Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра Обчислювальної Техніки

**Лабораторна робота №2**

з дисципліни "Розробка ігрових застосувань. Unity рішення"

Тема: "Дослідження базового патерну ігрового рушія Unity на прикладі тривимірного ігрового застосунку"

Виконав: Перевірив:

студент групи ІП-93 Катін Павло Юрійович

Домінський Валентин

Олексійович

Київ 2022

Зміст

[Мета: 3](#_Toc114645098)

[Завдання до роботи: 3](#_Toc114645099)

[Хід роботи: 4](#_Toc114645100)

[Рух: 4](#_Toc114645101)

[Середовище: 5](#_Toc114645102)

[Камера: 5](#_Toc114645103)

[Висновки: 6](#_Toc114645104)

[Додатки: 6](#_Toc114645105)

[Вихідний код: 6](#_Toc114645106)

[Посилання: 8](#_Toc114645107)

# Мета:

1. Полягає у набутті знань, умінь та навичок з технології розроблення основ проекту з використанням обраної мови програмування у обраній парадигмі. Надається досвід створення репозиторію у системі контролю версій і знання елементів середовища розробки і основи вихідного коду для управління грою

# Завдання до роботи:

1. Репозиторій у системі контролю версій. Створити проект 2D. Загальні вимоги. Акаунт на GitHab, на даному етапі за бажанням. Репозиторій на GitHab з проектом. Назва GameProgLab2GroupNum, де зафарбовано номер групи.
2. Установка ігрового рушія. Створений проект IDE (2D) на основі рушія, що містить 1 сцену, ігровий персонаж. Можуть бути включені інші елементи. Розроблений і налагоджений скрипт для управління ігровим персонажем. Достатньо продемонструвати рух ліворуч, праворуч, стрибки, коректну фізику, зупинку перед перешкодою. Проект розташовано у репозиторій на GitHab

# Хід роботи:

Прізвище –> Домінський

Ім’я –> Валентин

Шифр групи –> ІП-93

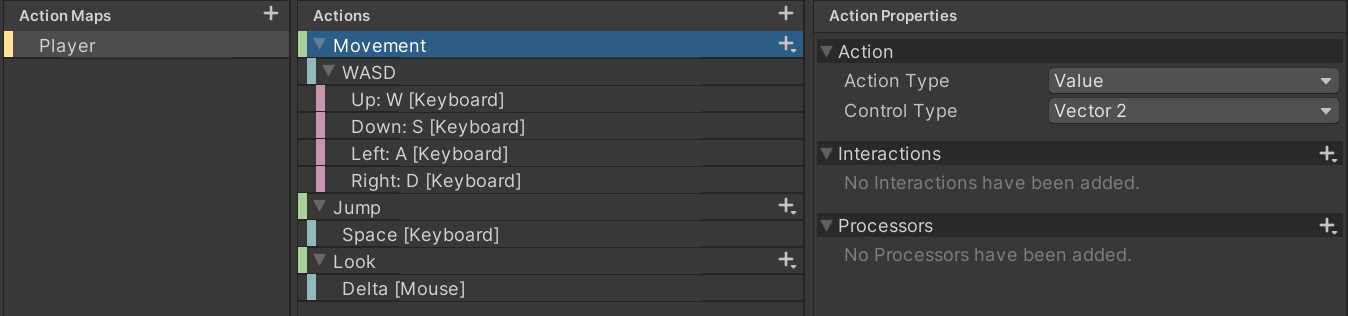
Назва факультету –> ФІОТ

Варіант –> 8 mod 5 = 3 -> 2022 PBR XfrogPlants Sampler та шар

## Рух:

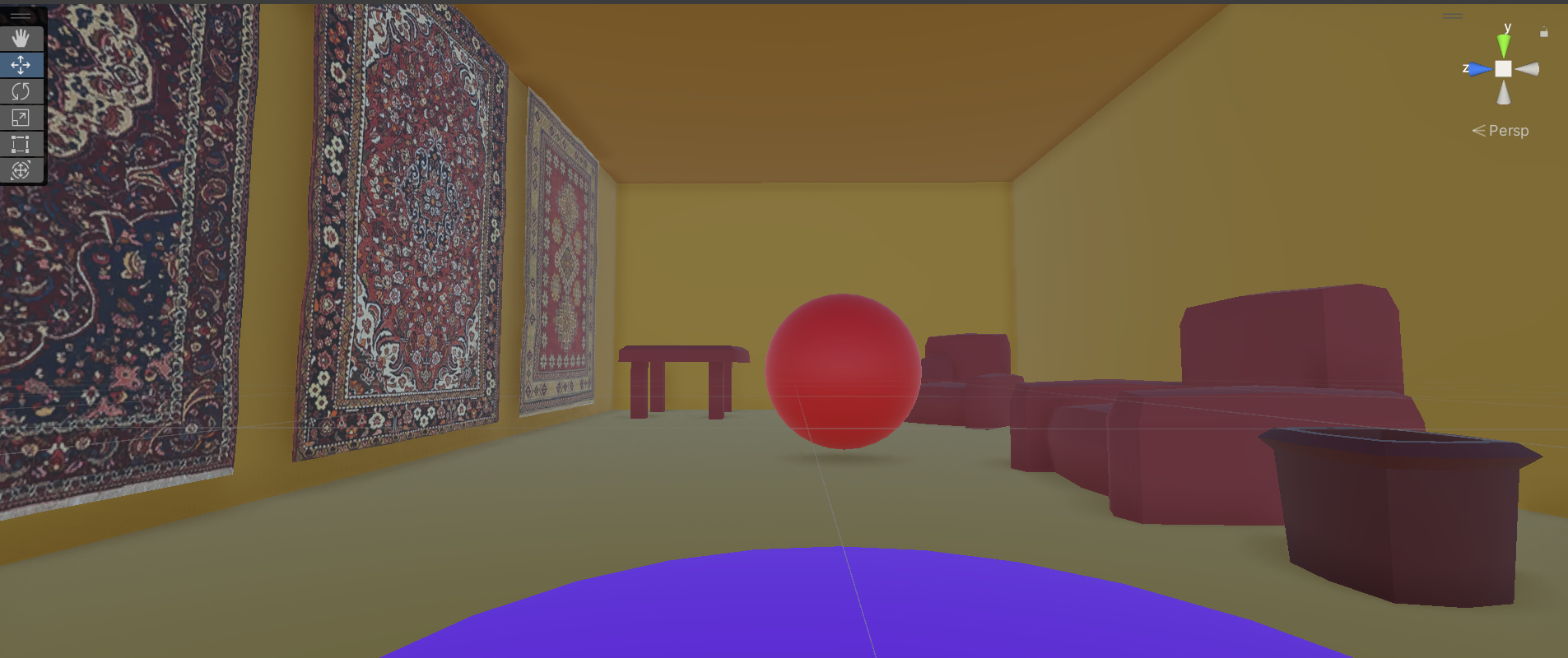
Для даного завдання Я вирішив використати нову систему руху, яка заснована на подіях та вже готовий прототип керування гравця:



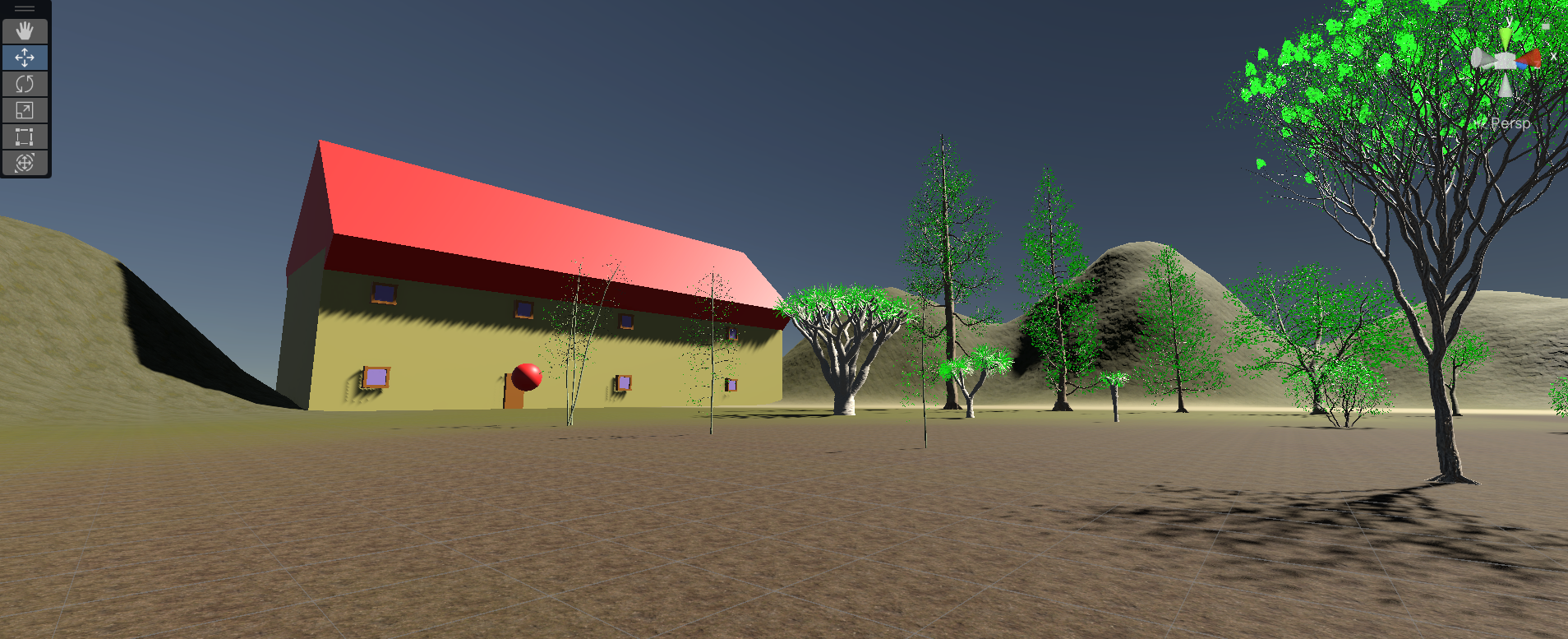


## Середовище:

Асету, який Мені випав, не вистачало для побудови кількох рівнів, тому Я вирішив взяти ще кілька для інтер’єру.



Усюди додав колайдери, а для кимимів зробив його трігером, щоб можна дуже близько до них підхоидити та отримувати за це бали. Як тільки гравець тричі дотикається до килимів, його відправляє на другу сцену – садочок.



## Камера:

Використав Я пакет Cinemachine. Завдяки йому та різним віртуальним камерам Я можу дуже кінематографічно слідувати за гравцем

# Висновки:

Я Спробував попрацювати з 3D простором. Додавав рух камери, нову систему інпутів, лічильник

# Додатки:

# Вихідний код:

PlayerController.cs:

**using** UnityEngine;

[RequireComponent(typeof(CharacterController))]

**public** **class** PlayerController : MonoBehaviour

{

[SerializeField]

**private** Transform \_cameraTransform;

[SerializeField]

**private** **float** \_playerSpeed = 2.0f;

[SerializeField]

**private** **float** \_jumpHeight = 1.0f;

[SerializeField]

**private** **float** \_fallScale = -3f;

**private** CharacterController \_controller;

**private** PlayerMovement \_playerMovement;

**private** Vector3 \_playerVelocity;

**private** **void** Awake()

{

\_playerMovement = new();

\_controller = gameObject.GetComponent<CharacterController>();

Cursor.lockState = CursorLockMode.Locked;

Cursor.visible = **false**;

}

**private** **void** OnEnable()

{

\_playerMovement.Enable();

}

**private** **void** OnDisable()

{

\_playerMovement.Disable();

}

**void** Update()

{

**var** isGrounded = \_controller.isGrounded;

**if** (\_playerVelocity.y < 0 && isGrounded)

{

\_playerVelocity.y = 0f;

}

Vector2 movement =

\_playerMovement.Player.Movement.ReadValue<Vector2>();

Vector3 controllerMovement = new(movement.x, 0, movement.y);

controllerMovement =

\_cameraTransform.forward \* controllerMovement.z +

\_cameraTransform.right \* controllerMovement.x;

controllerMovement.y = 0f;

\_controller.Move(\_playerSpeed \* Time.deltaTime \* controllerMovement);

**if** (\_playerMovement.Player.Jump.triggered && isGrounded)

{

\_playerVelocity.y += Mathf.Sqrt(\_jumpHeight \*

\_fallScale \* Physics.gravity.y);

}

\_playerVelocity.y += Physics.gravity.y \* Time.deltaTime;

\_controller.Move(\_playerVelocity \* Time.deltaTime);

}

}

RugCounter.cs:

**using** System;

**using** UnityEngine;

**public** **class** RugCounter : MonoBehaviour

{

[SerializeField]

**private** **string** \_rugTag = "Rug";

**public** **static** **event** Action<**int**> RugCollisionAdded;

**private** **int** \_rugCount;

**private** **const** **int** RugCountTrigger = 3;

**private** **void** OnTriggerEnter(Collider other)

{

**if** (other.CompareTag(\_rugTag))

{

\_rugCount++;

RugCollisionAdded?.Invoke(\_rugCount);

}

**if** (\_rugCount == RugCountTrigger)

{

SceneChanger.ChangeScene();

}

}

}

RugCounterText.cs:

**using** TMPro;

**using** UnityEngine;

**public** **class** RugCounterText : MonoBehaviour

{

[SerializeField]

**private** **string** \_rugCountText = "Score is: ";

**private** TextMeshProUGUI \_text;

**private** **void** Start()

{

\_text = GetComponent<TextMeshProUGUI>();

RugCounter.RugCollisionAdded += UpdateRugCounter;

}

**private** **void** UpdateRugCounter(**int** rugCount)

{

\_text.text = \_rugCountText + rugCount;

}

**private** **void** OnDestroy()

{

RugCounter.RugCollisionAdded -= UpdateRugCounter;

}

}

SceneChanger.cs:

**using** UnityEngine;

**using** UnityEngine.SceneManagement;

**public** **class** SceneChanger : MonoBehaviour

{

**private** **static** **readonly** **string** \_nextScene = "Garden";

**public** **static** **void** ChangeScene()

{

SceneManager.LoadScene(\_nextScene);

}

}

# Посилання:

Проект - [посилання](https://github.com/VsIG-official/Development-Of-Gaming-Applications.Unity-Solutions)