Specyfikacja wymagań aplikacji „Arcade Jump”

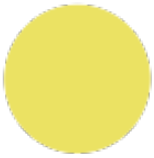
Autor: Krzysztof Bądkowski

Data utworzenia dokumentu: 12 listopada 2016
Data ostatniej modyfikacji: 2 grudnia 2016

Spis treści

1 Streszczenie

Głównym celem projektu jest nauka programowanie z wykorzystaniem zewnętrznych bibliotek stworzonych dla wybranego języka programowania (C/C++). Aplikacją będzie gra platformowa (zręcznościowa) o nazwie „Arcade Jump". Główna idea gry będzie polegała na tym aby używać spacji w odpowiednim momencie żeby skoczyć naszym awatarem nad przeszkodami. Planowane są również takie miejsca, gdzie musielibyśmy używać strzałek góra-dół aby unikać przeszkód (np. unikanie ostrzału z wieżyczek).



Rysunek 1: Animacja prototypu gry (kliknięcie uruchamia animację)

2 Ogólny opis

2.1 Relacje do bieżących projektów

Nie dotyczy.

2.2 Relacje do wcześniejszych i następnych projektów

Jest to pierwszy, bardziej złożony projekt. Wcześniej autor pracował z mniejszymi projektami, które zakończyły się sukcesem. Zdobyte doświadczenie podczas realizacji tego projektu będzie pomocne przy wykonywaniu następnych projektów.

2.3 Funkcje i cele

Celem aplikacji jest dostarczanie rozrywki użytkownikom.

2.4 Ustalenia dotyczące środowiska

Aplikacja zostanie napisana w języku C/C++ w środowisku systemu Microsoft Windows 10. Jako IDE służyć będzie Microsoft Visual Studio 2015.

2.5 Relacje do innych systemów

Gra będzie wykorzystywała zewnętrzną bibliotekę Allegro 5.

2.6 Ogólne ograniczenia

Gotowa aplikacja będzie działać wyłącznie w środowisku systemowym Microsoft Windows.

2.7 Opis modelu

Oprogramowanie będzie tworzone zgodnie z modelem ewolucyjnym

3 Specyficzne wymagania

3.1 Wymagania dotyczące funkcji systemu

Po naciśnięciu klawisza spacji ma nastąpić skok naszego awatara. Jeśli nastąpi kolizja z jakąś przeszkodą gra się kończy. Wtedy możemy rozpocząć poziom od nowa lub wyjść do menu głównego.

3.2 Wymagania dotyczące wydajności systemu

Aplikacja ma działać płynnie i niezawodnie tzn. podczas rozgrywki nie powinny zdarzać się przypadkowe wyjścia z aplikacji.

3.3 Wymagania dotyczące zewnętrznych interfejsów

Wymagana klawiatura.

3.4 Wymagania dotyczące wykonywanych operacji

Nie dotyczy.

3.5 Wymagania dotyczące wymaganych zasobów

Nie dotyczy.

3.6 Wymagania dotyczące sposobów weryfikacji

Weryfikację aplikacji będzie przeprowadzał autor.

3.7 Wymagania dotyczące sposobów testowania

Do testów zostanie użyta biblioteka cppunit oraz samo testowanie działania aplikacji przez autora.

3.8 Wymagania dotyczące dokumentacji

Nie dotyczy.

3.9 Wymagania dotyczące ochrony

Nie dotyczy.

3.10 Wymagania dotyczące przenośności

Nie dotyczy.

3.11 Wymagania dotyczące jakości

Nie dotyczy.

3.12 Wymagania dotyczące niezawodności

Nie powinny zdarzać się kolizje z niepożądanymi obiektami jak również przypadkowe przerwania lub zakłócenia w działaniu aplikacji.

3.13 Wymagania dotyczące pielęgnacyjności

Nie dotyczy.

3.14 Wymagania dotyczące bezpieczeństwa

Nie dotyczy.

4 Dodatki

4.1 Harmonogram

Wstępny zarys harmonogramu:
21.11.2016 – Stworzenie menu
28.11.2016 – Stworzenie prostej, przesuwającej się mapy
05.12.2016 – Powiększenie mapy
12.12.2016 – Implementacja skoku
19.12.2016 – Poprawki do istniejących implementacji
09.01.2016 – Detekcja kolizji
16.01.2016 – Testy
23.01.2016 – Porządkowanie kodu

4.2 Historia zmian

~~28.11.2016 – Stworzenie obiektów kolizyjnych~~ → Stworzenie prostej, przesuwającej się mapy
~~05.12.2016 – Ruch obiektów kolizyjnych~~ → Powiększenie mapy