

GAD 176.1

Feature Specifications

Nik Walsh

Note: The diagrams are animated .gif files. I forgot that these have to be submitted as .pdf files. If you want to see them properly here is a link to the file:

https://docs.google.com/document/d/1YtsTG32as_UPllY9BzcG8cxwduFviwgWdKE1_ZVQ9B0/edit?usp=sharing

Feature 1: Player Class	1
Feature 2: Base Enemy Class	2
Feature 3: Slow Enemy	3
Feature 4: Dash Enemy	4
Feature 5: Shoot Enemy	5
Feature 6: Bullet	6
Feature 7: IEnemy	7
Feature 8: Display Health	8
Feature 9: Respawn Manager	9
Class Structure	10

Feature 1: Player Class

The player class will be the way for the user to interact with the game world. They will be represented in the game via a cube, where the rest of the enemies are triangles. The player will have the following features.

- Health system: Where the player's health can be set in the Unity Editor as a whole number.
- Movement: While the game is presented in 3D, the player will be limited to movement along a 2D axis. The player's speed can be adjusted in the editor.
- A function for adjusting the player's health is present, to be accessed by the enemy classes when they decide the player should take damage.
- A function to destroy the player GameObject when their health reaches 0.
 - A function inside of this will set the player's health to 0 if it reaches below that value. This is for a neater display in the UI.

Flowchart:

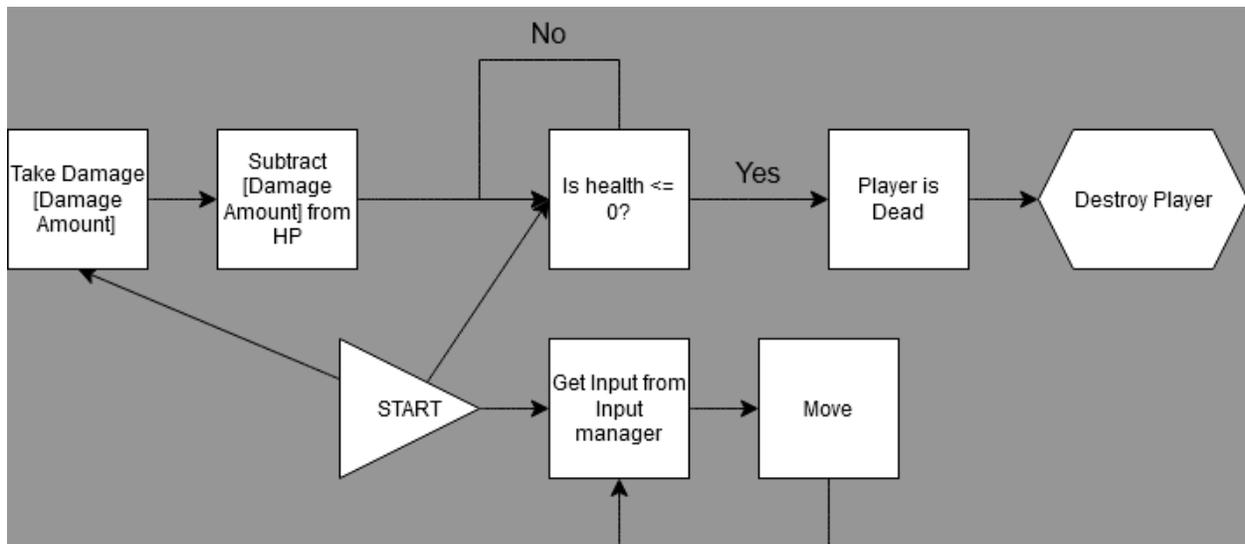
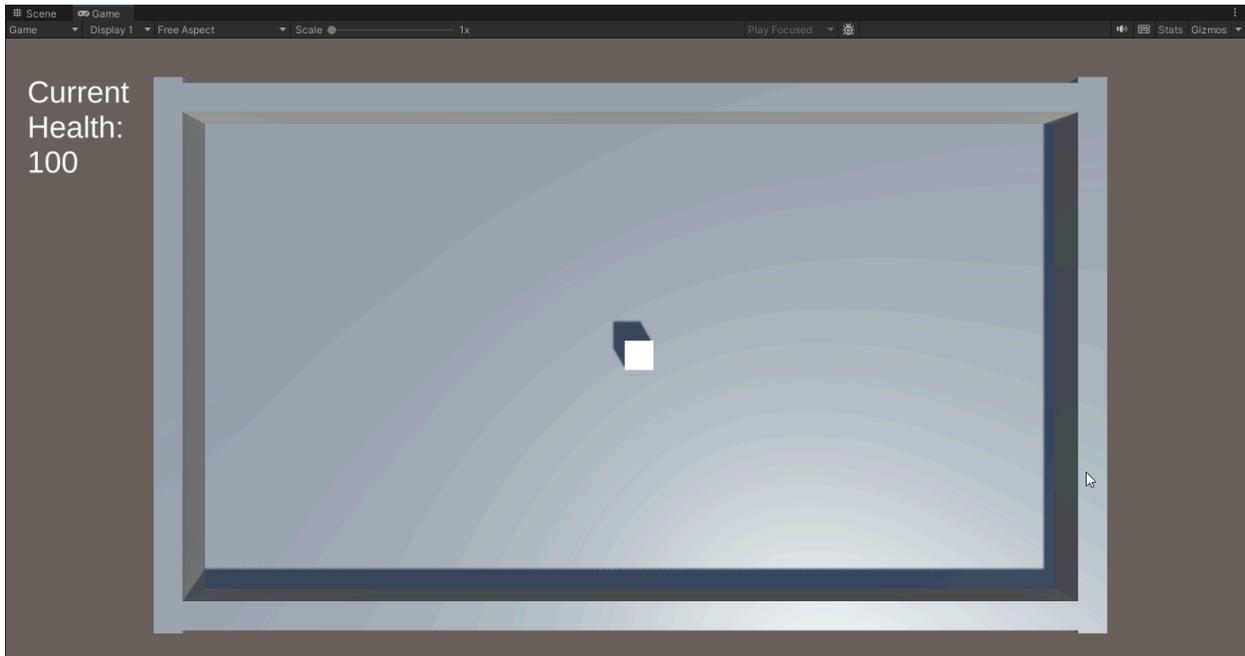


Diagram:

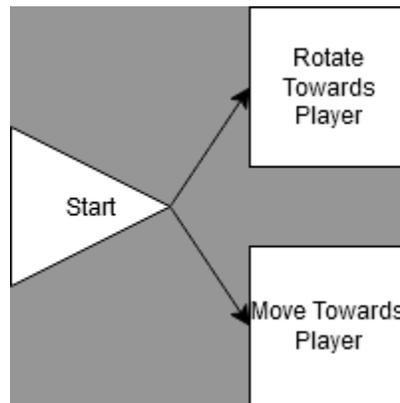


Feature 2: Base Enemy Class

The base enemy class will contain all the basic functions and variables that each child enemy class will use. This includes functions that relate to movement, rotation, the player, and the Respawn Manager.

- Individual values for movement speed, rotation speed, and respawning.
 - These are adjusted individually per enemy child class.
 - Enemy respawning can be toggled on and off manually, and the delay between respawns can be adjusted.
- A function that moves the enemy in the direction of the player.
- A function that rotates the enemy toward the player.
- A function that gets the magnitude (distance) between the enemy and the player, for use in child classes.
- A function that gets a reference to the player, in order to access the player's health setter.
- A function that gets a reference to the respawn manager, to respawn the enemy if so desired.

Flowchart:



Feature 3: Slow Enemy

Inherits from Base Enemy class (Enemy). The slow enemy will follow the player around at a constant speed, always facing towards it. If the slow enemy touches the player, it will destroy itself, and deal damage to the player.

- Inherits all functions from the base enemy class.
- Contains an integer to represent the indexed location of the enemy's prefab in the respawn array.
- Adjustable values for the speed of the enemy, and the amount of damage it deals.
- Unique functions are for checking if the enemy is touching the player, by comparing the player's coordinates as a Vector3, with the enemy's coordinates as a Vector3.
- The explode function will run if the distance between the player and the enemy is too low.
 - The explode function will delete the enemy object, access the player class and use the setter to adjust the player's health, and (if checked) will tell the respawn manager to respawn the enemy.

Flowchart:

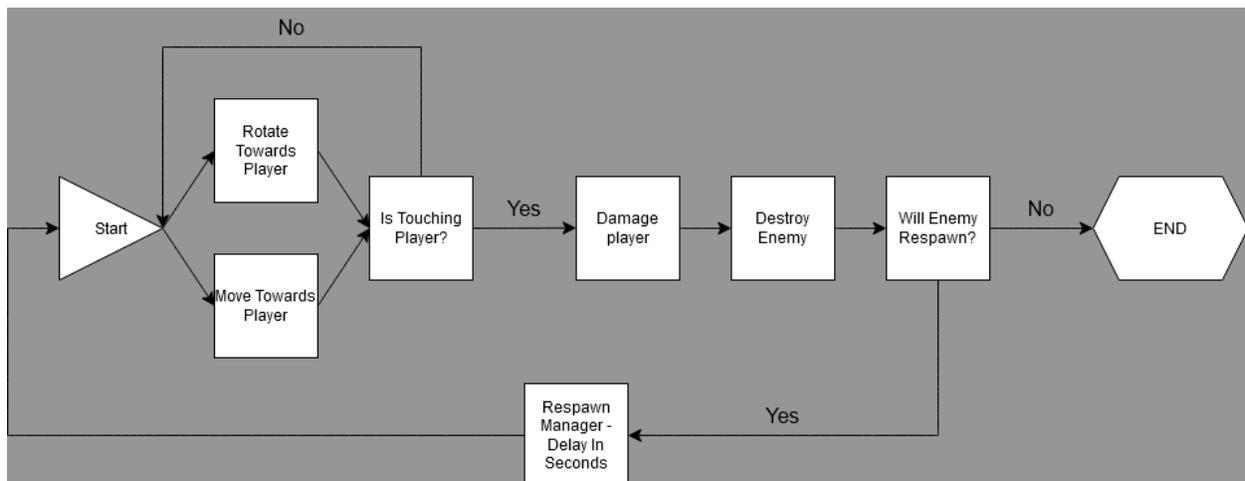
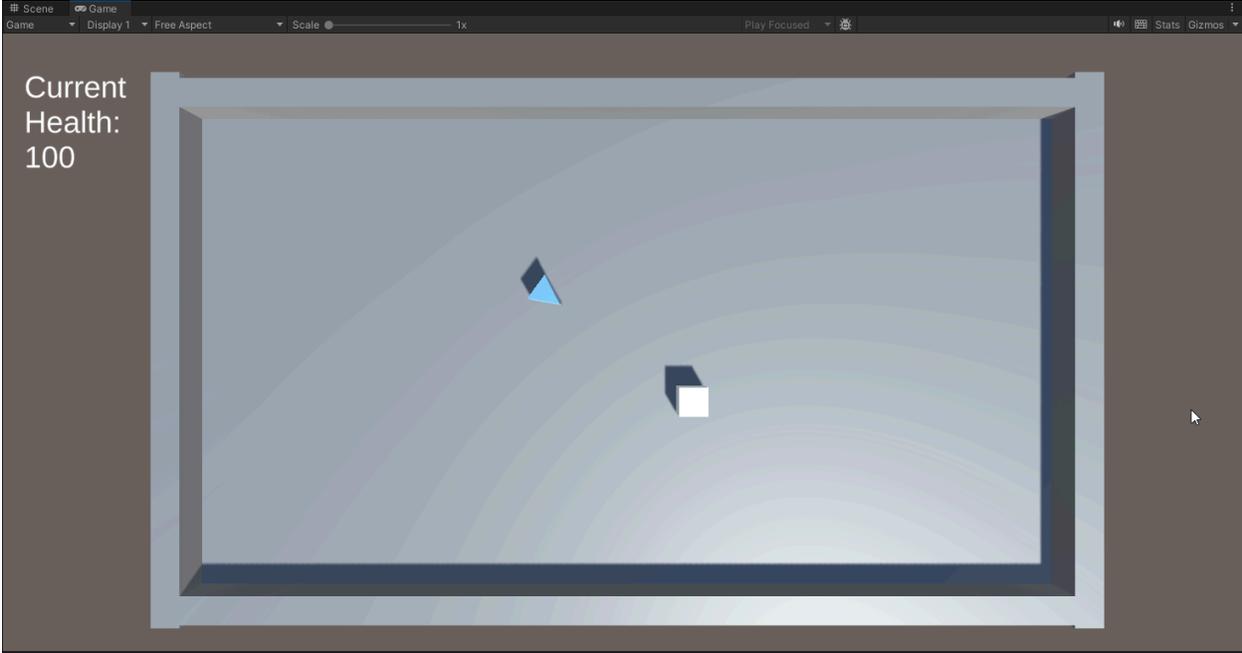


Diagram:



Feature 4: Dash Enemy

The dash enemy behaves similarly to the Slow Enemy, but instead of constantly moving toward the player, it will quickly move to the player's stored coordinates before waiting for a duration, damaging the player if it touches them.

- Inherits all functions from the base enemy class.
- Contains an integer to represent the indexed location of the enemy's prefab in the respawn array.
- Adjustable values for the speed of the enemy, and the amount of damage it deals.
- Unique function for "dashing" towards the player.
 - The enemy stays in place, but rotates toward the player.
 - After a short duration, the enemy will "lock" the coordinates of the player, storing them in a Vector3, and store its current coordinates as a separate Vector3.
 - After another short duration, the enemy will Lerp between the two coordinate locations, "dashing" toward where the player once was.
- This enemy checks if it is touching the player by using the function that checks if their colliders are touching. If they are, it runs the explode function.
- The explode function will run if the distance between the player and the enemy is too low.
 - The explode function will delete the enemy object, access the player class and use the setter to adjust the player's health, and (if checked) will tell the respawn manager to respawn the enemy.

Flowchart:

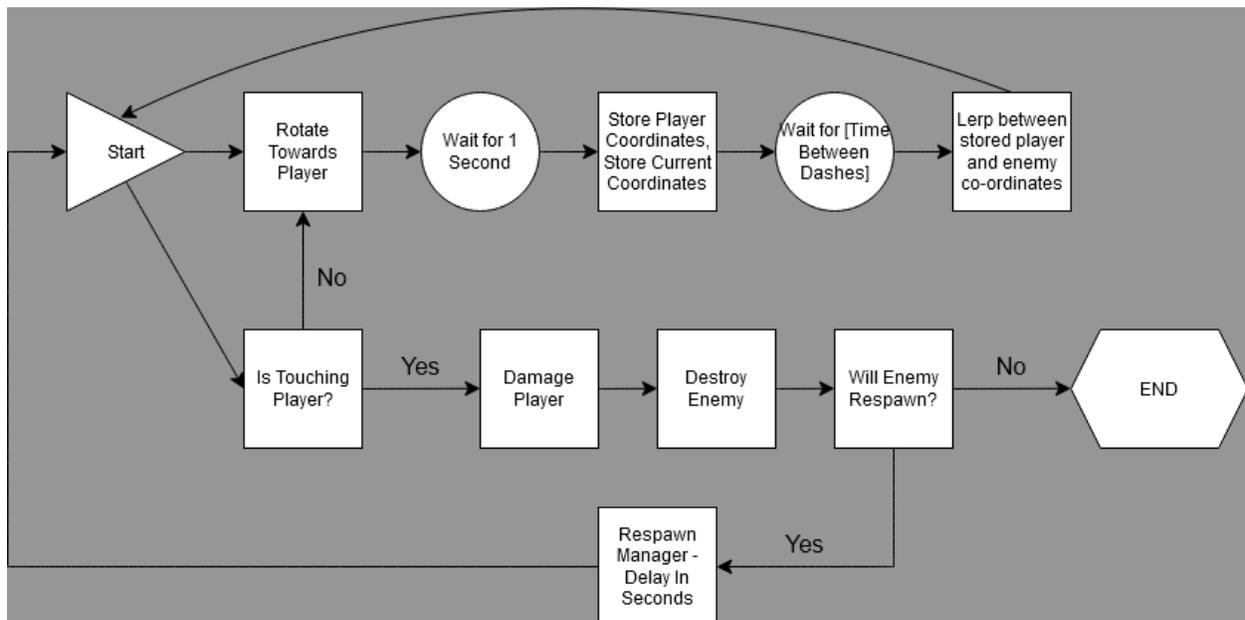
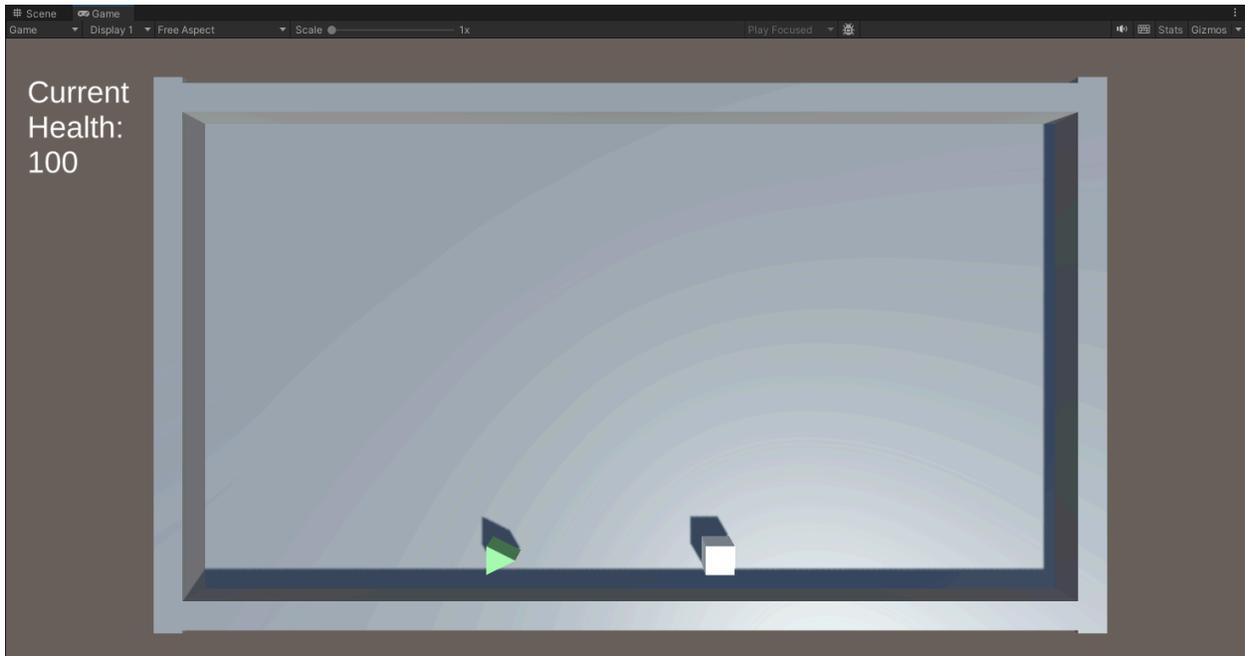


Diagram:



Feature 5: Shoot Enemy

The Shoot Enemy functions differently from the previous two. Instead of constantly moving toward the player, and damaging it when they touch, the Shoot Enemy will stay at an adjustable distance from the player, moving toward them as the player moves away, and vice versa. The Shoot Enemy will shoot bullets at the player, the speed, frequency, and damage of which are all adjustable.

- Inherits all functions from the base enemy class.
- Contains an integer to represent the indexed location of the enemy's prefab in the respawn array.
- Stores a reference to the bullet prefab for instantiation.
- Uses a method to keep a fixed distance from the player.
 - This combines the MoveTowardPlayer method inherited from the base class, and a new method called MoveAwayFromPlayer, which functions identically, just in reverse.
 - If the distance between the player reaches below the set distance + 0.5, then the enemy will move toward the player, and vice versa.
- The enemy will shoot at the player at a frequency set in the Unity Editor.
 - The bullet prefab is instantiated with the damage and travel speed set in the Shoot Enemy class.
 - The bullets will travel in the direction that the enemy is currently facing, i.e.: toward the player.

Flowchart:

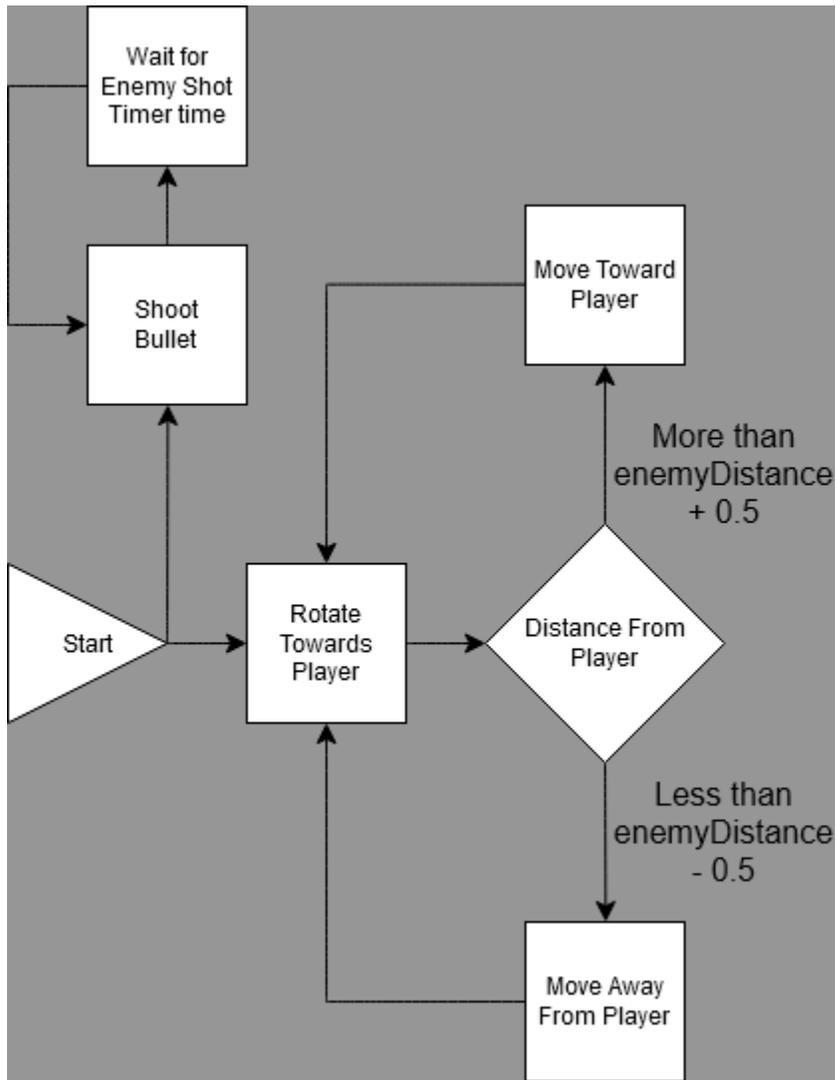
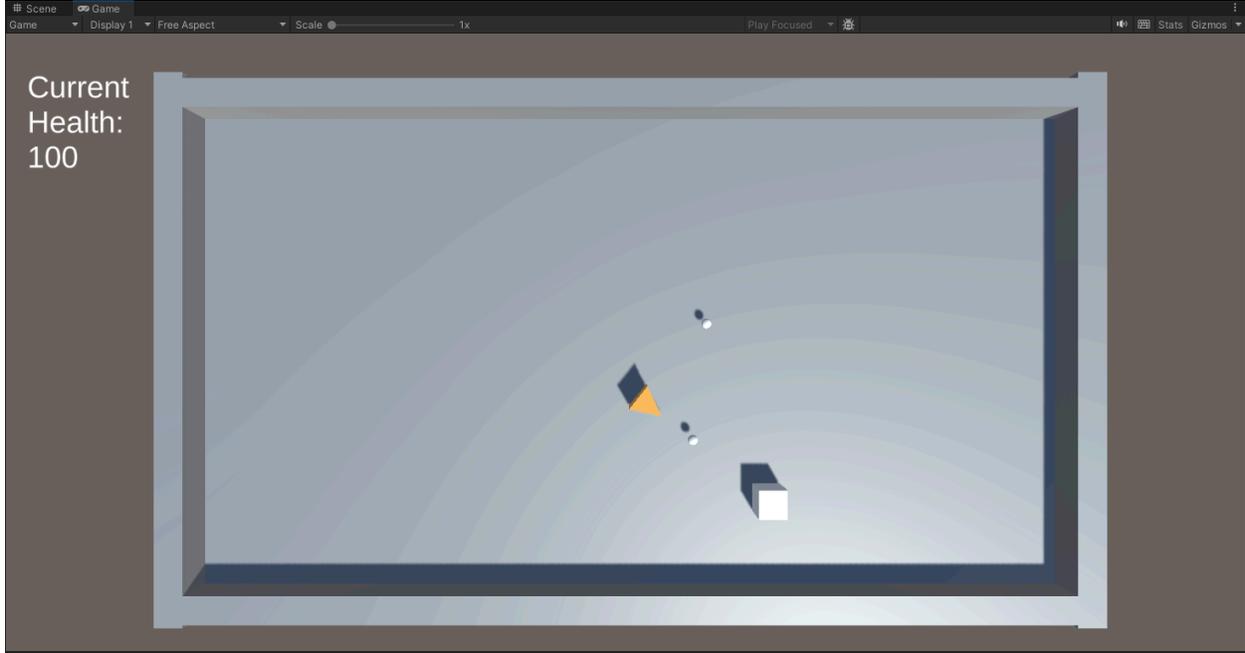


Diagram:



Feature 6: Bullet

The bullet is instantiated by the Shoot Enemy, and travels in a straight line. If it hits the player, it damages them, taking health away. If it misses, it will continue to travel along its path until it is no longer visible by the game camera, at which point it will destroy itself.

- Inherits from the ShootEnemy class.
- Instantiated at a set interval, and travels in a straight line in the direction that the enemy is facing.
- If the bullet collider touches the player collider, the bullet is destroyed, and the player takes damage via the Setter from the player class. The damage the bullet deals is set in the ShootEnemy class through the Unity Editor.
- If the bullet exits the view of the main camera, it is destroyed.

Flowchart:

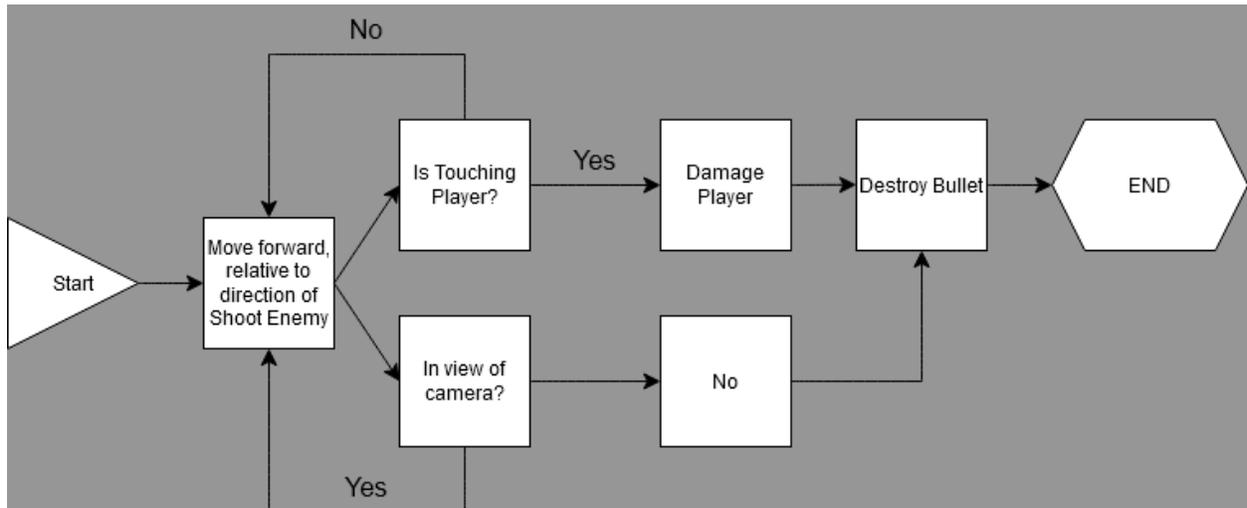
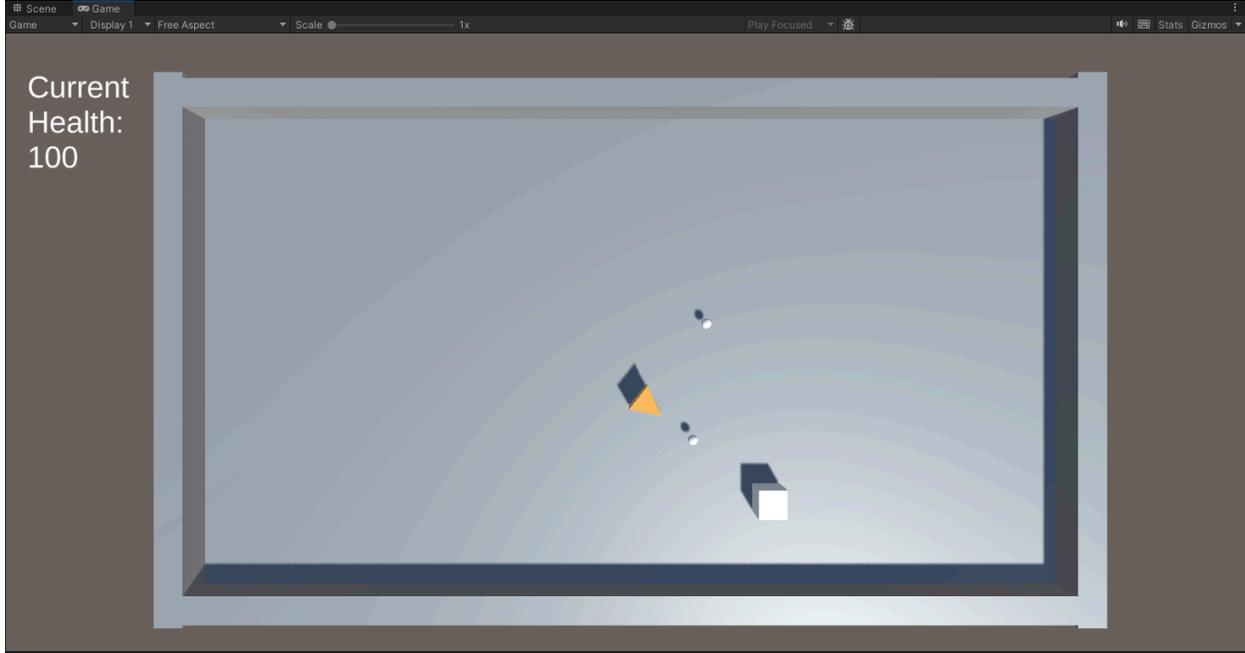


Diagram:



Feature 7: IEnemy

IEnemy is an interface used by all enemy child classes. It contains a number of functions essential for the enemy class to operate.

- Method to set the enemy type to be accessed by the respawn manager.

Feature 8: Display Health

The Display Health script displays the player's current health as an element of the User Interface.

- Gets a reference to the Text Field in the UI.
- Uses the getter in the Player Class to get the current HP value.
- Converts the value to a string to be displayed in the UI.

Flowchart:

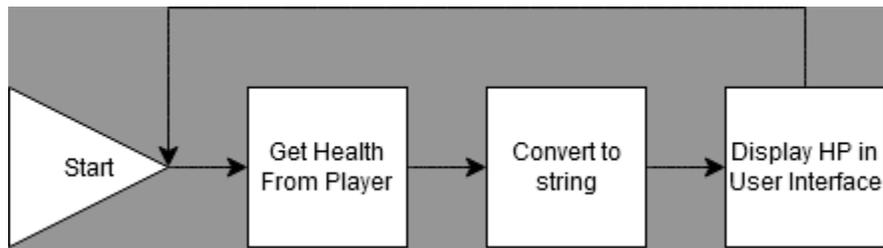


Diagram:



Feature 9: Respawn Manager

The Respawn Manager stores all of the enemy prefabs in an array, and uses the values stored by the individual player classes to respawn them. It uses a random location on the game plane as a location for instantiation. It also stores each respawned enemy in a list, so their individual functions can be adjusted after instantiation.

- Creates a new array with each of the enemy prefabs stored inside.
- Creates a new empty list.
- When accessed by an enemy subclass, the respawn manager picks a random location, starts a timer and instantiates an enemy prefab at the location when the timer ends.
 - The prefab and the timer are set by the class accessing the respawn manager.
- The instantiated prefab is loaded into a list, and has its respawn bool set to true, ensuring that each subsequent enemy is respawned.

Flowchart:

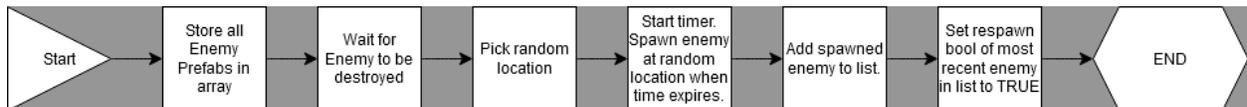
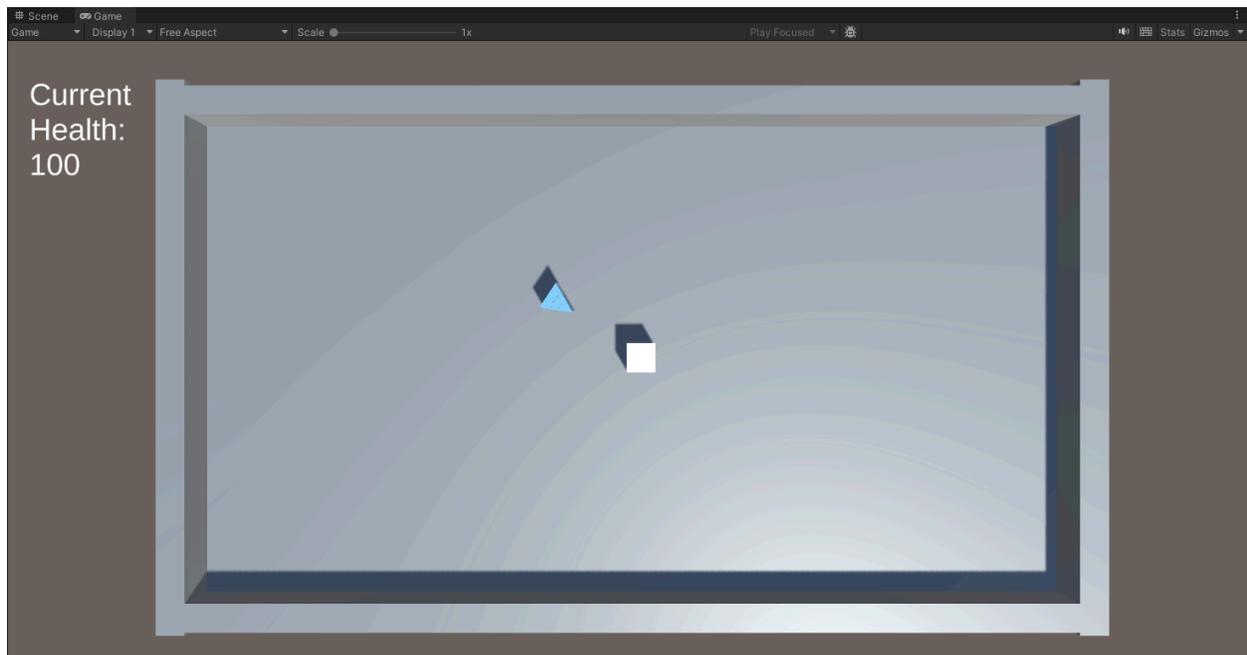


Diagram:



Class Structure

Class	Inherits from	Parent of
Player		
Enemy		SlowEnemy, DashEnemy, ShootEnemy
SlowEnemy	Enemy, IEnemy	
DashEnemy	Enemy, IEnemy	
ShootEnemy	Enemy, IEnemy	Bullet
Bullet	ShootEnemy	
IEnemy		SlowEnemy, DashEnemy, ShootEnemy
DisplayHealth		
RespawnManager		