Министерство науки и высшего образования Российской Федерации

федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский экономический университет имени Г.В. Плеханова»

**МОСКОВСКИЙ ПРИБОРОСТРОИТЕЛЬНЫЙ ТЕХНИКУМ.**

Специальность: 09.02.07 Информационные системы и программирование

Квалификация: Программист

ОТЧЕТ

ПО ОБЩЕПРОФЕССИОНАЛЬНОЙ ДИСЦИПЛИНЕ: МДК 01.01 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ

Создание проекта, передвижение персонажа.

Листов: 10

|  |  |
| --- | --- |
| Выполнил студент  Группы П50-4-21  Игошев Р. В.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2023 года | Проверил преподаватель  \_\_\_\_\_\_\_\_\_\_\_А.Ю.Бушин  «\_\_\_\_» \_\_\_\_\_\_\_2023 года |

Москва 2023

Цель работы: При помощи базовых знаний интерфейса Unity и языка программирования С#, реализовать передвижение персонажа.

Начать следует с создания проекта. Версия Unity – 2022.3.8f1, вид проекта – 3д.

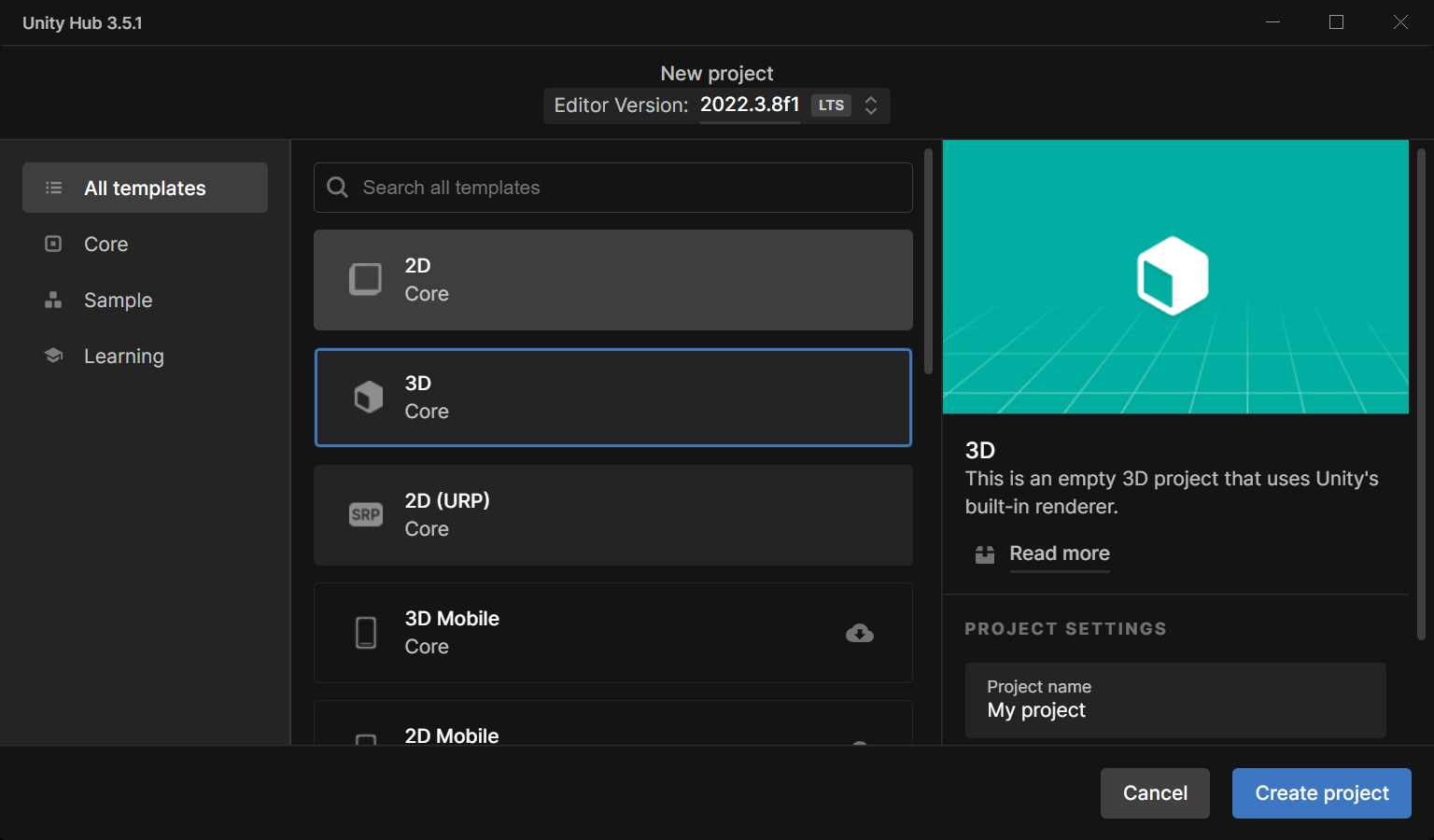


Рисунок 1 – Создание проекта

Теперь очередь создать площадь для нахождения на ней будущих объектов. Для этого нужно создать объект Plane и задать ему следующие настройки:



Рисунок 2 – Компоненты и настройки площадки

В данном случае важен Mesh Collider, который позволит регистрировать столкновения с другими объектами, а в данном случае, позволит стоять на площадке.

Теперь необходимо создать пустой объект, который будет выступать в роли игрока и задать ему следующие настройки:

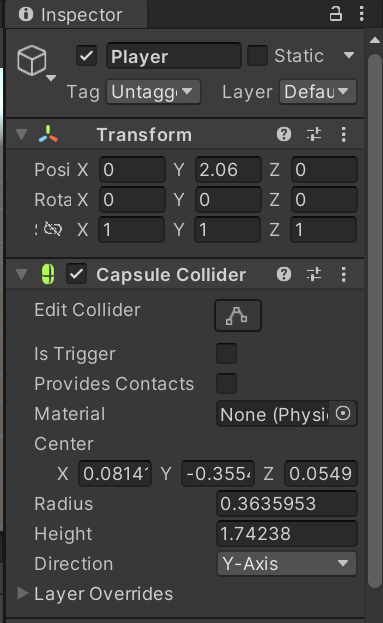


Рисунок 3 – Настройки коллайдера игрока

Коллайдер здесь нужен по что бы по его границам было выстроено взаимодействие с другими объектами.

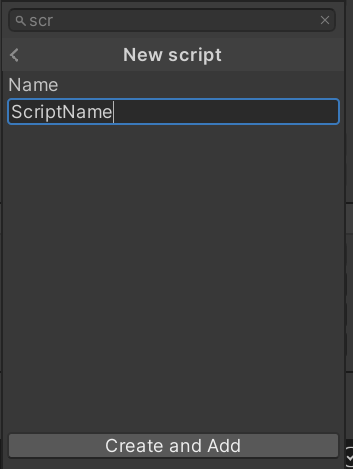


Рисунок 4 – Меню создания скрипта

В компоненты нужно также добавить скрипт, который впоследствии будем программировать.

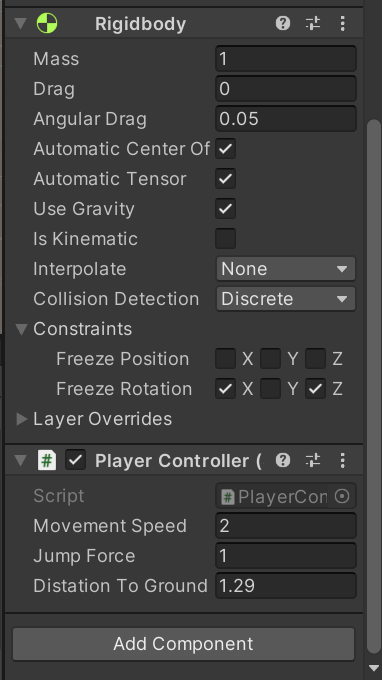


Рисунок 5 – Настройки физики игрока и подключение скрипта к нему

RigidBody здесь нужен для симуляции столкновения и для придачи физических свойств объекту.

После этого на сцене видно следующую картину:

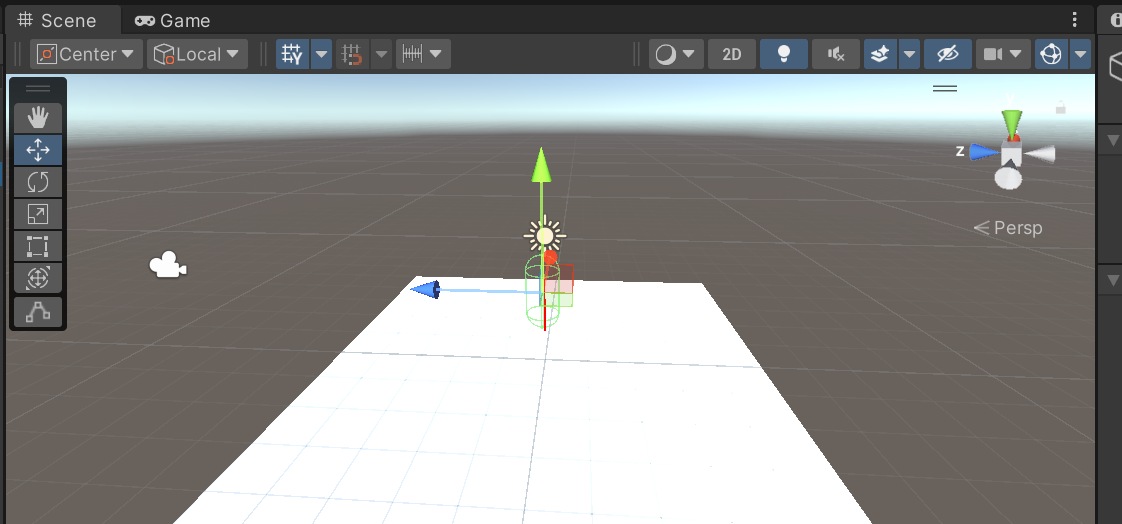


Рисунок 6 – Текущий результат

Для визуального отображения игрока следует добавить в него модель. Выглядит это вот так:

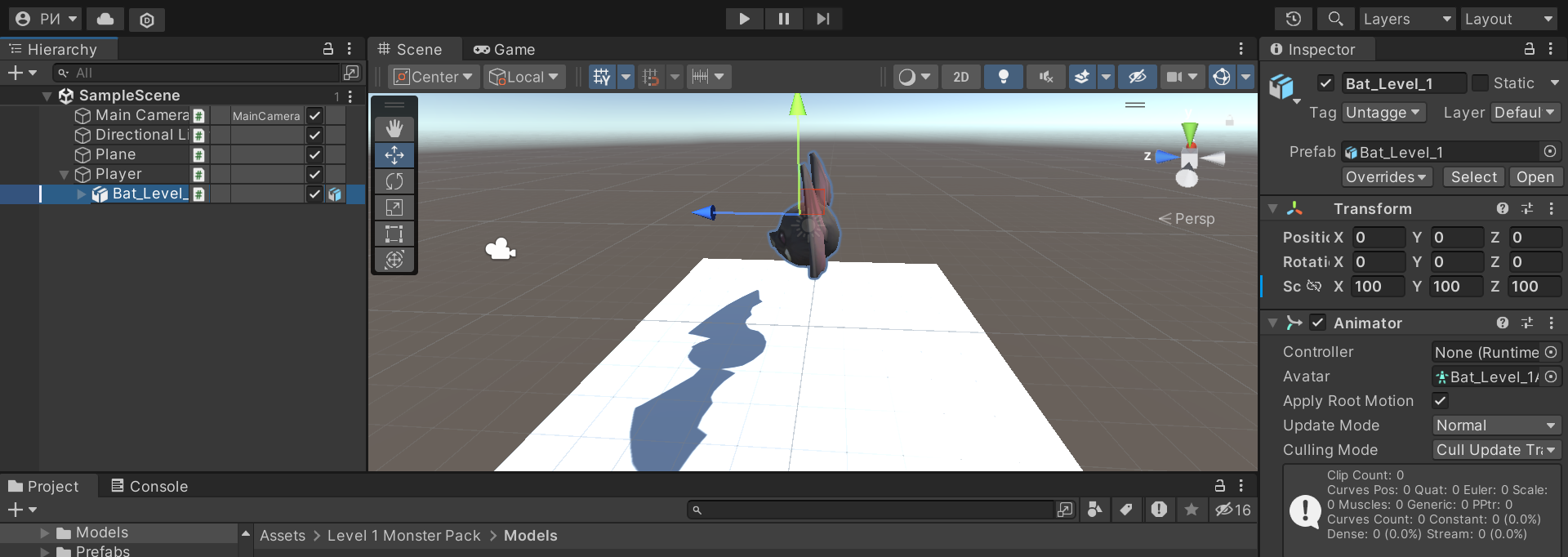


Рисунок 7 – Результат с моделью

Время написать код. Код скриптов Unity пишется на языке C#, который рекомендуется писать в Visual Studio.

Код работы:

using UnityEngine;

public class PlayerController : MonoBehaviour

{

public float MovementSpeed = 2.0f;

public float JumpForce = 5.0f;

public float DistationToGround = 0.1f;

private Rigidbody \_Rigidbody;

private bool IsGrounded;

void Start()

{

\_Rigidbody = GetComponent<Rigidbody>();

}

void FixedUpdate()

{

GroundCheck();

if (Input.GetKey(KeyCode.Space) && isGrounded)

{

Jump();

}

\_Rigidbody.MovePosition(CalculateMovement());

}

private Vector3 CalculateMovement()

{

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

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

Vector3 Move = transform.right \* HorizontalDirection + transform.forward \* VerticalDirection;

return \_Rigidbody.transform.position + Move \* Time.fixedDeltaTime \* MovementSpeed;

}

private void Jump()

{

\_Rigidbody.AddForce(Vector3.up \* JumpForce, ForceMode.Impulse);

}

private void GroundCheck()

{

isGrounded = Physics.Raycast(transform.position, Vector3.down, DistationToGround);

}

private void OnDrawGizmosSelected()

{

Gizmos.color = Color.red;

Gizmos.DrawLine(transform.position, transform.position + (Vector3.down \* DistationToGround));

}

}

Описание функций:

Start – выполняется при запуске сцены, здесь нужен для получения RigidBody игрока.

FixedUpdate – выполняется при отрисовке кадра, нужен для вызова прыжка или перемещения.

CalculateMovement – просчитывает передвижение игрока и возвращает новую позицию.

Jump – выполняет прыжок игрока.

GroundCheck – проверка того, находится ли на земле игрок.

OnDrawGizmosSelected – косметическая функция, позволяющая нам наглядно рассмотреть расстояние, необходимое для выполнения условия нахождения на земле при выделении объекта.

Результаты работы:

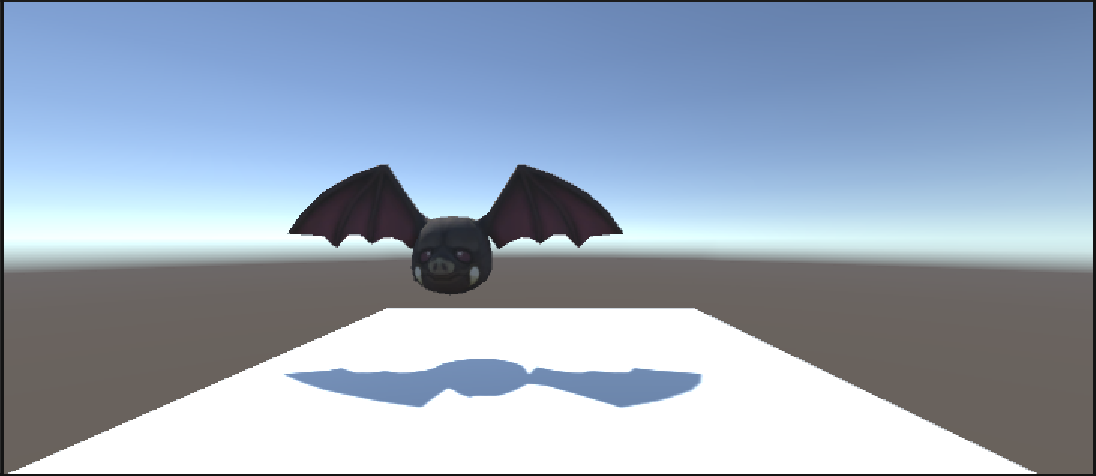


Рисунок 8 – Объект до перемещения

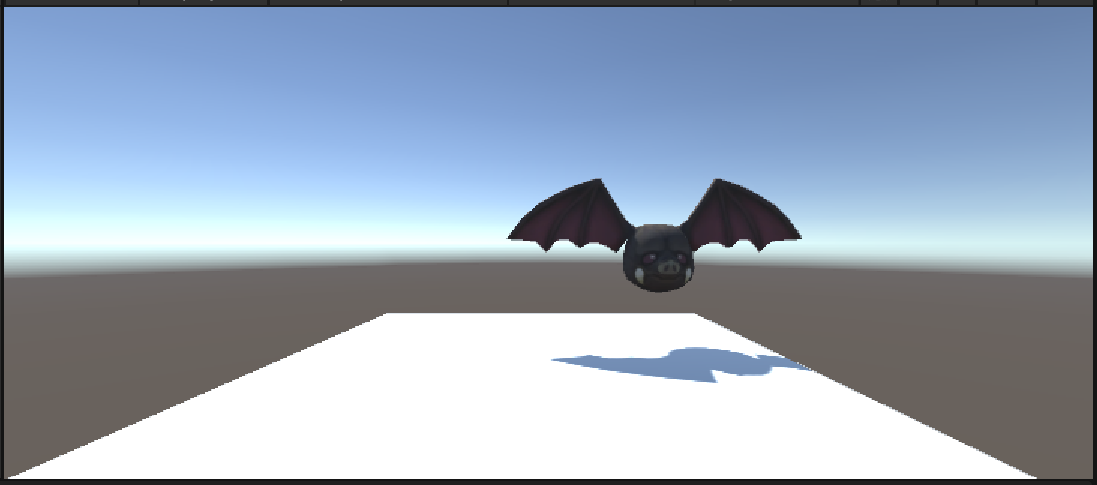


Рисунок 9 – Объект после перемещения

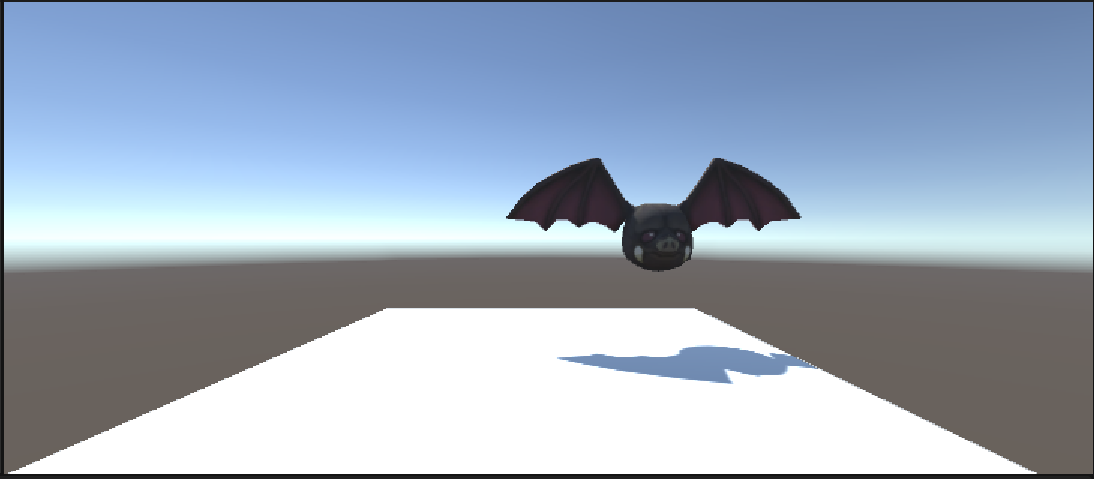


Рисунок 10 – Объект в момент прыжка

Вывод: При помощи базовых знаний интерфейса Unity и языка программирования С#, было реализовано передвижение персонажа.