Studenckie RPG - dokumentacja końcowa projektu

Antoni Przybylik
PROI 23L
GRUPA 103

Zoja Hordyńska
PROI 23L
GRUPA 103

30 maja 2023

Opis gry

Studencka gra RPG to wyzwanie dla miłośników przygód, którzy chcą zmierzyć się z największą zmorą inżynierów - Politechniką. Czy uda Ci się przeżyć te studia? Przejdź się korytarzami uczelni wśród gąszczu książek, pustych biurek i płaczących nad kodami studentów, na ścianach widząc istny koszmar programisty – wypisany czerwonymi literami ERROR. Niech nie zgasi Twego zapału oschłość Pani z dziekanatu, dbałość o szczegóły prowadzącego czy bezwzględność, wobec Twego zdrowia sychicznego, sesji. Legenda miejsca głosi, iż nikt jeszcze nie przeżył Politechniki bez uszczerbku na zdrowiu. Czy będziesz pierwszym, który dokona niemożliwego?

Instrukcja obsługi

Po uruchomieniu programu, na ekranie ukaże się menu startowe. Aby wybrać jedną z dostępnych opcji należy użyć strzałek (górnej lub dolnej), a następnie wcisnąć klawisz ENTER. „Play” rozpocznie grę, „Options” da możliwość dostosowania parametrów gry, „Exit” za to spowoduje zamknięcie okna menu.

Po wciśnięciu „Play” i pojawieniu się postaci na mapie, możesz swobodnie się po niej poruszać, używając, intuicyjnie działających, strzałek. Po spotkaniu na swej drodze jednego z trzech bossów, kliknij na niego, aby rozpocząć quiz. Wybierz odpowiedź za pomocą strzałek, a następnie zatwierdź swój wybór klawiszem ENTER. Wciśnij spację, aby przejść do kolejnego pytania. Po odpowiedzeniu na wszystkie pytania (bądź utracie wszystkich punktów zdrowia) zamknij okno za pomocą klawisza ESC. Nie tracąc wszystkich punktów zdrowia, odpowiedz na wszystkie pytania i przeżyj studia!

W pierwotnym planie gry gracz miał walczyć z przeciwnikami, używając do tego przedmiotów, jednak jako zwolennicy polubownego rozwiązywania konfliktów, stwierdziliśmy, że dużo ciekawsze i bardziej pasujące do klimatu studiów będzie przeprowadzenie mini quizów z, odpowiednimi dla każdego bossa, pytaniami.

Architektura

Na początku założyliśmy podział projektu na następujące cztery moduły:

- Silnik gry
- Ciało gry
- Launcher
- Aplikacja gry

Ostatecznie, nie zrobiliśmy osobnej aplikacji gry. UI i ciało gry są zintegrowane. Zostały natomiast wyodrębnione dodatkowo moduły **questions_ui** i **questions_core**. Ostatecznie podział projektu na moduły wygląda w ten sposób:

- Silnik gry
- Ciało gry
- Launcher
- Questions UI
- Questions Core

Silnik

Silnik gry dostarcza interfejs zaprojektowany w paradygmacie obiektowym. Jest on oparty o model „Sprite'owy". Użytkownik tworzy „duszki" (ang. sprite) i określa ich interakcje z otoczeniem. Następnie dodaje duszki do silnika i uruchamia silnik. Silnik symuluje zachowanie duszków podane przez użytkownika. Polimorfizm w języku C++ pozwala na nadpisywanie w klasach dziedziczących po klasie Sprite (duszek) funkcji wirtualnych i dzięki temu można w duszkach umieszczać kod który zostanie wykonany na odpowiednich zdarzeniach takich jak zderzenie dwóch duszków lub kodu, który jest wykonywany w każdym tyknięciu zegara.

Silnik został zaimplementowany tak jak założono w dokumentacji wstępnej.

Ciało Gry

Ciało gry to kod implementujący właściwą rozgrywkę. Znajdują się w nim klasy takie jak **PlayerSprite**, **TileSprite**, **BossSprite**, **LabelSprite** które odpowiadają kolejno za duszka gracza, duszki bloków, duszki przeciwników, duszki będące napisami wyświetlanymi na ekranie.

Poza nimi znajduje się tam kod odpowiedzialny za inicjalizację gry (min. wygenerowanie mapy), oraz klasa Game.

Launcher

Launcher jest programem służącym do uruchomienia gry. Z jego pomocą rozpoczyna się rozgrywkę, zapisuje stan gry do pliku, a także wczytuje stan wcześniej zapisanej gry.

Ostatecznie przybrał on formę menu gry.

Questions UI

Aplikacja „Questions UI" jest odpowiedzialna za wyświetlanie zadań znajdujących się w grze. Wyświetla się tam treść pytań wraz z dostępnymi odpowiedziami. Zła odpowiedź odbiera życie graczowi, a odpowiedzenie na wszystkie pytania przeciwnika sprawia, że znika z mapy.

Ten moduł jest odpowiedzialny tylko za wyświetlanie pytań. Klasy przechowujące pytania znajdują się w Questions Core.

Questions Core

Klasy odpowiedzialne za pytania. Jest możliwość ich serializacji, zapisu i wczytania z pliku. W tym module znajdują się między innymi następujące klasy:

Klasa Player przechowuje liczbę punktów zdrowia gracza.

Klasa Enemy jest klasą, opisującą wrogów. Każdy wróg posiada wektor indywidualnych pytań oraz punktów zdrowia, które zabierają graczowi za niepoprawną odpowiedź. Metody klasy pozwalają na zarządzanie zestawem pytań, wyświetlaniem ich treści oraz odpowiedzi, a także modyfikację parametrów gracza.

Klasa Question przechowuje informacje dotyczące pytań - ich treść oraz dostępne odpowiedzi.

Wymagania

Gra wymaga zainstalowanej biblioteki SFML i jsoncpp. Działa na Linuksie.

Biblioteki i Narzędzia

Projekt w całości został zrealizowany w języku C++, w standardzie C++20. Nie używaliśmy żadnych rozszerzeń kompilatora, ale skorzystaliśmy z jednej niestandardowej biblioteki **unistd.h** ze standardu POSIX, który jest implementowany przez wszystkie powszechnie używane systemy operacyjne (Linux, MacOS, Windows).

Skrypty pozwalające na zbudowanie projektu mają są kompatybilne z programem make i do zbudowania projektu ma jest używany toolchain GCC. Wszystkie moduły można zbudować uruchamiając Makefile w katalogu game/. Ten Makefile rekursywnie wywoła pliki Makefile wszystkich modułów.

Projekt został zrealizowany przy użyciu biblioteki SFML, biblioteki standardowej języka C++ (libstdc++), biblioteki standardowej języka C (libc) i biblioteki matematycznej (libm). Do tego skorzystaliśmy jeszcze z biblioteki jsoncpp do serializacji obiektów.

Podział Pracy

Antoni

- Silnik
- Ciało Gry
- Wybrane części wszystkich innych modułów

Zoja

- Launcher (Menu)
- Aplikacja do wyświetlania pytań
- Klasy przechowywujące stan gry
- Tekstury i obrazki