A totally unrelated picture from https://rare-gallery.com/934961-battle-charge-knight-fighting-fantasy-art-fantasy-men.html

Endless Ascension

Sequence Diagram

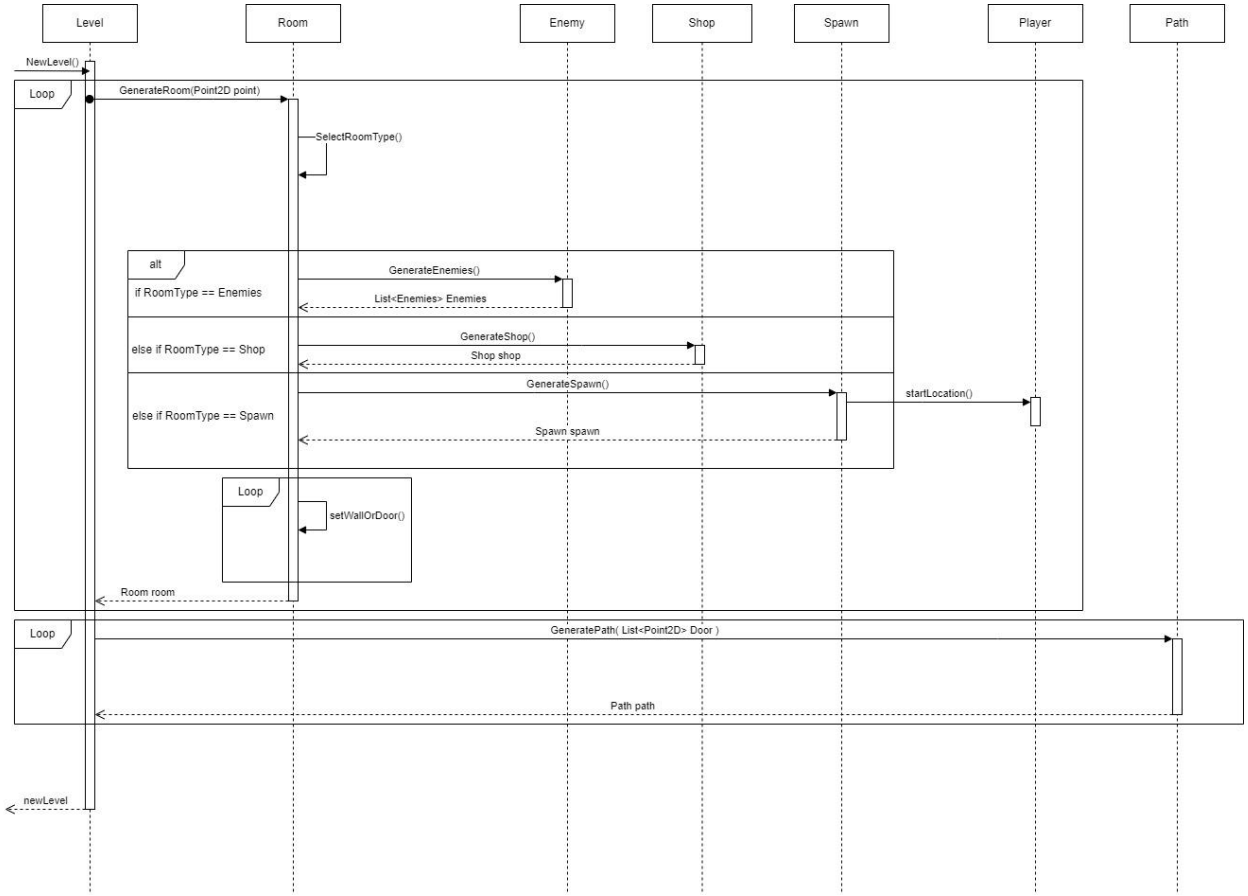
Distinction Task - A portfolio project for COS20007

12.05.2023

—

Leonardo Liew
102781996

UML Sequence Diagram



This document will discuss how a complex function works in my project. The complex function presented in this document is “Map Generation”. The discussion below is to be followed with the UML sequence diagram provided above or in the assignment upload. Note that this design is not finalized and I will change it based on what is needed.

The map generation starts with a function call - NewLevel(), invoking the start of map generation. It will move into a loop where it will generate one room at a time by calling the initialization of the room class. To make each level somewhat unique, room locations are randomized. A center location of the room is chosen. With the randomizer choosing the random width and length as well, every room generated will be of different size.

The room class will then call a function in itself and generate a room type based on the RoomType enumeration that leads to the Room inheritance where it will execute related functions. A level will definitely have two spawn room types but the amount of shop and enemy room type may vary. The reason for two spawn room types is for the spawn of the

player, and the exit of the level. This is not split into two different classes as I feel like they have almost the same design.

If the room type is “Enemies”, it will call the Enemy class to generate and initialize enemies and return a list of “Enemy” to the room class. Else if room type is “Shop”, it will call the shop class to generate and initialize a shop in the room and return a Shop object. Else if the room type is “Spawn”, it will call the spawn class to generate a spawn for the player to appear in the level. This will set the location for the player at the start of the level by calling the Player class to pass relevant fields over. If this room type is chosen a second time during the level generation, it will still call spawn class but to generate an exit for the player to go to the next level. This will then return a spawn object to the room class. The edges of the room will then be considered “wall” tile type with at least two random wall tiles being the “door” tile type.

The loop of generating rooms will go on for five to ten times to make the levels unique. The map generation function will then generate paths based on where the “door” tile type of each room is. It will call the path class where a path generation function resides. It will connect two random doors and return a path object to the level class. This will loop through a few times until all rooms are connected by going through the doors one by one.