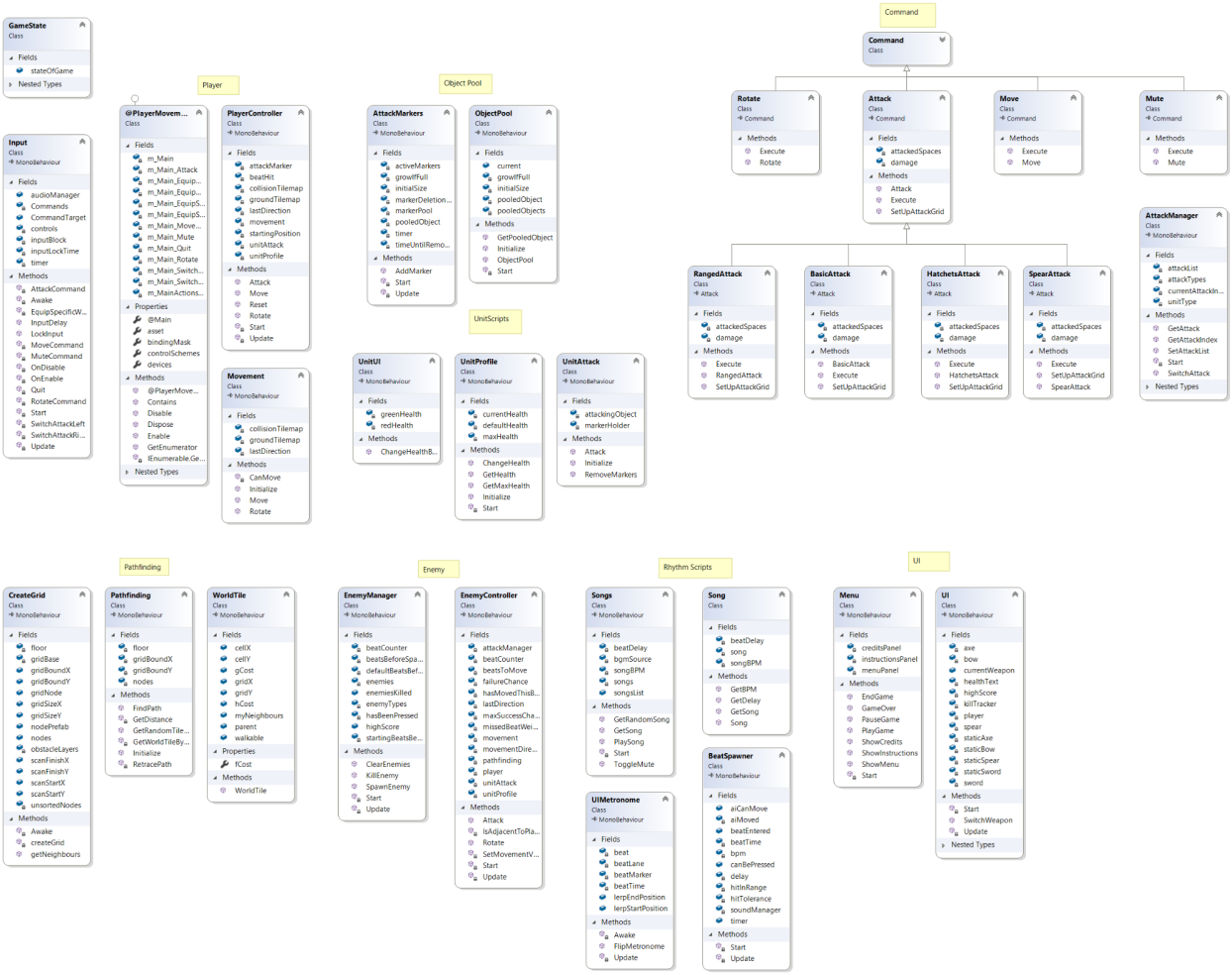
Postmortem:



Patterns and Systems:

- Command Pattern: For player movement and attacking
- A* Search Algorithm: For AI pathfinding
- Object Pool: For attack markers.

Challenges:

- Originally I used Unity's colliders and rigid body movement to control beats and when the player could hit, but this was relatively inaccurate and unclear. I changed this to be based on time so that it was easier to make changes to timing and it was more accurate.
- I also struggled to figure out a way to easily make multiple attacks with different ranges and damage. To do so, I made it so that each command holds an array of floats. When the player makes an attack, the script iterates through the array differently based on which direction the player is facing. Damage is dealt based on the floats in the array, with healing also being possible if numbers are negative.

Reusability:

- My game does use prefabs for the enemies and player, which causes reliance on Unity. Outside of that, some scripts are reusable in groups, though few are completely reusable alone.

Maintainability:

It is very easy to add new enemies and attacks. The enemy and the player use the same scripts except for their controller. When creating a new enemy prefab, you can select from all the attacks that are currently scripted, and the same for the player. You can also adjust health via the UnitProfile script. Adding new attacks via the commands are also simple, as they all use the same code, with the only thing changing being the names and the attackGrid array, which controls where the attack hits and how much damage it deals.

Completion:

I would consider this game finished, though improvements could still be made. In terms of technical debt, the BeatSpawner could be further refactored to remove some things that other classes should instead control. There are also bugs involving the enemy movement that I was unable to solve.