EduMaze - opis projektu

Wiktor Ogrodnik

June 23, 2021

1 Opis programu

Eud Maze to eukacyjna gra typu labirynt, w której gracz wybiera zestaw pytań, a następnie musi pokonać labirynt odpowiadając na pytania. Ilość błędyncyh odpowiedzi jest mocno ograniczona :).

1.1 Technologie

Program został napisany w języku C# 9.0. Jako środowiska uruchomieniowego korzytsałem z .NET 5.0.

Użyte biblioteki:

- SFML.NET `https://www.nuget.org/packages/SFML.Net/`
- Newsoft Json.NET `https://www.nuget.org/packages/Newtonsoft.Json/`

2 Kompilacja ze źródła i uruchamianie

2.1 Windows

- Pobierz .NET SDK z oficjalnej stony Microsoft: `https://dotnet.microsoft.com/download`
- Korzystając z powłoki PowerShell przejdź do folderu z kodem i wykonaj polecenie: 'dotnet run'
- Jeżeli chcesz skompilować program, by później używać go jako pliku wykonywalnego możesz skorzytać z polecenia: 'dotnet build'

2.2 Linux

- Pobierz najnowsze .NET SDK. W tym celu skorzystaj z instukcji zawartych na tej stronie: `https://docs.microsoft.com/pl-pl/dotnet/core/install/linux` dla twojej dystrybucji. Możesz też pobrać wersję dostępną w repozytorium twojej dystrybucji.
- Korzystając z terminala wejdź do folderu z kodem i wykonaj polecenie: 'dotnet run'
- Jeżeli chcesz skompilować program, by później używać go jako pliku wykonywalnego możesz skorzytać z polecenia: 'dotnet build'

3 Szczegółowy opis gry

Gracz (po wyborze zestawu pytań) pojawia się w punkcie (0, 0) w proceduralnie wygenerowanym labiryncie. Może poruszać się za pomocą strzałek lub popularnego WSADu. Musi przedostać się na przeciwległy kraniec labiryntu. Cały labirynt usiany jest znakami zapytania, po wejściu na niego pojawia się monit z pytaniem i maksymalnie czterema odpowiedziami. Po poprawnej opowiedzi znak zapytania oraz monit znikają i można kontynuować grę. Po złej odpowiedzi gracz traci jedno życie, ale pytanie zostaje uznane za "zaliczone".

Zestaw pytań może zostać załadowany jako argument konsoli (nazwa pliku). Jeżeli nie podamy nazwy pliku z pytaniami program spróbuje załadować plik 'examples/set1.json' Format pliku to JSON, którego układ wygląda nastepująco [pytanie1, pytanie2, pytanie3, itd.]. Natomiast pytanie składa się z [Pytanie, [odp0, odp1, odp2, odp3], numer poprawnej odpowiedzi].

Pytań w pliku może być dowolna ilość*, a do każdego pytania musi być od 2 do 4 odpowiedzi. Przykładowe zestawy pytań znajdują się w folderze examples.

*Co najmniej 2.

4 Opis kodu

Główną częścią kodu programu jest klasa Game. Tutaj znajdują się wszystkie ważne obiekty gry takie jak okno programu, okno pytań, labirynt, gracz, itd.

Znajdują się tu również dwie ważne metody: Update i Render. Wykonują się w każdej klatce i aktualizaują logikę programu, a następnie renderuja wszystkie obiekty. Większość aktualizacji logiki wykonuję się jednak asynchronicznie za pomocą C-sharpowych delegatów i są obsługiwane przez "Handlery" w Game.cs.

Wyświetlanie z metody Render jest natomiast zaprojektowane na wzór warstw z programów graficznych takich jak GIMP. Render po prostu po kolei rysuje warstwy (obiekty typu "Layer"). Każda warstwa przechowuję listę obiektów implementujących interfejst IDrawable.

Wartwy mają jeszcze jedno zastosowanie, wyszukują obiekt, na który najeżdża myszka. Taki obiekt musi implementować interfejs ISelectable.

W kalsie Game znajdują się dodatkowo metody obsługujące wbudowane sygnały w SFMLa, takie jak kliknięcie myszą, wprowadzone z klawaitury znaki, polecenie resize czy zamknięcie programu.

Równie ważnym elementem programu jest klasa Maze, która zarządza planszą gry. Na początku labirynt jest generowany automatycznie z wykorzystaniem algorytmu DFS. Umieszane są również na nim znaki zapytania. Później klasa oprócz tego, że implementuje intefejs IDrawable wysyła do klasy Game sygnały QuestionEntered i FinishEntered, które odpowiednio informują o wejściu na pytanie i wejściu na metę.

Klasa Question to okienko z pytaniem. Posiada pola tekstowe i cztery pyszciski, które implementują interejsy IDrawable i ISelectable. Okienko tak jak przyski jest tylko jedno na cały program, a potem zmieniana jest tylko jego zawartość. Klasa ta ma również wbudowany algorytm zawijania tekstu. W krytycznych momentach może również zmieniać czcionkę, żeby zmieścić się w okienku. Klasa wysyła sygnał do klasy Game o tym, że odpowiedzano na pytanie (wraz z rezultatem) oraz obsługuje sygnał od przycisku, który wysyła sygnał o kliknięciu.

Zestaw pytań znajduje się natomiast w klasie QuestionSet, klasa ta na początku programu parsuje Jsona z pytaniami i przechowuje pytania w specjalnej klasie QuestionPrototype. Klasa zajmuje się też losowaniem kolejnych pytań oraz podaje je klasie Game, która z kolei aktualizuje zawartość klasy Question.

Istnieje jeszcze parę mniejszych klas takie jak EndWindow, jest to okienko z informacją o końcu gry (z porażką lub nie), jest też klasa Player, która wyświetla gracza oraz jego ilość żyć. Gdy zostanie 0 żyć to klasa wyśle sygnał do klasy Game. Zostaje jeszcze klasa finish, która jest po prostu kwadratem z teksturą.

5 Dalszy rozwój

Po zakończeniu projektu planuję dalsze prace nad programem. Oprócz rozwoju samego kodu, zaplanowane są również następujące funkcjonalności:

- Menu główne z wyborem pliku z pytaniami oraz rozmiarem labiryntu,
- Menu ustawień i "Co nowego",
- Moduł automatycznych aktualizacji.

6 Licencjonowanie

Program wykresy będzie darmowym oprogramowaniem open-source na licencji MIT. Kod jest dostępny w publicznym repozytorium na GitHubie.