Politechnika Śląska

Wydział Automatyki, Elektroniki i Informatyki

PROGRAMOWANIE KOMPUTERÓW 4

SFML_RPG

Autor: Jakub Lachman
Prowadzący: Dr. Anna Gorawska
Rok Akademicki: 2020/2021
Rodzaj Studiów: SSI
Semestr: 4
Termin Laboratoriów: Poniedziałek 9:00-10:30 Wtorek 8:00-9:30
Sekcja: 11

---

1. Treść Zadania

Napisać za pomocą języka C++ oraz biblioteczki graficznej SFML prostą grę turową, opartą na walce z nieprzerwanie pojawiącymi się przeciwnikami.

2. Analiza Tematu

Głównym celem projektu jest poznanie podstaw biblioteki graficznej SFML oraz przećwiczenie nowych tematów poznanych w czwartym semestrze PK.

2.1 Ogólna Idea Pisania Programu

Program został napisany przede wszystkim w taki sposób, aby był jak najbardziej „modularny”. Dzięki temu można relatywnie łatwo dodawać nowych przeciwników, umiejętności oraz wydarzenia.

2.2 Wybór Bibliotek I Klas

Jako głównym celem jest użycie biblioteki graficznej, wybór SFML był dość oczywisty.

Aby program był najbardziej jak modularny i móc dodawać nowe klasy/wydarzenia/przeciwników program został napisany głównie za pomocą obiektowości i wskaźników aby dało się dodawać się pochodne klas bazowych, reprezentujące nową zawartość.

2.3 Klasy

Tworząc program, utworzone zostały poniższe klasy:

Clickable - Opisująca obiekty „klikalne”
Button i Unitframe - Dziedziczące z Clickable, opisują one poszczególnie przycisk jak i „ramkę” na jednostkę

Entity- Opisująca szkielet działania jednostek, czy to gracza czy to przeciwnika
Player i Enemy - Dziedziczące z Entity, opisują dokładniej sposób działania danej jednostki

Event - Będąca szkieletem to działania „wydarzeń”
GenDMGUpgrade - Przykładowe wydazenie, dziedziczy z Event

EventInstance - Przetrważającą wydarzenie obecnie mające miejsce na ekran gry.
FightInstance - Przetrważającą walkę obecnie mającą miejsce na ekran gry.

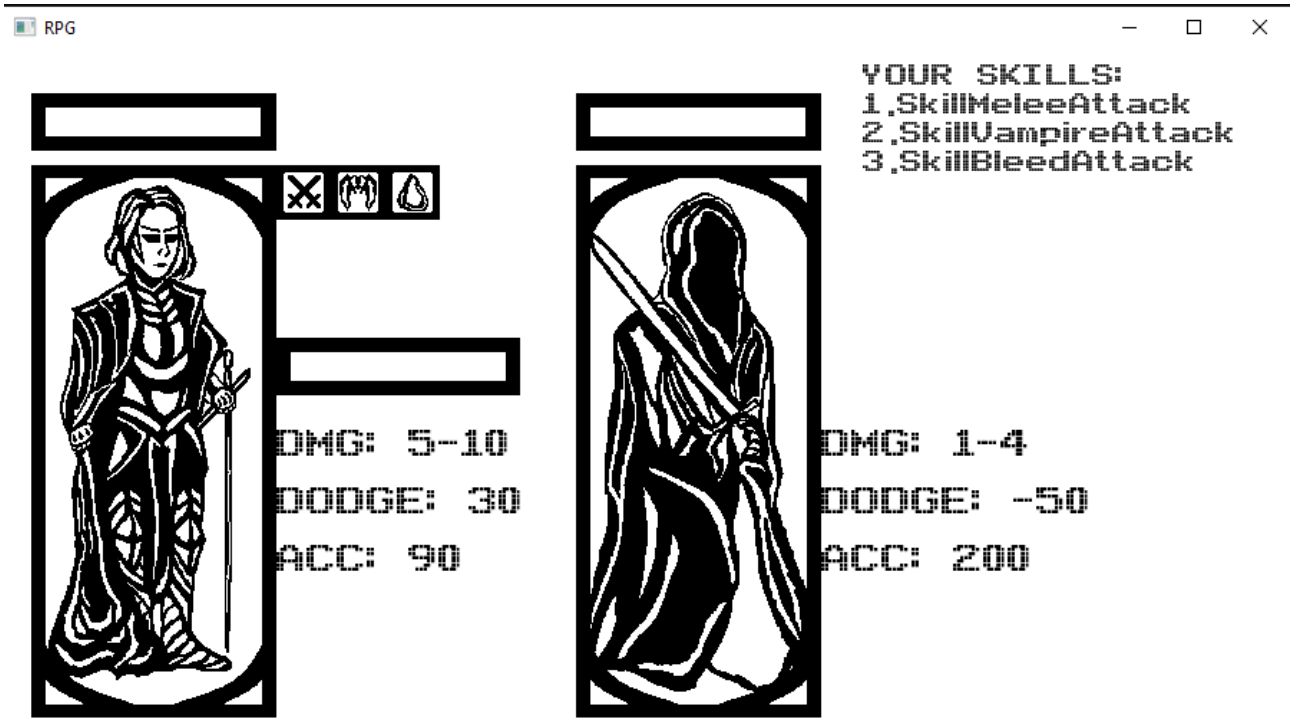
Skill - Opisująca szkielet działania umiejętności
BleedAttack, MeleeAttack i VampireAttack - Opisujące przykładowe umiejętności

HealtBar - Opisuje działanie pasku zdrowia
Random - Służy do generowania liczb losowych

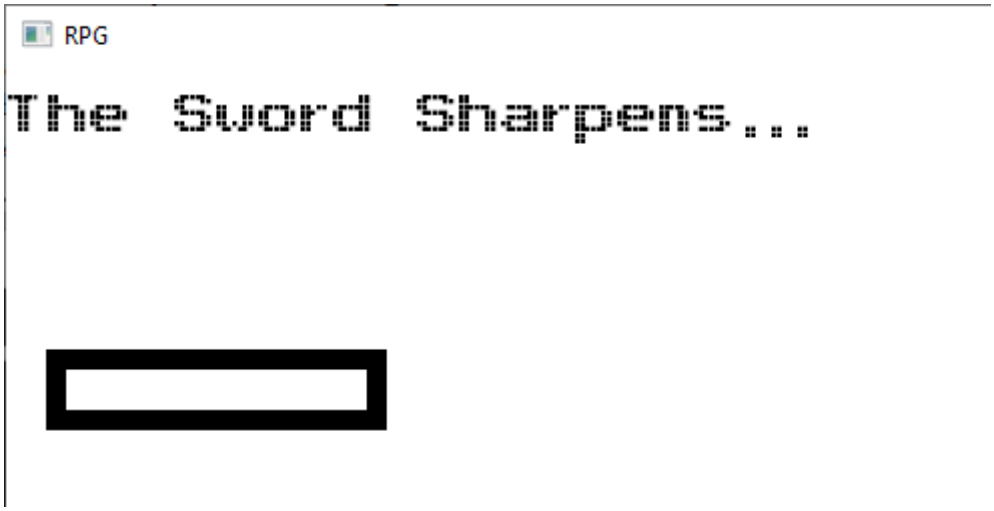
4. Specyfikacja zewnętrzna

4.1 Krótki Opis Działania Programu

Program toczy się w nieskończonej pętli, do czasu utraty przez gracza wszystkich punktów zdrowia lub zamknięcia okna programu. Gra składa się z dwóch naprzemiennie pojawiających się ekranów: wydarzeń i walki.



Ekran Walki



Ekran Wydarzenia

4.2 Krótka Instrukcja Użytkowania

Ekran walki jest stosunkowo prosty w obyciu. Wystarczy kliknąc na obraz przeciwnika (lub siebie, jeśli jest taka ochota), wybierając go, postąpić tak samo z wyborem ikonki ataku a następnie zatwierdzić akcję prostokątnym guzikiem, wykonując tym samym atak. Przeciwnik wykona następnie kontratak i wszystko zacznie się od nowa. Ekran ten kończy się zgonem jednej ze stron.

Ekran wydarzenia wystarczy jedynie zatwierdzić prostokątnym guzikiem.

Do poprawnego uruchomienia wymagane jest folder z assetami (zasobami) oraz pliki biblioteki sfml 2.5

5. Specyfikacja Wewnętrzna

Program wymaga zainstalowanej biblioteki SFML 2.5 aby móc skutecznie działać.

5.1 Podział Plików

Program został utworzony zgodnie z paradygmatem strukturalnym. Został on podzielony na pliki:

Clickable.h / Clickable.ccp
Button.h / Button.ccp
Unitframe.h / Unitframe.cpp

Entity.h / Entity.cpp
Enemy.h /Enemy.cpp
Player.h / Player.cpp
Stats.h - Plik nagłówkowy zawierający strukturę statystyk.
Status.h - Plik nagłówkowy zawierający strukturę opisów danych.

Event.h / Event.cpp
GenDMGUpgrade.h / GenDMGUpgrade.cpp

Skill.h / Skill.cpp
VampireAttack.h / VampireAttack.cpp
MeleeAttack.h / MeleeAttack.cpp
BleedAttack.h / BleedAttack.cpp
enums.h - Pliki z typami wyliczeniowymi.

Random.h / Random.cpp

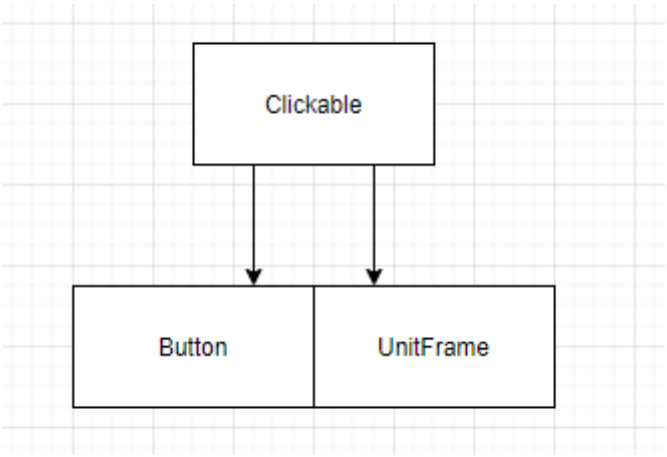
Healthbar.h / Healthbar.Cpp

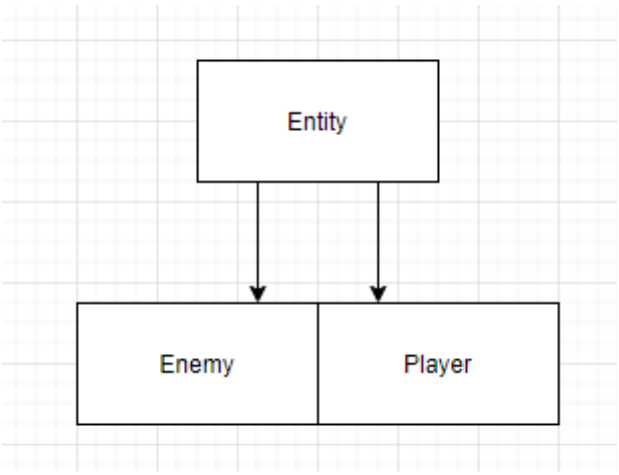
FightInstance.h / Eventintstance.cpp
EventInstance.h / EventInstance.cpp

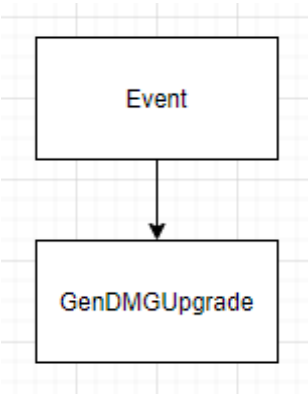
LoadPaths.h - Plik nagłówkowy zawierający zmienne stałe ze ścieżkami ładowania.

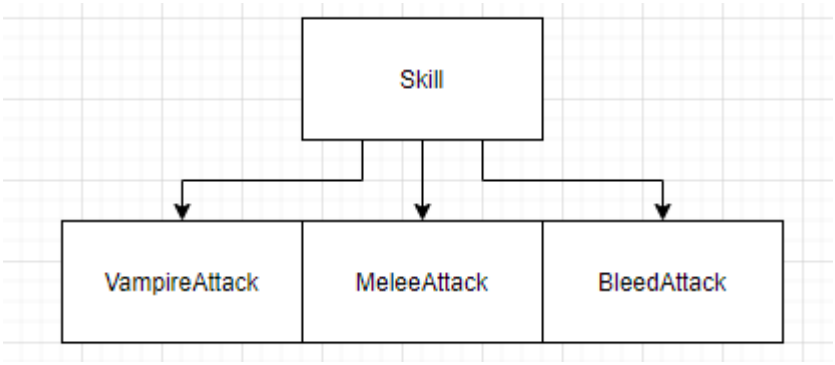
main.cpp - Plik z funkcją głowną.

5.2 Diagramy Klas









5.3 Wykorzystane Techniki Obiektowe

Najważniejszymi wykorzystanymi technikami obiektowymi jest przede wszystkim dziedziczemie oraz polimorfizm funkcji.

5.4 Ogólny Schemat Działania Programu

Po wygenerowaniu postaci, program tworzy na przemian instancję wydarzenia, która generuje i przetrwarza event, a następnie walki która generuje przeciwnika i przetwarza walkę.

5.5 Wykorzystanie Nowości Poznanie Na Zajęciach

W projekcie wykorzystano cztery nowo poznane tematy:

- Kontenery STL

```
std::vector<std::unique_ptr<Button>> buttons;
std::vector<bool> prevPressed;
std::vector<bool> nowPressed;
```

- Inteligetne Wskaźniki

```
this->unitFrame = std::unique_ptr<UnitFrame>(new UnitFrame(x, y, path + name + spriteSuffix));
this->healthBar = std::unique_ptr<HealthBar>(new HealthBar(x, y - 50));
```

- Wątki

```
std::atomic<bool> mainThreadRunning;
int secondsPassed;

void timeCounter()
{
    while (mainThreadRunning)
    {
        tt::sleep_for(ch::seconds{ 1 });
        secondsPassed++;
    }
}
```

```
countTime.join();

std::cout << "Game ended in " << secondsPassed << " seconds!";
char a = _getch();
```

- Algorytmy STL

```
for (std::vector<std::unique_ptr<Button>>::iterator it = buttons.begin(); it != buttons.end(); ++it)
{
    (*it)->Render(renderTarget);
}
```

5.6 Dokładniejszy Opis Metod Klas

Ze względu na częste występowanie pominięte zostaną metody
Update - Odpowiedzialna za aktualizowanie danego obiektu
Render - Odpowiedzialna za renderowanie danego obiektu.

Clickable/Button/Unitframe:
SetButtonState - Ustawienie stanu naciśnięcia klikalnego obiektu.
Ispressed - Zwrócenie informacji czy obiekt jest naciśnięty

Entity/Enemy/Player
AddStatus - Dodanie statusu okresowego (np. krwawienia)
ProcessStatuses - Obsłużenie obecnie istniejących statusów.

Player
CheckCooldown - Sprawdzenie czy można ponownie uzyć tej samej umiejętności
DecrementCooldowns - Dekrementacja cooldownów.
InitializeSkills - Initializacja umiejętności.
SetCooldown - ustalenie cooldownu.

Enemy
InitializeSkill - Initializacja umiejętności.

Event/GenDMGUpgrade
Option1 - Przetworzenie (nadpisywanej) wybranej opcji pierwszej.

Skill/BleedAttack/MeleeAttack/VampireAttack
Function - Przetworzenie (nadpisywanej) funkcji danego ataku.

Random
RandomInt - Zwrócenie losowej liczby całkowitej należącej do podanego zakresu
IfHitLanded - Obliczenie czy dany atak się powiódł.

EventIstance
GenerateNewEvent - Generacja nowego eventu.
ProcessEvent - Przetworzenie eventu.

FightInstance
UpdateEntities - Aktualizacja jednostek.
Fight - Przetworzenie walki.
OnPressedAttackButton - Wywołanie ataku spowodowane naciśnięciem guziku zatwierdzającego atak.
CounterAttack - Wywołanie kontrataku przeciwnika.
CheckIfFinished - Sprawdzenie czy walka się zakończyła.
InitializeText - Initializacja tekstu.
GenerateEnemy - Stworzenie nowego przeciwnika.

6. Testowanie I Uruchamianie

Program został przetestowany na każdy mozliwy sposób. Wszystkie wycieki pamięci zostały usunięte.

Głównymi problemami z którymi trzeba było się zmierzyć było upewnienie się w kwestii odpowiedniej hiearchii importowani oraz upewnienie się że nigdy nie będzie miało miejsca wykorzystanie „nieistniejących” danych. - Na przykład takich poza zakresem tablicy.

Postawiono też nacisk na uczynienie kod czytelniejszym oraz podzielenie go na poszczególne metody, czego brakowało w początkowych fazach projektu.

Aby uruchomić plik potrzebny jest plik exe.