RPG Game

1.0

Generated by Doxygen 1.8.13

Contents

Chapter 1

MSW_Undefined

A program CLI-ből indítható, jelenleg 2 heroval. A 2 hero adatait 2 fileban kell megadni json formátumba, name, hp, dmg, attackcooldown sorrendben. Ezután ezeket a fileokat kell beadni argumentumként. ### Pl:

```
{
"name":"Valaki",
"hp":50,
"dmg":30
"attackcooldown":3.0
}
```

```
./a.out 1.json 2.json
```

Ezek után a program lejátsza a 2 karakter közötti csatát, ahol minden karakter a hp-jából az ellenfél dmg-jének megfelelő sebzést szenved el, ha a cooldown lejárt, ameddig az egyik meg nem hal.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 Hero Class Reference

Hero class.

`#include <Hero.h>`

Public Member Functions

- Hero (std::string name_, int hp_, int damage_, double cd_)

  This is a constructor for Hero.
- std::string getName ()

  This is a simple getter for getting the Hero's name.
- int getDamage ()
- int getHp ()
- std::string getStringvar ()
- double getCooldown ()
- bool endGame (Hero *h2_)

  This method is for checking if one of the two Hero died while attacking eachother, and if one of them died, then it just puts who won as a string in the stringvar variable.
- void Attack (Hero *h2_)

  This method is for the Heroes attacking eachother. It contains the cooldown logic as well. The first two round both of the Heroes attack, but if one of them dies, it just quits from a while loop and endGame() will be called. If noone dies in the first round, then it continues to run in the else statement and there is 4 different cases. First it checks if the first Hero has lower cooldown, then it reduces the second Hero's cooldown with the first Hero's cooldown, and first Hero attacks second Hero. After that the second Hero's cooldown will remain in reduced state and the first Hero will get its original cooldown again. There is another if statement if the second Hero has lower cooldown. It does the same thing as the first one. And then it checks if both of them has the same cooldown, but it is not zero, and it will change both Hero's cooldown to Zero. And the last if statement for the case when both Hero has zero cooldown. The first Hero will start the attack, and there is an if statement for if the second Hero dies while first Hero attacked and second Hero has 0 hp, this if statement will do a continue, which will break out, and the engGame() will be called. If the second Hero doesn't die while the first Hero attacking the second Hero, then it continues to that part, when the second Hero attacks the first Hero, and at the end of the if statement, both of the Hero's cooldown will be the original cooldown again, and the while loop continues until one of them dies.

Static Public Member Functions

- static Hero parseUnit (std::string fname)

  This method is for parsing the json files. It reads in the file totally and finds: name, hp, damage, attackcooldown. And then it returns an object. There is an exception if it can't find the file.

3.1.1 Detailed Description

Hero class.

This is a Hero class. This contains the name, health, damage, and cooldown of the Hero. The Hero can attack the other Hero, but every Hero has its own attackcooldown, which makes the fight more interesting.

Author

LeviG9901

Version

1.0

Date

2020.10.13. 12:00

Created on: 2020.10.13. 12:00

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Hero()

```
Hero::Hero (
            std::string name_,
            int hp_,
            int damage_,
            double cd_ )
```

This is a constructor for Hero.

Parameters

in	*name_*	The Hero's name
in	*hp_*	The Hero's health
in	*damage_*	The Hero's damage
in	*cd_*	The Hero's attackcooldown

3.1.3 Member Function Documentation

3.1.3.1 Attack()

```
void Hero::Attack (
            Hero * h2_ )
```

This method is for the Heroes attacking eachother. It contains the cooldown logic as well. The first two round both of the Heroes attack, but if one of them dies, it just quits from a while loop and endGame() will be called. If noone dies in the first round, then it continues to run in the else statement and there is 4 different cases. First it checks if the first Hero has lower cooldown, then it reduces the second Hero's cooldown with the first Hero's cooldown, and first Hero attacks second Hero. After that the second Hero's cooldown will remain in reduced state and the first Hero will get its original cooldown again. There is another if statement if the second Hero has lower cooldown. It does the same thing as the first one. And then it checks if both of them has the same cooldown, but it is not zero, and it will change both Hero's cooldown to Zero. And the last if statement for the case when both Hero has zero cooldown. The first Hero will start the attack, and there is an if statement for if the second Hero dies while first Hero attacked and second Hero has 0 hp, this if statement will do a continue, which will break out, and the engGame() will be called. If the second Hero doesn't die while the first Hero attacking the second Hero, then it continues to that part, when the second Hero attacks the first Hero, and at the end of the if statement, both of the Hero's cooldown will be the original cooldown again, and the while loop continues until one of them dies.

Parameters

in	*h2_*	The enemy Hero as parameter

3.1.3.2 endGame()

```
bool Hero::endGame (
            Hero * h2_ )
```

This method is for checking if one of the two Hero died while attacking eachother, and if one of them died, then it just puts who won as a string in the stringvar variable.

Returns

The game is ended

Parameters

in	*h2_*	The enemy Hero as parameter

3.1.3.3 getCooldown()

```
double Hero::getCooldown ( )
```

Returns

The Hero's cooldown

3.1.3.4 getDamage()

```
int Hero::getDamage ( )
```

Returns

The Hero's damage

3.1.3.5 getHp()

```
int Hero::getHp ( )
```

Returns

The Hero's Hp

3.1.3.6 getName()

```
std::string Hero::getName ( )
```

This is a simple getter for getting the Hero's name.

Returns

The Hero's name

3.1.3.7 getStringvar()

```
std::string Hero::getStringvar ( )
```

Returns

The stringvar variable

3.1.3.8 parseUnit()

```
Hero Hero::parseUnit (
            std::string fname )  [static]
```

This method is for parsing the json files. It reads in the file totally and finds: name, hp, damage, attackcooldown. And then it returns an object. There is an exception if it can't find the file.

Exceptions

std::invalid_argument	file cannot opened

Parameters

in	*fname*	Name of the file

The documentation for this class was generated from the following files:

- Hero.h
- Hero.cpp

Index