

Grafika Komputerowa - Dokumentacja Projektu

Łukasz Jezapkowicz

30 grudnia 2020

1 Projekt - informacje ogólne

1.1 Co przedstawia projekt?

Stworzony przeze mnie projekt jest implementacją prostej gry stworzonej przy pomocy *physi.js*. Gra polega na strzelaniu z armaty do obiektów (tutaj pudełek) i strąceniu wszystkich do wody przy użyciu jak najmniejszej ilości strzałów (trudno o bardziej klasyczną grę). Gra ma tylko kilka poziomów, ale dodanie nowych jest banalnie łatwe. Poniżej widoczne przykładowe zdjęcie z gry:



1.2 Skąd pomysł?

- Pomysł na grę polegającą na strzelaniu do celu wpadł mi do głowy podczas gry w Moorhuhn - wszystkim dobrze znaną grę o strzelaniu do kurczaków.
- Inspiracja morzem i armatą wpadła mi do głowy podczas oglądania Piratów z Karaibów w te święta.
- Wzory na armacie od zawsze kojarzyły mi się z amerykańskimi komediami.
- Muzyka powinna być relaksująca.

1.3 Inspiracje:

- Kod odpowiedzialny za "poruszanie się" morza zapożyczyłem z następującej [strony](#).
- Mechanika strzału inspirowana jest wieloma projektami znalezionymi w Internecie związanymi z m.in. strzelaniem do bramki (piłka nożna).
- Wszelakie fora takie jak [Stack Overflow](#).
- Poprzednie laboratoria.

1.4 Użyte biblioteki:

- *ammo.js*
- *physi.js*
- *physijs_worker.js*
- *three.js*

1.5 Jakie podobne projekty można zrobić na podstawie tego projektu?

- Okręty wojenne (piraci).
- Strzelanie do bramki.
- Siatkówka.
- Koszykówka.
- Bilard.
- I wiele innych gier tego rodzaju.

1.6 Co można dodać/poprawić?

- Dodać ciekawsze tekstury.
- Dodać bardziej skomplikowane obiekty do zbijania.
- Dodać bardziej skomplikowane tworzenie poziomów.
- Rozbudować grę, tworząc z niej np. bitwy pirackich statków.

1.7 Jak zagrać?

UWAGA: Podczas gry leci muzyka.

Gra jest dostępna pod następującym [linkiem](#).

2 Projekt - opis kodu

Poniższy opis streszcza co robią poszczególne kawałki kodu/funkcje napisane w Javascript. Bardziej szczegółowy opis kodu znajduje się w plikach źródłowych.

2.1 cannon.js

Pomijam opis stałych oraz zmiennych globalnych - opisuje tylko funkcje.

2.1.1 Funkcja 'interpolate'

Funkcja interpolująca używana przy obracaniu armatą/odrzućcie armaty.

2.1.2 Funkcja 'secondsSince'

Funkcja pomocnicza zwracająca ilość sekund, która upłynęła od danego czasu.

2.1.3 Funkcja 'removeObjects'

Funkcja usuwająca ze sceny wszystkie obiekty spełniające dany warunek.

2.1.4 Funkcja 'AudioPool'

Funkcja tworząca kilka kopii danego pliku audio.

2.1.5 Funkcja 'addOceanPlane'

Funkcja tworząca poruszające się morze.

2.1.6 Funkcja 'addCannon'

Funkcja dodająca do sceny armatę (jako złożenie kuli oraz walca).

2.1.7 Funkcja 'createTarget'

Funkcja tworząca pojedyncze pudełko.

2.1.8 Funkcja 'addTargets'

Funkcja dodająca do sceny pudełka (zależnie od poziomu). Pudełka są początkowo "zamrożone".

2.1.9 Funkcja 'unfreezeTargets'

Funkcja "odmrażająca" pudełka (dzięki temu możemy powalić konstrukcję z pudełek).

2.1.10 Funkcja 'createBall'

Funkcja dodająca do sceny piłkę, którą strzelamy.

2.1.11 Funkcja 'shootBall'

Funkcja strzelająca stworzoną piłką.

2.1.12 Funkcja 'addSplash'

Funkcja dodająca do sceny jedno "chłapnięcie wodą".

2.1.13 Funkcja 'loadResources'

Funkcja inicjalizująca, wczytująca większość tekstur.

2.1.14 Funkcja 'initScene'

Funkcja inicjalizująca, wczytująca scenę, światło, kamerę i muzykę.

2.1.15 Funkcja 'initEvents'

Funkcja inicjalizująca, wczytująca wszystkie zdarzenia (event listenery).

2.1.16 Funkcja 'init'

Funkcja inicjalizująca całą grę.

2.1.17 Funkcja 'startNextLevel'

Funkcja usuwająca poprzedni poziom oraz wczytująca nowy.

2.1.18 Funkcja 'levelWon'

Funkcja uruchamiana w przypadku przejścia poziomu (zapisuje wyniki).

2.1.19 Funkcja 'gameWon'

Funkcja uruchamiana w przypadku przejścia gry (podsumowuje wyniki).

Funkcja dodatkowo włącza spadające z nieba pudełka (żeby móc dalej strzelać).

2.1.20 Funkcja 'dropRandomTarget'

Funkcja dodająca do sceny pudełko w losowym miejscu.

2.1.21 Funkcja 'updateCannon'

Funkcja aktualizująca armatę (rotację, położenie, kamerę).

2.1.22 Funkcja 'updateOcean'

Funkcja aktualizująca morze (symulująca ruch morza).

2.1.23 Funkcja 'updateSplashes'

Funkcja aktualizująca chlapanie wody (przemieszczanie poszczególnych elementów wody).

2.1.24 Funkcja 'checkForWinCondition'

Funkcja sprawdzająca czy poziom został ukończony.

2.1.25 Funkcja 'removeUnderwaterObjects'

Funkcja usuwająca obiekty znajdujące się pod wodą.

2.1.26 Funkcja 'render'

Funkcja renderująca.

2.2 index.html

W pliku 'index.html' znajdują się dwa zapożyczone skrypty odpowiedzialne za symulowanie poruszającego się morza. Jeden z nich to Fragment Shader, drugi zaś Vertex Shader.

Dokładniejszy opis znajduje się w poszczególnych plikach źródłowych.