

Grafika 3D i systemy multimedialne

## POKÓJ STRACHÓW

**Autorzy:**

Piotr Osipa, nr indeksu:

Paweł Andziul, nr indeksu:

Maciej Kiedrowski, nr indeksu: 200105

Joanna Piątek, nr indeksu: 199966

**Grupa:** Czwartek TN/13:15

**Data oddania:** 19.01.2017

**Prowadzący:** dr inż. Jan Nikodem

**Ocena pracy:**

## Spis treści

<b>1</b>	<b>Instrukcja obsługi</b>	<b>3</b>
1.1	Uruchomianie aplikacji . . . . .	3
1.2	Sterowanie . . . . .	3
<b>2</b>	<b>Implementacja projektu</b>	<b>4</b>
2.1	Definicja sterowania . . . . .	4
2.2	Sterowanie postacią . . . . .	4
2.3	Poruszanie kijem . . . . .	4
2.4	Piłka . . . . .	4
2.5	Wazon i jego rozbicie . . . . .	4
2.5.1	Modelowanie . . . . .	4
2.5.2	Import do środowiska Unity . . . . .	4
2.5.3	Skrypt . . . . .	5
2.6	Świeczka . . . . .	5

# 1 Instrukcja obsługi

TestTestTest

*Test*

**Test**

## 1.1 Uruchomianie aplikacji

Aplikacja jest uruchamiana za pomocą pliku o rozszerzeniu .exe.

## 1.2 Sterowanie

		Funkcja	
		Sterowanie postacią	Poruszanie kijem
Urządzenie	Klawiatura	W, S, A, D lub strzałki	Q, E, R, F
	Pad do Xbox360	Lewa gałka analogowa	Prawa gałka analogowa

Tabela 1: Sterowanie

- a. Pierwszy punkt
- b. Drugi punky
- c. Trzeci punkt
- d. ...

## 2 Implementacja projektu

### 2.1 Definicja sterowania

### 2.2 Sterowanie postacią

Sterowanie postacią zostało zaimportowane z Standard Assets zawierających m.in. moduł `FirstPersonCharacter`, w którym zostało zdefiniowane całe sterowanie postacią: kiwanie się kamery przy chodzeniu, dźwięki kroków, postać w kształcie kapsuły, itp.

### 2.3 Poruszanie kijem

### 2.4 Piłka

### 2.5 Wazon i jego rozbicie

W celu zademonstrowania możliwości interakcji i destrukcji otoczenia, w pomieszczeniu umieszczona została waza. Przy mocnym uderzeniu, upadku, rozpada się na mniejsze fragmenty, z którymi użytkownik może przeprowadzać dalszą interakcję.

#### 2.5.1 Modelowanie

**Waza** Waza została zamodelowana w środowisku *blender*. Podstawowym kształtem dla wazy jest okrąg, z którego za pomocą narzędzia *Extrude* i skalowania w prosty sposób można utworzyć symetryczne, owalne przedmioty.

Krawędzie utworzonego modelu zostały wygładzone za pomocą narzędzia *Smooth* dostępnego w oprogramowaniu.

**Fragmenty** Modele rozbitej wazy - zarówno całej, jak i już rozbitych elementów które podlegają dalszej destrukcji zostały również utworzone w narzędziu *blender*. W tym celu skorzystano z dodatkowego narzędzia *Cell Fracture* dostępnego dla *blendera*. Dodatek ten pozwala na podział utworzonego modelu na  $n$  fragmentów za pomocą tesselacji Woronoja. Środki obszarów mogą zostać wygenerowane automatycznie, lub zostać wybrane przez użytkownika. Korzystając z narzędzia *Cell Fracture* użytkownik ma do dyspozycji szereg opcji, m.in. wybór maksymalnej liczby fragmentów na które zostanie podzielony model, czy wielkość odstępów pomiędzy nowo utworzonymi fragmentami. Z jednej strony pogarsza to wizualne wrażenia przy rozbiciu przedmiotu, jednakże pozwala ograniczyć ryzyko "wybuchu" szczątków na skutek błędu silnika fizycznego. Ma to szczególnie duże znaczenie przy podziale modelu na kilkadziesiąt lub kilkaset fragmentów.

#### 2.5.2 Import do środowiska Unity

Po zaimportowaniu modelu do *Unity*, należy wygenerować *collidery*, poprzez zaznaczenie opcji *Generate Colliders* w Inspektorze modelu. Na poszczególnych fragmentach należy zaznaczyć opcję *Convex* w komponencie **Mesh Collider** w celu umożliwienia wykrywania kolizji pomiędzy tymi obiektami. Aby zminimalizować odbicia obiektu od podłoża po upadku, materiał powinien być ustawiony na materiał z małym współczynnikiem odbicia, np. metal. Następnie każdemu fragmentowi należy dodać komponent **RigidBody**, który daje możliwość ustawienia takich cech obiektu jak masa czy opory powietrza wpływające na

zachowanie obiektu na scenie. Tak przygotowany obiekt w postaci *prefabu* jest gotowy do użycia na scenie.

### 2.5.3 Skrypt

W celu symulacji zniszczenia obiektu w środowisku *Unity*, konieczne jest przygotowanie modeli przed i po zniszczeniu. W momencie wykrycia działania na obiekt siły wystarczającej do jego zniszczenia, należy podmienić model na scenie. // Dla wazy został utworzony skrypt *DesctrucibleVase.cs*, jeżeli w momencie kolizji z innym obiektem zostanie przekroczona uprzednio zdefiniowana wartość progowa, dynamicznie tworzona jest nowa instancja wazy posiadająca za model wazę zniszczoną. Następnie dla każdego fragmentu nowej fazy przypisywana jest rotacja i pęd istniejącego obiektu, po czym obiekt ten jest usuwany ze sceny i zastępowany nowo utworzonym. W ten sposób użytkownik nie widzi podmiiany modelu, a jedynie efekt rozpadającej się na kawałki wazy. Ta sama technika może zostać wykorzystana przy dowolnym obiekcie w środowisku *Unity*.

## 2.6 Świeczka