**SPRAWOZDANIE**

**Zajęcia: Grafika i Multimedia**

**Prowadzący: prof. dr hab. inż. Vasyl Martsenyuk**

|  |  |
| --- | --- |
| Laboratorium Nr 1.  Data 07.03.2020  Temat: "Modelowanie gry 3D. Roll-a-ball ".  Wariant 3. | Jacek Adamczyk  Informatyka  II stopień, stacjonarne,  1 semestr, gr. A. |

<https://github.com/jacekaGIT/ATH-1g/tree/master/GIM-g/Lab1>

**1. Polecenie: wariant 3 zadania.**

Stworzenie gry Roll-a-ball w oparciu o materiały z zajęć oraz tutorial dostępny pod adresem:  
[https://learn.unity.com/tutorial/collecting-scoring-and-building-the-game?projectId=5c51479fedbc2a001fd5bb9f#](https://learn.unity.com/tutorial/collecting-scoring-and-building-the-game?projectId=5c51479fedbc2a001fd5bb9f)

* Szczegółowe wytyczne dla wariantu 3:

Kolor materialu „playera” – zielony,

obiekty „pick up” dwóch typów

1 typ obiekta „pick up” - capsule,

2 typ obiekta „pick up” - cylinder,

ilość obiektów „pick up” typu 1 - 7,

ilość obiektów „pick up” typu 2 - 5,

Kolor materialu obiektów „pick up” typu 1 – żółty,

Kolor materialu obiektów „pick up” typu 2 – brązowy,

Kolor materialu „ścian” - niebieski

Reguły gry:

trafianie w obiekt typu 1 – 1 punkt

trafianie w obiekt typu 2 – 4 punkty

warunek zakończenia gry – 9 punktów

**2. Opis opracowanego programu (kody źródłowe, zrzuty ekranu)**

Grę wykonano w środowisku Unity 2018.4.18f1 Personal, na podstawie Instrukcji Lab1\_pl.pdf, wymagań szczegółowych oraz tutoriala internetowego firmy Unity.

Instrukcja była opracowywana dla jednej z poprzednich wersji środowiska Unity i w związku z tym występują pewne różnice w stosunku do opisu. Przede wszystkim dotyczy to skryptów. W szczególności:

- Obsługę ruchu gracza zawarto w metodzie **FixedUpdate** skryptu **PlayerController**,

- Obsługę kolizji i licznika gry zawarto w skrypcie **PlayerController**,

- Nastąpiły niewielki zmiany i modyfikacje w skryptach **Rotator i CameraScript**.

* Skrypt **PlayerController**:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class PlayerController : MonoBehaviour

{

public float speed;

public Text countText;

public Text winText;

private Rigidbody rb;

private int count;

//Start is called before the first frame update

void Start()

{

rb = GetComponent<Rigidbody>();

count = 0;

SetCountText();

winText.text = "";

}

// Update is called once per frame

void Update()

{

/\*

float moveHorizontal = Input.GetAxis("Horizontal");

float moveVertical = Input.GetAxis("Vertical");

Vector3 movement = new Vector3(moveHorizontal, 0.0f, moveVertical);

rigidbody.AddForce(movement \* speed \* Time.deltaTime);

\*/

}

void FixedUpdate ()

{

float moveHorizontal = Input.GetAxis("Horizontal");

float moveVertical = Input.GetAxis("Vertical");

Vector3 movement = new Vector3(moveHorizontal, 0.0f, moveVertical);

rb.AddForce(movement \* speed);

}

private void OnTriggerEnter(Collider other)

{

if (other.gameObject.CompareTag ("Pick Up"))

{

other.gameObject.SetActive(false);

count = count + 1;

SetCountText();

}

if (other.gameObject.CompareTag("Pick Cyl"))

{

other.gameObject.SetActive(false);

count = count + 4;

SetCountText();

}

}

void SetCountText ()

{

countText.text = "Count: " + count.ToString();

if (count >= 9)

{

winText.text = "You Win !!!";

//Time.timeScale = 0;

}

}

}

* Skrypt **Rotator**:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Rotator : MonoBehaviour

{

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

transform.Rotate(new Vector3(15, 30, 45) \* Time.deltaTime);

}

}

* Skrypt **CameraSript**:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class CameraScript : MonoBehaviour

{

public GameObject player;

private Vector3 offset;

// Start is called before the first frame update

void Start()

{

offset = transform.position - player.transform.position;

}

// Update is called once per frame

void LateUpdate()

{

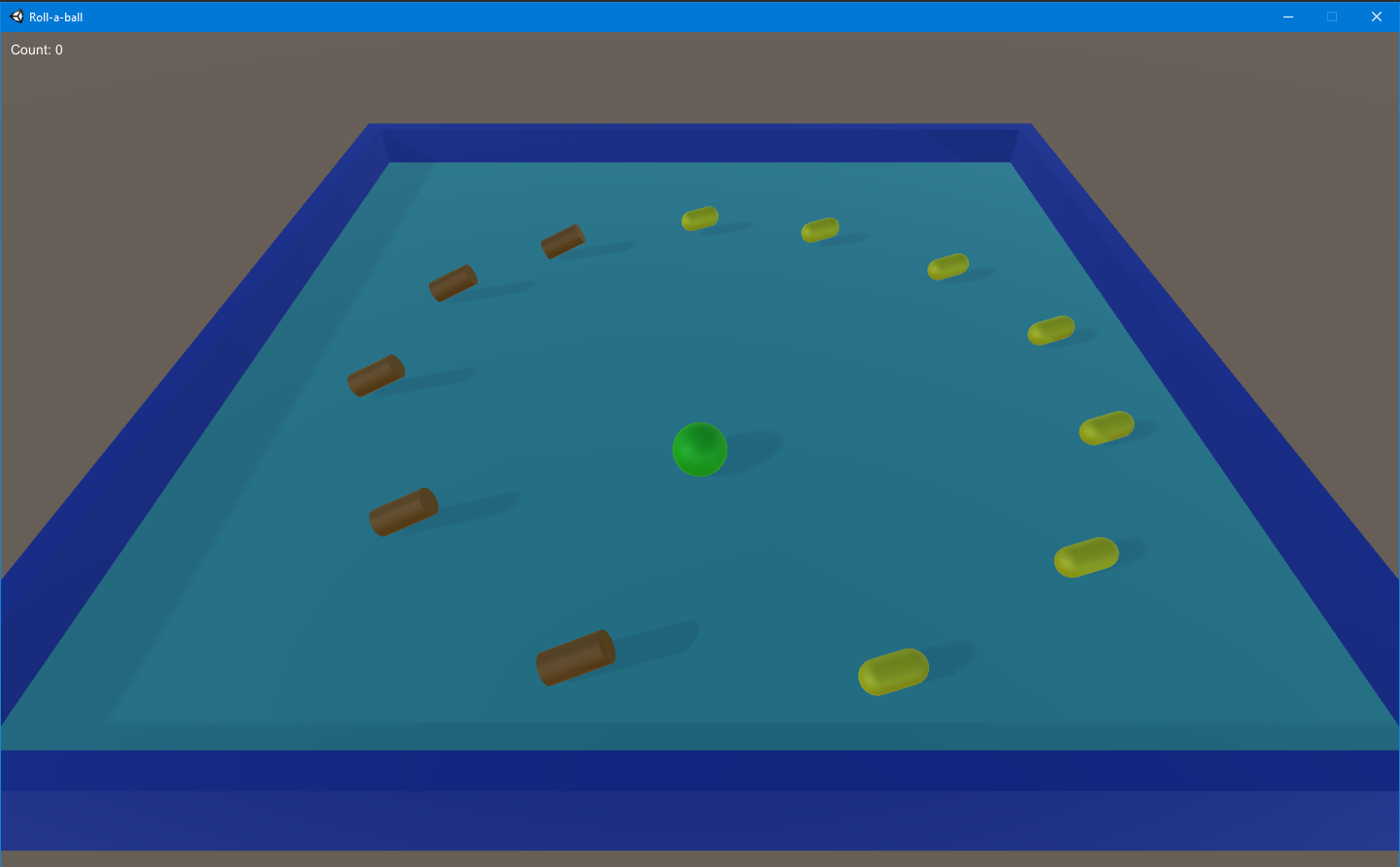
transform.position = player.transform.position + offset;

}

}

* Efekt końcowy:

Plansza po uruchomieniu gry:



* Nagranie z przebiegu gry:

Dwukrotne naciśnięcie na poniższy obiekt uruchomi odtwarzanie (działa w dokumencie \*.docx, nie działa w dokumencie \*.pdf).



**3. Wnioski**

Wykonana gra „Roll-a-bal” zawiera wszystkie założone funkcje i spełnia wymagania postawione w zadaniu.

Szczególnej uwagi, podczas opracowywania gry wymagało przygotowanie i pozycjonowanie obiektów w przestrzeni trójwymiarowej. Podczas tych kroków łatwo o błędy które skutkują nieoczekiwanym zachowaniem obiektów i koniecznością powtarzania wielu operacji wstecz.

Jeżeli wystąpiły rozbieżności pomiędzy instrukcją w .pdf a tutorialem internetowym to wykonywano operacje zgodnie z tutorialem internetowym.

Sprawozdanie, tutoriale oraz pliki źródłowe gry oraz wersję skompilowaną można znaleźć w repozytorium GITHUB pod adresem:

<https://github.com/jacekaGIT/ATH-1g/tree/master/GIM-g/Lab1>