CS-355 Project 4 Documentation

Members: Gavin Hopper, Zach Handel, Zack West

Data Structures and Algorithms

Dr. Daniel Ray

8 November, 2022

## Description

The goal of this project is to implement personalized functionality into Dr. Daniel Ray’s Interactive Fiction Game Engine. We are tasked with adding 3 new features by the due date of November 8th, 2022. The new features will be added onto the most updated version of the IF engine (project 3).

## Timeline

October 12, 2022: This is when Gavin Hopper, Zach Handel, and Zach West decided they wanted to work together on this project.

October 17, 2022: The group agreed that Zach Handel would fix any issues that they did not finish from the previous project. They would then use the program that he fixed as a starting point for this project.

October 18, 2022: This is when the group members decided they want to add the following features: AI combat system, GUI, and a backpack storage system.

## Difficulties

## 

## Discoveries

## GUI System

* Creating many new classes

## AI Combat System

Inside the IFD file, there will be a tag called <combat> and inside the tag there will be a number (1 or 0) indicating if it is a combat area or not. 1 is yes, 0 is no. Both basic and hpsp players will have the ability to enter combat. If the player defeats the enemy, they will be able to enter the area. We can even make a new Area that is a Combat Area. When parser reads if the area is a combat area, you can dynamically allocate into the vector a combatArea instead of a regular area. NOTE: Combat Areas will NEVER be instant-death areas, but they can be goal areas.

1. Allow the parser to read if it is a combat area or not
   1. If it is, trigger the combat() function.
   2. If not, do nothing
2. When the combat() function is triggered, create the fighting stuff inside of there.
3. Create a combat rule check list when “help” is typed.

USE COMBAT ID TO TRIGGER EVENT INSIDE GAME.H THAT CAN ACCESS THE PLAYERS STATS AHHHHH

STRUGGLES:

* Trying to use inheritance, but trying to link combat area links and area links is very complicated, so I am treating it as a goal or ID event.
* Tried to access an HPSP player from the combat object, so I needed to find a way to use the player polymorphism implemented in order for it to work
  + This is where I would define some new pure virtual methods for it to work
* Created a delay function to get different values each combat turn
* Now I need to implement that the player can exit and reset during combat, so this means I need to read in a string instead of an int so it is more dynamic with user input
* Have to find a way to terminate the game reading the room being a combat room after an enemy is defeated
* Added a new flag system to detect if the player hits the enemy or not
* While polishing, I found issues such as when a false input is applied and trying to throw an error. I fixed the error but then it would still take damage from the player because hit was set to ‘false’ still.
  + I am going to try to create a ‘swing’ variable to detect if the player actually swung at the enemy.
  + THIS WORKED WOOHOO!!!!
* DID IF NOT INPUT FIX FOR STRING ISSUE!

## Backpack System

# Presentation Notes

What additions/improvements your group decided to make to the game engine

* The additions we have added into the game are all functional game features. First is our chance combat feature. Then, there is our ASCII GUI feature. Finally, there is our backpack item feature.

What design decisions motivated your changes

* Because there is an HPSP player option, we wanted to add a feature that utilizes the health system with something else other than items that can positively or negatively effect the player.
* It also allows for more interaction from the player as well as another obstacle for players to go through to access other rooms. It gives an extra dynamic of tensity and is easy to integrate by people trying to add combat to the game. It only takes a single tag to turn combat on.

How your design changes started and how they evolved over time

* It started within the IFD file. I wanted to make it easily integrated for people to be able to add combat to any room they want with only a single tag.
* Following this, it started with the parsing system. In our MapV2.h file, reading in the combat tag started in the makeArea() function. This is where a “combat ID” is set inside the info of our Area.h file. It is just another attribute created following the insta-death number and the goal room number.
* Then, I attempted to create a new combatArea.h, but decided it would be too complicated to try to link different kinds of areas. This is where the simple one-tag system came to play.
* I then had to add flags to detect if the player both swung or hit so that you cannot just spam any other number and give damage to the player.
* Anyways, back to Game.h. After the ‘if’ statement that checks to see if the game is over, the program then checks to see what the combat-ID is. If the ID is equal to 1, then a combat event will be triggered (there is a combat object in Game.h).
* This is where all the magic happens in Combat.h
* After combat has taken place, the combat ID is internally set to 0 so that the player and move in and out of the rooms after combat is over. It will not detect the 1.

Display and review the code changes you made and be prepared to answer questions about your decisions

Show your working game