



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

The Wrong Dungeon

Distinction Task - A portfolio project for COS20007

25.04.2023

—

Leonardo Liew

102781996

Overview

This project will mainly be a top down RPG game with idea inspiration from "Soul Knight" by ChillyRoom. This game will feature similar game mechanics and system from the inspired game but is implemented by myself without referring to the actual source code. All ideas and inspiration comes from the gameplay itself.

Goals

1. To make a simplified playable RPG game similar to Soul Knight focusing on the mechanics and not so much on the graphics.
2. This project aims to challenge how close a real game can be created within the short time limit of the semester.

Lore

In a world with portal travel, the main character travels to the grocery store to buy ingredients only to find that the destination does not look like a grocery store. The portal travel is assumed to be intercepted and now the main character is in some crack of dimensions that look like a dungeon. He will now need to look for an exit to continue his portal travel.


Portals in this world function like a line, you go in one end, come out the other. In this case, the line broke and turned into two. The main character will need to find the second one which hopefully leads him to the grocery store.

Specifications

This part will go through briefly the scope of the game and mechanics planned during the design stage.

The game will feature a map-progressing gameplay where the player will fight their way through the map, looking for an exit. The exit marks the end of the game. However, the player will have to defeat a boss before even escaping. The plan is to fight your way through the dungeon to escape it.

A player will have means to fight with a weapon that is somehow near the spawn and can get a new weapon afterwards through a "vending machine" that gives random weapons.

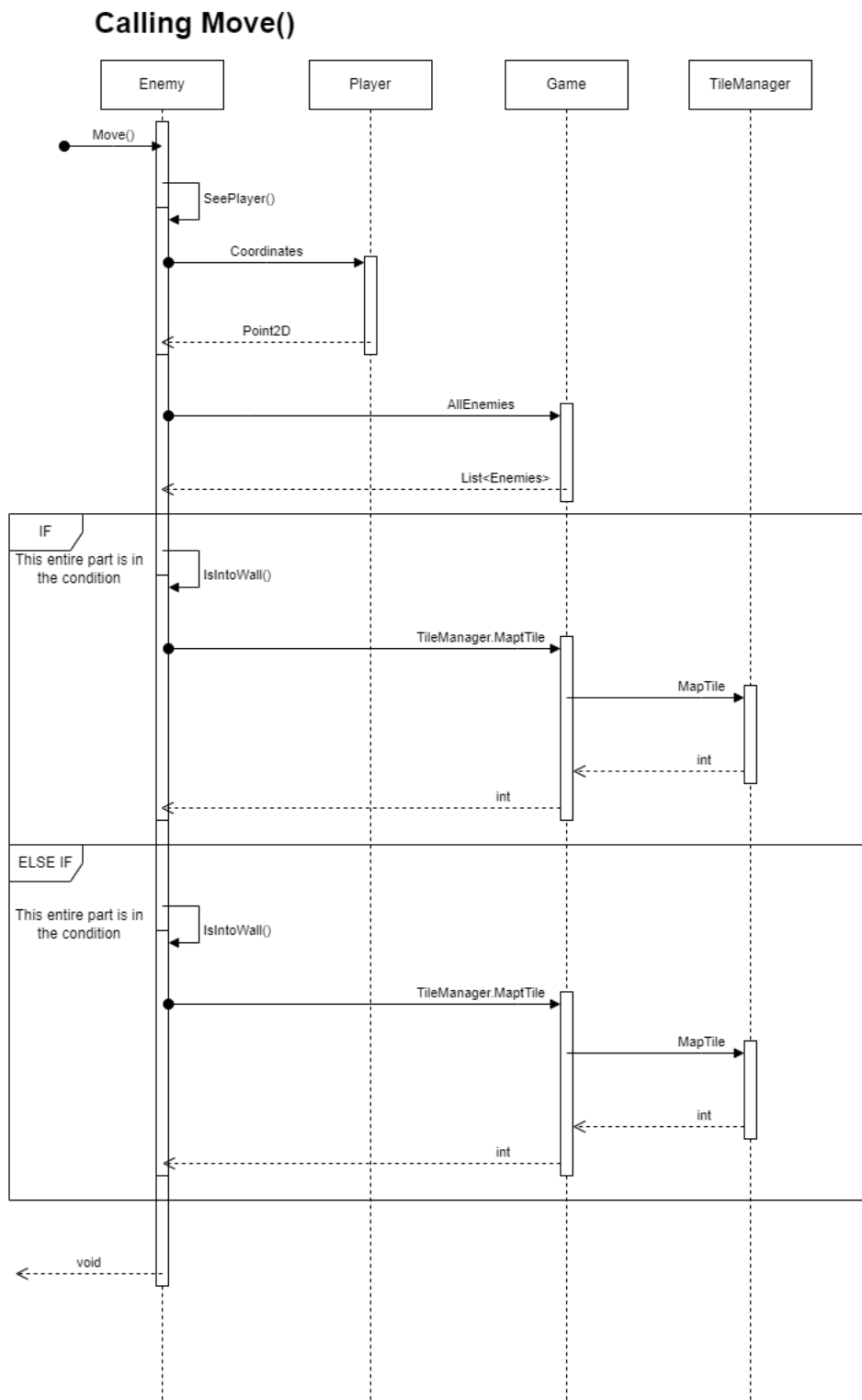



The dungeon enemies will drop coins which will be the currency system used in this game to buy weapons. Each weapon type comes with unique attack patterns.

Enemies will fill the floor and fight the player with unique attacks. Upon death, they will drop related items such as health and coins.

The main expandable part of this project is the enemy and weapon variations. Level design may also be one of them. Due to time constraints, there will only be a total of three weapons and enemy types implemented.

UML Sequence Diagram





The diagram above shows the Move() function inside the Enemy class. When Move() is called, it will first check if it sees the player by calling the SeePlayer() function.

In the SeePlayer() function, it calls the player class' property of Coordinates to obtain a player location in Point2D. If the player is in sight range, the function will change the field _seePlayer to true.

In our case, it will then get all the enemies there are in the game from the Game class property to check for collisions between them. We do not want them to stack on top of each other when chasing the player.

It will then enter the if statement where in the condition, it checks for _collision (collision between enemies) and it calls a function from the parent class of itself, IsIntoWall(). This checks if the coordinates passed is a wall. This checks by calling the property of the MapTile of TileManager object that currently exists in the game class. It should return an integer value which will then be processed into a boolean in the IsIntoWall() function. If it passes, the code block inside is executed and fields are updated accordingly.

If the statement above does not pass the conditions, it will then be directed to this where it checks if it is inside the wall with the same function discussed above. If it passes, the code block inside is executed and fields are updated accordingly.

If the statements above do not pass, the function will still return, and not update any position during the next cycle of Update() function in the Game object.

Game Screenshots



The main menu for the game. Features the title and a castle image as well as buttons for navigation. Controls are at the bottom of the screen.



Example of gameplay. It features the health and coins of the player on the top left of the screen.

The blue portal is where the player first comes out and enemies can be seen behaving hostile towards the player. Note that not all enemy and structure types are present in the screenshot.



Game over screen. Easily get this when your health reaches 0.