condition within it alongside the death check, and now it works perfectly.

Entry 5: 01/03/22

Tutorial Used: "Enemy Item Drop Upon Death - Intro To Arrays" by Skitziod Studios - https://www.youtube.com/watch?v=OtJklb2kKtg

With my enemy now working properly, I wanted to add one last function to make it feel more complete - a random item drop. I went through 3 tutorials before this one, as they were either too complicated in their method or too simple and not making the drop randomised. The tutorial I used showed me how to use arrays to randomise what item is instantiated when the enemy is destroyed, something I have used before, but I am still not entirely confident with. Unlike the other tutorials I tried, I had no issues with it at all, and it worked on my first attempt. This completes my enemy component, and I will move on to my next one which will be based around the items I just made.

Entry 6: 08/03/22

Before moving on, I wanted to do one final test of my enemy component to make sure it could slot in easily with my player controller component as I have not tested them together for a couple of weeks. Luckily, it all fitted together smoothly, and I only had to edit the references to script names and assign layers and a couple of objects - no major changes were needed.

Tutorial Used: "Unity 2D Collecting Coins Tutorial" by Xlaugts - https://www.youtube.com/watch?v=DZ-3g31jk90

In this tutorial I learned how to interact with objects using triggers and keeping score of them using a score manager and updating this score on the UI. I had a small problem where the player was picking up coins as they were destroyed when interacted with, but the score wouldn't update. I fixed this by changing the player's tag to the tag the coin script was expecting. This confused me as the coins were still being destroyed even though the player was untagged.

Tutorial Used: "HOW TO MAKE HEART/HEALTH SYSTEM - UNITY TUTORIAL" by Blackthornprod - https://www.youtube.com/watch?v=3uyolYVsiWc

In this tutorial I learned how to make a heart display system that updates depending on the amount of health. It was straightforward and I had no issues with it. It didn't show everything I needed for my heart pickup but it was useful and I think I will be able to make the other parts myself.

I adapted the previous code to tie in with the health script on my player component so now the health integer would be the same as the health integer the player uses when it gets damaged. I then made a new script similar to the coin collect script and I created variables I then matched up with on the health script, and wrote an if statement to check if the object touching the heart was the player and if the player's health was below the maximum health, which would then add 1 on to the health and destroy the heart. The heart counter would update when the player took damage, but I couldn't pick up the hearts. I realised this was because the if statement was checking if the health was more than the maximum health, so I fixed that, so it picked up the heart, but the health wouldn't update. I fixed this by getting rid of the named health reference that was getting added to so that it was just taking the health from the health script. This completes my pickups component, but I will do some more testing to make sure they work as prefabs.

Entry 7: 15/3/22

I did a quick test of my collectibles to make sure they work as prefabs dropped after an enemy is killed. Luckily it worked first try, so nothing needs to be changed and I can move on to my next component.

Tutorial used: "Pokemon Mystery Dungeon Level Generation | Unity Tutorial" by Slug Glove - https://www.youtube.com/watch?v=WK_qNkz4ZlA

For my next component, I decided to look at random dungeon generation. I haven't finished with this tutorial yet so I haven't gotten to a point where I can test it, but I will continue with it next time.

Entry 8: 22/3/22

Tutorial used: "Pokemon Mystery Dungeon Level Generation | Unity Tutorial" by Slug Glove - https://www.youtube.com/watch?v=WK_qNkz4ZlA

I continued with the tutorial from last time, and I got all the scripts written, however when it got to setting everything up, it became too confusing as the video was sped up too fast and had skipped some parts, and I just found it too difficult to understand. It's disappointing, but I will have to find another tutorial that covers this, even if it's not exactly how I wanted the final result to look.

Tutorial used: "RANDOM DUNGEON GENERATOR - EASY UNITY TUTORIAL - #1" by Blackthornprod - https://www.youtube.com/watch?v=qAf9axsyijY&t=205s

I found this tutorial series which is has the same sort of result I want but only has one sized room and no hallways or objects in it, so it's less complex. This tutorial mostly showed me how to set everything up; the rooms are made up of tiles set up precisely, so they snap together, and at each doorway there is a spawn point which has a trigger collider and a rigidbody, as well as a script that will handle whether or not a room can be spawned and what type. There wasn't much code writing in this tutorial, so I haven't been able to test anything yet, but I will carry on with this next time.

Entry 9: 29/3/22

Tutorial used: "RANDOM DUNGEON GENERATOR - EASY UNITY TUTORIAL - #2" by Blackthornprod - https://www.youtube.com/watch?v=eR74EjkA_4s

In this tutorial I learned how to use multiple arrays to classify rooms by door direction, so they can be used by the spawn points to decide what rooms can be spawned there. Each spawn point has an integer corresponding to a direction which determines what array to use. If it collides with another spawn point, it will be destroyed so it cannot spawn a room and overlap with an existing one. However, when I came to test it out, the rooms were spawning, but it wouldn't stop, and they would overlap, and it would keep going and eventually crash if I didn't stop it. I checked my tiles to make sure they were all placed in line with each other and each room's tiles were identical, which they were, so I went over all the spawn points and ensured they all had the right directional value, but this still didn't work. I'm not sure how to fix this, so when I have more time I will try this again, but if that doesn't work, I will look at finding something else to do for a component.

Entry 10: 5/4/22

Tutorial used: "RANDOM DUNGEON GENERATOR - EASY UNITY TUTORIAL - #1" by Blackthornprod - https://www.youtube.com/watch?v=qAf9axsyijY&t=205s

Using this tutorial I redid my room prefabs in hopes of solving my problem. Though nothing is different, I did set them up against the scene grid to make sure the sizes and positions were exactly like what was shown in the video, rather than align them with the main camera in the centre, which is what I did last time.

Tutorial used: "RANDOM DUNGEON GENERATOR - EASY UNITY TUTORIAL - #2" by Blackthornprod - https://www.youtube.com/watch?v=eR74EjkA_4s

I kept the code from last time as I had made sure when following the video that my code was exact, so I just assigned the different rooms I just remade to the arrays and tested it. I was having the same problem as before, but this time, everything was at a diagonal offset. This led me to believe that the issue was something to do with the positions of the whole rooms themselves, as I made them in a different space than what I did last time. I got all my prefabs into the scene and aligned them to the centre with the main camera, which changed their parent objects positions from 0. I took apart the parent objects to reset their positions back to 0 and put them back together, so the parent objects holding the rooms themselves, the walls and the spawn points were all aligned with the centre. When I tested it, my solution worked and now my dungeon generated a clear set of rooms from the entry room, to a final room where no more rooms could be spawned. I can now move onto the last part of the tutorial.

Tutorial used: "RANDOM DUNGEON GENERATOR - EASY UNITY TUTORIAL - #3" by Blackthornprod - https://www.youtube.com/watch?v=CUdKdHmT8xA

In this tutorial, I was shown how to clean up the dungeon generator by spawning closed rooms when two spawners collide and destroy each other before they can generate a room, leaving doors empty. As this then generates a closed room at the entry point, another point is added to the centre with a small script to destroy them to fix this issue. Lastly, I learned how to use lists that add each generated room to it in runtime, in order to find the final room which is the exit, so it can spawn a boss.

I also made the spawners destroy themselves after a little while as once they have finished spawning there is no need for them to be in the scene. I had no issues with this part of the tutorial, so I will move on to my final component next.

Entry 11: 12/4/22

Tutorial used: "Unity Switch Camera on Trigger Enter - Cinemachine Trigger Zone Tutorial" by Dan Pos - Game Dev Tutorials! - https://www.youtube.com/watch?v=nkL-VvMmWHg&t=501s

In this tutorial, I learned how to set up Cinemachine cameras that switch between each other when the player enters their respective trigger zones. I had a small issue where if I moved the player to one zone and quickly went to the other before it had completed the transition, it would remain in the other zone, so my player was off the screen in the other zone. I fixed this by reducing the size of the triggers, so they weren't touching, and had a clear gap between them, so they wouldn't both be triggered and override one another if the player moves between them too quickly. This is my final component package done, meaning that I can start combining them after I've done some more tests.

I wanted to test my camera scripts with my dungeon generator, by setting up camera zones for the room prefabs to see if they would work when instantiated. I made a small edit to the camera code so that it would automatically find the cinemachine camera as a child object of the zone rather than dragging and dropping to make it easier. When I tested it, the rooms were spawning but the entry room's zone would disappear as well as my test player. I realised this was because the destroyer in the centre of it was detecting the zone trigger and destroying it as the code will destroy anything that collides with it, so to fix this I changed the destroyer code to only destroy objects tagged as spawn points. I had another problem however, where I couldn't see any of my rooms in the game view despite them being present in the scene and the being in the range of the camera, and this only happened when the main camera had the cinemachine brain. After looking around, I found that changing the near clip plane of the virtual cameras to 0 made the objects visible again, however when they were being spawned, their near clip planes were being set to 0.01, which made everything invisible in the game view again. To fix this, I added some code to set the near clip plane of each zone's virtual camera to 0 at every frame.

The last issue I had was that the main camera was focusing on the last room instantiated, rather than staying in the entry room where the player was. I was stuck on it for a while, but then I thought to have each virtual camera be inactive upon instantiating, and then set active when the player enters its trigger, however this caused another error as because the trigger assigned the camera at the start, it couldn't find it anymore because it was inactive. To fix this, I used the following for loop:

```
for (int a = 0; a < transform.childCount; a++)
{
    transform.GetChild(a).gameObject.SetActive(true);
}
```

which I found in this help forum - https://answers.unity.com/questions/894211/set-objects-child-to-activeinactive.html - in order to locate the child camera and set it active when the player entered the trigger. Once the camera was set active, I then got the trigger to assign it and carry out the camera switching. So, all the virtual cameras apart from the starting one are inactive, and are set active when the player enters the zone, from which point they stay active as they only mess up the main camera focus if they are active when instantiated.

I am happy that I got these two components to work together as I was unsure if they were going to work. I want to also test my dungeon generator with my enemy and pickup components, which I will do next time.

Entry 12: 19/4/22

Today I tested my dungeon generator with my enemy. Instead of having lots of different room prefabs with different enemy layouts for each room, I thought it would be more convenient to have each room contain a number of spawners that will spawn an enemy if a randomised integer is a handful of certain values. I had an issue where my enemy component wouldn't work when instantiated as a prefab as the player had to be assigned in the inspector, but I fixed this by having the player be assigned via code. Another issue I had was that too many enemies were spawning at once, so I had to play around with the range until I found something that was a bit more balanced. Finally, I realised that my enemy when returning to its home position would struggle if it went to another room. So, rather than remove the returning function, I decided to add colliders to the doors with layers that only the enemy will detect, so enemies will be confined to one room. It's not the best solution, but I would have to rework my entire enemy again otherwise.

I also set up my packages, arranging everything into the correct folders and writing the documentation, so they are ready to be submitted as well as being able to be easily implemented into my game project.

Entry 13: 26/4/22

Today I started work on my game project. I started by importing my player package from last semester, and imported the other packages one by one, doing quick tests as I did so. I didn't encounter any major issues, other than having to edit the names of the scripts as some clashed or referred to different names, but these were easily fixed. I had to edit some prefabs to combine components from other packages and I had to set up the UI, but again this was easy to do. Now I have a working level, I just need to make some menus and transitions to make everything flow together.

I added a game over menu so that when the player's health reaches 0, it pulls up a menu which stops the game running and allows you to quit or restart, giving the game a losing condition. I also made an edit to the score to allow it to be kept over in other scenes and the game over screen can display it. I didn't have any issues with this. Next I will add the main menu and add functionality to the buttons, and then I just need to make level transitions and the end screen.

Entry 14: 3/5/22

Tutorial used: "PAUSE MENU in Unity" by Brackeys - https://www.youtube.com/watch?v=JivuXdrIHK0&t=377s

Today I continued making my game project by setting up a pause menu and a main menu and allowing the game to switch between them. I had a small issue where you could pause the game even when the game over screen was up, but I fixed this by having a bool be set to true when the game over screen is up and disable the ability to pause when this bool is true.

However once I got all the buttons working I realised that if you get a game over and try to restart the level, the dungeon doesn't spawn, the player detects movement but won't move, and enemies are spawned but won't move. I am unsure of what is causing it as I tried a few different things; I tried the restart function in a scene with the dungeon alone and it worked, so I knew that it wasn't a problem with reloading the scene, it was something about getting a game over, so I edited the player controller so that the input enable bool was removed, but the error still occurred. Strangely, I managed to get it to work when I removed the game over screen setting the timescale to zero, even though the restart button was changing it back to 1. After a quick test of setting the timescale to 1 in the start function in another script, the game over worked with the timescale intact, meaning the issue was that the timescale wasn't being set to 1 with the restart for whatever reason. I also ended up having to do this with the game over bool as I couldn't pause after restarting.

Finally, I added an end screen to my game and allowed it to be triggered by entering the prefab that is spawned in the last room. It wasn't registering at first, but then I realised I was using OnTriggerEnter rather than OnTriggerEnter2D, so once I changed it, the trigger then worked. With that, I think I have everything I need for my game, though I will do a little more testing before I hand it in.

Entry 15: 8/5/22

I went back to my game project to run through it and make sure everything works before I hand it in. Last time, I noticed that my player attack wasn't working, but I found out that it was because the enemies weren't on the right layer, so I was able to fix that issue. I wanted to balance the game a bit as when you move to a different room enemies will rush at you even if you can't see them yet as the camera is still transitioning, so I edited the enemy code slightly so that they only start following the player when they are within the view of the camera, to make it fairer. I also took away the enemy spawns in the starting room, so that they don't rush at you when you start the game, however I noticed that after I did this, I was having the same issue as last time when if you reload the level after beating it or getting a game over, nothing spawns and the player is stuck. After some testing, I found that the problem was due to the code restoring the timescale upon starting was on those enemy spawners which I had just removed from the starting room, so I took that piece of code and put it on the score manager script as it was in the scene from the start and never goes away, which fixed the issue. With my game project checked over, I went back to my packages to make sure they were all formatted correctly and worked as they should, and I found no issues with these, so now everything is ready to be handed in.