МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания по выполнению лабораторных работ

по дисциплине «Искусственный интеллект в игровых приложениях»

для студентов направления подготовки
09.03.02 «Информационные системы и технологии»
Профиль «Прикладное программирование в интеллектуальных информационных системах»

Ставрополь, 2026

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

1. Цели и задачи освоения дисциплины. Основная цель – дать студентам теоретическое представление о современных проблемамах в области систем искусственного интеллекта и машинного обучения, а также познакомить с путями их решения.

Пособие предназначено для студентов, обладающих теоретическими знаниями в области проектирования приложений и практическими навыками программирования (предпочтительно языки C, C++, C#, Python, R). Цель учебного пособия: сформировать у студентов целостный взгляд на современные тенденции в областях машинного обучения, искусственного интеллекта, анализа данныз; сформировать систему компетенций.

Учебное пособие (лабораторный практикум) по дисциплине «Искусственный интеллект в игровых приложениях» для студентов направления 09.03.02 «Информационные системы и технологии». Пособие охватывает теоретические аспекты построения информационных систем на основе методов искусственного интеллекта, а также предлагает студентам практические рекомендации по разработке систем на основе методов искусственного интеллекта. Основное внимание уделяется теории обучения машин (машинное обучение, machine learning).

В пособии рассмотрены практические аспекты проектирования и разработки информационных систем для решения различных задач с использованием следующих подходов: линейных методов классификации, аппарата нейронных сетей, байесовских методов классификации, алгоритмов восстановления регрессии, алгоритмов логической классификации.

Многие задачи, возникающие в практических приложениях, не могут быть решены заранее известными методами или алгоритмами. Это происходит по той причине, что нам заранее не известны механизмы порождения исходных данных или же известная нам информация недостаточна для построения модели источника, генерирующего поступающие к нам данные.

Как говорят, мы получаем данные из «черного ящика». В этих условиях ничего не остается, как только изучать доступную нам последовательность исходных данных и пытаться строить предсказания совершенствуя нашу схему в процессе предсказания. Подход, при котором про- шлые данные или примеры используются для первоначального формирования и совершенствования схемы предсказания, называется методом машинного обучения (Machine Learning). Машинное обучение – чрезвычайно широкая и динамически развивающаяся область исследований, использующая огромное число теоретических и практических методов.

ЛАБОРАТОРНАЯ РАБОТА 1. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА АРХИТЕКТУРЫ ИГРЫ

Цели и задачи

Цель лабораторной работы: знакомство с игровым движком Unity; изучение основных инструментов разработки игр; приобретение практических навыков работы со сценой и игровыми объектами.

Основные задачи:

- познакомиться с инструментарием игрового движка Unity;
- научиться размещать игровые объекты на сцене;
- познакомиться с игровой физикой (Collider, RigidBody).

Теоретическое обоснование

Перед выполнением практической работы необходимо ознакомиться с базовыми принципами работы с Unity.

Основные горячие клавиши:

Q	Pan (перемещение камеры сцены)
W	Move (перемещение)
E	Rotate (вращение)
R	Scale (масштабирование)
CTRL+D	Дублировать
CTRL+SHIFT+F	Выровнять по виду
Зажатая ПКМ	Вращение на сцене
Зажатое колесико мыши	Перемещение по сцене

С расширенным списком горячих клавиш вы можете ознакомиться по ссылке: https://docs.unity3d.com/ru/530/uploads/Main/Unity_HotKeys_Win.pdf

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько програмных продуктов для практической реализации задач лабораторной работы): MS Visual Studio 2019 и выше; среда разработки Java, интерпретатор Python (Anaconda).

Методика и порядок выполнения работы

Перед выполнением индивидуального задания настоятельно рекомендуется ознакомиться с учебной задачей.

Проект учебной задачи находится по ссылке:

https://github.com/RedInHat/DPO_practices/blob/main/practice1.unitypackage

1.1. Учебная задача

1. Откройте Unity Hub, выберите вкладку projects и создайте новый проект.

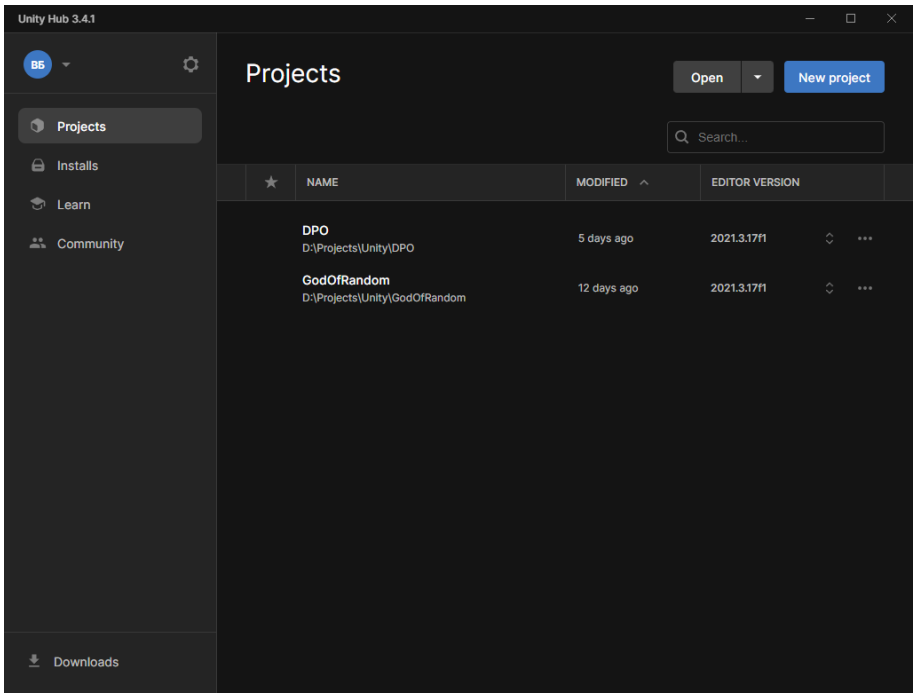


Рисунок 1 – вкладка Projects в Unity Hub

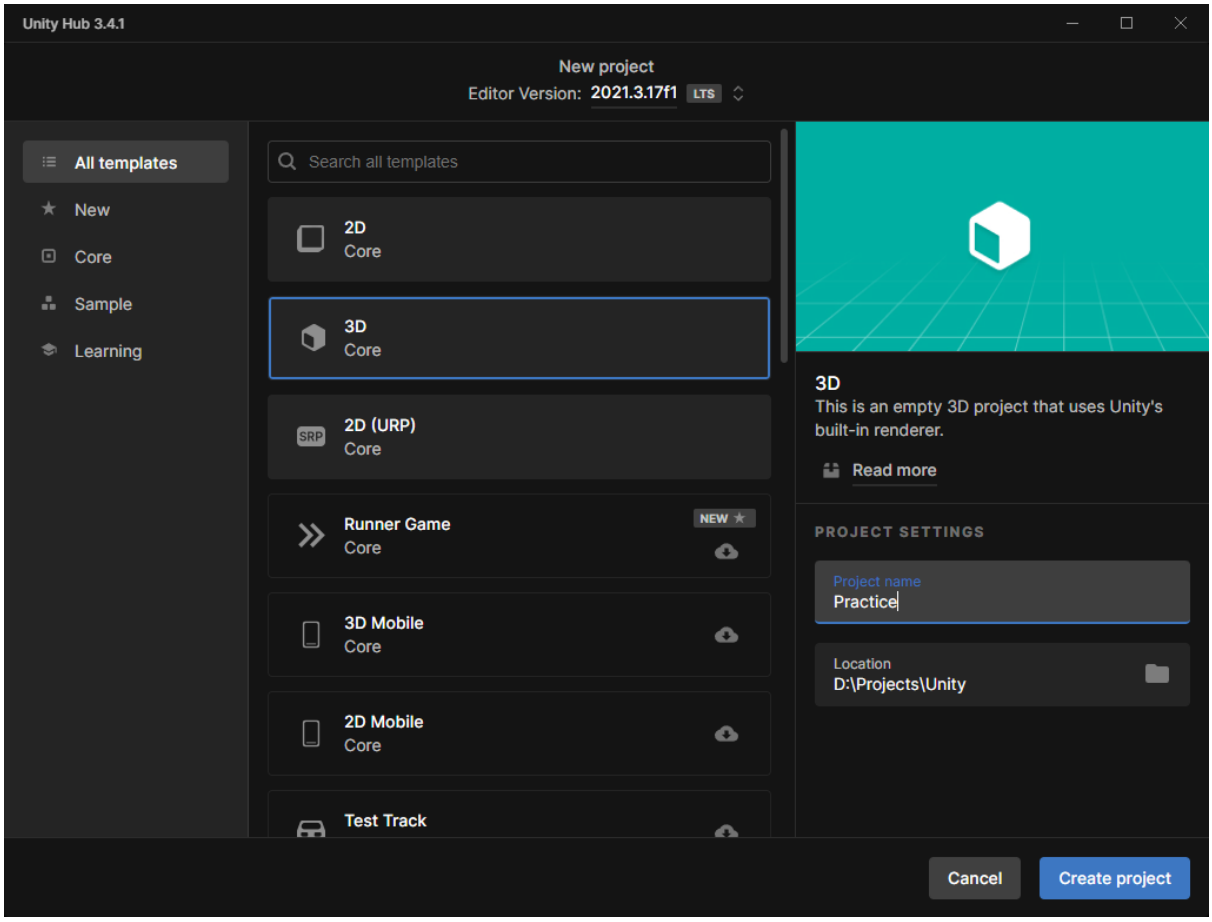


Рисунок 2 – Создание 3d проекта

После загрузки и создания проекта откроется окно Unity, которое будет выглядеть следующим образом:

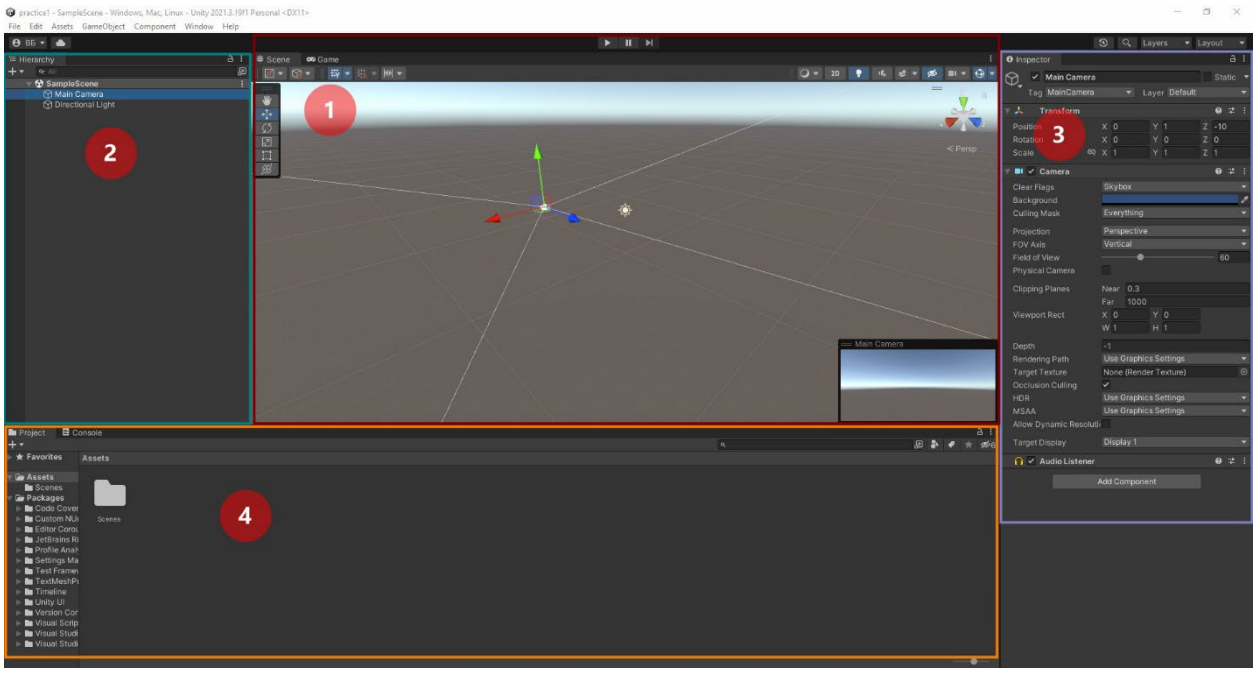


Рисунок 3 – Интерфейс начального окна Unity

На начальном экране вы увидите следующие элементы:

1. Окно редактора сцены. В этом окне вы будете размещать объекты на сцене. Здесь же есть вкладка Game, в которой отображается то, как вашу игру будет видеть игрок. Чуть выше есть 3 кнопки для запуска/паузы проекта.
2. Окно иерархии объектов на сцене.
3. Окно инспектора. В этом окне отображаются все компоненты выбранного объекта.
4. Окно проекта и консоли. Все элементы (папки, материалы, префабы, скрипты и т.д.) будут отображаться здесь.

2. Добавьте примитив Plane на сцену.

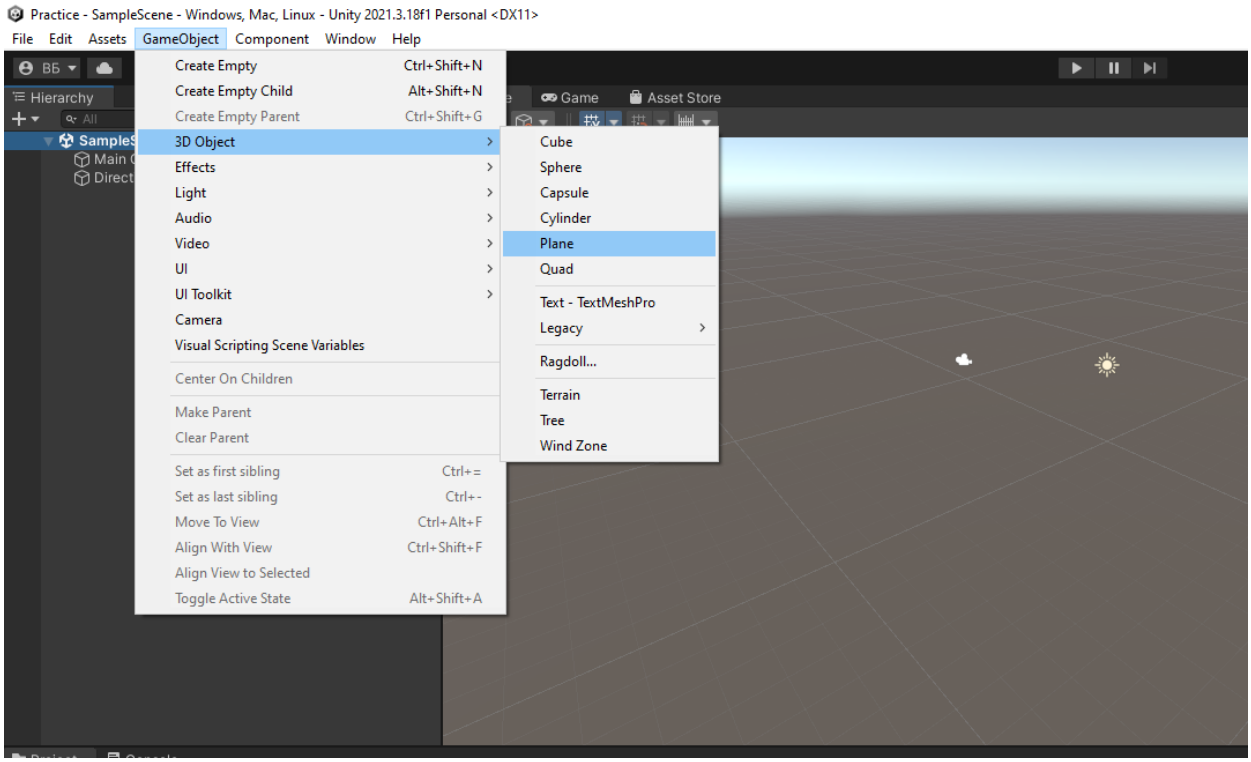


Рисунок 4 – Добавление примитива Plane

Чтобы перемещать объект по сцене можно тянуть за оси в нужном направлении (красная ось – Ось X, синяя ось – ось Z, зеленая ось – ось Y).

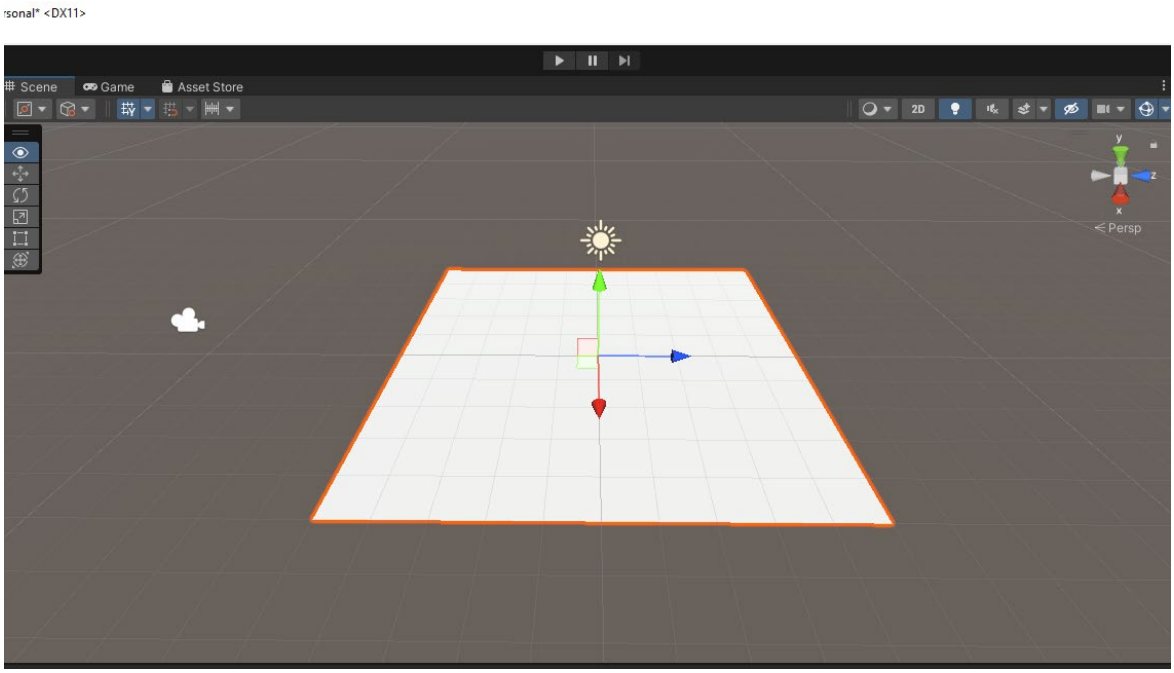


Рисунок 5 – Оси Plane

Или изменять числовые значения компонента Transform.

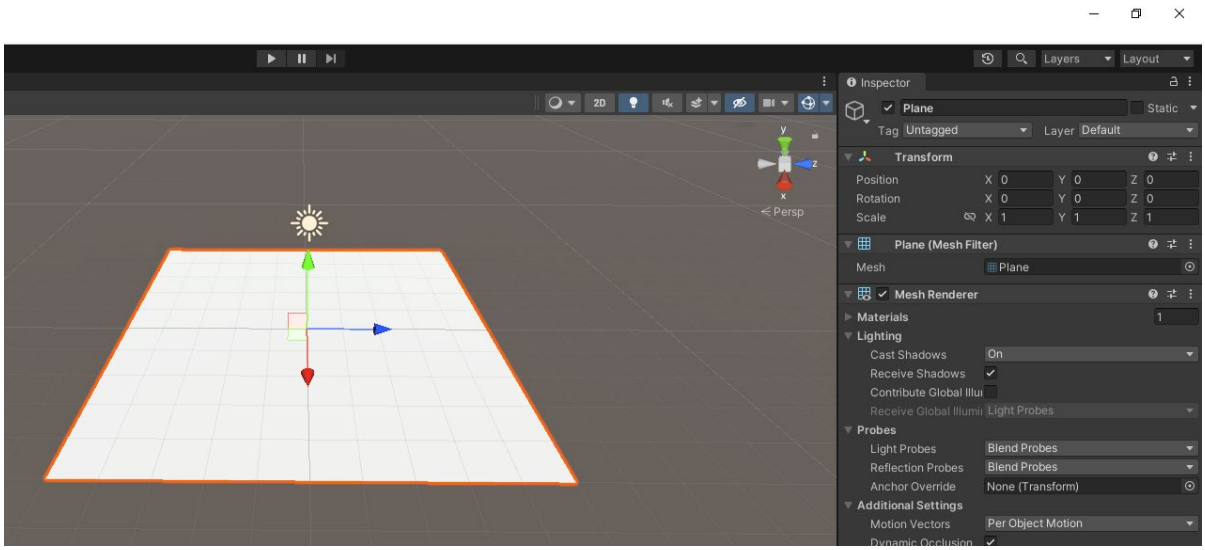


Рисунок 6 – Transform

3. Чтобы поменять цвет объекта, нужно добавить материал. Для этого в окне Project создайте пустую папку и назовите ее Materials.

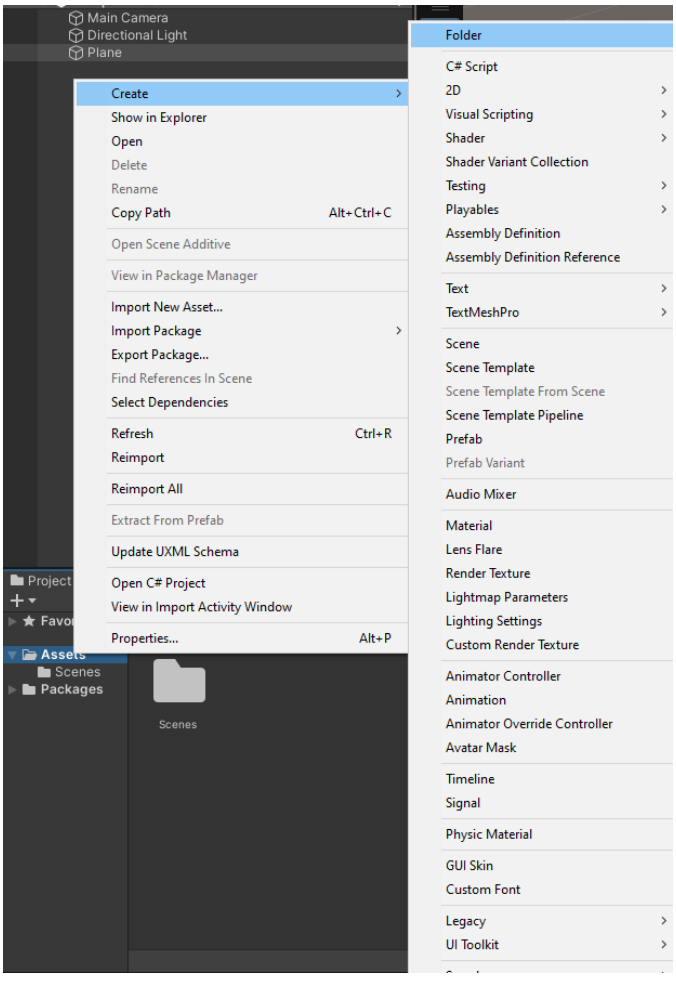


Рисунок 7 – Добавление новой папки

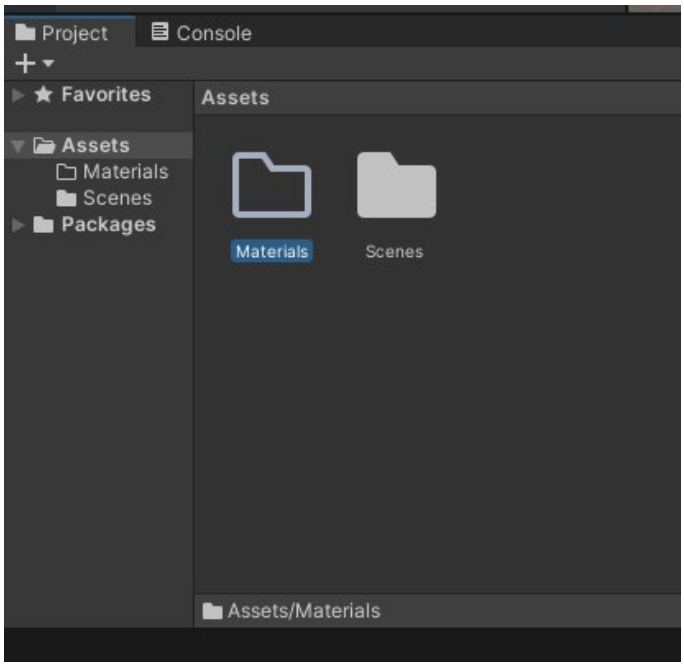


Рисунок 8 – Новая папка в окне Project

4. В папке Materials создайте материал. Назовите, например, floor. Или оставьте название по умолчанию.

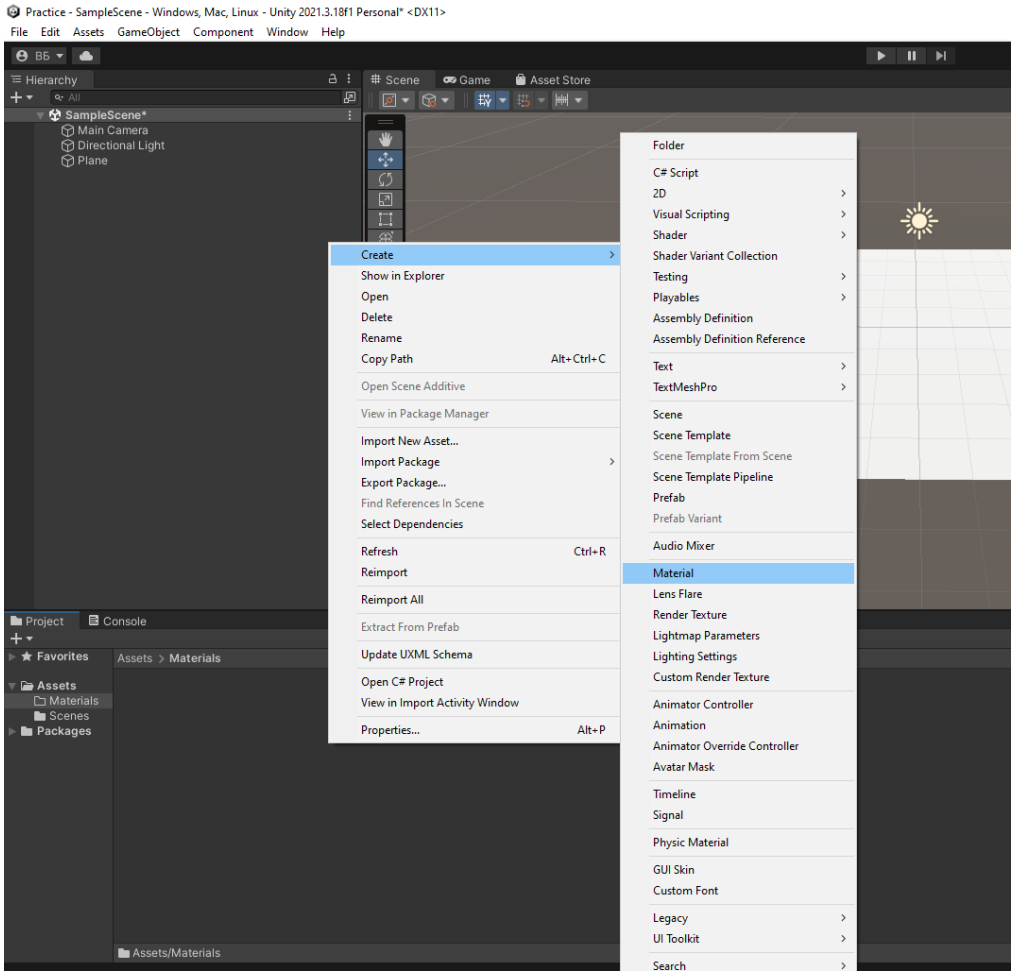


Рисунок 9 – Создание материала

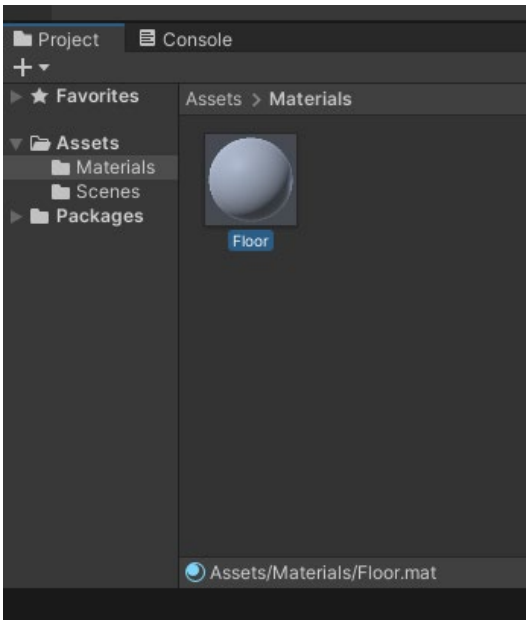


Рисунок 10 – Новый материал

5. Выделите материал в окне Project и в Inspector поменяйте цвет.

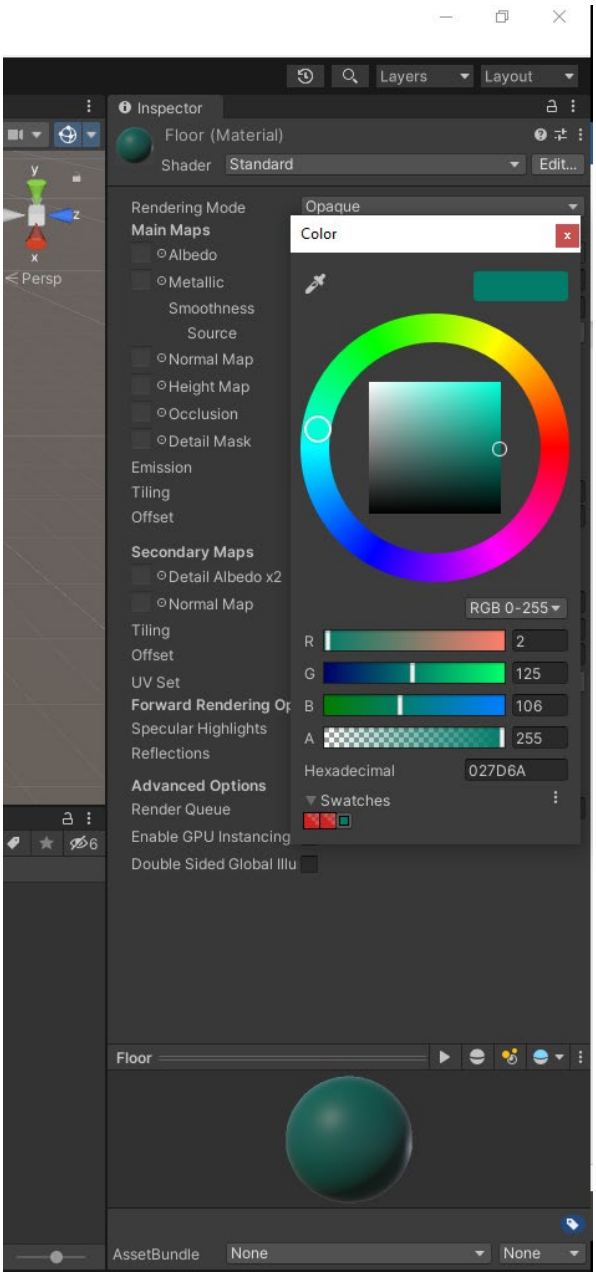


Рисунок 11 – Изменение цвета материала

6. Для того, чтобы Plane принял цвет Floor, перетащите материал из окна Project в окно Editor на Plane. Если вы всё сделали правильно, то Plane поменял цвет, на выбранный вами.

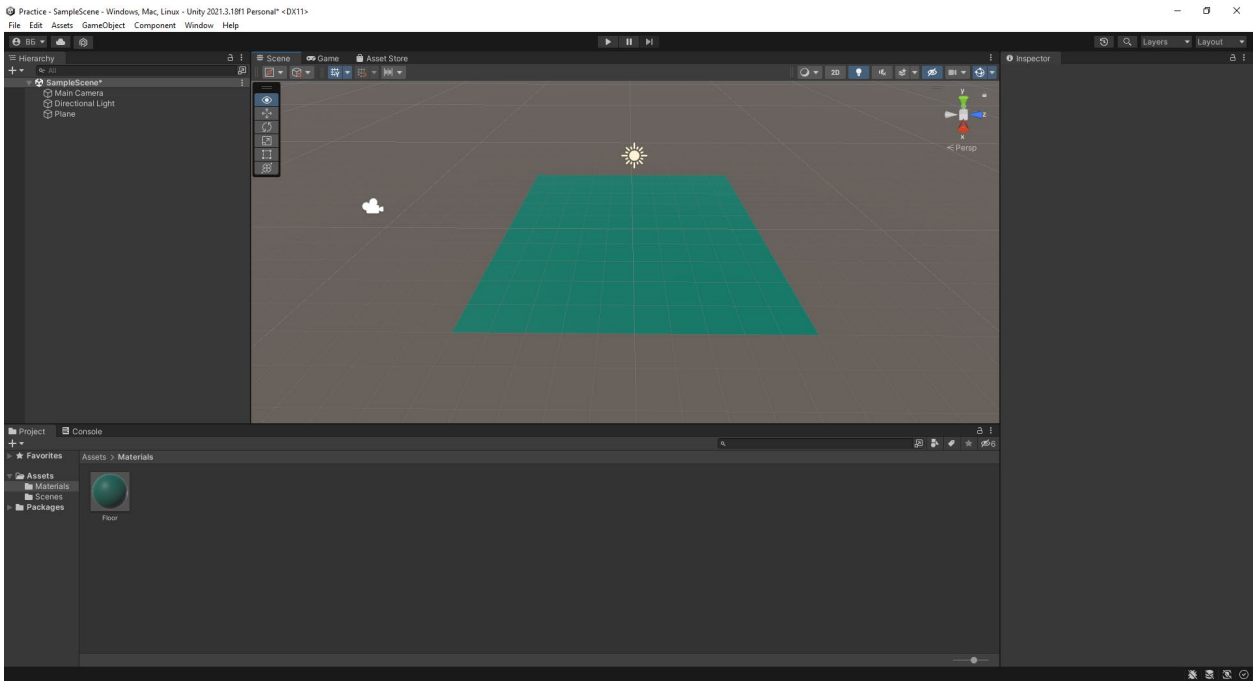


Рисунок 12 – Добавление материала к Plane

Чтобы не переключаться между вкладками Editor и Game, вкладку Game можно перетащить в удобное вам место.

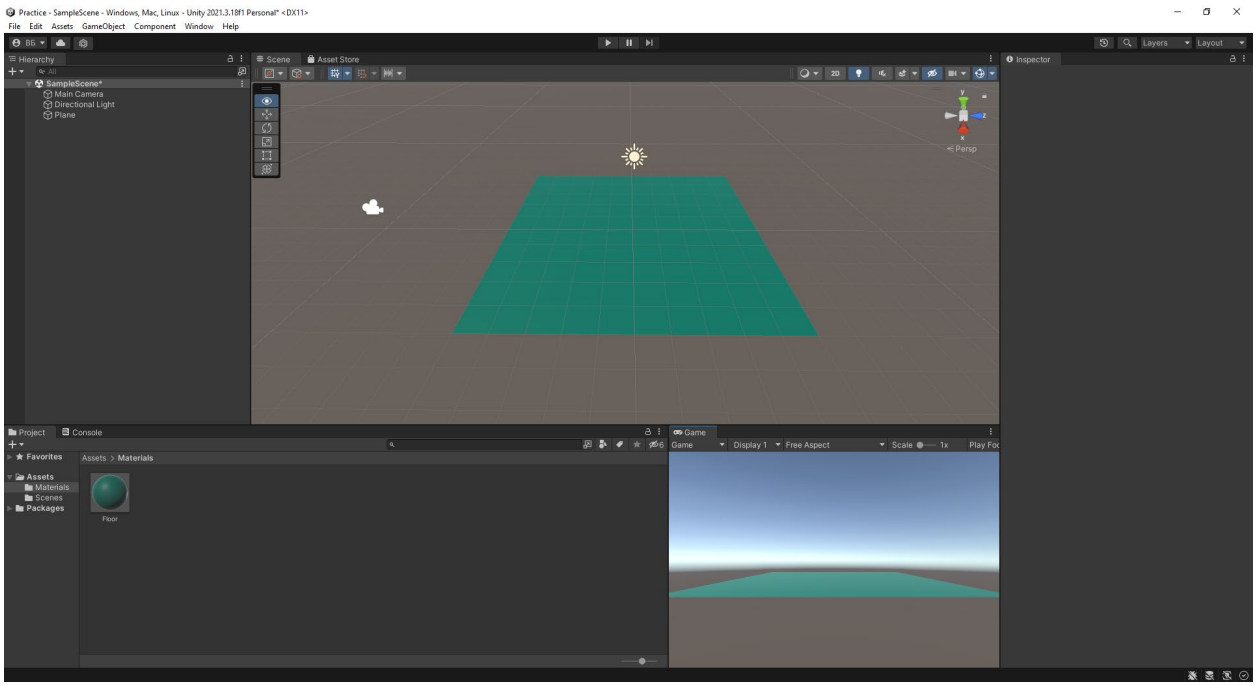


Рисунок 13 – Перемещение вкладки Game

То, как ваша игра будет отображаться зависит от объекта MainCamera. Перемещая ее, можно увидеть, что картинка в вкладке Game меняется. Слева

в окне Editor есть панель с инструментами, которые позволяют перемещать, вращать и масштабировать объекты. Перемещаясь по сцене, добейтесь подходящего для вас вида Plane в Editor (как показано на рисунке), затем выделите MainCamera в иерархии объектов и нажмите Ctrl+Shift+F.

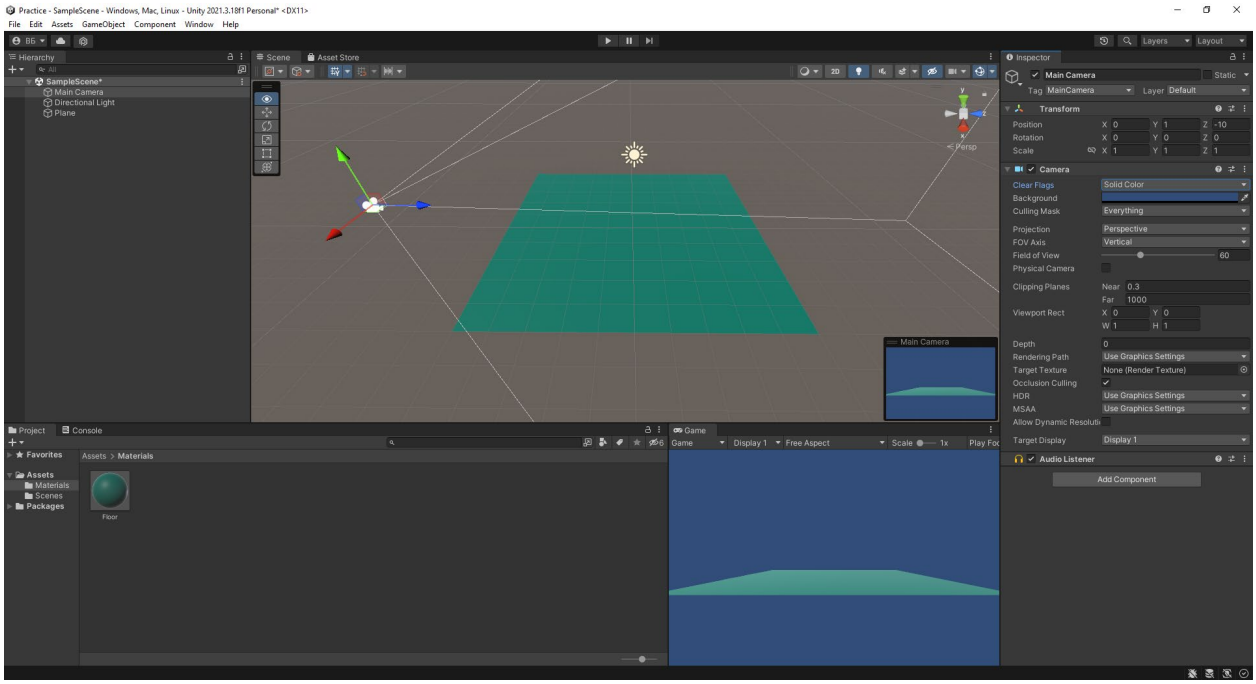


Рисунок 14 – Изменение вида в редакторе

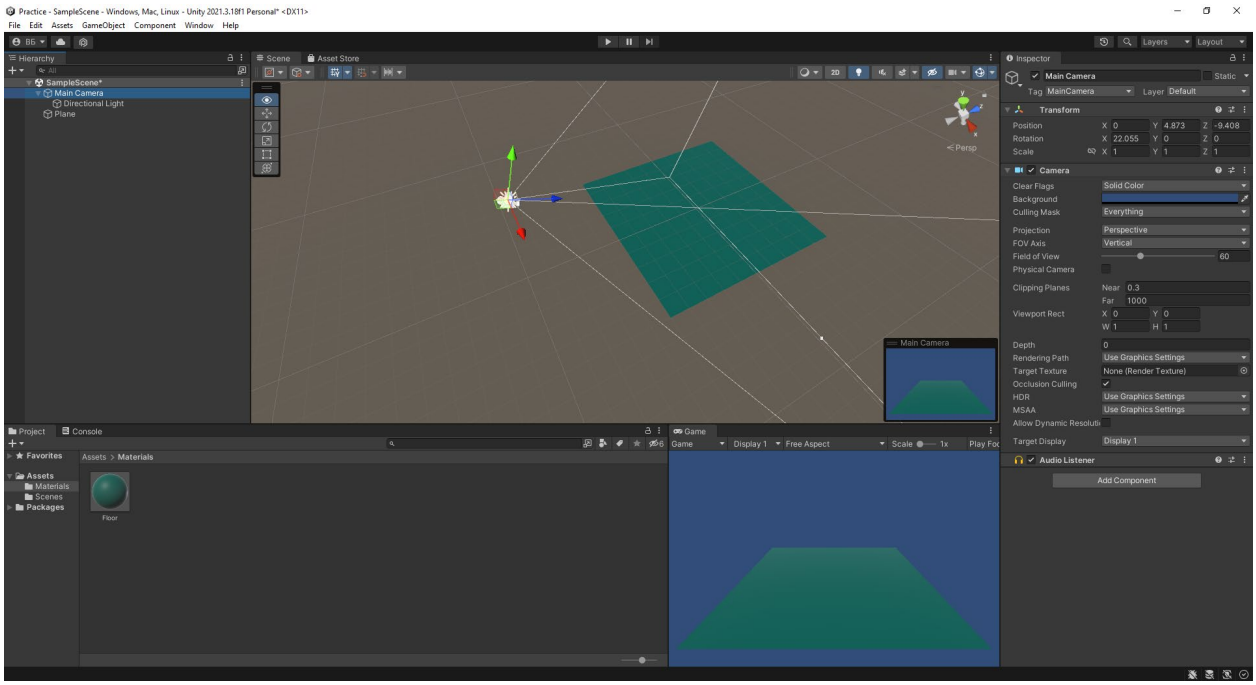


Рисунок 15 – Постановка камеры в позицию вида

7. Добавьте новый примитив (Sphere) на сцену. В инспекторе нажмите Add Component и добавьте RigidBody. Это позволит сфере обрести физические свойства, такие как: масса, гравитация и т.д. Запустите игру и посмотрите, что произошло.

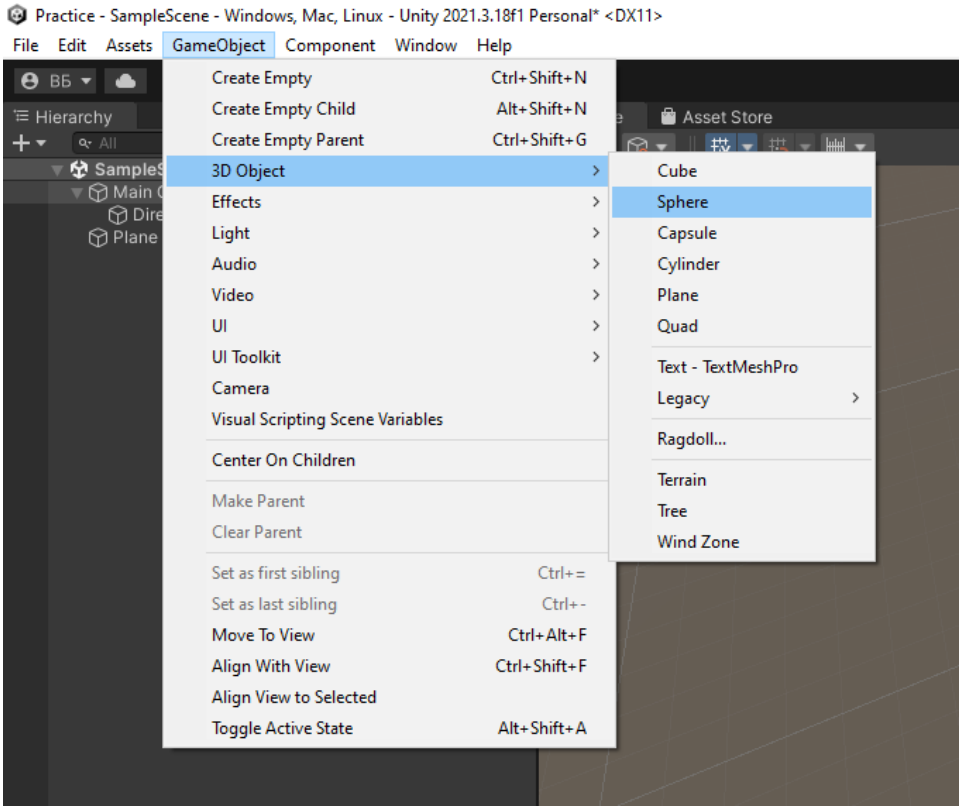


Рисунок 16 – Добавление примитива Sphere

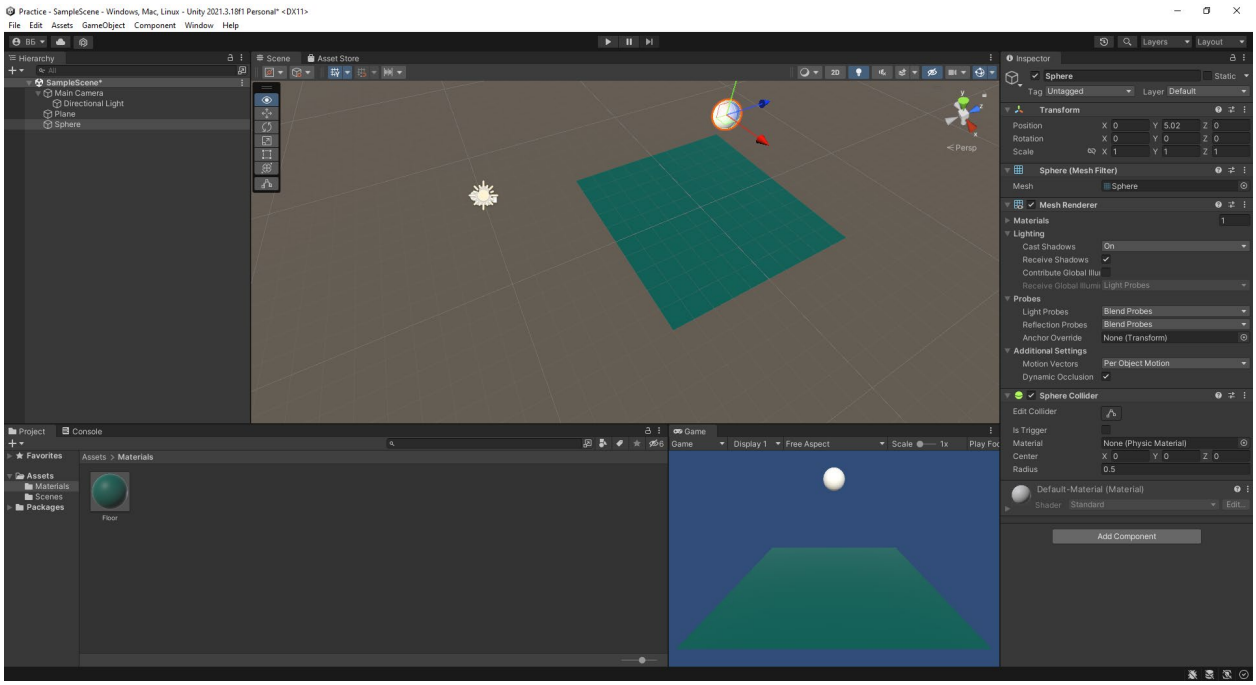


Рисунок 17 – Перемещение сферы по сцене

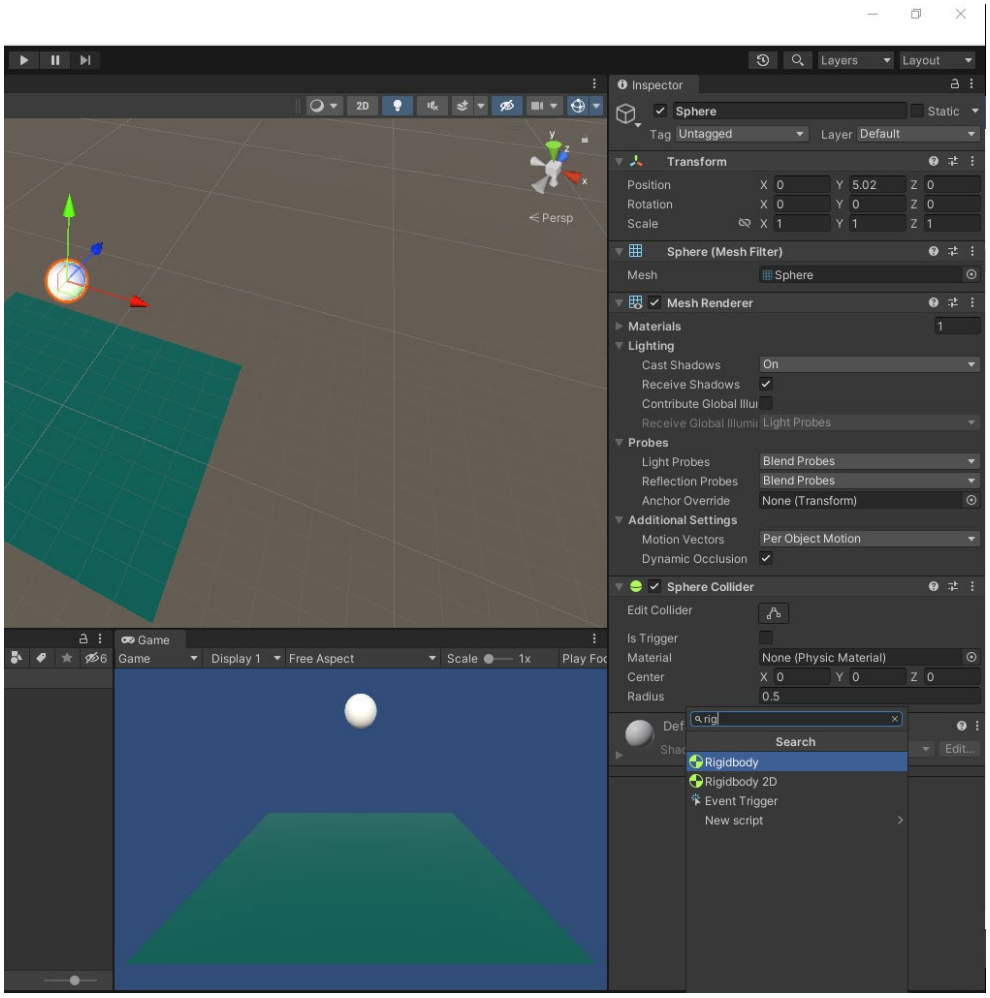


Рисунок 18 – Добавление компонента RigidBody к примитиву Sphere

Рисунок 19 – Свойства RigidBody

8. В папке Materials создайте Physical Material. В инспекторе у этого материала поменяйте значение параметра Bounciness на 1. Перетащите этот материал на сферу. Запустите игру и посмотрите, что поменялось.

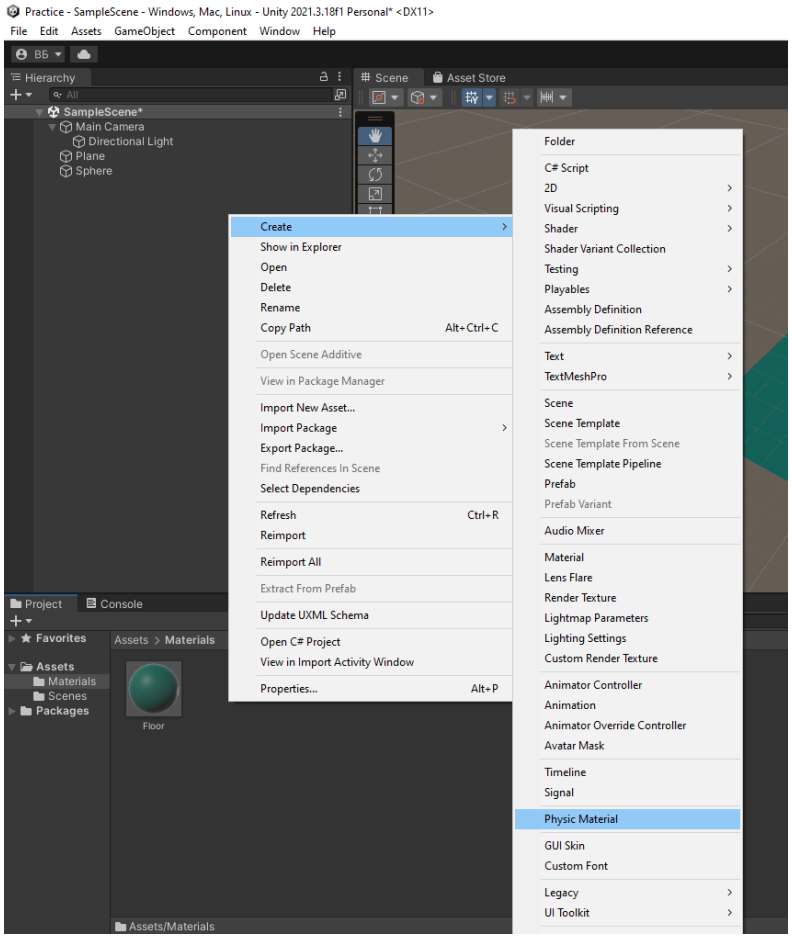


Рисунок 20 – Создание физического материала

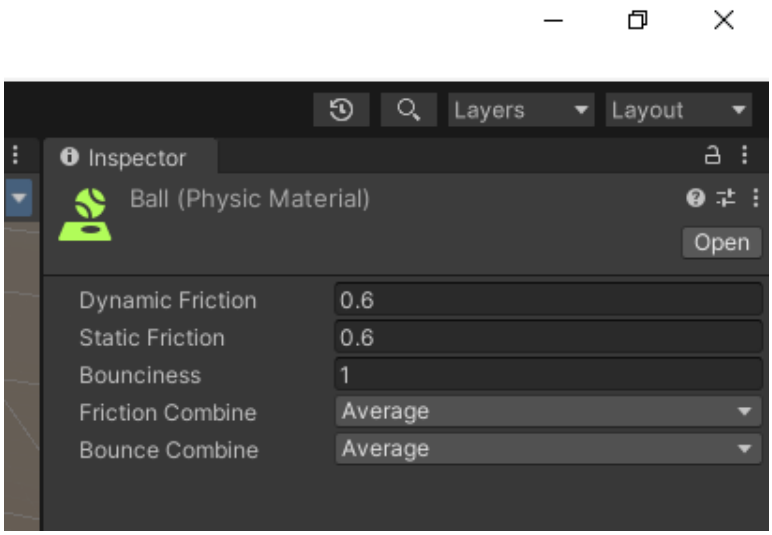


Рисунок 21 – Свойства физического материала

9. Добавьте несколько кубов на сцену. Разместите их на некоторой высоте. Добавьте всем RigidBody. У одного куба увеличьте массу. Запустите игру. Проверьте как работает свойство isKinematic компонента RigidBody.

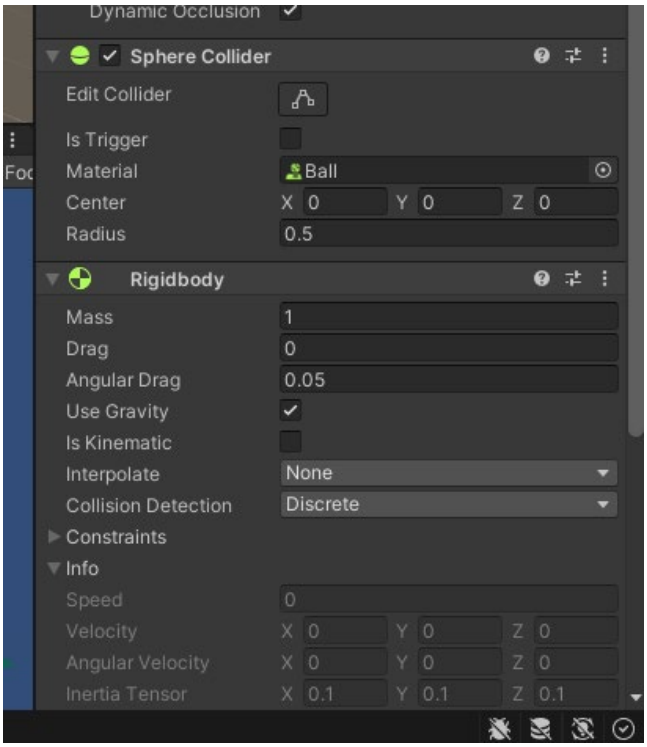


Рисунок 22 – Добавление физического материала к сфере

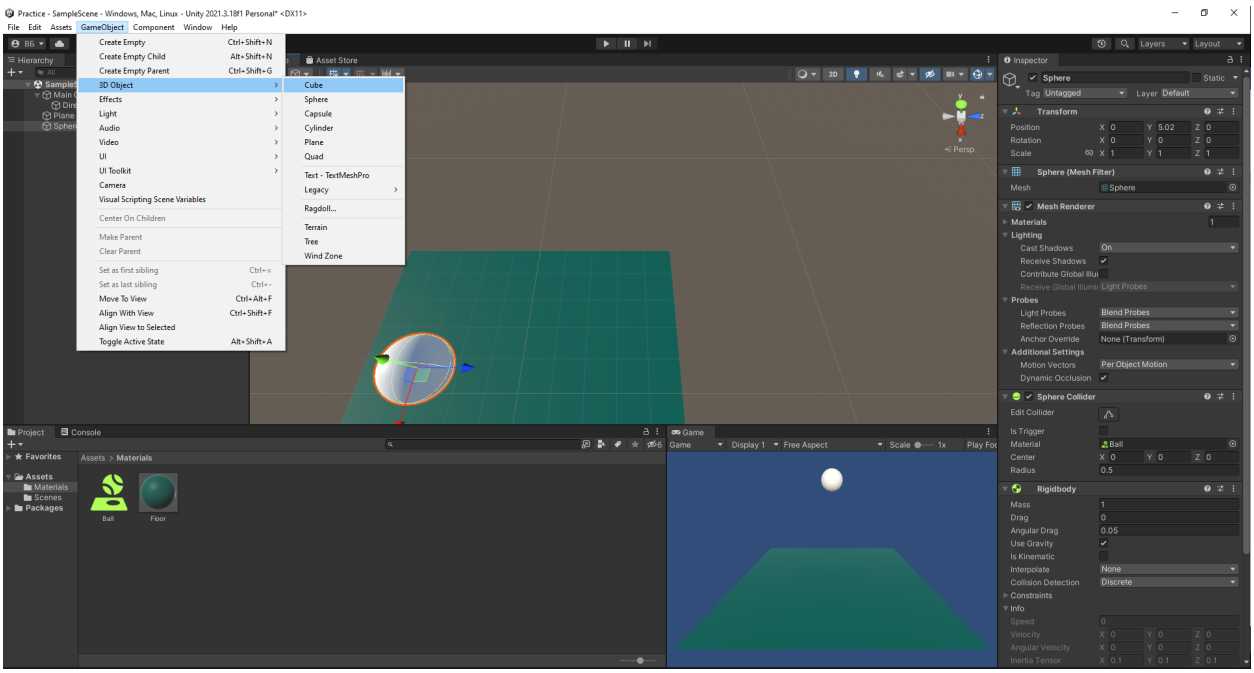


Рисунок 23 – Добавление примитива Cube

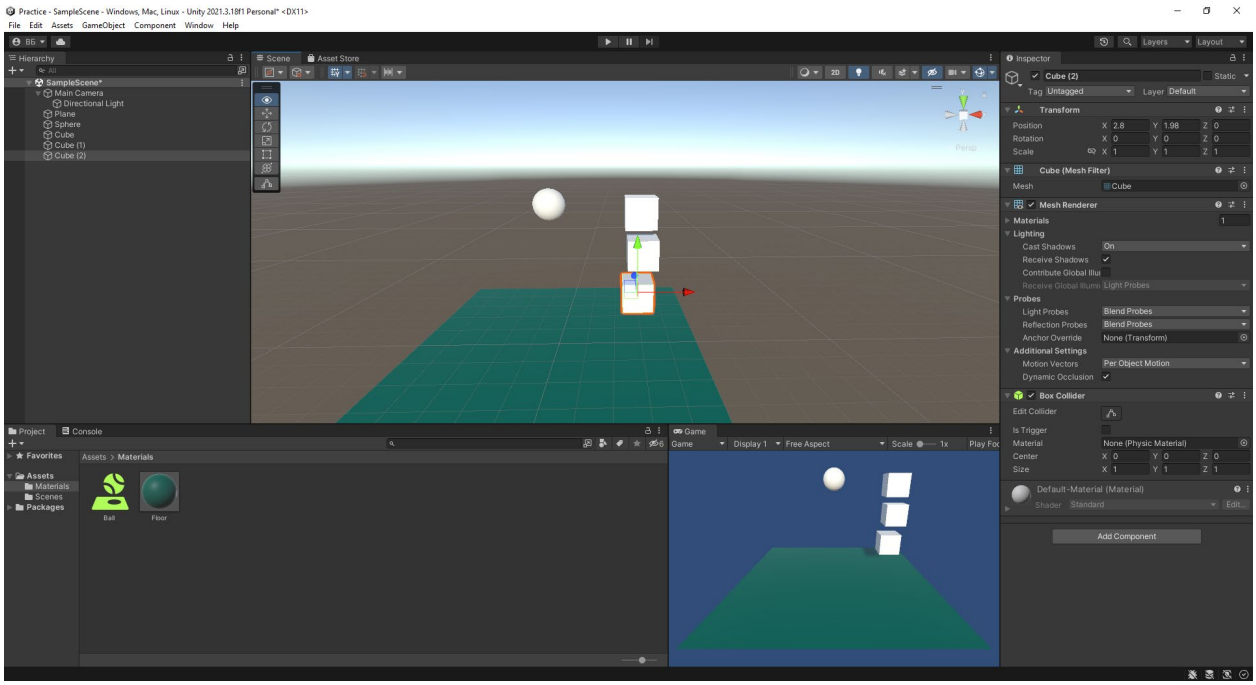


Рисунок 24 – Добавление несколько кубов на сцену

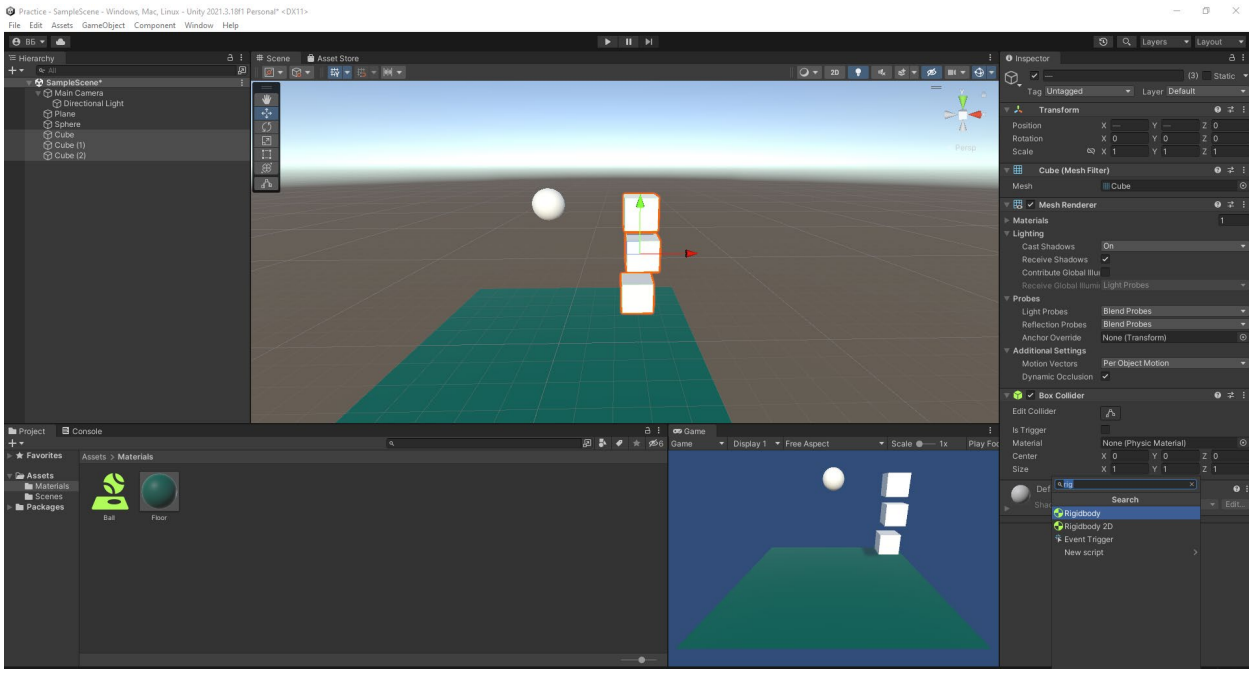


Рисунок 25 – Добавление к кубам компонента RigidBody

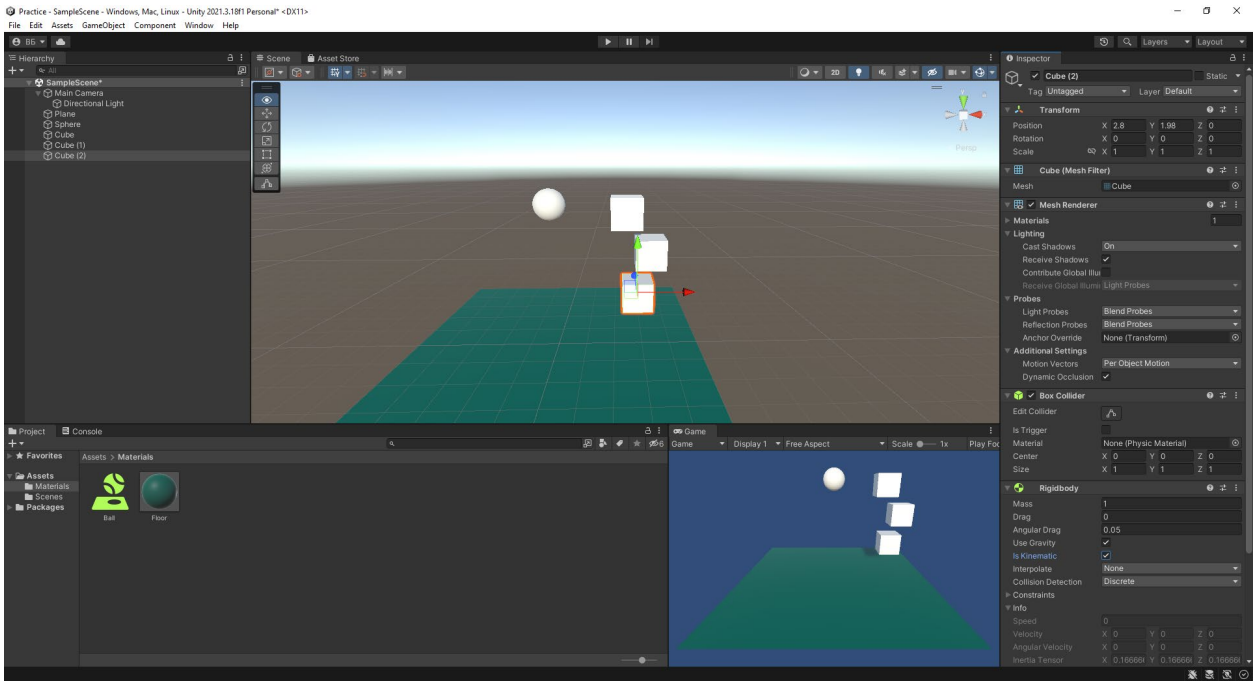


Рисунок 26 – Выравнивание кубов и изменение свойств RigidBody

Индивидуальное задание

1. Создайте новый проект в Unity и спроектируйте сцену таким образом, чтобы на ней были следующие элементы:
   - пол;
   - круглый объект;
   - квадратный объект.
2. Задействуйте игровую физику. Например, шар, который сбивает композицию из кубов (имитация дорожки в боулинге) и т.д.
3. Создайте платформу с препятствиями, реализуйте механику вращения платформы по нажатию на определенные кнопки. Реализуйте «конец игры».

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого пункта подраздела «Индивидуальное задание» с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

4. Экранные формы (консольный вывод) и листинг программного кода с комментариями, показывающие порядок выполнения лабораторной работы, и результаты, полученные в ходе её выполнения.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Что из перечисленного является специализированным игровым движком: UnrealEngine, Godot, Constract3, Microsoft Word, CCleaner, Unity?
2. Что такое коллизия (collision)?
3. Что такое RigidBody?
4. Для чего в компоненте RigidBody используется свойство isKinematic?

Список литературы

Для выполнения лабораторной работы, при подготовке к защите, а также для ответа на контрольные вопросы рекомендуется использовать следующие источники: [1-4].

ЛАБОРАТОРНАЯ РАБОТА 2. ПОЛУЧЕНИЕ ИНФОРМАЦИИ И ПРИНЯТИЕ РЕШЕНИЙ

Цели и задачи

Цель лабораторной работы: приобретение практических навыков работы со сценой и игровыми объектами; приобретение практических навыков реализации отслеживания персонажа другим объектом.

Основные задачи:

- познакомиться с инструментарием игрового движка Unity;
- научиться создавать скрипты;
- научиться передвигать персонажа по сцене с помощью кнопок;
- научиться поворачивать объекты в сторону персонажа.

Теоретическое обоснование

Перед выполнением практической работы необходимо ознакомиться с базовыми принципами работы с Unity.

Основные горячие клавиши:

Q	Pan (перемещение камеры сцены)
W	Move (перемещение)
E	Rotate (вращение)
R	Scale (масштабирование)
CTRL+D	Дублировать
CTRL+SHIFT+F	Выровнять по виду
Зажатая ПКМ	Вращение на сцене
Зажатое колесико мыши	Перемещение по сцене

С расширенным списком горячих клавиш вы можете ознакомиться по ссылке: https://docs.unity3d.com/ru/530/uploads/Main/Unity_HotKeys_Win.pdf

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько програмных продуктов для практической реализации задач лабораторной работы): MS Visual Studio 2019 и выше; игровой движок Unity или Unreal Engine.

Методика и порядок выполнения работы

Перед выполнением индивидуального задания настоятельно рекомендуется ознакомиться с учебной задачей.

Проект учебной задачи находится по ссылке: https://github.com/RedInHat/DPO_practices/blob/main/practice2.unitypackage

Отдельно скрипты вы можете скачать здесь: https://github.com/RedInHat/DPO_practices/tree/main/Scripts

Учебная задача

1. Откройте Unity Hub, выберите вкладку projects и создайте новый проект.
2. Измените у Main Camera следующие свойства.

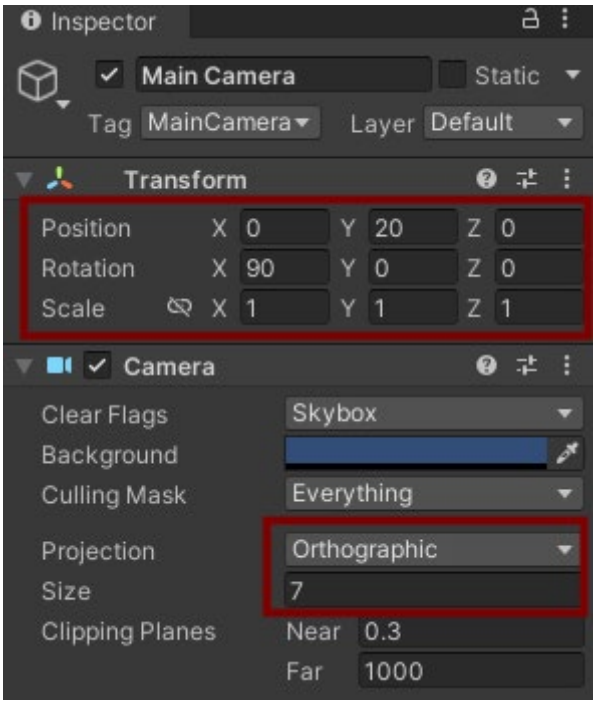


Рисунок 3 – Main Camera

3. Добавьте примитив Plane на сцену.

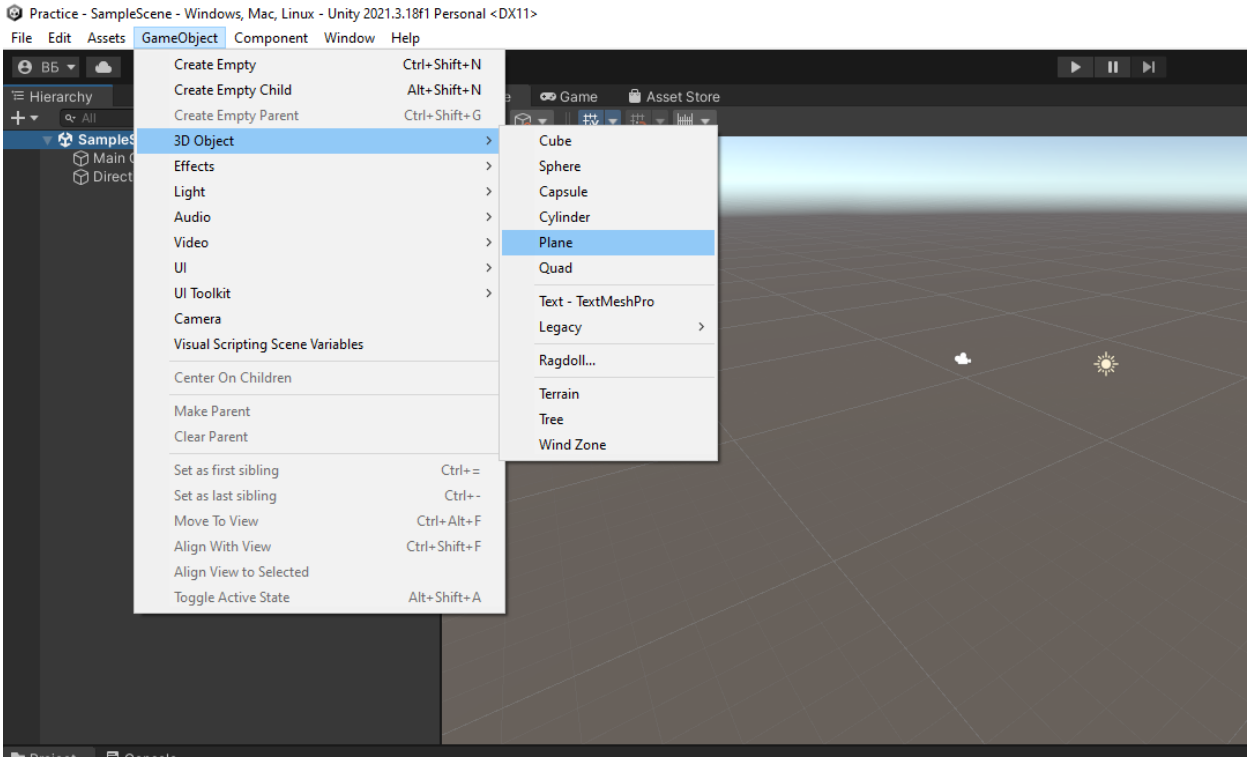


Рисунок 4 – Добавление примитива Plane

Измените значения Plane таким образом, чтобы в окне Game было видно только Plane.

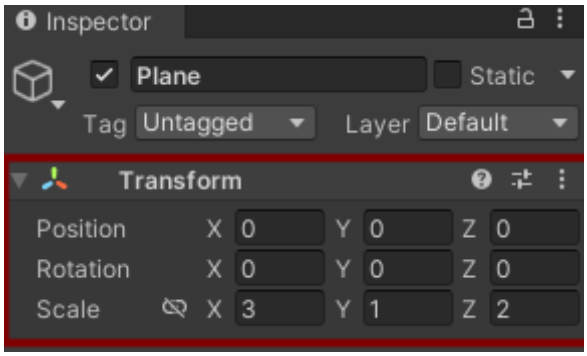


Рисунок 5 – Plane

4. Измените цвет Plane через материал. В окне Project создайте пустую папку и назовите ее Materials. В папке Materials создайте новый материал.

Выделите материал в окне Project и в Inspector поменяйте цвет.

Перетащите материал на Plane.

5. Добавьте на сцену примитивы Cube и разместите их таким образом, чтобы в окне Game был виден только край кубов (создайте стены и разместите их по периметру зоны видимости камеры).

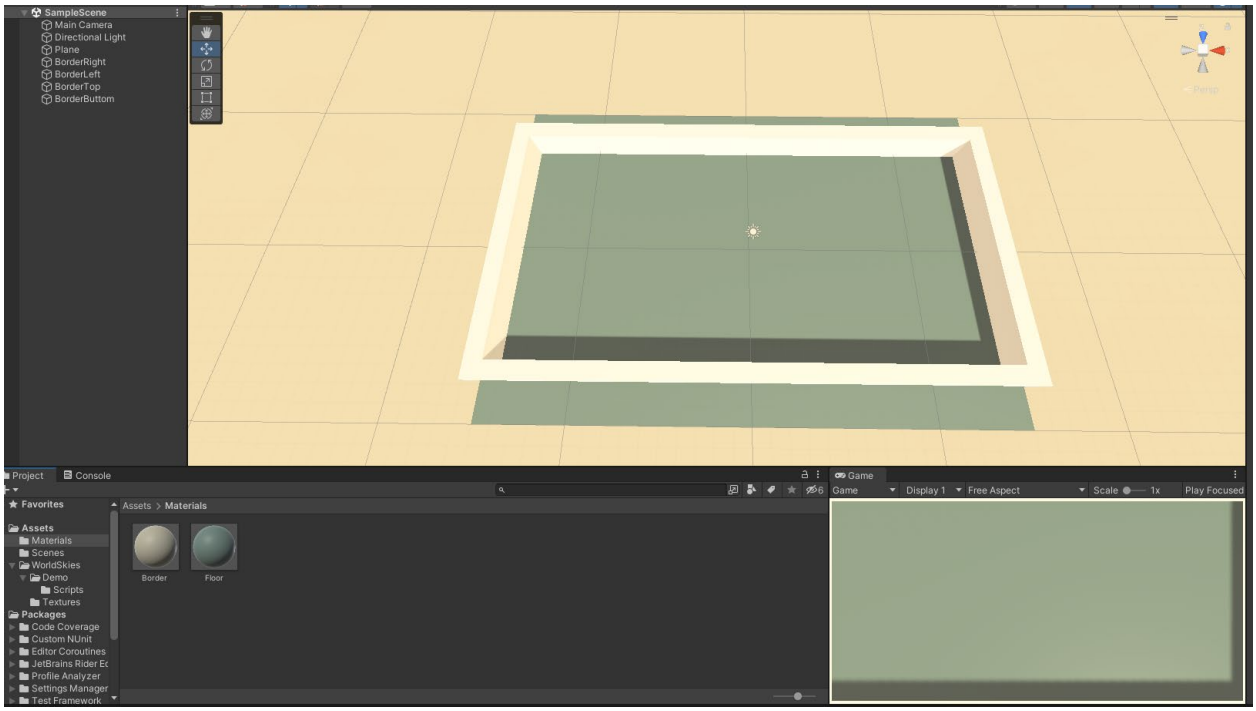


Рисунок 10– Периметр

6. Добавьте персонажа. Для этого создайте примитив Capsule. Нареките капсулу гордым именем Player.

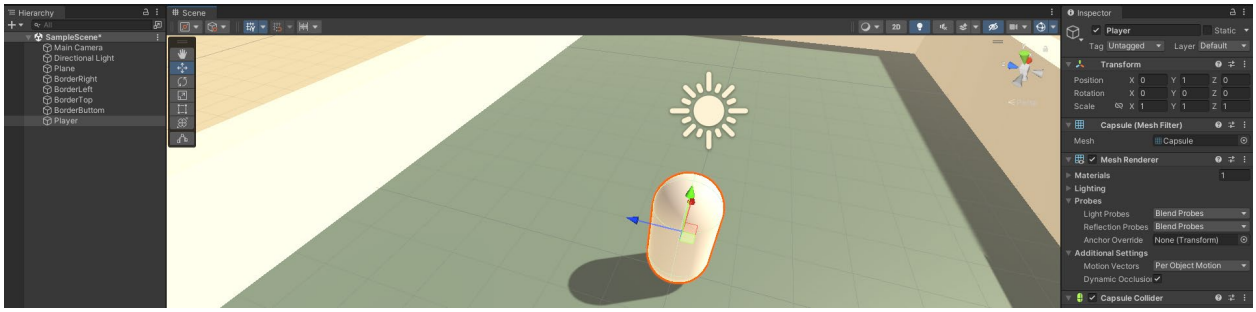


Рисунок 12 – Capsule aka Player

Давайте добавим нашему персонажу лицо. Для этого добавьте какой-нибудь примитив (например, Cube) и сделайте его дочерним объектом Player. Для этого достаточно перетащить Face в Player в окне иерархии. Разместите «лицо» персонажа в, по вашему мнению, анатомически правильном месте. Но, учитывайте тот факт, что, для корректной работы скриптов, «лицевая» часть персонажа расположена вдоль оси Z (куда стремится синяя стрелка).

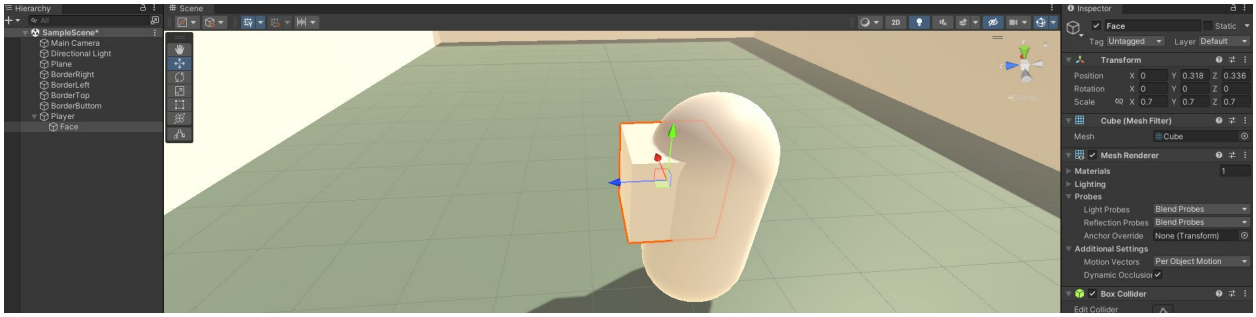


Рисунок 13 – Экспонат №71 «Персонаж обретает лицо»

7. Пора научить нашего персонажа подчиняться нашей воле. Для этого нам понадобится скрипт. Создайте в окне Project в папке Assets новую папку Scripts. В этой папке создайте C# Script и назовите его, например, PlayerController.

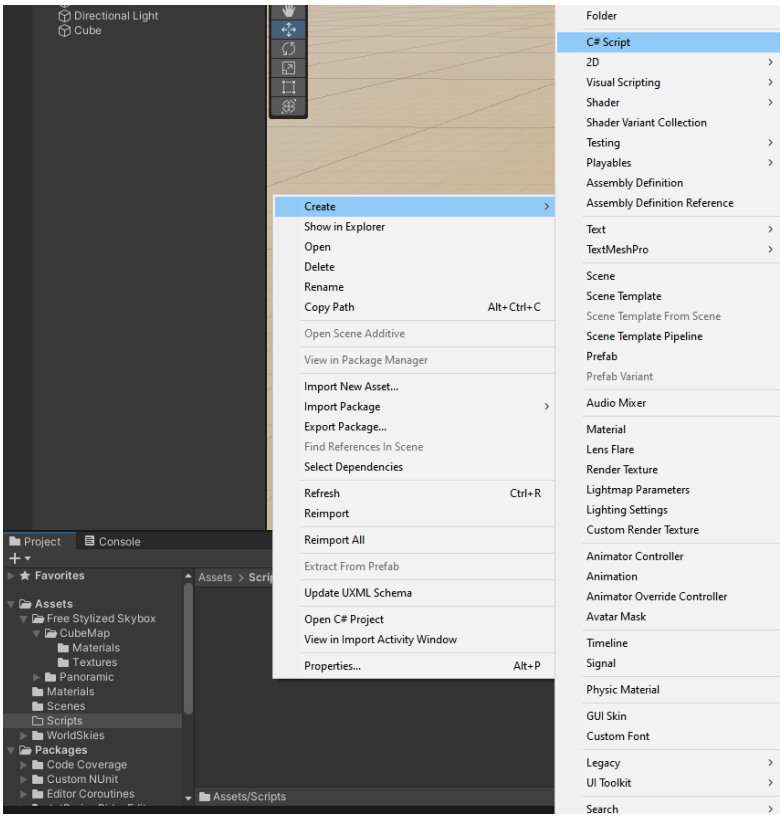
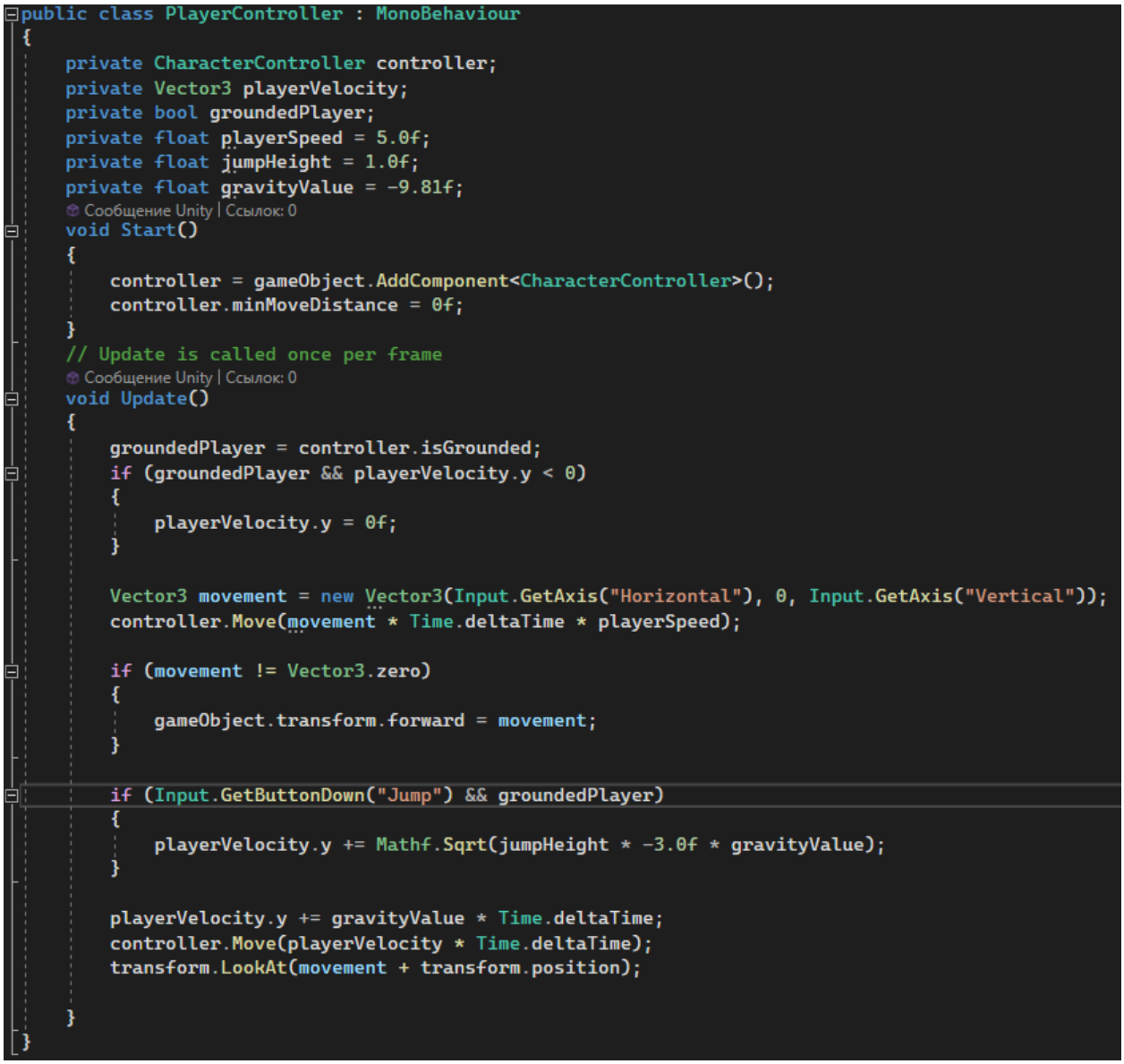


Рисунок 14 – Создание скрипта

Перейдите по Edit -> Preferences -> External Tools и удостоверьтесь, что в External Script Editor выбрана Visual Studio.

После этого откройте скрипт и добавьте следующий код:



```
public class PlayerController : MonoBehaviour
{
    private CharacterController controller;
    private Vector3 playerVelocity;
    private bool groundedPlayer;
    private float playerSpeed = 5.0f;
    private float jumpHeight = 1.0f;
    private float gravityValue = -9.81f;
    Сообщение Unity | Ссылок: 0
    void Start()
    {
        controller = gameObject.AddComponent<CharacterController>();
        controller.minMoveDistance = 0f;
    }
    // Update is called once per frame
    Сообщение Unity | Ссылок: 0
    void Update()
    {
        groundedPlayer = controller.isGrounded;
        if (groundedPlayer && playerVelocity.y < 0)
        {
            playerVelocity.y = 0f;
        }

        Vector3 movement = new Vector3(Input.GetAxis("Horizontal"), 0, Input.GetAxis("Vertical"));
        controller.Move(movement * Time.deltaTime * playerSpeed);

        if (movement != Vector3.zero)
        {
            gameObject.transform.forward = movement;
        }

        if (Input.GetButtonDown("Jump") && groundedPlayer)
        {
            playerVelocity.y += Mathf.Sqrt(jumpHeight * -3.0f * gravityValue);
        }

        playerVelocity.y += gravityValue * Time.deltaTime;
        controller.Move(playerVelocity * Time.deltaTime);
        transform.LookAt(movement + transform.position);

    }
}
```

Рисунок 16 – Листинг

Сохраните скрипт и перетащите его на игрока из окна Project. Запустите проект и проверьте работает ли он вообще.

8. Добавьте Spotlight на сцену. Разместите спот в центре экрана и добейтесь имитации прожектора. Выключите Directional Light (снимите флажок в инспекторе в самом верху рядом с именем.

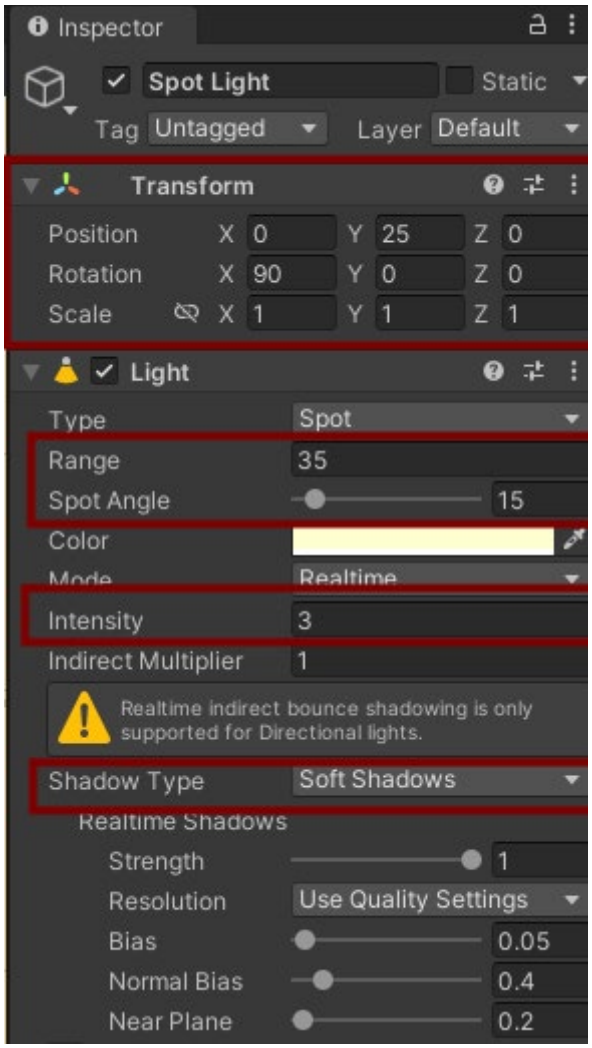


Рисунок 19 – Свойства компонентов Spotlight

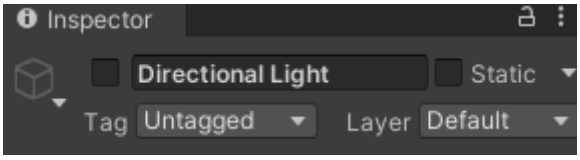


Рисунок 20 – Выключение Directional Light

Теперь добавим отслеживание игрока. Создайте новый скрипт и добавьте код.

```
public class PlayerTracking : MonoBehaviour
{
    [SerializeField] private Transform player;
    // Start is called before the first frame update
    Сообщение Unity | Ссылок: 0
    void Start()
    {
        player = GameObject.FindGameObjectWithTag("Player").transform;
    }

    // Update is called once per frame
    Сообщение Unity | Ссылок: 0
    void Update()
    {
        transform.LookAt(player);
    }
}
```

Рисунок 21– Листинг

Добавьте игроку тег «Player» и повесьте скрипт на Spotlight.

Если всё сделано так, как задумывалось, то, при передвижении игрока, прожектор будет следовать за персонажем.

Рисунок 26 – Результат

Рисунок 27 – Результат

Индивидуальное задание

1. Создайте новый проект в Unity и спроектируйте сцену таким образом, чтобы на ней были следующие элементы:
   - пол;
   - стены;
   - персонаж;
   - прожектор.
2. Реализуйте управление персонажем по нажатию соответствующих кнопок. Добавьте отслеживание игрока прожектором.
3. Создайте лабиринт, из которого персонаж должен выбраться. Реализуйте следующие механики:

- Игрок сначала должен найти кнопку, которая откроет какую-то дверь.
- Если персонаж попадёт в свет прожектора, то все прожекторы начинают отслеживать положение персонажа.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого пункта подраздела «Индивидуальное задание» с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

4. Экранные формы (консольный вывод) и листинг программного кода с комментариями, показывающие порядок выполнения лабораторной работы, и результаты, полученные в ходе её выполнения.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Что из перечисленного является специализированным игровым движком: UnrealEngine, Godot, Constract3, Microsoft Word, CCleaner, Unity?
2. Что такое коллизия (collision)?
3. Что такое скрипт?
4. Для чего в компоненте Collider используется свойство isTrigger?

Список литературы

Для выполнения лабораторной работы, при подготовке к защите, а также для ответа на контрольные вопросы рекомендуется использовать следующие источники: [1-4].

ЛАБОРАТОРНАЯ РАБОТА 3. ПЕРЕДВИЖЕНИЕ В ПРОСТРАНСТВЕ

Цели и задачи

Цель лабораторной работы: приобретение практических навыков работы со сценой и игровыми объектами; приобретение практических навыков реализации отслеживания персонажа другим объектом.

Основные задачи:

- познакомиться с инструментарием игрового движка Unity;
- научиться создавать скрипты;
- научиться поворачивать объекты в сторону персонажа;
- научиться реализовывать преследование персонажа.

Теоретическое обоснование

Перед выполнением практической работы необходимо ознакомиться с базовыми принципами работы с Unity.

Основные горячие клавиши:

Q	Pan (перемещение камеры сцены)
W	Move (перемещение)
E	Rotate (вращение)
R	Scale (масштабирование)
CTRL+D	Дублировать
CTRL+SHIFT+F	Выровнять по виду
Зажатая ПКМ	Вращение на сцене
Зажатое колесико мыши	Перемещение по сцене

С расширенным списком горячих клавиш вы можете ознакомиться по ссылке: https://docs.unity3d.com/ru/530/uploads/Main/Unity_HotKeys_Win.pdf

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько програмных продуктов для практической реализации задач лабораторной работы): MS Visual Studio 2013 и выше; среда разработки Java, интерпретатор Python (Anaconda).

Методика и порядок выполнения работы

Перед выполнением индивидуального задания настоятельно рекомендуется ознакомиться с учебной задачей.

Проект учебной задачи находится по ссылке: https://github.com/RedInHat/DPO_practices/blob/main/practice3.unitypackage

Отдельно скрипты вы можете скачать здесь: https://github.com/RedInHat/DPO_practices/tree/main/Scripts

Учебная задача

1. Откройте Unity Hub, выберите вкладку projects и создайте новый проект.
2. Разместите Plane на сцене. Увеличьте объект (например, (2,1,2)).

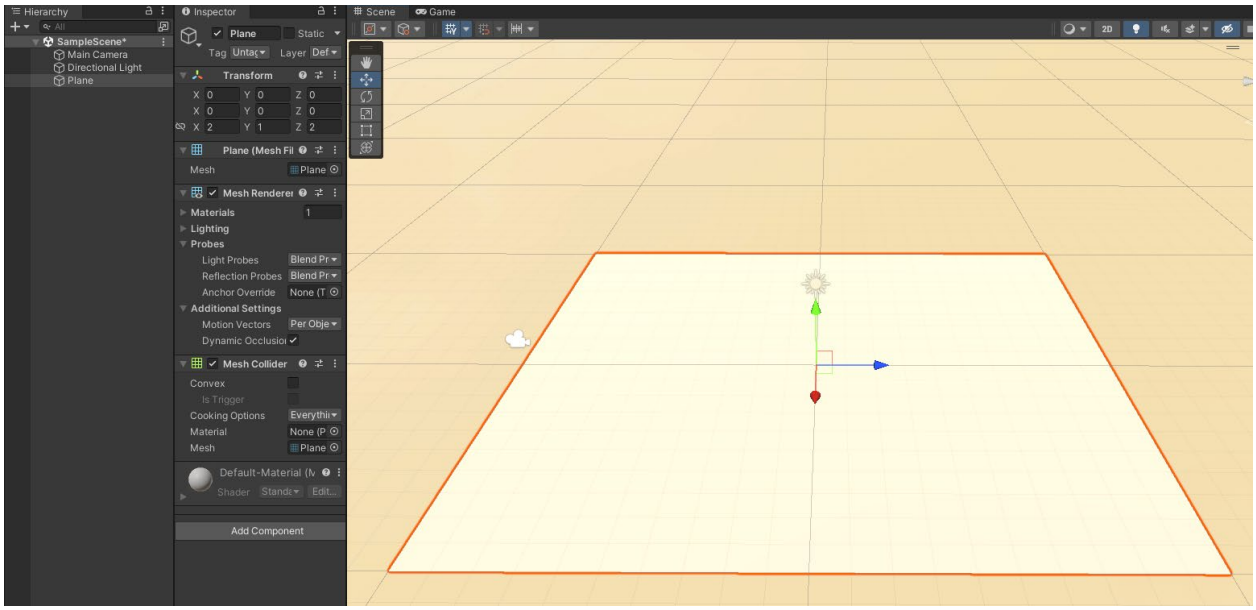


Рисунок 3 – Plane

3. Добавьте персонажа и создайте скрипт для управления. Как и в предыдущих работах для простоты будем использовать объект Capsule. Для понимая того, где находится лицо персонажа, разместите дочерний объект Cube. Лицо персонажа должно располагаться в направлении оси Z (синяя стрелочка). Добавьте игроку тег «Player».

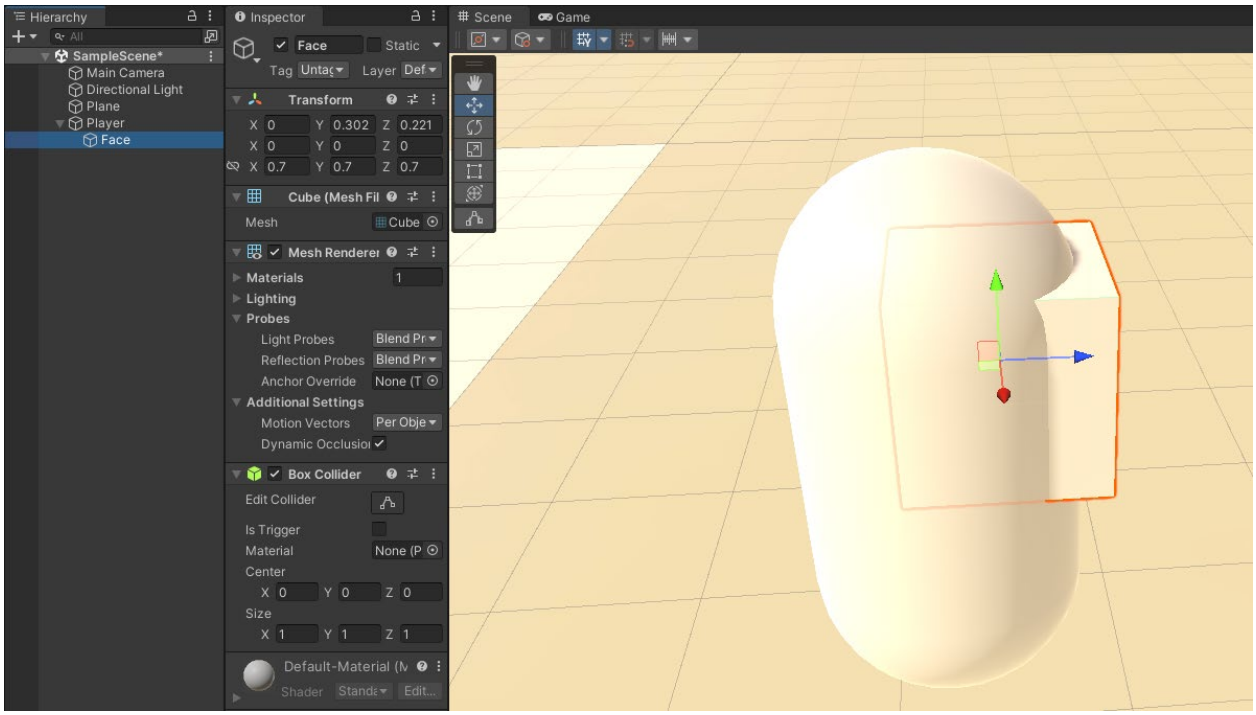


Рисунок 4 – Персонаж

Добавьте скрипт управления персонажем и перетащите его на капсулу.

Рисунок 5 – компонент Player Movement

```
using UnityEngine;

Скрипт Unity | Ссылок: 0
public class PlayerMovement : MonoBehaviour
{
    [SerializeField] private float _playerSpeed = 5.0f;
    [SerializeField] private float _jumpHeight = 1.0f;

    private readonly float _gravityValue = -9.81f;

    private CharacterController _controller;
    private Vector3 _playerVelocity;
    private bool _groundedPlayer;


    Сообщение Unity | Ссылок: 0
    void Start()
    {
        _controller = gameObject.AddComponent<CharacterController>();
        _controller.minMoveDistance = 0f;
    }

    Сообщение Unity | Ссылок: 0
    void Update()
    {
        _groundedPlayer = _controller.isGrounded;

        if (_groundedPlayer && _playerVelocity.y < 0)
        {
            _playerVelocity.y = 0f;
        }

        Vector3 movement = new Vector3(Input.GetAxis("Horizontal"), 0, Input.GetAxis("Vertical"));
        _controller.Move(movement * Time.deltaTime * _playerSpeed);

        if (movement != Vector3.zero)
        {
            gameObject.transform.forward = movement;
        }

        if (Input.GetButtonDown("Jump") && _groundedPlayer)
        {
            _playerVelocity.y += Mathf.Sqrt(_jumpHeight * -3.0f * _gravityValue);
        }

        _playerVelocity.y += _gravityValue * Time.deltaTime;
        _controller.Move(_playerVelocity * Time.deltaTime);
        transform.LookAt(movement + transform.position);

    }
}
```

Рисунок 6 – Листинг

4. Создайте врага. Добавьте ему компонент NavMeshAgent. Добавьте тег «Enemy».

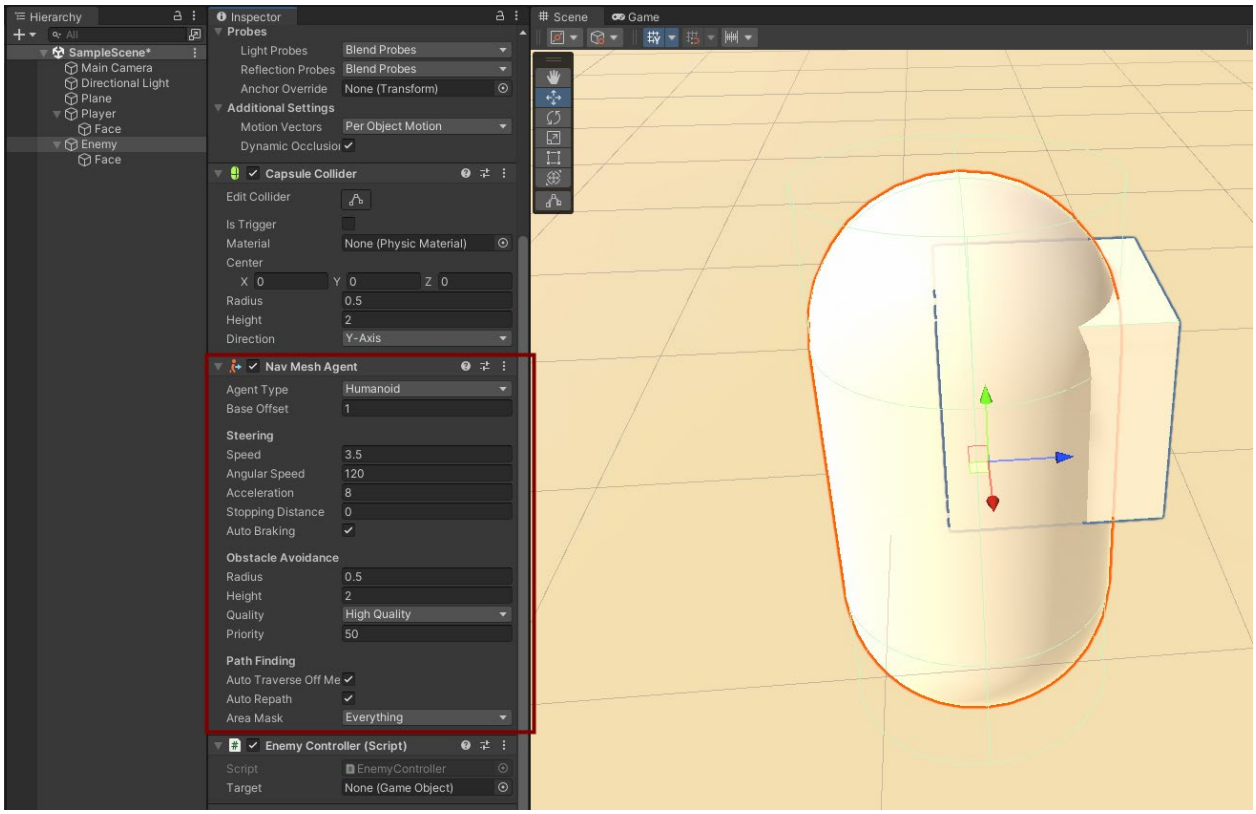


Рисунок 7 – Enemy

```
using UnityEngine;
using UnityEngine.AI;

Скрипт Unity | Ссылок: 0
public class EnemyController : MonoBehaviour
{
    [SerializeField] private GameObject _target;
    private NavMeshAgent _agent;
    Сообщение Unity | Ссылок: 0
    void Start()
    {
        _target = GameObject.FindGameObjectWithTag("Player");
        _agent = GetComponent<NavMeshAgent>();
    }


    Сообщение Unity | Ссылок: 0
    void Update()
    {
        _agent.transform.LookAt(_target.transform);
        _agent.SetDestination(_target.transform.position);
    }
}
```

Рисунок 8 – Листинг

5. Добавьте цвета и разместите персонажа и врага на Plane. В моей случае, желтый – это игрок, розовый – это враг.

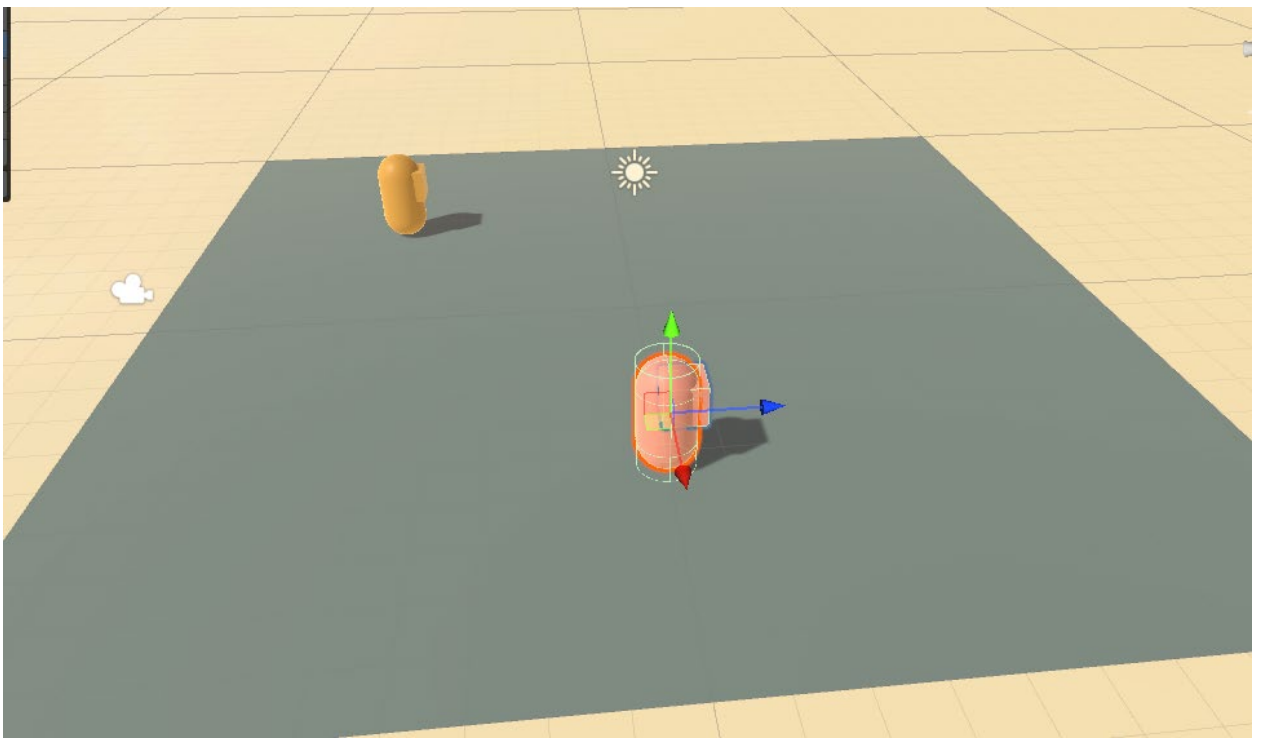


Рисунок 9 – Оцветление

Запустите и проверьте работу. На данный момент мы можем управлять персонажем, а вот с врагом есть трудности. Всё, что сейчас может враг – поворачиваться в сторону игрока, хотя в логике его скрипта уже есть возможность бежать за персонажем. Вот здесь:

```
Сообщение Unity | Ссылок: 0
void Update()
{
    _agent.transform.LookAt(_target.transform);
    _agent.SetDestination(_target.transform.position);
}
```

Рисунок 10 – Фрагмент кода

В метод SetDestination передаются координаты игрока, и враг стремится их занять, т.е. догнать нашего игрока. Но, по факту, этого не происходит. Чтобы все заработало перейдите Windows->AI->Navigation. Именно это окно позволяет настроить поведение врагов.

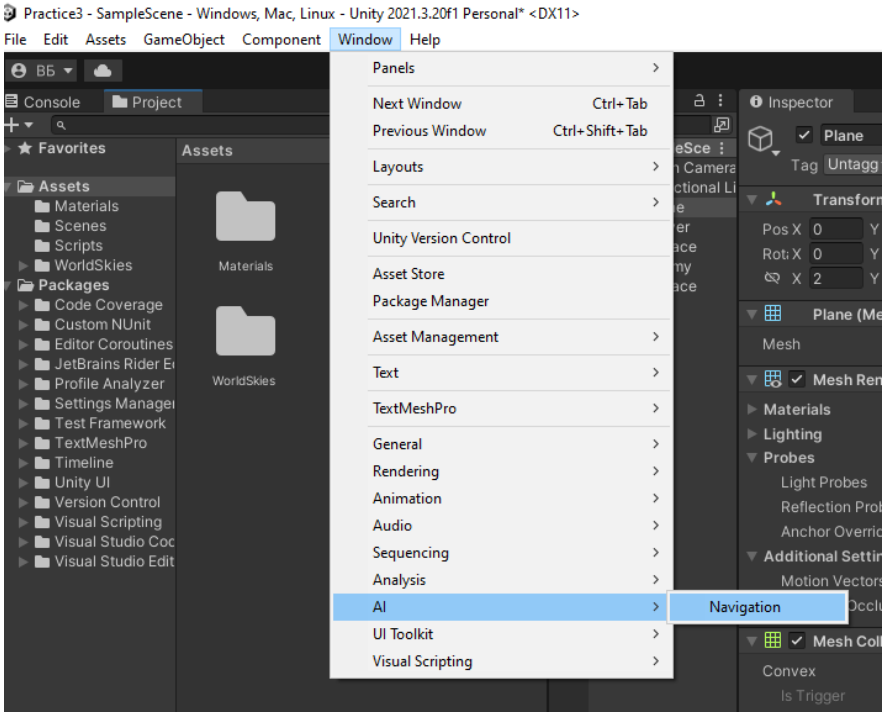


Рисунок 11 – Окно Navigation

Врагу нужно по чему-то перемещаться, поэтому в иерархии объектов выберите Plane и перейдите в окно Navigation. Здесь можно настроить агента, но в данный момент нас интересует вкладка Object. Отметьте Plane как

статический объект. Затем перейдите во вкладку Bake и нажмите на кнопку Bake.

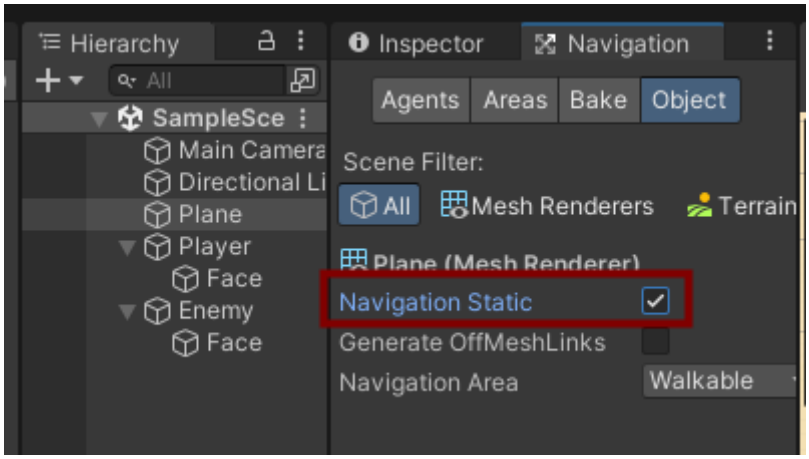


Рисунок 12 – Вкладка Object

Если всё сделано правильно, то на Plane появится голубая область, которая будет показывать, где сможет перемещаться наш агент (враг).

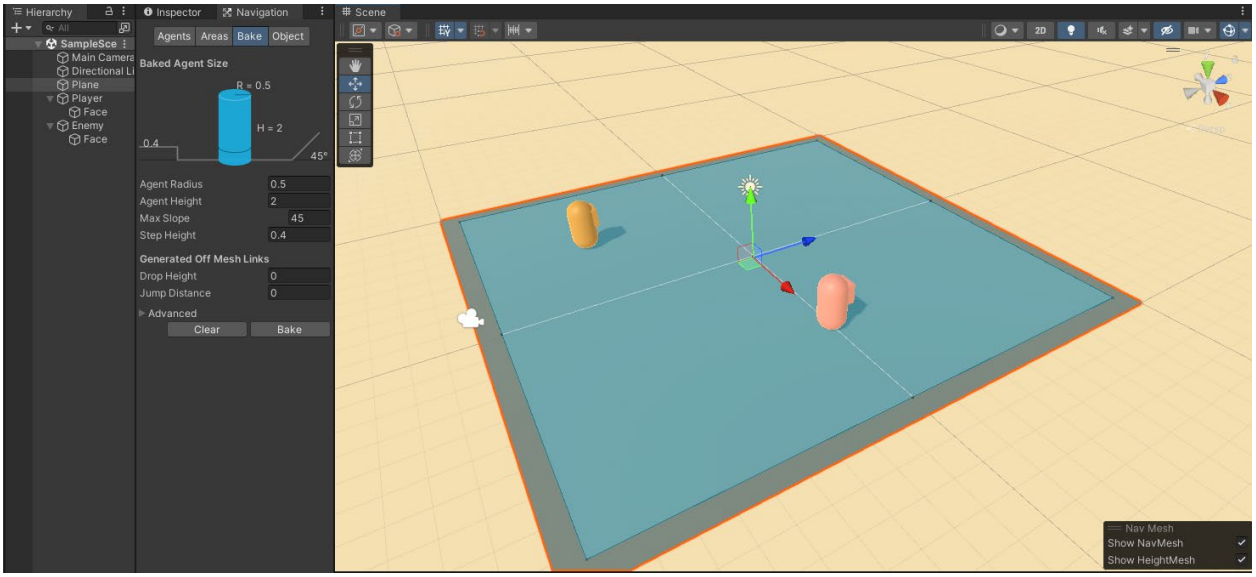


Рисунок 13 – Вкладка Bake

Теперь, если вы запустите проект, то враг будет бежать к персонажу и даже будет его толкать, потому что остановка не предусмотрена.

6. Давайте создадим две платформы (на одной враг сможет нас достичь, а на другой – нет).

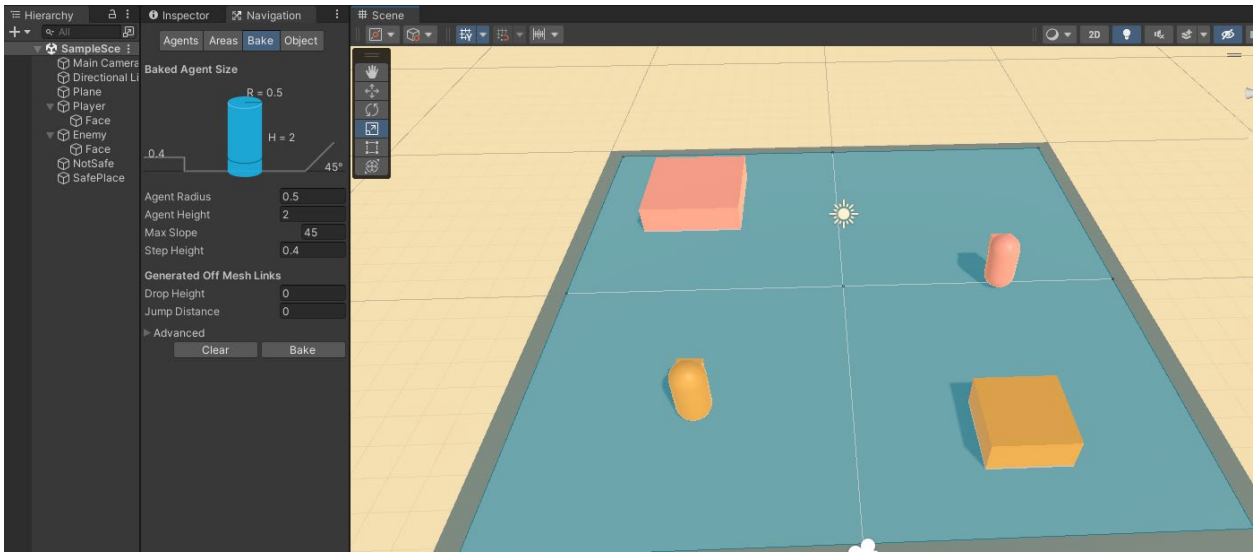


Рисунок 14 – Вкладка Bake

Желтая платформа – безопасная, розовая – нет.

Эти платформы нужно отметить, как статические объекты (пункт 5 для Plane). С той лишь разницей, что для желтой платформы нужно в NavigationArea указать NotWalkable

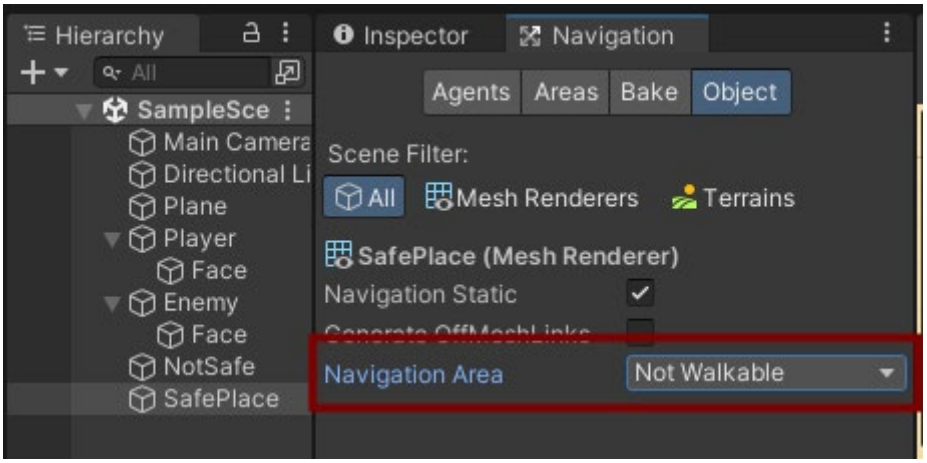


Рисунок 15 – Вкладка Object

И все это снова запечь во вкладке Bake. Если всё сделано правильно, то получится вот такая область.

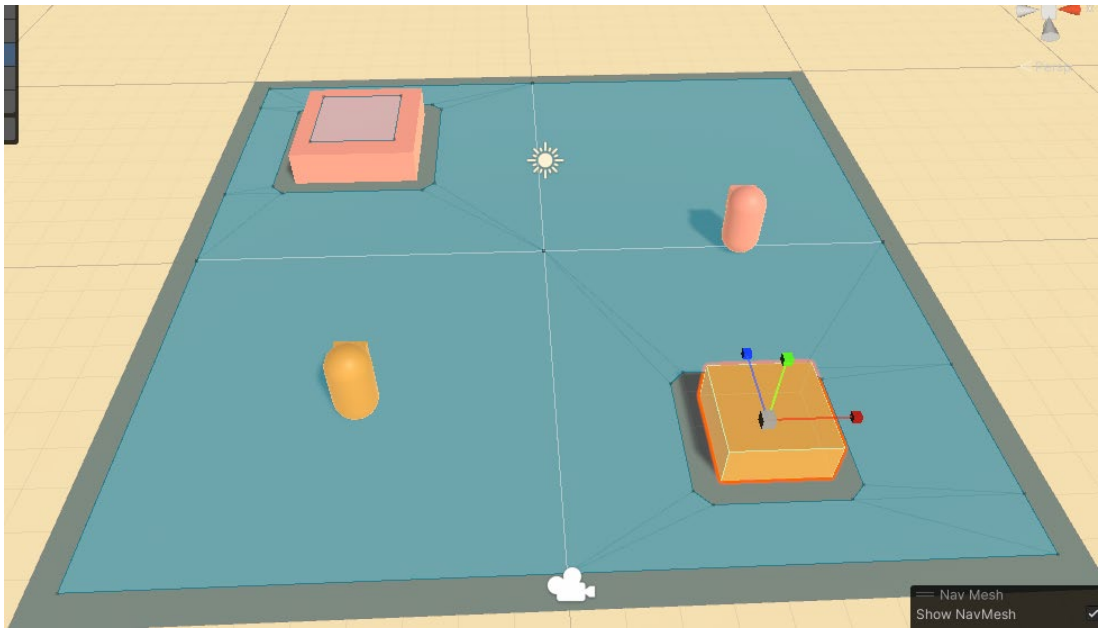


Рисунок 16 – Область навигации агента

Но, если мы сейчас это протестируем, то враг всё равно не может достать нас на розовом кубе.

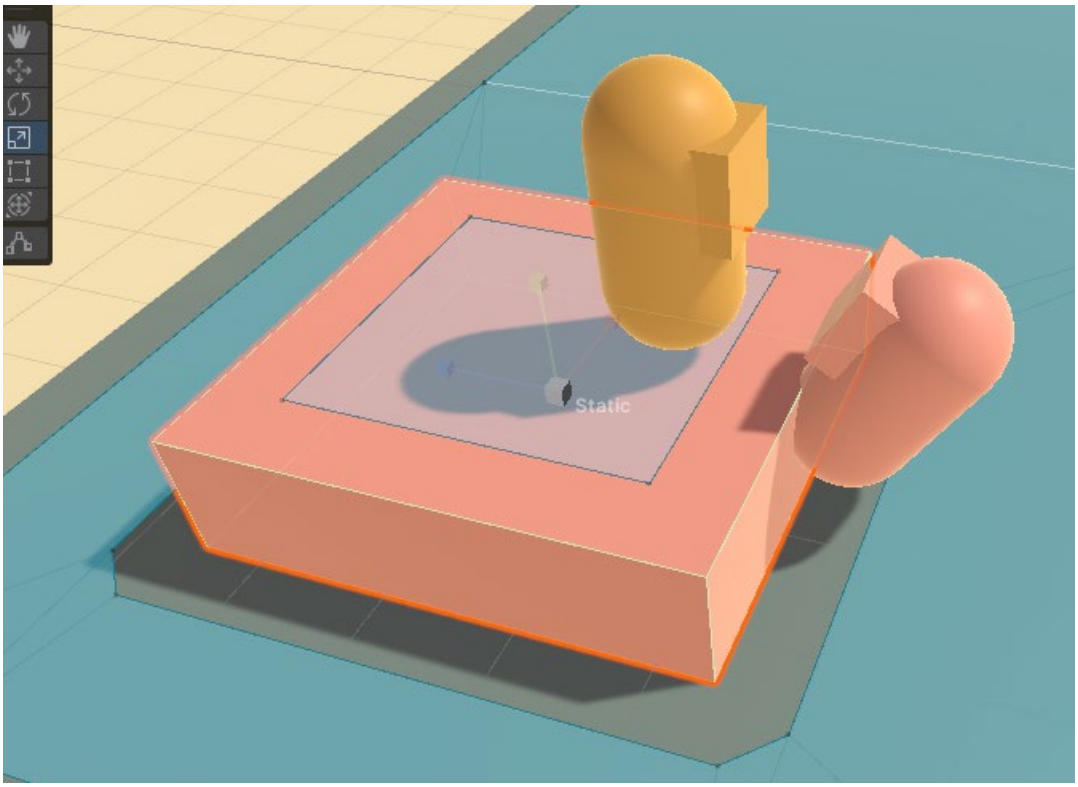


Рисунок 17 – Платформа

Происходит это потому, что платформа слишком высокая для врага. Можно либо сделать платформу ниже, либо изменить высоту шага агента. Для этого можно перейти во вкладку Bake и Step Height поставить 1 вместо 0.4. И снова это запечь. Не забудьте выбрать нужную платформу в иерархии. И теперь на этой платформе от врага не спастись.

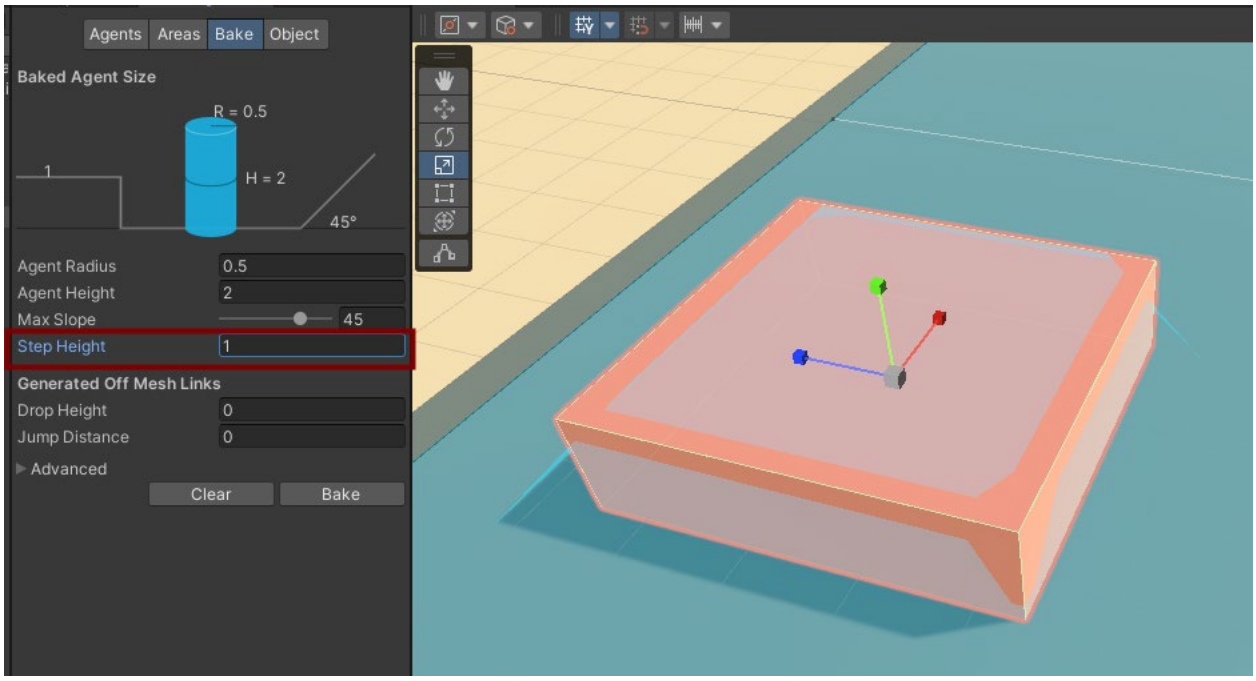


Рисунок 18 – Платформа

Давайте научим нашего врага атаковать персонажа. Реализуем простую механику: при столкновении с коллайдером врага, у персонажа будет отниматься здоровье. В качестве здоровья будет выступать цвет. С каждым ударом игрок будет становиться всё светлее.

Также, раздробим логику на несколько скриптов:

– Health – здесь будет логика здоровья (какой урон и как наносится);
– Attacker – повесим на объект, который умеет атаковать (наш враг);
– Damageable – повесим на объект, который может получить урон (игрок).

Но перед этим давайте немного изменим арену. Добавим стены и еще какие-нибудь препятствия. Как-нибудь вот так:

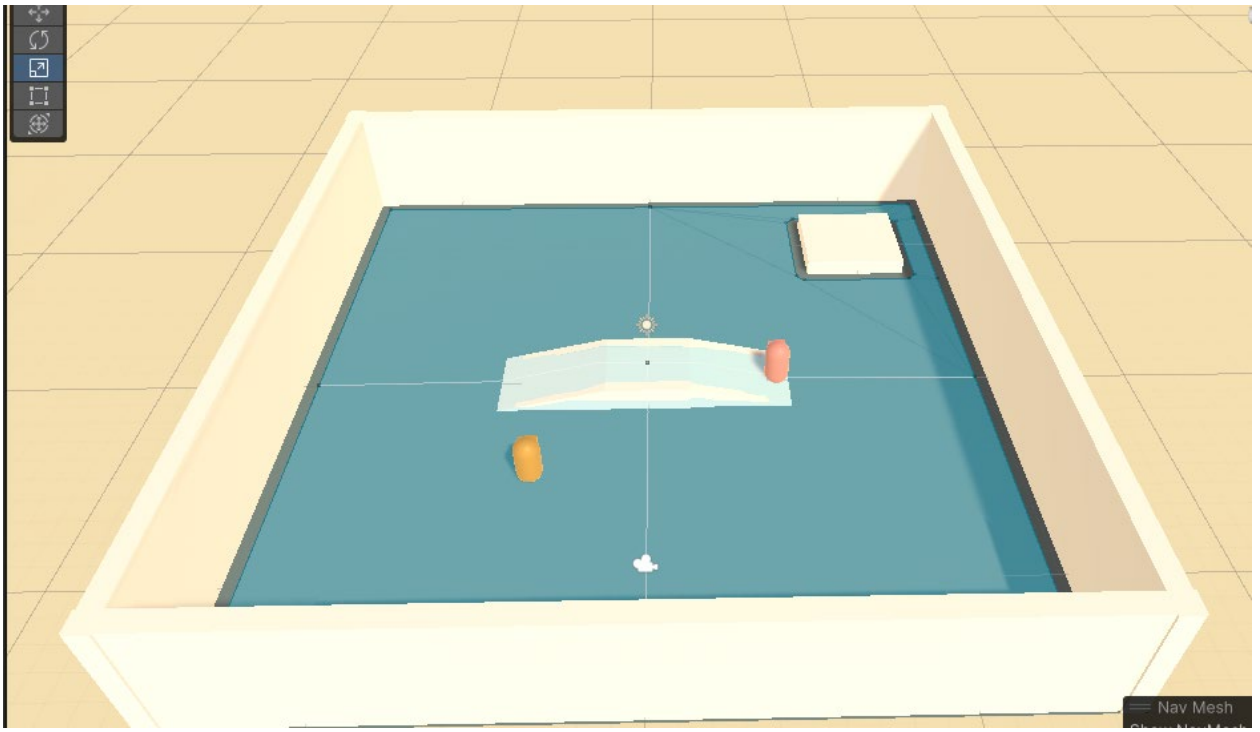


Рисунок 19 – Арена

Не забудьте все изменения запечь в Navigation. Теперь создадим скрипты.

```
using UnityEngine;

Скрипт Unity | Ссылок: 0
public class Attacker : MonoBehaviour
{
    [SerializeField] private float _damage;
    Ссылок: 0
    public float Damage => _damage;
}
```

Рисунок 20 – Attacker

```
using UnityEngine;
using UnityEngine.Events;

Скрипт Unity | Ссылок: 0
public class Damageable : MonoBehaviour
{
    [SerializeField] private UnityEvent<float> DamageGot;
    Сообщение Unity | Ссылок: 0
    private void OnCollisionEnter(Collision collision)
    {

        if (collision.collider.TryGetComponent<Attacker>(out var attacker))
        {
            DamageGot?.Invoke(attacker.Damage);


        }
    }
}
```

Рисунок 21 – Damageable

```
public class Health : MonoBehaviour
{
    [SerializeField] private Material _material;
    private Color _color;

    Сообщение Unity | Ссылок: 0
    private void Start()
    {
        _material = GetComponent<MeshRenderer>().material;
        _color = _material.color;
    }
    Ссылок: 0
    public void GetDamage(float damage)
    {
        _color.r = Mathf.Clamp(_color.r + damage, 0, 1);
        _color.g = Mathf.Clamp(_color.g + damage, 0, 1);
        _color.b = Mathf.Clamp(_color.b + damage, 0, 1);

        _material.color = _color;
        transform.Find("Face").GetComponent<MeshRenderer>().material.color= _color;
    }
}
```

Рисунок 22 – Health

Скрипты Health и Damageable перетащите на игрока. Игроку накиньте RigidBody и заморозьте передвижение и повороты по всем осям. В компоненте Damageable у игрока появилось событие.

Рисунок 23 – Событие

Перетащите Health в поле под Runtime Only и в поле No Function выберите Health.GetDamage.

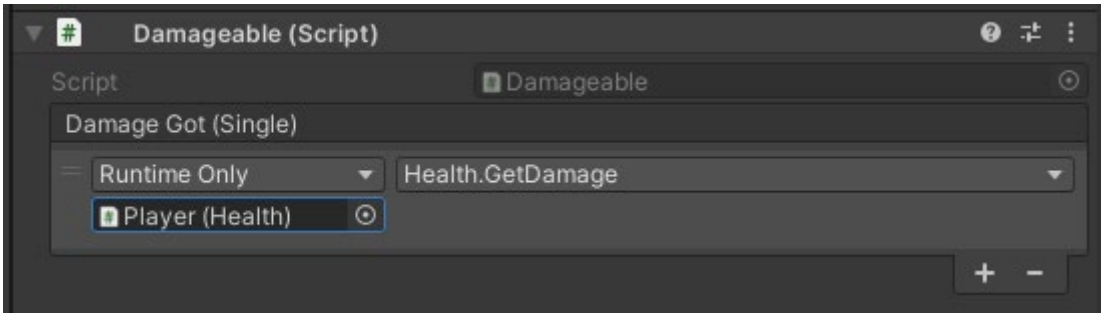


Рисунок 24 – Событие

На врага накиньте скрипт Attacker и не забудьте выставить Damage в инспекторе (0.01).

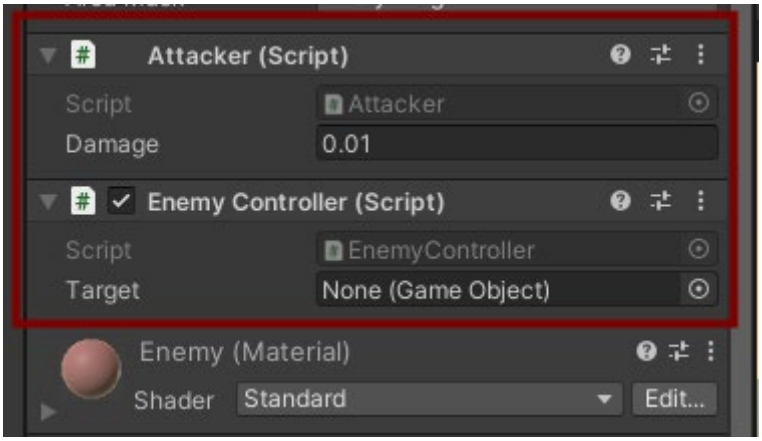


Рисунок 25 – Урон

Теперь при столкновении с врагом у игрока будет прибавляться урон (0.01) к каждому каналу (RGB). (1,1,1) – это белый цвет. Эквивалентно (255,255,255). Таким образом, возникает иллюзия, что, при столкновении с врагом, цвет игрока «вымывается».

Индивидуальное задание

1. Добавьте окружение, персонажа и врага.
2. Реализуйте управление персонажем по нажатию соответствующих кнопок.
3. Реализуйте преследование игрока врагом.
4. Реализуйте атаку персонажа и нанесение урона.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого пункта подраздела «Индивидуальное задание» с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

4. Экранные формы (консольный вывод) и листинг программного кода с комментариями, показывающие порядок выполнения лабораторной работы, и результаты, полученные в ходе её выполнения.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Что из перечисленного является специализированным игровым движком: UnrealEngine, Godot, Constract3, Microsoft Word, CCleaner, Unity?
2. Что такое коллизия (collision)?
3. Что такое скрипт?
4. Для чего в компоненте Collider используется свойство isTrigger?

Список литературы

Для выполнения лабораторной работы, при подготовке к защите, а также для ответа на контрольные вопросы рекомендуется использовать следующие источники: [1-5].

ЛАБОРАТОРНАЯ РАБОТА 4. НАВИГАЦИЯ

Цели и задачи

Цель лабораторной работы: приобретение практических навыков работы со сценой и игровыми объектами; приобретение практических навыков реализации навигации агентов.

Основные задачи:

- познакомиться с инструментарием игрового движка Unity;
- научиться создавать скрипты;
- научиться работать с объектом Terrain;
- научиться реализовывать патрулирование территории.

Теоретическое обоснование

Перед выполнением практической работы необходимо ознакомиться с базовыми принципами работы с Unity.

Основные горячие клавиши:

Q	Pan (перемещение камеры сцены)
W	Move (перемещение)
E	Rotate (вращение)
R	Scale (масштабирование)
CTRL+D	Дублировать
CTRL+SHIFT+F	Выровнять по виду
Зажатая ПКМ	Вращение на сцене
Зажатое колесико мыши	Перемещение по сцене

С расширенным списком горячих клавиш вы можете ознакомиться по ссылке: https://docs.unity3d.com/ru/530/uploads/Main/Unity_HotKeys_Win.pdf

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько програмных продуктов для практической реализации задач лабораторной работы): MS Visual Studio 2013 и выше; среда разработки Java, интерпретатор Python (Anaconda).

Методика и порядок выполнения работы

Перед выполнением индивидуального задания настоятельно рекомендуется ознакомиться с учебной задачей.

Проект учебной задачи находится по ссылке:
https://github.com/RedInHat/DPO_practices/blob/main/practice4.unitypackage

Отдельно скрипты вы можете скачать здесь:
https://github.com/RedInHat/DPO_practices/tree/main/Scripts

Учебная задача

1. Откройте Unity Hub, выберите вкладку projects и создайте новый проект.

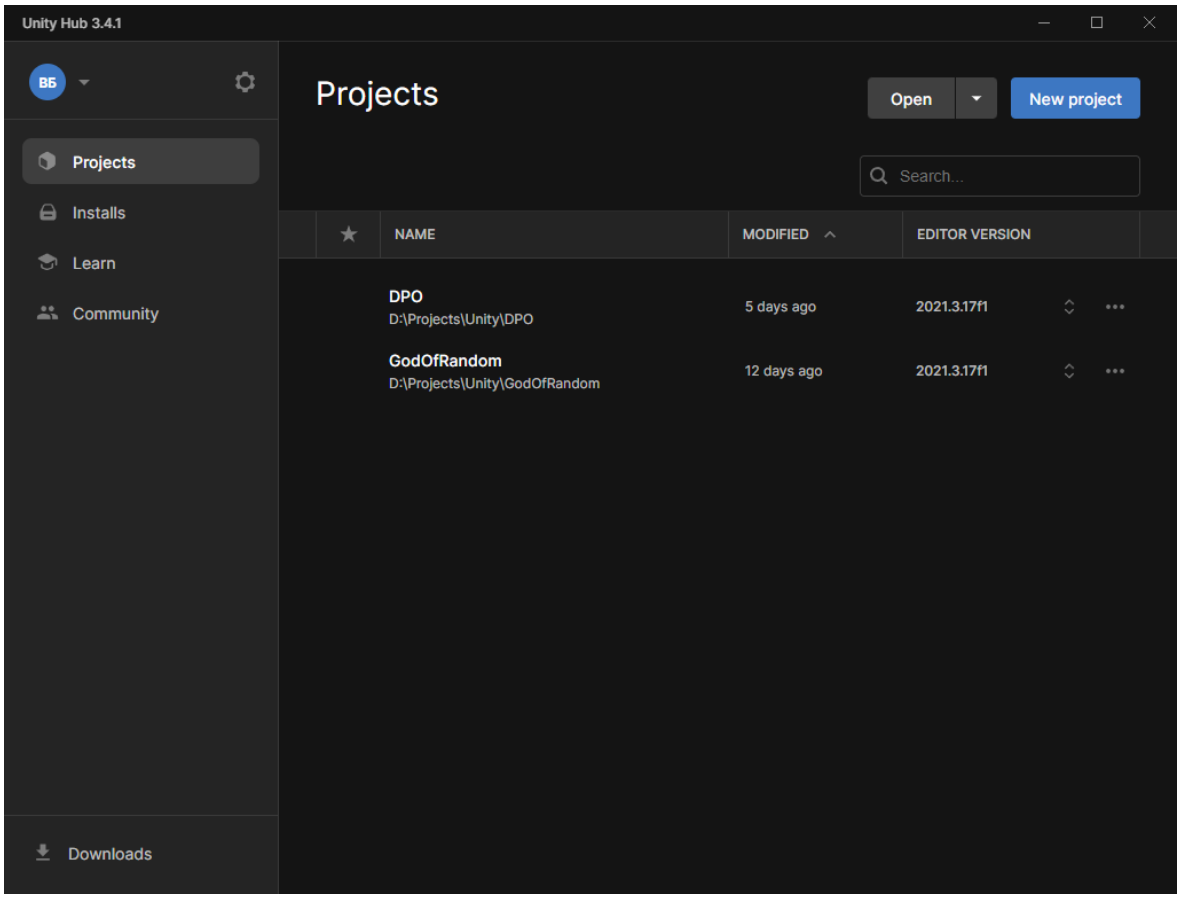


Рисунок 1 – Вкладка Projects в Unity Hub

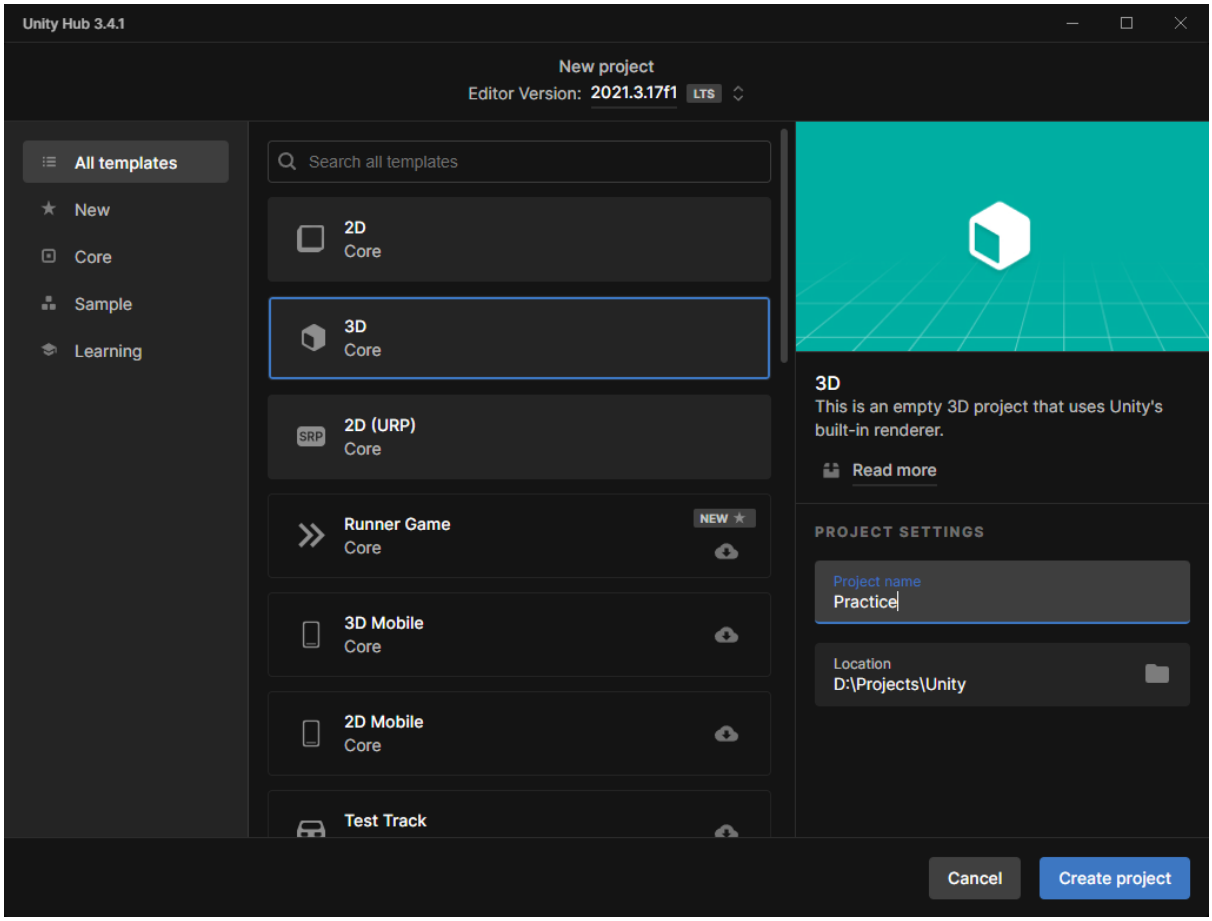


Рисунок 2 – Создание 3d проекта

2. Добавьте на сцену объект Terrain.

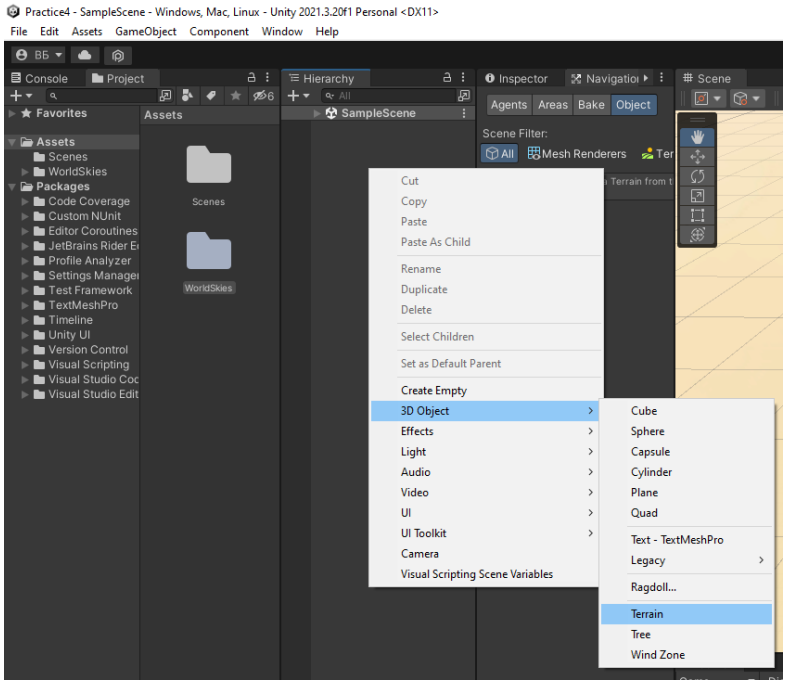
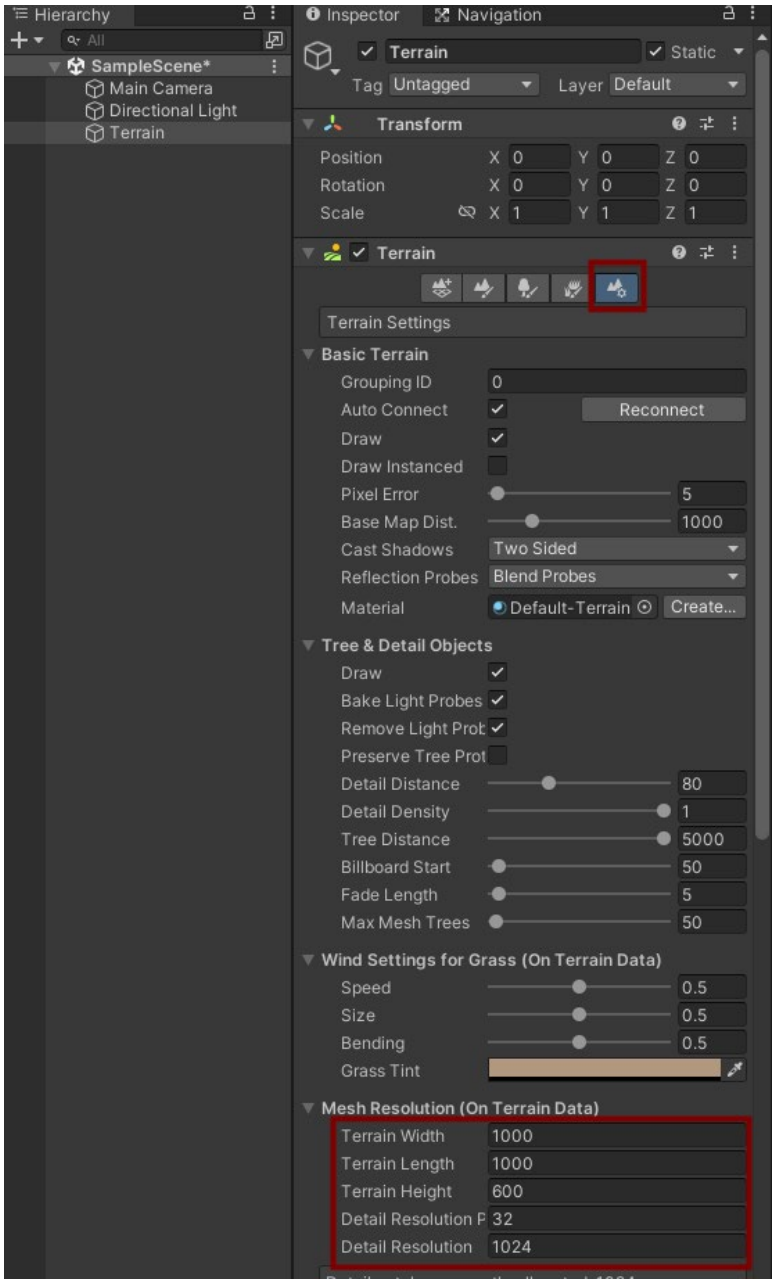


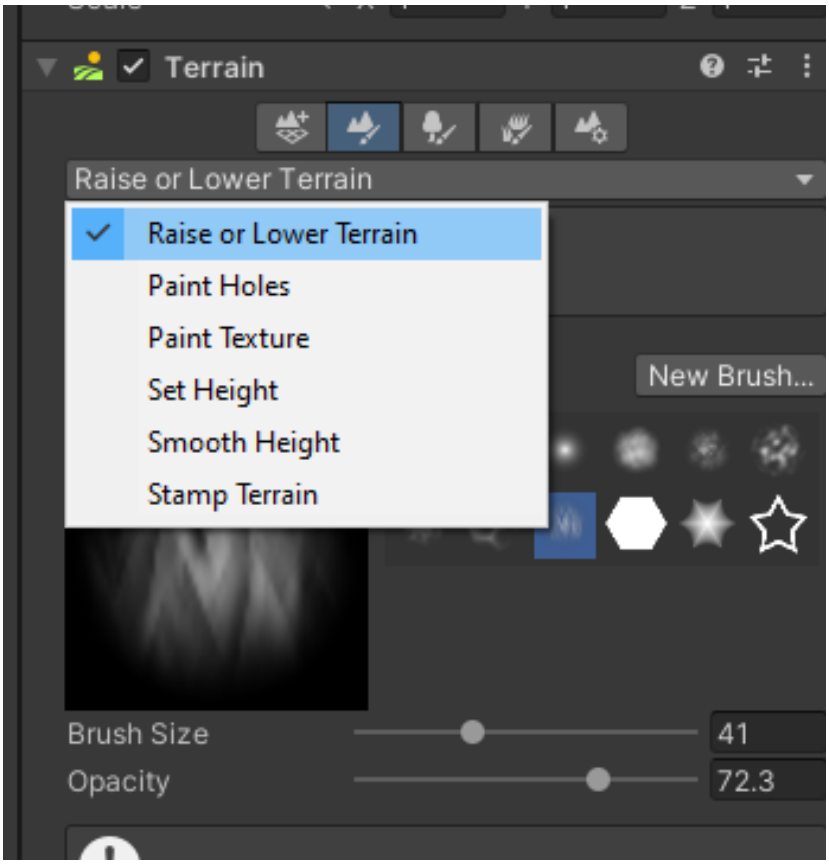
Рисунок 3 – Добавление объекта Terrain

Terrain – очень гибкий инструмент, который позволяет создавать ландшафт. Возвышенности, неровности и дыры.

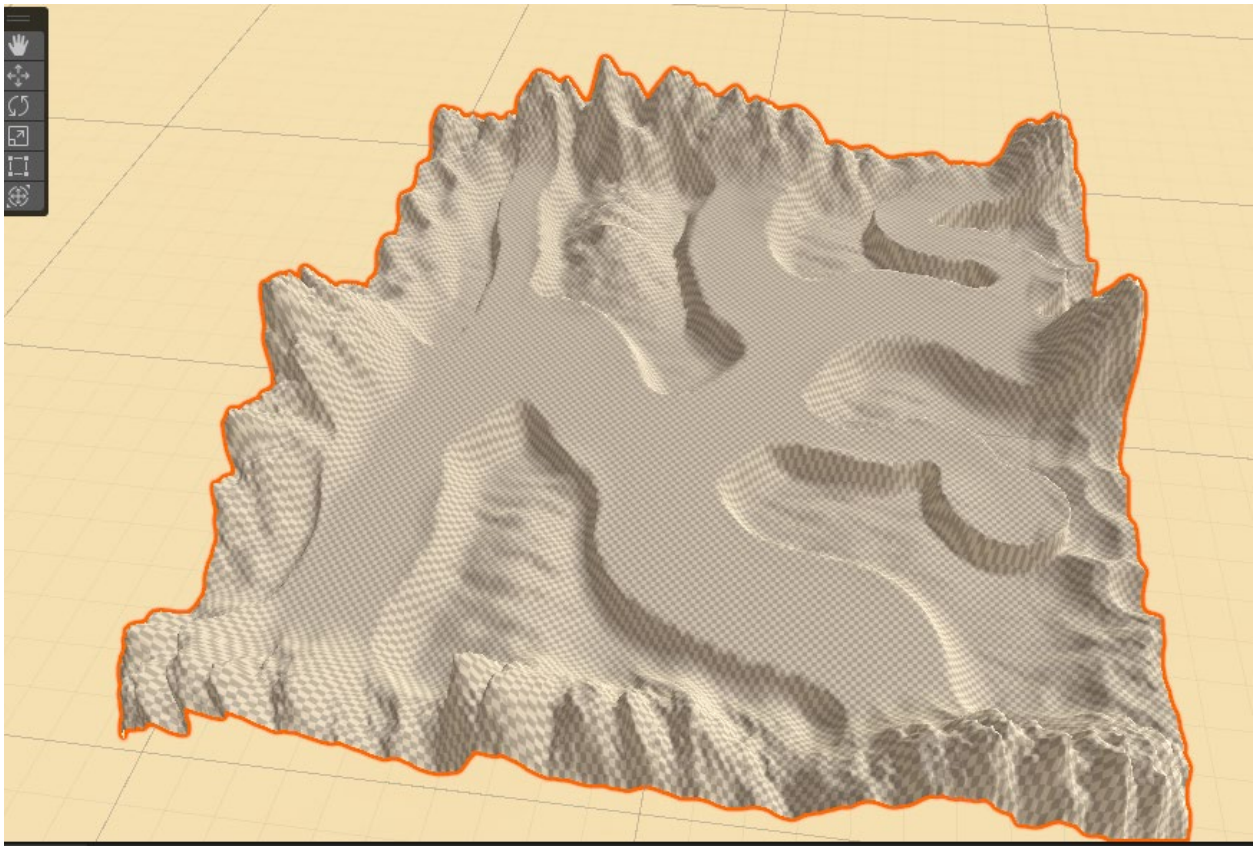
Чтобы изменить размер Terrain перейдите в его настройки.



Перейдите в Paint Terrain и выберите Raise or Lower Terrain. Выберите кисть (Brushes) и нарисуйте ландшафт.

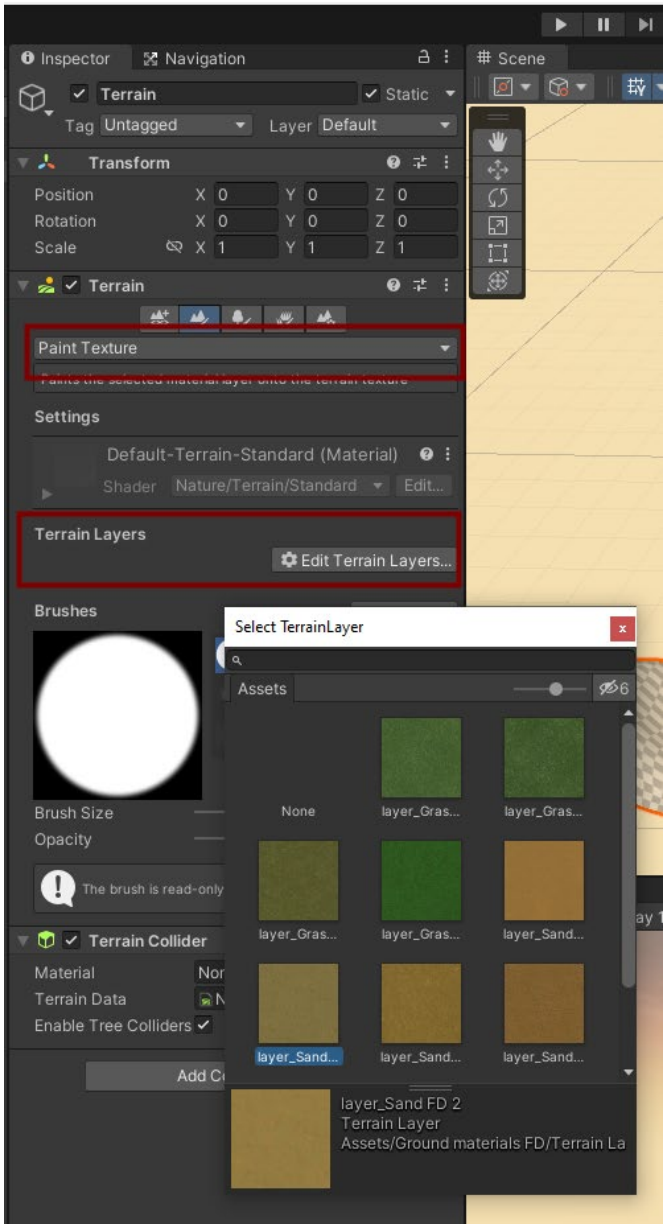


Получается что-то вроде такого:

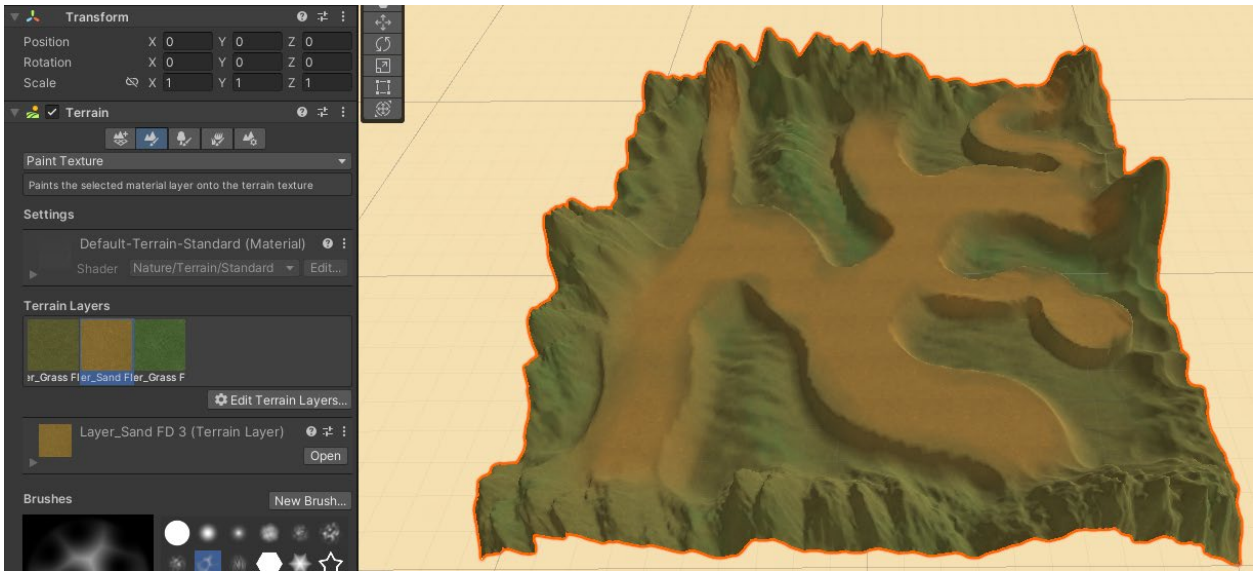


Чтобы покрасить terrain просто материала уже будет недостаточно. Нам нужна текстура. Поэтому идем в asset Store и скачиваем подходящую текстуру травы, например.

После того, как мы выбрали текстуру и загрузили ее в проект, то в Paint Terrain выбираем Paint Texture, и в Terrain Layers добавляем слои.

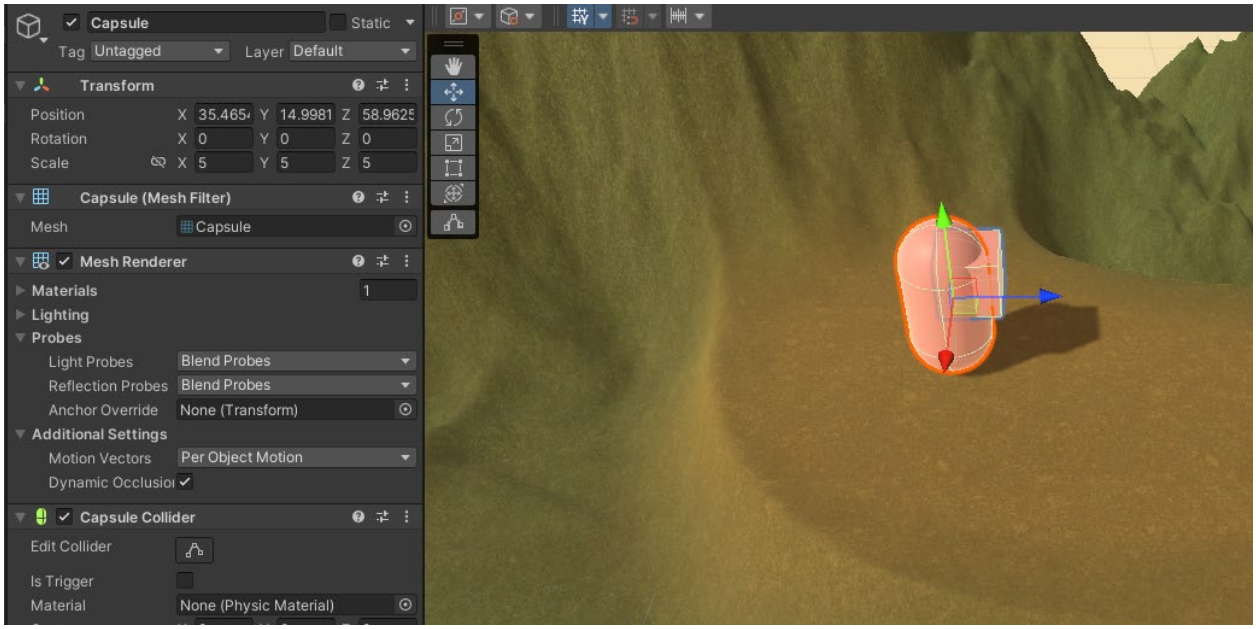


Terrain стал однородного цвета. Добавляя несколько разных слоев, можно раскрашивать нужные места кистью.

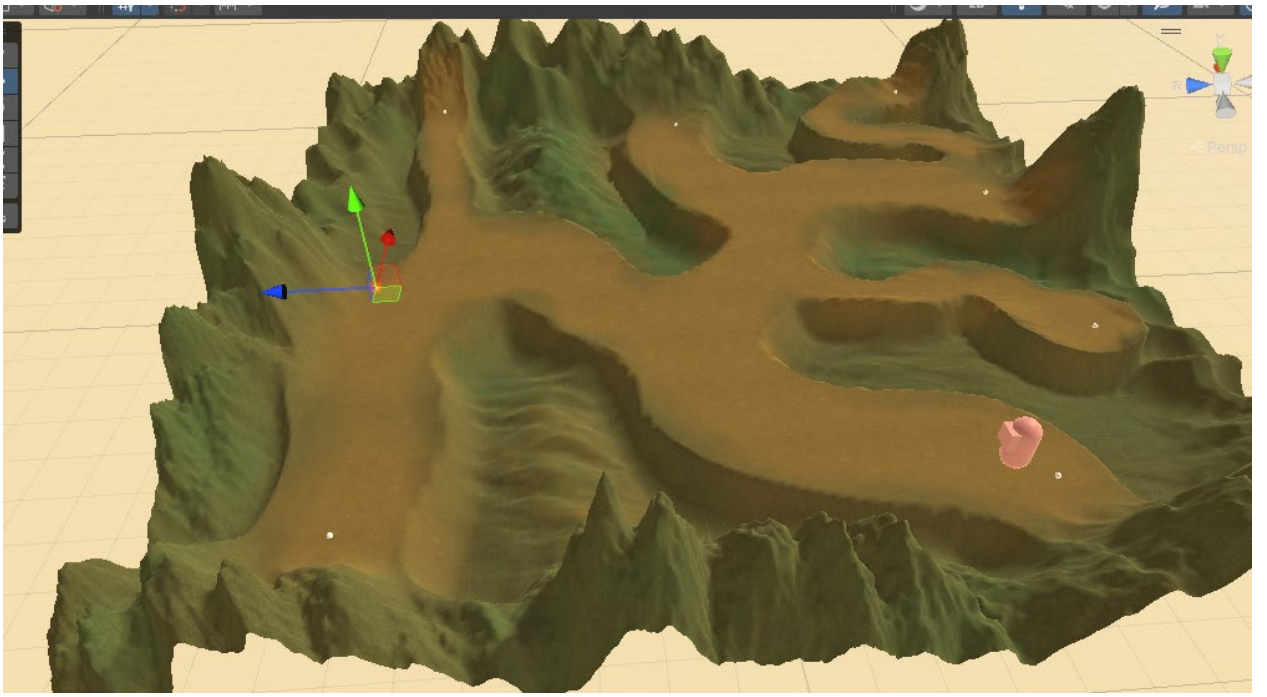


Запеките terrain в окне Navigation, чтобы агент мог перемещаться.

3. Добавим агента (врага) на сцену. Добавьте агенту компонент NavMeshAgent.



Теперь создадим точки маршрута. Можно создать пустой объект или сферы. Расставьте их по сцене.



Логика следующая: мы уже знакомы с методом SetDistanation. Теперь в качестве параметра вместо игрока мы будем передавать эти точки. Но чтобы отыскать все точки на сцене мы не будет использовать поиск по тегу. Создайте скрипт PathPoint и добавьте всего одну строчку:

```
public class PathPoint : MonoBehaviour
{
    Сообщение Unity | Ссылок: 0
    void Start()
    {
        GetComponent<MeshRenderer>().enabled = false;
    }


}
```

Перетащите этот скрипт на каждую точку (можно в иерархии сразу выделить все точки и одни раз перетащить скрипт, он добавится ко всем точкам). Скрипт отключает рендер точек при запуске приложения (точки есть на сцене, но не отображаются). Но самое главное, искать точки на сцене мы будет по этому компоненту.

Создайте скрипт Patrol.

```
public class Patrol : MonoBehaviour
{
    [SerializeField] private List<PathPoint> _points;
    [SerializeField] private NavMeshAgent _agent;

    private int _indexPoint;
    Сообщение Unity | Ссылок: 0
    void Start()
    {
        _points = FindObjectsByType<PathPoint>(FindObjectsSortMode.None).ToList();
        _agent = GetComponent<NavMeshAgent>();
    }

    Сообщение Unity | Ссылок: 0
    void Update()
    {
        if (_agent.remainingDistance <= _agent.stoppingDistance)
        {
            _indexPoint = Random.Range(0,_points.Count);
            _agent.SetDestination(_points[_indexPoint].transform.position);
        }
    }
}
```

Накиньте скрипт на агента и, при необходимости, измените скорость передвижения агента в NavMeshAgent.

Теперь агент перемещается по сцене от одной точки к другой. При желании, можно создать последовательный маршрут, а не случайный выбор точек. Подумайте, как это можно реализовать.

Индивидуальное задание

1. Добавьте terrain и создайте ландшафт местности.
2. Добавьте на сцену точки маршрута.
3. Реализуйте патрулирование местности.
4. Реализуйте связный маршрут. Реализуйте возможность менять цвет целевой точки.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого пункта подраздела «Индивидуальное задание» с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

4. Экранные формы (консольный вывод) и листинг программного кода с комментариями, показывающие порядок выполнения лабораторной работы, и результаты, полученные в ходе её выполнения.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Что из перечисленного является специализированным игровым движком: UnrealEngine, Godot, Constract3, Microsoft Word, CCleaner, Unity?
2. Что такое коллизия (collision)?
3. Что такое скрипт?
4. Для чего в компоненте Collider используется свойство isTrigger?

Список литературы

Для выполнения лабораторной работы, при подготовке к защите, а также для ответа на контрольные вопросы рекомендуется использовать следующие источники: [1–4].

ЛАБОРАТОРНАЯ РАБОТА 5. МЕХАНИЗМЫ ОБУЧЕНИЯ ИИ

Цели и задачи

Цель лабораторной работы: приобретение практических навыков работы со сценой и игровыми объектами; приобретение практических навыков реализации и интеграции нейронной сети в Unity.

Основные задачи:

- познакомиться с инструментарием игрового движка Unity;
- научиться создавать скрипты.

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с теорией построения регрессионных моделей машинного обучения, используя следующие источники: [1-4].

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько програмных продуктов для практической реализации задач лабораторной работы): MS Visual Studio 2013 и выше; среда разработки Java, интерпретатор Python (Anaconda).

Методика и порядок выполнения работы

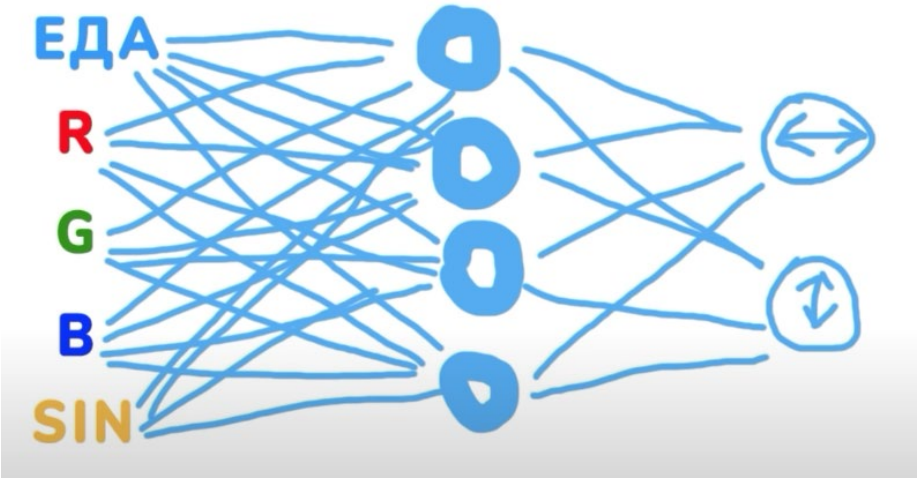
Учебная задача

Для симуляции естесственного отбора в Unity необходимо:

1. Разработать и реализовать нейронную сеть.

2. Интегрировать нейронную сеть в Unity.

Архитектура нейронной сети может выглядеть следующим образом:



Входы:

- Еда (x,y) – центр масс окружающей еды;
- R (x,y) – концентрация красных бактерий вокруг;
- G (x,y) – концентрация зеленых бактерий вокруг;
- B (x,y) – концентрация синих бактерий вокруг;
- SIN – синус от времени для разнообразия поведения.

Концентрация синих, красных, зеленых бактерий. Для каждого цвета есть вектор, который к нему направлен. Этот вектор и будет подаваться на вход нейросети. Вектор двумерный.

Таким образом, получается 9 входов.

Выходы:

Координаты x,y (куда нужно идти). Т.е. вправо-влево, вверх-вниз.

Для скрытого слоя можно использовать 5 нейронов.

Для реализации можно использовать следующую струкруру классов:

- Genome – для хранения весов нейронной сети и способностей бактерий;
- Layer – слои нейронной сети;
- NN – архитектура нейронной сети с обучением.

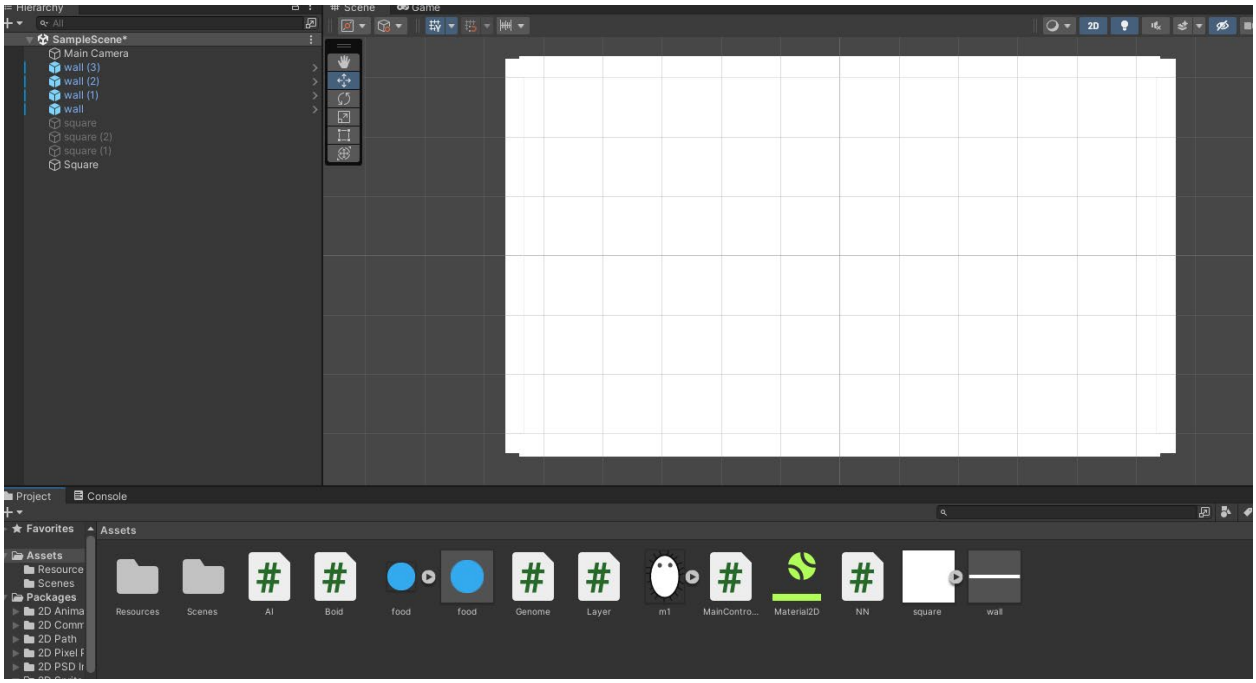
Эти классы не связаны с Unity, поэтому при их создании можно убрать наследование от класса MonoBehaviour.

Для того, чтобы связать эту нейронную сеть с Unity необходимо создать соотвествующие скрипты:

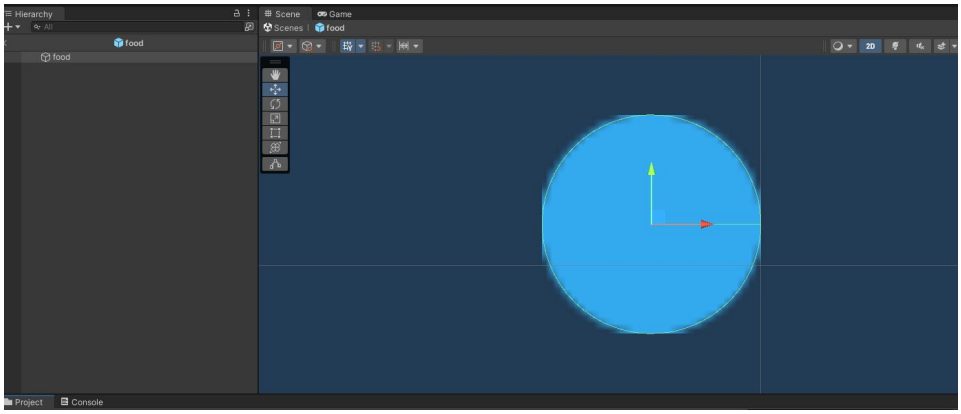
MainController – скрипт для инициализации и управления проектом.

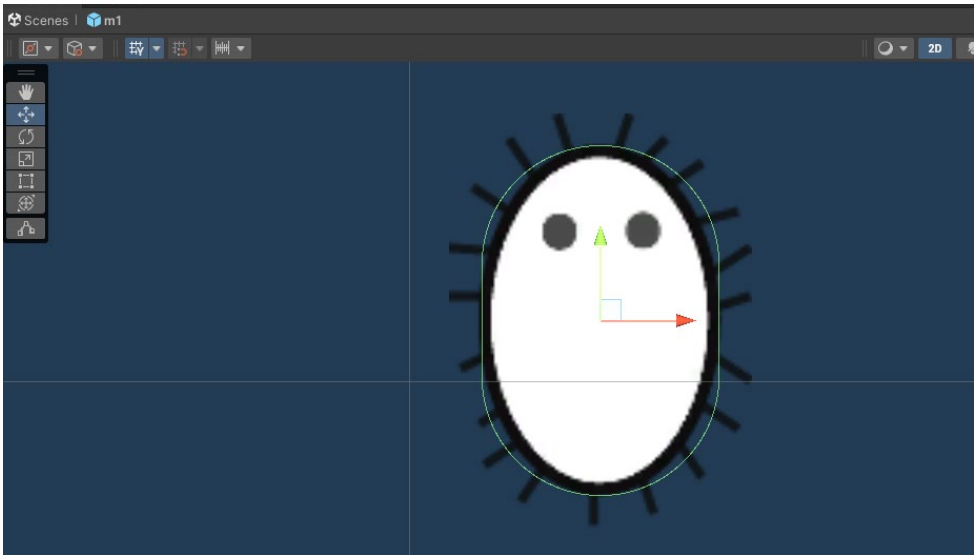
AI – для реализации естесственного отбора. В этом скрипте будет связана нейронная сеть с префабами бактерий. Можно сказать, искусственный интеллект бактерий.

В самом проекте необходимо создать сцену. Добавить барьеры и создать префабы бактерий и еды.



У префаба еды и бактерии обязательно добавьте коллайдеры.





В качестве примера данный проект реализован в проекте 2D.

Пример реализации проекта находится по ссылке:

https://github.com/RedInHat/DPO_practices/blob/main/practice6.unitypackage

Отдельно скрипты вы можете скачать здесь: https://github.com/RedInHat/DPO_practices/tree/main/Scripts

Индивидуальное задание

1. Реализуйте нейронную сеть для симуляции естественного отбора.
2. Интегрируйте нейронную сеть в проект Unity.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого пункта подраздела «Индивидуальное задание» с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

4. Экранные формы (консольный вывод) и листинг программного кода с комментариями, показывающие порядок выполнения лабораторной работы, и результаты, полученные в ходе её выполнения.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Что из перечисленного является специализированным игровым движком: UnrealEngine, Godot, Constract3, Microsoft Word, CCleaner, Unity?
2. Что такое коллизия (collision)?

3. Что такое скрипт?
4. Для чего в компоненте Collider используется свойство isTrigger?

Список литературы

Для выполнения лабораторной работы, при подготовке к защите, а также для ответа на контрольные вопросы рекомендуется использовать следующие источники: [1–5].

ЗАКЛЮЧЕНИЕ

Учебное пособие (лабораторный практикум) по дисциплине «Искусственный интеллект в игровых приложениях» для студентов направления 09.03.02 «Информационные системы и технологии». Пособие охватывает теоретические аспекты построения информационных систем на основе методов искусственного интеллекта, а также предлагает студентам практические рекомендации по разработке систем на основе мтеодов искусственного интеллекта. Основное внимание уделяется теории обучения машин (машинное обучение, machine learning).

В пособии рассмотрены практические аспекты проектирования и разработки информационных систем для решения различных задач с использованием следующих подходов: линейных методов классификации, аппарата нейронных сетей, байесовских методов классификации, алгоритмов восстановления регрессии, алгоритмов логической классификации.

Многие задачи, возникающие в практических приложениях, не могут быть решены заранее известными методами или алгоритмами. Это происходит по той причине, что нам заранее не известны механизмы порождения исходных данных или же известная нам информация недостаточна для построения модели источника, генерирующего поступающие к нам данные. Как говорят, мы получаем данные из «черного ящика». В этих условиях ничего не остается, как только изучать доступную нам последовательность исходных данных и пытаться строить предсказания совершенствуя нашу схему в процессе предсказания. Подход, при котором про- шлые данные или примеры используются для первоначального формирования и совершенствования схемы предсказания, называется методом машинного обучения (Machine Learning). Машинное обучение – чрезвычайно широкая и динамически развивающаяся область исследований, использующая огромное число теоретических и практических методов.

СПИСОК ЛИТЕРАТУРЫ

Список основной литературы

1. Мэннинг Дж., Батфилд-Эддисон П. Unity для разработчика. Мобильные мультиплатформенные игры/ Мэннинг Дж., Батфилд-Эддисон П. –М.:Спб.: Питер, 2018. 304 с.

2. Торн А. Искусство создания сценариев в Unity [Электронный ресурс]. М. : ДМК Пресс, 2016. - 360 с. Режим доступа: http://www.studentlibrary.ru/book/ISBN9785970603819.html

Список дополнительной литературы

3. Торн А. Основы анимации в Unity [Электронный ресурс]. М. : ДМК Пресс, 2016. - 176 с. Режим доступа: http://www.studentlibrary.ru/book/ISBN9785970603772.html

4. Шейдеры и эффекты в Unity. Книга рецептов [Электронный ресурс] / Кенни Ламмерс - М.: ДМК Пресс, 2014. -274 с. Режим доступа: http://www.studentlibrary.ru/book/ISBN9785940747376.html

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания

по организации и проведению самостоятельной работы
по дисциплине
«Искусственный интеллект в игровых приложениях»
для студентов направления подготовки
09.03.02 «Информационные системы и технологии»
направленность (профиль)
«Прикладное программирование в интеллектуальных информационных системах»

Ставрополь, 2026

1. Общие положения

Самостоятельная работа – планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное (аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (при частичном непосредственном участии преподавателя, оставляющем ведущую роль за работой студентов).

Самостоятельная работа студентов (СРС) в ВУЗе является важным видом учебной и научной деятельности студента. Самостоятельная работа студентов играет значительную роль в рейтинговой технологии обучения.

К основным видам самостоятельной работы студентов относятся:

− формирование и усвоение содержания конспекта лекций на базе рекомендованной лектором учебной литературы, включая информационные образовательные ресурсы (электронные учебники, электронные библиотеки и др.);

− написание докладов;

− подготовка к семинарам, практическим и лабораторным работам, их оформление;

− составление аннотированного списка статей из соответствующих журналов по отраслям знаний (педагогических, психологических, методических и др.);

− выполнение учебно-исследовательских работ, проектная деятельность;

− подготовка практических разработок и рекомендаций по решению проблемной ситуации;

− выполнение домашних заданий в виде решения отдельных задач, проведения типовых расчетов, расчетно-компьютерных и индивидуальных работ по отдельным разделам содержания дисциплин и т.д.;

− компьютерный текущий самоконтроль и контроль успеваемости на базе электронных обучающих и аттестующих тестов;

− выполнение курсовых работ (проектов) в рамках дисциплин;

− выполнение выпускной квалификационной работы и др.

Методика организации самостоятельной работы студентов зависит от структуры, характера и особенностей изучаемой дисциплины, объема часов на ее изучение, вида заданий для самостоятельной работы студентов, индивидуальных качеств студентов и условий учебной деятельности.

Процесс организации самостоятельной работы студентов включает в себя следующие этапы:

- подготовительный (определение целей, составление программы, подготовка методического обеспечения, подготовка оборудования);
- основной (реализация программы, использование приемов поиска информации, усвоения, переработки, применения, передачи знаний, фиксирование результатов, самоорганизация процесса работы);
- заключительный (оценка значимости и анализ результатов, их систематизация, оценка эффективности программы и приемов работы, выводы о направлениях оптимизации труда).

Самостоятельная работа по дисциплине «Искусственный интеллект в игровых приложениях» направлена на формирование следующих **компетенций**:

Код, формулировка компетенции	Код, формулировка индикатора
ПК-1 Способен разрабатывать программное обеспечение интеллектуальных информационных систем	ПК-1$_{ИД-1.1}$ – Демонстрирует знания в области применения алгоритмов и их адаптации к практическим задачам

	ПК-1$_{ИД-1.2}$ – Разрабатывает приложения на языке высокого уровня в том числе с использованием интеллектуальных технологий
	ПК-1$_{ИД-1.3}$ – Применяет современные автоматизированные технологии тестирования и отладки программного обеспечения
	ПК-1$_{ИД-1.4}$ – Применяет современные технологии непрерывного развертывания и доставки программного обеспечения
	ПК-1$_{ИД-1.5}$ – Разрабатывает приложения для различных целевых платформ (мобильные, серверные, встраиваемые, оконные)
ПК-5 Способен осуществлять автоматизированное проектирование информационных систем, в том числе интеллектуальных	ПК-5$_{ИД-5.1}$ – Проектирует новые и выполняет реинжиниринг существующих информационных систем в рамках решения профессиональных задач основываясь на данных предметной области и перспективах применения интеллектуальных моделей
	ПК-5$_{ИД-5.2}$ – Применяет автоматизированные инструментальные средства проектирования интеллектуальных информационных систем
	ПК-5$_{ИД-5.3}$ – Применяет технологии моделирования интеллектуальных информационных систем

2. Цель и задачи самостоятельной работы

Ведущая цель организации и осуществления СРС совпадает с целью обучения студента – формирование набора общенаучных, профессиональных и специальных компетенций будущего бакалавра по соответствующему направлению подготовки

При организации СРС важным и необходимым условием становятся формирование умения самостоятельной работы для приобретения знаний, навыков и возможности организации учебной и научной деятельности. Целью самостоятельной работы студентов является овладение фундаментальными знаниями, профессиональными умениями и навыками деятельности по профилю, опытом творческой, исследовательской деятельности. Самостоятельная работа студентов способствует развитию самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровня.

Задачами СРС являются:

- систематизация и закрепление полученных теоретических знаний и практических умений студентов;
- углубление и расширение теоретических знаний;
- формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;
- развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;
- формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;

- развитие исследовательских умений;
- использование материала, собранного и полученного в ходе самостоятельных занятий на семинарах, на практических и лабораторных занятиях, при написании курсовых и выпускной квалификационной работ, для эффективной подготовки к итоговым зачетам и экзаменам.

3. Технологическая карта самостоятельной работы студента

Коды реализуемых компетенций	Вид деятельности студентов	Средства и технологии оценки	Объем часов, в том числе (акад.)		
			СРС	Контактная работа с преподавателем	Всего
4 семестр					
ПК-1, ПК-5	Самостоятельное изучение литературы и источников	Собеседование	10	2	12
ПК-1, ПК-5	Подготовка к лабораторным занятиям	Защита ЛР	48	16	64
Итого за 4 семестр			58	18	76
Итого			58	18	76

4. Порядок выполнения самостоятельной работы студентом

4.1. Методические рекомендации по работе с учебной литературой

При работе с книгой необходимо подобрать литературу, научиться правильно ее читать, вести записи. Для подбора литературы в библиотеке используются алфавитный и систематический каталоги.

Важно помнить, что рациональные навыки работы с книгой - это всегда большая экономия времени и сил.

Правильный подбор учебников рекомендуется преподавателем, читающим лекционный курс. Необходимая литература может быть также указана в методических разработках по данному курсу.

Изучая материал по учебнику, следует переходить к следующему вопросу только после правильного уяснения предыдущего, описывая на бумаге все выкладки и вычисления (в том числе те, которые в учебнике опущены или на лекции даны для самостоятельного вывода).

При изучении любой дисциплины большую и важную роль играет самостоятельная индивидуальная работа.

Особое внимание следует обратить на определение основных понятий курса. Студент должен подробно разбирать примеры, которые поясняют такие определения, и уметь строить аналогичные примеры самостоятельно. Нужно добиваться точного представления о том, что изучаешь. Полезно составлять опорные конспекты. При изучении материала по учебнику полезно в тетради (на специально отведенных полях) дополнять

конспект лекций. Там же следует отмечать вопросы, выделенные студентом для консультации с преподавателем.

Выводы, полученные в результате изучения, рекомендуется в конспекте выделять, чтобы они при перечитывании записей лучше запоминались.

Опыт показывает, что многим студентам помогает составление листа опорных сигналов, содержащего важнейшие и наиболее часто употребляемые формулы и понятия. Такой лист помогает запомнить формулы, основные положения лекции, а также может служить постоянным справочником для студента.

Чтение научного текста является частью познавательной деятельности. Ее цель – извлечение из текста необходимой информации. От того на сколько осознанна читающим собственная внутренняя установка при обращении к печатному слову (найти нужные сведения, усвоить информацию полностью или частично, критически проанализировать материал и т.п.) во многом зависит эффективность осуществляемого действия.

Выделяют ***четыре основные установки в чтении научного текста***:

информационно-поисковый (задача – найти, выделить искомую информацию)

усваивающая (усилия читателя направлены на то, чтобы как можно полнее осознать и запомнить как сами сведения излагаемые автором, так и всю логику его рассуждений)

аналитико-критическая (читатель стремится критически осмыслить материал, проанализировав его, определив свое отношение к нему)

творческая (создает у читателя готовность в том или ином виде – как отправной пункт для своих рассуждений, как образ для действия по аналогии и т.п. – использовать суждения автора, ход его мыслей, результат наблюдения, разработанную методику, дополнить их, подвергнуть новой проверке).

Основные виды систематизированной записи прочитанного:

Аннотирование – предельно краткое связное описание просмотренной или прочитанной книги (статьи), ее содержания, источников, характера и назначения;

Планирование – краткая логическая организация текста, раскрывающая содержание и структуру изучаемого материала;

Тезирование – лаконичное воспроизведение основных утверждений автора без привлечения фактического материала;

Цитирование – дословное выписывание из текста выдержек, извлечений, наиболее существенно отражающих ту или иную мысль автора;

Конспектирование – краткое и последовательное изложение содержания прочитанного.

Конспект – сложный способ изложения содержания книги или статьи в логической последовательности. Конспект аккумулирует в себе предыдущие виды записи, позволяет всесторонне охватить содержание книги, статьи. Поэтому умение составлять план, тезисы, делать выписки и другие записи определяет и технологию составления конспекта.

Методические рекомендации по составлению конспекта:

1. Внимательно прочитайте текст. Уточните в справочной литературе непонятные слова. При записи не забудьте вынести справочные данные на поля конспекта;
2. Выделите главное, составьте план;
3. Кратко сформулируйте основные положения текста, отметьте аргументацию автора;
4. Законспектируйте материал, четко следуя пунктам плана. При конспектировании старайтесь выразить мысль своими словами. Записи следует вести четко, ясно.
5. Грамотно записывайте цитаты. Цитируя, учитывайте лаконичность, значимость мысли.

В тексте конспекта желательно приводить не только тезисные положения, но и их доказательства. При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и

выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной последовательности, отвечающей логической структуре произведения. Для уточнения и дополнения необходимо оставлять поля.

Овладение навыками конспектирования требует от студента целеустремленности, повседневной самостоятельной работы.

4.2. Методические рекомендации по подготовке к практическим и лабораторным занятиям

Для того чтобы практические и лабораторные занятия приносили максимальную пользу, необходимо помнить, что упражнение и решение задач проводятся по вычитанному на лекциях материалу и связаны, как правило, с детальным разбором отдельных вопросов лекционного курса. Следует подчеркнуть, что только после усвоения лекционного материала с определенной точки зрения (а именно с той, с которой он излагается на лекциях) он будет закрепляться на практических занятиях как в результате обсуждения и анализа лекционного материала, так и с помощью решения проблемных ситуаций, задач. При этих условиях студент не только хорошо усвоит материал, но и научится применять его на практике, а также получит дополнительный стимул (и это очень важно) для активной проработки лекции.

При самостоятельном решении задач нужно обосновывать каждый этап решения, исходя из теоретических положений курса. Если студент видит несколько путей решения проблемы (задачи), то нужно сравнить их и выбрать самый рациональный. Полезно до начала вычислений составить краткий план решения проблемы (задачи). Решение проблемных задач или примеров следует излагать подробно, вычисления располагать в строгом порядке, отделяя вспомогательные вычисления от основных. Решения при необходимости нужно сопровождать комментариями, схемами, чертежами и рисунками.

Следует помнить, что решение каждой учебной задачи должно доводиться до окончательного логического ответа, которого требует условие, и по возможности с выводом. Полученный ответ следует проверить способами, вытекающими из существа данной задачи. Полезно также (если возможно) решать несколькими способами и сравнить полученные результаты. Решение задач данного типа нужно продолжать до приобретения твердых навыков в их решении.

4.3. Методические рекомендации по самопроверке знаний

После изучения определенной темы по записям в конспекте и учебнику, а также решения достаточного количества соответствующих задач на практических занятиях и самостоятельно студенту рекомендуется, провести самопроверку усвоенных знаний, ответив на контрольные вопросы по изученной теме.

В случае необходимости нужно еще раз внимательно разобраться в материале.

Иногда недостаточность усвоения того или иного вопроса выясняется только при изучении дальнейшего материала. В этом случае надо вернуться назад и повторить плохо усвоенный материал. Важный критерий усвоения теоретического материала - умение решать задачи или пройти тестирование по пройденному материалу. Однако следует помнить, что правильное решение задачи может получиться в результате применения механически заученных формул без понимания сущности теоретических положений.

4.4. Методические рекомендации по написанию научных текстов (докладов, докладовов, эссе, научных статей и т.д.)

Перед тем, как приступить к написанию научного текста, важно разобраться, какова истинная цель вашего научного текста - это поможет вам разумно распределить свои силы и время.

Во-первых, сначала нужно определиться с идеей научного текста, а для этого необходимо научиться либо относиться к разным явлениям и фактам несколько критически (своя идея – как иная точка зрения), либо научиться увлекаться какими-то известными идеями, которые нуждаются в доработке (идея – как оптимистическая позиция и направленность на дальнейшее совершенствование уже известного). Во-вторых, научиться организовывать свое время, ведь, как известно, свободное (от всяких глупостей) время – важнейшее условие настоящего творчества, для него наконец-то появляется время. Иногда именно на организацию такого времени уходит немалая часть сил и талантов.

Писать следует ясно и понятно, стараясь основные положения формулировать четко и недвусмысленно (чтобы и самому понятно было), а также стремясь структурировать свой текст. Каждый раз надо представлять, что ваш текст будет кто-то читать и ему захочется сориентироваться в нем, быстро находить ответы на интересующие вопросы (заодно представьте себя на месте такого человека). Понятно, что работа, написанная «сплошным текстом» (без заголовков, без выделения крупным шрифтом наиболее важным мест и т, п.), у культурного читателя должна вызывать брезгливость и даже жалость к автору (исключения составляют некоторые древние тексты, когда и жанр был иной и к текстам относились иначе, да и самих текстов было гораздо меньше – не то, что в эпоху «информационного взрыва» и соответствующего «информационного мусора»).

Объем текста и различные оформительские требования во многом зависят от принятых в конкретном учебном заведении порядков.

Доклад - это самостоятельное исследование студентом определенной проблемы, комплекса взаимосвязанных вопросов.

Доклад не должна составляться из фрагментов статей, монографий, пособий. Кроме простого изложения фактов и цитат, в доклад е должно проявляться авторское видение проблемы и ее решения.

Рассмотрим основные этапы подготовки

а студентом.

Выполнение доклада начинается с выбора темы.

Затем студент приходит на первую консультацию к руководителю, которая предусматривает:

- обсуждение цели и задач работы, основных моментов избранной темы;
- консультирование по вопросам подбора литературы;
- составление предварительного плана.

Следующим этапом является работа с литературой. Необходимая литература подбирается студентом самостоятельно.

После подбора литературы целесообразно сделать рабочий вариант плана работы. В нем нужно выделить основные вопросы темы и параграфы, раскрывающие их содержание.

Составленный список литературы и предварительный вариант плана уточняются, согласуются на очередной консультации с руководителем.

Затем начинается следующий этап работы - изучение литературы. Только внимательно читая и конспектируя литературу, можно разобраться в основных вопросах темы и подготовиться к самостоятельному (авторскому) изложению содержания доклада. Конспектируя первоисточники, необходимо отразить основную идею автора и его позицию по исследуемому вопросу, выявить проблемы и наметить задачи для дальнейшего изучения данных проблем.

Систематизация и анализ изученной литературы по проблеме исследования позволяют студенту написать работу.

Рабочий вариант текста доклада предоставляется руководителю на проверку. На основе рабочего варианта текста руководитель вместе со студентом обсуждает возможности доработки текста, его оформление. После доработки доклад сдается на кафедру для его оценивания руководителем.

Требования к написанию доклада

Написание 1 доклада является обязательным условием выполнения плана СРС по любой дисциплине профессионального цикла.

Тема доклада может быть выбрана студентом из предложенных в рабочей программе или фонде оценочных средств дисциплины, либо определена самостоятельно, исходя из интересов студента (в рамках изучаемой дисциплины). Выбранную тему необходимо согласоваться с преподавателем.

Доклад должен быть написан научным языком.

Объем доклада должен составлять 20-25 стр.

Структура доклада:

- Введение (не более 3-4 страниц). Во введении необходимо обосновать выбор темы, ее актуальность, очертить область исследования, объект исследования, основные цели и задачи исследования.
- Основная часть состоит из 2-3 разделов. В них раскрывается суть исследуемой проблемы, проводится обзор мировой литературы и источников Интернет по предмету исследования, в котором дается характеристика степени разработанности проблемы и авторская аналитическая оценка основных теоретических подходов к ее решению. Изложение материала не должно ограничиваться лишь описательным подходом к раскрытию выбранной темы. Оно также должно содержать собственное видение рассматриваемой проблемы и изложение собственной точки зрения на возможные пути ее решения.
- Заключение (1-2 страницы). В заключении кратко излагаются достигнутые при изучении проблемы цели, перспективы развития исследуемого вопроса
- Список использованной литературы (не меньше 10 источников), в алфавитном порядке, оформленный в соответствии с принятыми правилами. В список использованной литературы рекомендуется включать работы отечественных и зарубежных авторов, в том числе статьи, опубликованные в научных журналах в течение последних 3-х лет и ссылки на ресурсы сети Интернет.
- Приложение (при необходимости).

Требования к оформлению:

- текст с одной стороны листа;
- шрифт Times New Roman;
- кегль шрифта 14;
- межстрочное расстояние 1,5;
- поля: сверху 2,5 см, снизу – 2,5 см, слева - 3 см, справа 1,5 см;
- доклад должен быть представлен в сброшюрованном виде.

Порядок защиты доклада:

Защита доклада проводится на практических занятиях, после окончания работы студента над ним и исправления всех недочетов, выявленных преподавателем в ходе консультаций. На защиту доклада отводится 5-7 минут времени, в ходе которого студент должен показать свободное владение материалом по заявленной теме. При защите доклада приветствуется использование мультимедиа-презентации.

Оценка доклада

Доклад оценивается по следующим критериям:

• соблюдение требований к его оформлению;

• необходимость и достаточность для раскрытия темы приведенной в тексте доклада информации;

• умение студента свободно излагать основные идеи, отраженные в докладе;

• способность студента понять суть задаваемых преподавателем и сокурсниками вопросов и сформулировать точные ответы на них.

Критерии оценки:

Оценка «отлично» выставляется студенту, если в докладе студент исчерпывающе, последовательно, четко и логически стройно излагает материал; свободно справляется с

задачами, вопросами и другими видами применения знаний; использует для написания доклада современные научные материалы; анализирует полученную информацию; проявляет самостоятельность при написании доклада.

Оценка «хорошо» выставляется студенту, если качество выполнения доклада достаточно высокое. Студент твердо знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопросы по теме доклада.

Оценка «удовлетворительно» выставляется студенту, если материал доклада излагается частично, но пробелы не носят существенного характера, студент допускает неточности и ошибки при защите доклада, дает недостаточно правильные формулировки, наблюдаются нарушения логической последовательности в изложении материала.

Оценка «неудовлетворительно» выставляется студенту, если он не подготовил доклад или допустил существенные ошибки. Студент неуверенно излагает материал доклада, не отвечает на вопросы преподавателя.

4.5. Методические рекомендации по выполнению исследовательских проектов

Исследовательская проектная работа – это групповая работа, для выполнения которой необходим выбор и приложение научной методики к поставленной задаче, получение собственного теоретического или экспериментального материала, на основании которого необходимо провести анализ и сделать выводы об исследуемом явлении. Выполнение проекта – это всегда коллективная, творческая практическая работа, предназначенная для получения определенного продукта или научно-технического результата. Такая работа подразумевает четкое, однозначное формирование поставленной задачи, определение сроков выполнения намеченного, определение требований к разрабатываемому объекту.

Выполнение 1 группового проекта является обязательным условием выполнения самостоятельной работы по любой дисциплине профессионального цикла. Тема проектного задания может быть выбрана студентом из предложенных в рабочей программе или фонде оценочных средств дисциплины, либо определена самостоятельно, исходя из интересов студента (в рамках изучаемой дисциплины). Выбранную тему необходимо согласоваться с преподавателем.

Требования по выполнению и оформлению проекта

При выполнении проекта приветствуется работа в группе (2-3 человека). Проект – это исследовательская работа, в ходе которой студенты должны продемонстрировать владение навыками научного исследования, умения проводить анализ, обобщать информацию, делать выводы, предлагать свои решения проблемы, рассматриваемой в проекте.

При подготовке материалов проекта студенты должны продемонстрировать владение современными методами компьютерной обработки данных.

Критерии оценки работы участника проекта.

Для каждого из участников проекта оцениваются:

• профессиональные теоретические знания в соответствующей области;

• умение работать со справочной и научной литературой, осуществлять поиск необходимой информации в Интернет;

• умение работать с техническими средствами;

• умение пользоваться соответствующими выполняемому проекту информационными технологиями;

• умение готовить материалы проекта для презентации: составлять и редактировать тексты, формировать презентацию проекта;

• умение работать в команде;

• умение публично представлять результаты собственной деятельности;

• коммуникабельность, инициативность, творческие способности.

Критерии выставления оценки участникам проекта

Оценка	Профессиональные компетенции	Компетенции, связанные с использованием соответствующих выполняемому проекту технических средств и информационных технологий	Иные универсальные компетенции (коммуникабельность, инициативность, умение работать в «команде», управленческие навыки и т.д.)	Отчетность
«Отлично»	Работа выполнена на высоком профессиональном уровне. Представленный материал в основном фактически верен, допускаются негрубые фактические неточности. Студент свободно отвечает на вопросы, связанные с проектом.	Технические средства и информационные технологии освоены и использованы для реализации проекта полностью	Студент проявил инициативу, творческий подход, способность к выполнению сложных заданий, навыки работы в коллективе, организационные способности.	Проект представлен полностью и в срок.
«Хорошо»	Работа выполнена на достаточно высоком профессиональном уровне. Допущено до 4–5 фактических ошибок. Студент отвечает на вопросы, связанные с проектом, но недостаточно полно.	Обнаруживаются некоторые ошибки в использовании соответствующих технических средств и информационных технологий	Студент достаточно полно, но без инициативы и творческих находок выполнил возложенные на него задачи.	Проект представлен достаточно полно и в срок, но с некоторыми недоработками.
«Удовлетворительно»	Уровень недостаточно высок. Допущено до 8 фактических ошибок. Студент может ответить лишь на некоторые из заданных вопросов, связанных с проектом.	Обнаруживает недостаточное владение навыками работы с техническими средствами и соответствующими информационными технологиями	Студент выполнил большую часть возложенной на него работы.	Проект сдан со значительным опозданием (более недели) и не полностью
«Неудовлетворительно»	Работа не выполнена или выполнена на низком уровне. Допущено более 8 фактических ошибок. Ответы на связанные с	Навыков работы с техническими средствами нет, информационные технологии не освоены	Студент практически не работал, не выполнил свои задачи или выполнил лишь	Проект не сдан.

Оценка	Профессиональные компетенции	Компетенции, связанные с использованием соответствующих выполняемому проекту технических средств и информационных технологий	Иные универсальные компетенции (коммуникабельность, инициативность, умение работать в «команде», управленческие навыки и т.д.)	Отчетность
	проектом вопросы обнаруживают непонимание предмета и отсутствие ориентации в материале проекта.		отдельные не существенные поручения в групповом проекте.	

Студенты должны: защитить проект в режиме презентации, предъявить файлы выполненного проекта, уметь рассказать о технологиях, использованных ими при выполнении проекта, дать оценку работы каждого члена группы (*если проект групповой*).

4.6. Методические рекомендации по подготовке к экзаменам и зачетам

Изучение многих общепрофессиональных и специальных дисциплин завершается экзаменом. Подготовка к экзамену способствует закреплению, углублению и обобщению знаний, получаемых, в процессе обучения, а также применению их к решению практических задач. Готовясь к экзамену, студент ликвидирует имеющиеся пробелы в знаниях, углубляет, систематизирует и упорядочивает свои знания. На экзамене студент демонстрирует то, что он приобрел в процессе обучения по конкретной учебной дисциплине.

Экзаменационная сессия - это серия экзаменов, установленных учебным планом. Между экзаменами интервал 3-4 дня. Не следует думать, что 3-4 дня достаточно для успешной подготовки к экзаменам.

В эти 3-4 дня нужно систематизировать уже имеющиеся знания. На консультации перед экзаменом студентов познакомят с основными требованиями, ответят на возникшие у них вопросы. Поэтому посещение консультаций обязательно.

Требования к организации подготовки к экзаменам те же, что и при занятиях в течение семестра, но соблюдаться они должны более строго. Во-первых, очень важно соблюдение режима дня; сон не менее 8 часов в сутки, занятия заканчиваются не позднее, чем за 2-3 часа до сна. Оптимальное время занятий - утренние и дневные часы. В перерывах между занятиями рекомендуются прогулки на свежем воздухе, неутомительные занятия спортом. Во-вторых, наличие хороших собственных конспектов лекций. Даже в том случае, если была пропущена какая-либо лекция, необходимо во время ее восстановить (переписать ее на кафедре), обдумать, снять возникшие вопросы для того, чтобы запоминание материала было осознанным. В-третьих, при подготовке к экзаменам у студента должен быть хороший учебник или конспект литературы, прочитанной по указанию преподавателя в течение семестра. Здесь можно эффективно использовать листы опорных сигналов.

Вначале следует просмотреть весь материал по сдаваемой дисциплине, отметить для себя трудные вопросы. Обязательно в них разобраться. В заключение еще раз целесообразно повторить основные положения, используя при этом листы опорных сигналов.

Систематическая подготовка к занятиям в течение семестра позволит использовать время экзаменационной сессии для систематизации знаний.

Контроль самостоятельной работы студентов

Контроль самостоятельной работы проводится преподавателем в аудитории.

Предусмотрены следующие виды контроля: собеседование, оценка доклада, оценка презентации, оценка участия в круглом столе, оценка выполнения проекта.

Подробные критерии оценивания компетенций приведены в Фонде оценочных средств для проведения текущей и промежуточной аттестации.

Список литературы для выполнения СРС

Перечень основной литературы:

1. Мэннинг Дж., Батфилд-Эддисон П. Unity для разработчика. Мобильные мультиплатформенные игры/ Мэннинг Дж., Батфилд-Эддисон П. –М.:Спб.: Питер, 2018. 304 с.

2. Торн А. Искусство создания сценариев в Unity [Электронный ресурс]. М. : ДМК Пресс, 2016. - 360 с. Режим доступа: http://www.studentlibrary.ru/book/ISBN9785970603819.html

Перечень дополнительной литературы:

1. Торн А. Основы анимации в Unity [Электронный ресурс]. М. : ДМК Пресс, 2016. - 176 с. Режим доступа: http://www.studentlibrary.ru/book/ISBN9785970603772.html

2. Шейдеры и эффекты в Unity. Книга рецептов [Электронный ресурс] / Кенни Ламмерс - М.: ДМК Пресс, 2014. -274 с. Режим доступа: http://www.studentlibrary.ru/book/ISBN9785940747376.html

Методическая литература:

1. Методические рекомендации для самостоятельной работы студентов по дисциплине «Искусственный интеллект в игровых приложениях»
2. Методические указания к лабораторным работам по дисциплине «Искусственный интеллект в игровых приложениях»

Интернет-ресурсы:

1. http://el.ncfu.ru/ – система управления обучением ФГАОУ ВО СКФУ. Дистанционная поддержка дисциплины «Искусственный интеллект в игровых приложениях».