

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Филиал федерального государственного бюджетного образовательного учреждения высшего образования
«Кубанский государственный университет»
в г. Новороссийске

Кафедра информатики и математики

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(БАКАЛАВРСКАЯ РАБОТА)**

**МЕТОДЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА В РАЗРАБОТКЕ
ИГР В UNITY 3D**

**Работу выполнил Качур Пантелеймон
Александрович**

Направление подготовки 01.03.02 Прикладная математика и информатика

Направленность (профиль) Системный анализ, исследование операций и управление (Математическое и информационное обеспечение экономической деятельности)

**Научный руководитель
Доцент, канд.физ.-мат.наук, доцент Дьяченко
Сергей Владимирович**

Краснодар
2022



Создавайте игры, общайтесь со своей аудиторией и добивайтесь успеха

СОДЕРЖАНИЕ

Введение

1 Искусственный интеллект в Unity 3D: фундаментальные основы и особенности игровых вариантов

1.1 Понятие, направления исследований и применение искусственного интеллекта

1.1.1 Понятие искусственного интеллекта

1.1.2 Направления в исследовании искусственного интеллекта

1.2 Unity и иные технологии создания игр с современным искусственным интеллектом

1.3 Методы и компоненты искусственного интеллекта в Unity 3D

2 Практическое применение методов для создания искусственного интеллекта в Unity 3D

2.1 Создание, обучение, настройка проекта и основных поведенческих возможностей игровых персонажей

2.2 Образование искусственного интеллекта игровых персонажей методом «простого скрипта»

2.3 Настройка искусственного интеллекта поведения игровых персонажей методом «машины состояний»

2.4 Программирование искусственного интеллекта поведения игровых персонажей методом «теории автоматизированного управления»

2.5 Совершенствование искусственного интеллекта поведения игровых персонажей методом «нейронных сетей»

Заключение

Список использованных источников

АКТУАЛЬНОСТЬ ТЕМЫ ВЫЗВАНА:

1) Огромным вниманием к искусственному интеллекту в России и других странах.

Мировой объем государственных инвестиций в искусственный интеллект (ИИ) достиг более **1,5 трлн. долларов**. В США в 2016 г. выделили на ИИ в период до 2030 г. - **100 млрд. долларов**. Президент России В.В. Путин в 2021 г. признал необходимость искусственного интеллекта для России и поставил задачу скорейшего его внедрения.

2) Значимостью искусственного интеллекта в процессе создания и совершенствования компьютерных игр.

Игровые компьютерные программы с ИИ **приносят доход разработчикам и отдельным странам, привлекают государственный и частный капитал**. Например, белорусская компания Wargaming, разработала игру World of Tanks и Беларусь получает прибыли от участия в виртуальных танковых сражениях более миллиона игроков из всего мира.

3) Направлениями развития и проблемами эффективности использования отдельных методов программирования искусственного интеллекта.

Основное направление – **совершенствование логики поведения виртуальных объектов и взаимодействия их между собой**. Основная проблема – **в эффективности применяемых методов, что необходимо для развития сверх – ИИ.**

Объект, предмет, цель и задачи исследования:

Объект исследования – искусственный интеллект.

Предмет исследования – теоретические знания и практические разработки, посвященные искусственному интеллекту в компьютерных играх.

- Цель исследования: - 1) изучить и проанализировать методы искусственного интеллекта компьютерных игр в Unity 3D;
- 2) с использованием различных методов создать в Unity 3D искусственный интеллект персонажей в эпизодах (сюжетах) компьютерной игры из жанра РПГ.

Задачи исследования:

- - рассмотреть понятие, направления исследования, применение искусственного интеллекта с использованием Unity 3D;
- - проанализировать методы и технологии разработки игр с современным искусственным интеллектом в Unity 3D;
- - отразить процессы создания, обучения, общей настройки проекта, выбора окружения и основных поведенческих возможностей в рамках практической деятельности при программировании искусственного интеллекта в Unity 3D;
- - по итогам исследования сформулировать выводы и предложения по совершенствованию практики применения методов искусственного интеллекта в Unity3D.

Методы, теоретическая и практическая значимость:

Методологическая основа: материалистический, диалектический, логические, математический, кибернетический и др. научные методы научного исследования.

Теоретическая значимость темы: вызвана активным ее обсуждением в научном мире. За три с половиной года в системе РИНЦ этим вопросам, в рамках исследований Unity 3D, было посвящено 318 работ российских и иностранных ученых. Это исследования А.Е. Безюка, В.А. Балаева, Т.А. Гришаниной, В.С. Гридчина, В.Ю. Дроздова, В.А. Зориной и других ученых, посвятивших свои труды этой сфере.

Практическая значимость:

1) Поэтапно был создан искусственный интеллект поведения игровых персонажей в Unity 3D, осуществлен анализ результатов использованных методов. Построен интеллект поведения на примерах персонажей (герой, враг, дрон).

2) полученные результаты способны наделить начинающих разработчиков компьютерных игр навыками успешного выбора необходимых игровых характеристик в процессе создания искусственного интеллекта.

Выводы:

1) Искусственный интеллект *определяется международно-правовыми стандартами*, поэтому имеет не только *техническую, но и правовую сущность, сопутствующую политической, социальной или культурной*.

Для разработчиков, ИИ - совокупность технических и программных средств, которые, в целом, составляют интеллектуальную систему, соединяющую в себе базу знаний, решатель задач и интеллектуальный интерфейс взаимодействия с человеком.

По нашему мнению, ИИ - это система или объект, который имитирует поведение человека при выполнении определенных (творческих) задач и способен учиться на основе получаемой информации. Он в результате обучения способен решать ряд задач без прописанного заранее алгоритма.

ПРОБЛЕМЫ ИИ:

- искусственный интеллект и его понятие постоянно развиваются;
- еще недостаточный уровень технологического развития, что тормозит возможности в моделировании поведения человека;
- обострение этических проблем.

2) Направления в развитии ИИ, сформулированные «Дартмутской группой»:

- создание автоматизированных систем;
- программирование компьютеров для использования человеческого языка;
- создание нейронных сетей;
- разработка теории размера вычислений;
- исследование возможностей самообучения машины;
- классификация абстракций;
- изучение творческого мышления.

Классификация ИИ по силе: 1) слабый, 2) сильный, 3) зарождающийся **сверх-искусственный интеллект**, который должен будет выйти за границы человеческого понимания.

Важнейшие НАПРАВЛЕНИЯ РАЗВИТИЯ ИИ в рамках социальных сетей:

- 1) большие данные (Big Data);
- 2) обработка естественного языка (NLP);
- 3) распознавание объектов по накопленным признакам;
- 4) разработка моделей представления знаний в экспертных системах.

3) Для внедрения искусственного интеллекта в компьютерные игры используются плагины, библиотеки, движки. Благодаря им происходит взаимодействие с другими технологиями.

Unity - межплатформенная среда (движок) для разработки компьютерных игр, которая имеет два основных преимущества: визуальную среду разработки и межплатформенную поддержку.

Преимущество Unity - в модульной системе компонентов.

Недостатки Unity: - 1) затруднена совместная работа, поскольку Unity не поддерживает подключение к внешним библиотекам и настраивается непосредственно программистом.

- 2) в Unity не решены **проблемы производительности и потребления памяти**, что связано с рядом архитектурных особенностей.

4) Unity 3D - система разработки 3D-приложений, которая содержит инструментарий для быстрого проектирования простейшего поведения объектов на базе искусственного интеллекта.

В Unity 3D используются **ОСНОВНЫЕ МЕТОДЫ РАЗРАБОТКИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА:**

- 1) простые скрипты;
- 2) машина состояний (State machine);
- 3) ТАУ Алгоритм (теория автоматического управления);
- 4) нейронные сети и др.

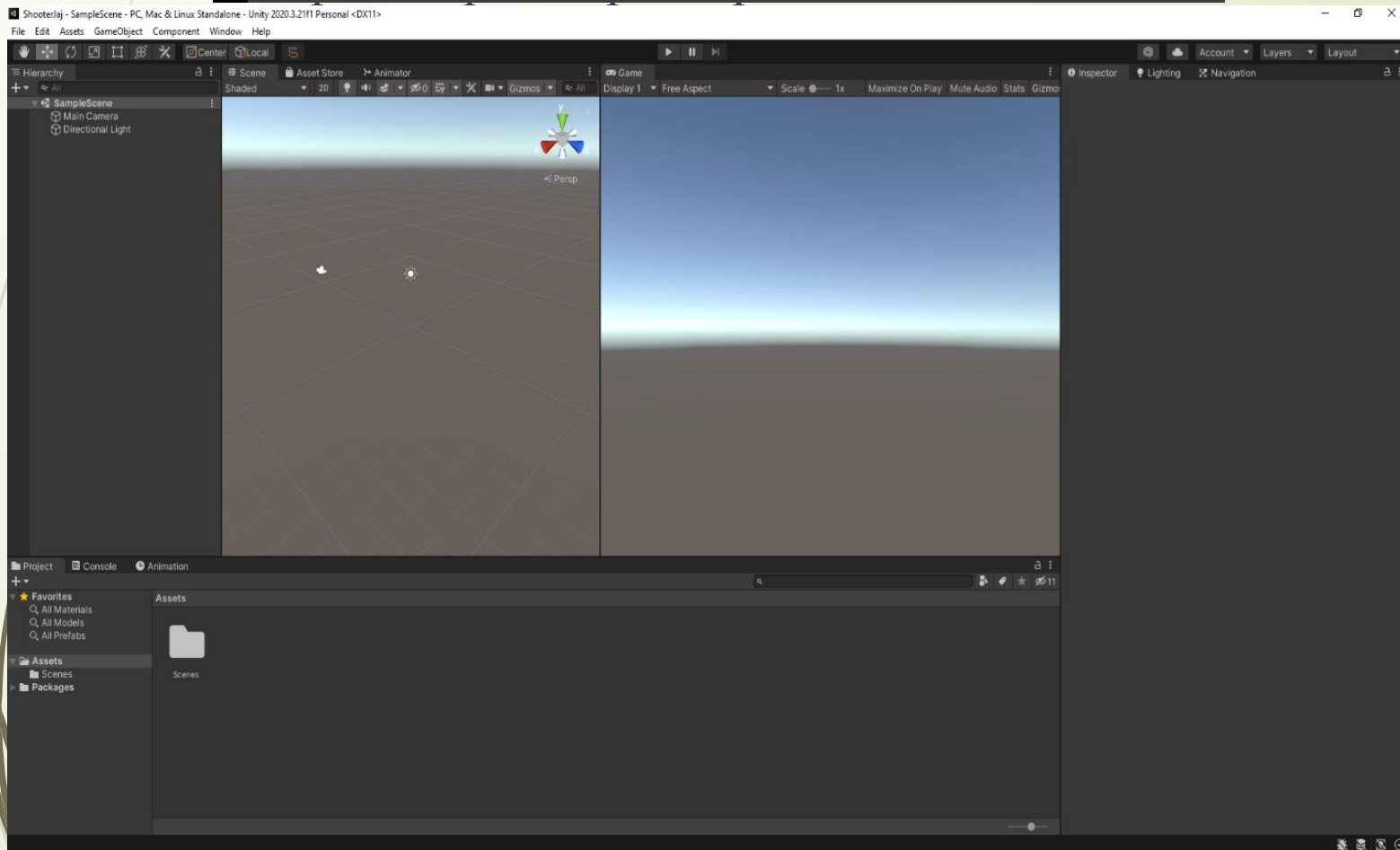
Основные компоненты системы Unity 3D:

- 1) **Компонент NavMesh** (Navigation Mesh) – это структура данных, которая описывает проходимые поверхности игрового мира и позволяет найти путь из одного места в другое.
- 2) **Компонент NavMesh Agent** - *помогает создать виртуальных персонажей, которые избегают друг друга при движении к своей цели.*
- 3) **Компонент Off-Mesh Link** - позволяет включать навигационные ярлыки, которые не могут быть представлены с помощью проходимой поверхности, позволяет описать элементы, не связанные с навигационной сеткой, но участвующие в перемещении (двери, лифты, др. препятствия и порталы).
- 4) **Компонент Nav Mesh Obstacle** - *позволяет описать движущиеся препятствия, которые агенты должны избегать во время навигации на сцене.*

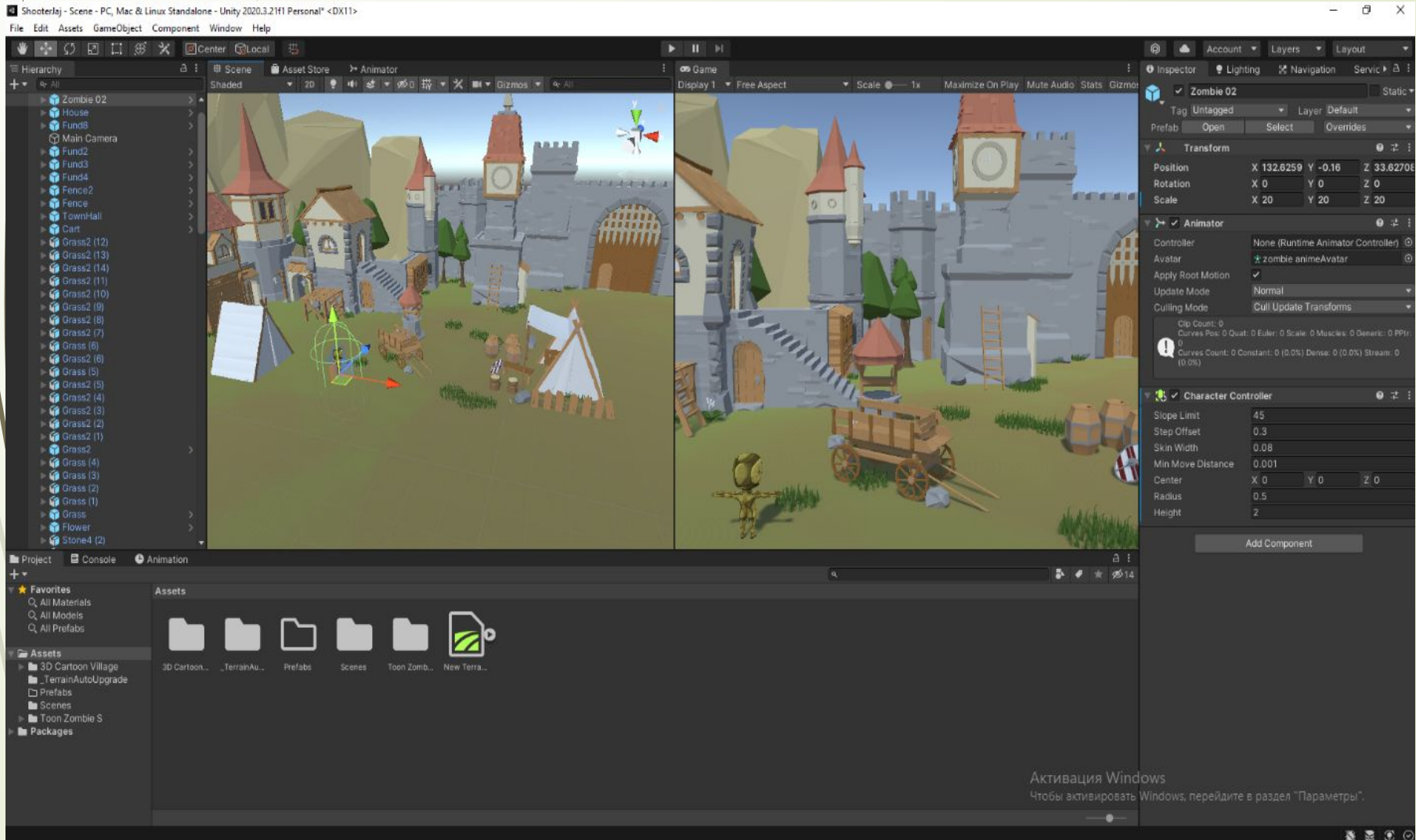
5) Во втором разделе описываем практические действия по созданию игры и применению методов программирования ИИ. Создаем «героя», «врага» и «дрона».

В первую очередь, через Unity Hub в 3D пространстве **создан проект, настроен, добавлены в этот проект ассеты.**

Настроена ориентировка работы на создание ИИ.



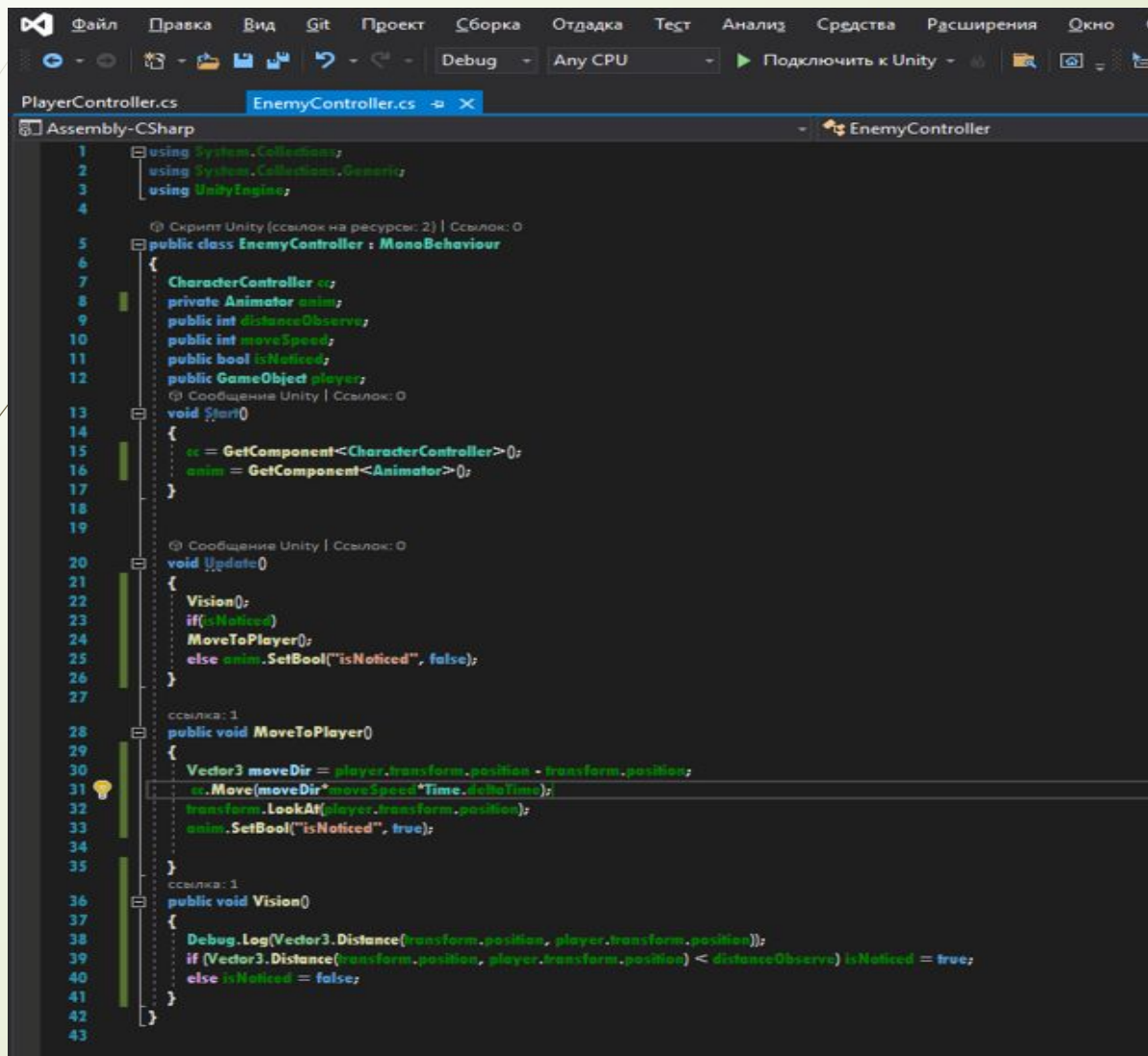
Была подготовлена сцена с окружением и новыми ассетами. Это ландшафт и модели объектов в том числе и ключевого объекта «героя» с предполагаемым искусственным интеллектом.



6) В рамках рассмотрения метода ПРОСТОЙ СКРИПТ:

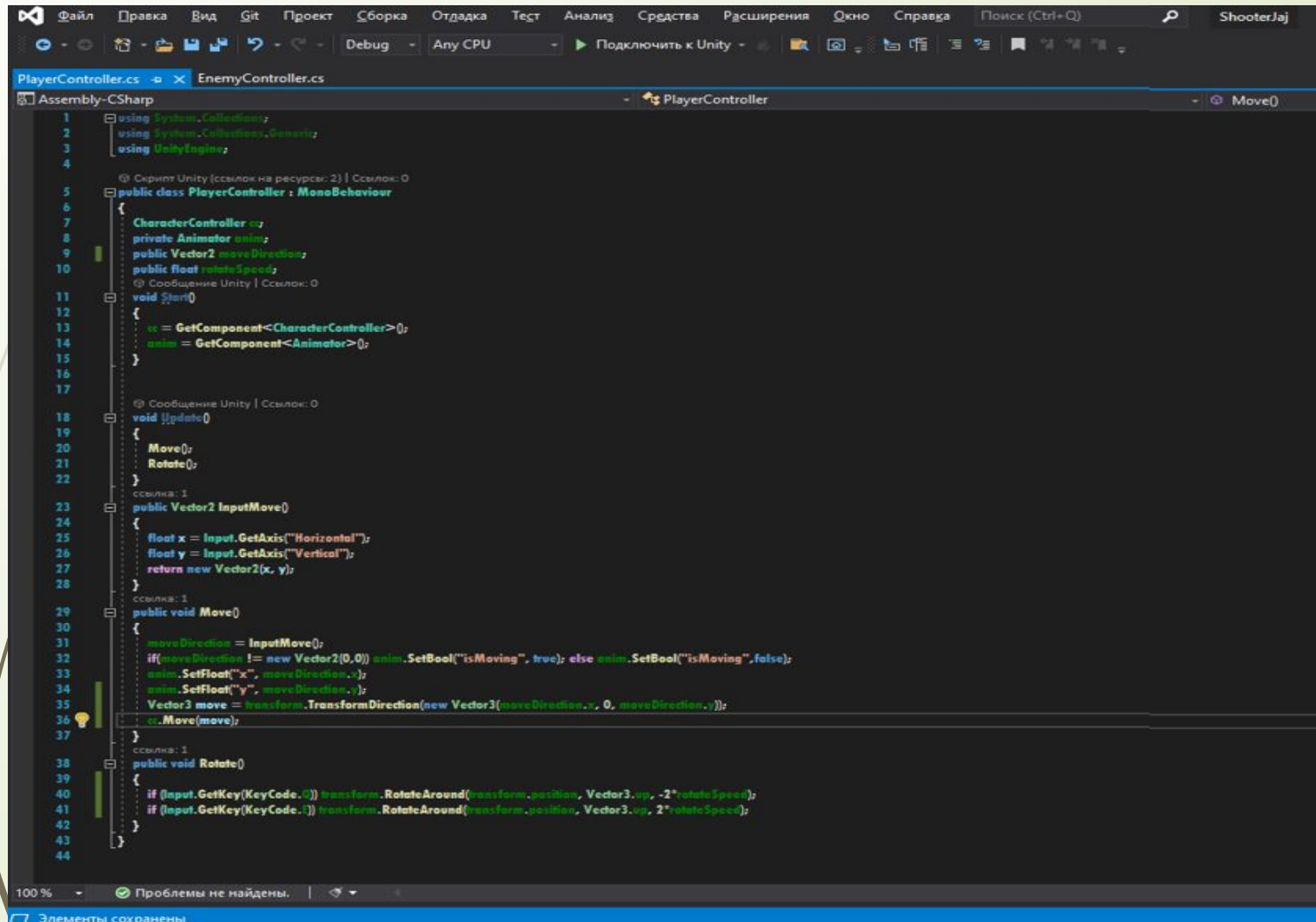
- открываем код в установленной программной среде Microsoft;
- Вводим необходимые свойства и компоненты;
- Переделываем код поиска персонажа, когда игрок уничтожен;
- Наделяем персонажа ИИ способностью хождения;
- Настраиваем редактор с инспектором «героя» и «врага» («зомби»).

Добавляем формы действий для ИИ «врага», чтобы он мог, при условии подхода к цели, ударять с анимацией и через секунду регистрировать удар:




```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class EnemyController : MonoBehaviour
6 {
7     CharacterController cc;
8     private Animator anim;
9     public int distanceObserve;
10    public int moveSpeed;
11    public bool isNoticed;
12    public GameObject player;
13
14    void Start()
15    {
16        cc = GetComponent<CharacterController>();
17        anim = GetComponent<Animator>();
18    }
19
20    void Update()
21    {
22        Vision();
23        if (isNoticed)
24            MoveToPlayer();
25        else anim.SetBool("isNoticed", false);
26    }
27
28    ссылка: 1
29    public void MoveToPlayer()
30    {
31        Vector3 moveDir = player.transform.position - transform.position;
32        cc.Move(moveDir * moveSpeed * Time.deltaTime);
33        transform.LookAt(player.transform.position);
34        anim.SetBool("isNoticed", true);
35    }
36
37    ссылка: 1
38    public void Vision()
39    {
40        Debug.Log(Vector3.Distance(transform.position, player.transform.position));
41        if (Vector3.Distance(transform.position, player.transform.position) < distanceObserve) isNoticed = true;
42        else isNoticed = false;
43    }
44 }
```


Показываем возможность произвести модификацию скрипта и делаем это. При подходе «героя» враг начинает за ним бежать с анимацией и поворотом тела:



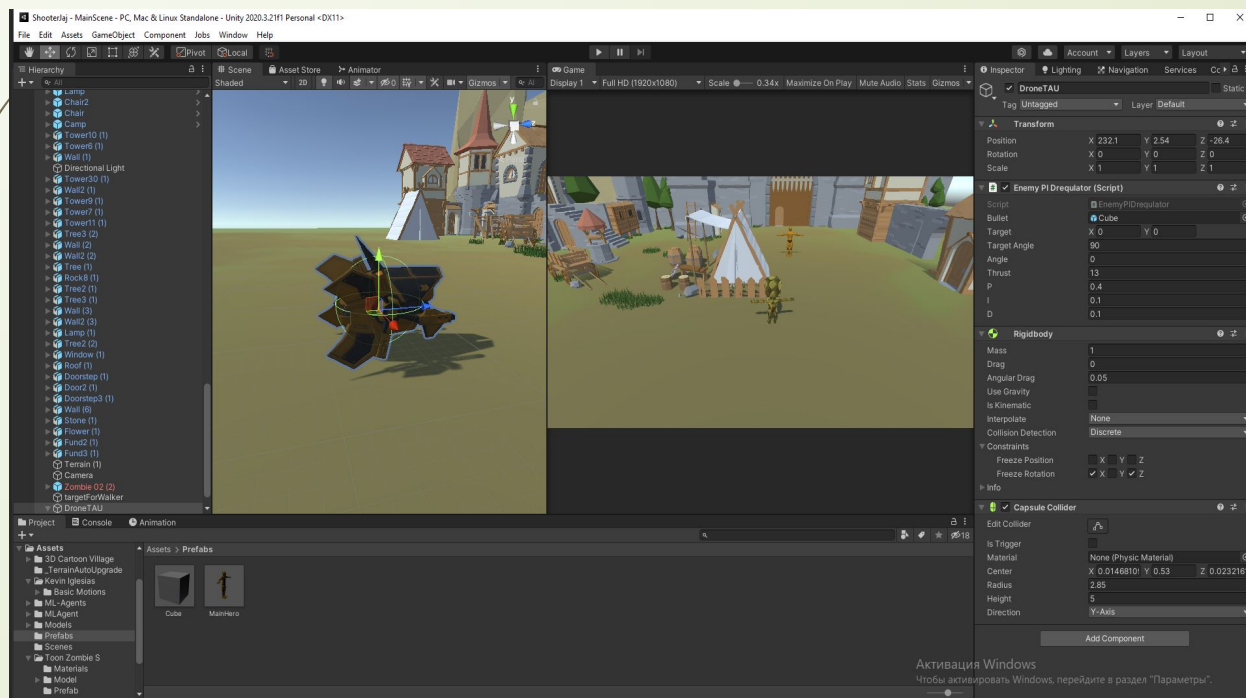
```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 // Скрипт Unity (ссылка на ресурс: 2) | Ссылка: 0
6 public class PlayerController : MonoBehaviour
7 {
8     CharacterController cc;
9     private Animator anim;
10    public Vector2 moveDirection;
11    public float rotateSpeed;
12
13    // Сообщение Unity | Ссылка: 0
14    void Start()
15    {
16        cc = GetComponent<CharacterController>();
17        anim = GetComponent<Animator>();
18    }
19
20    // Сообщение Unity | Ссылка: 0
21    void Rotate()
22    {
23        Move();
24        Rotate();
25    }
26
27    // Ссылка: 1
28    public Vector2 InputMove()
29    {
30        float x = Input.GetAxis("Horizontal");
31        float y = Input.GetAxis("Vertical");
32        return new Vector2(x, y);
33    }
34
35    // Ссылка: 1
36    public void Move()
37    {
38        moveDirection = InputMove();
39        if (moveDirection != new Vector2(0, 0)) anim.SetBool("isMoving", true); else anim.SetBool("isMoving", false);
40        anim.SetFloat("x", moveDirection.x);
41        anim.SetFloat("y", moveDirection.y);
42        Vector3 move = transform.TransformDirection(new Vector3(moveDirection.x, 0, moveDirection.y));
43        cc.Move(move);
44    }
45
46    // Ссылка: 1
47    public void Rotate()
48    {
49        if (Input.GetKey(KeyCode.D)) transform.RotateAround(transform.position, Vector3.up, -2*rotateSpeed);
50        if (Input.GetKey(KeyCode.A)) transform.RotateAround(transform.position, Vector3.up, 2*rotateSpeed);
51    }
52 }
```

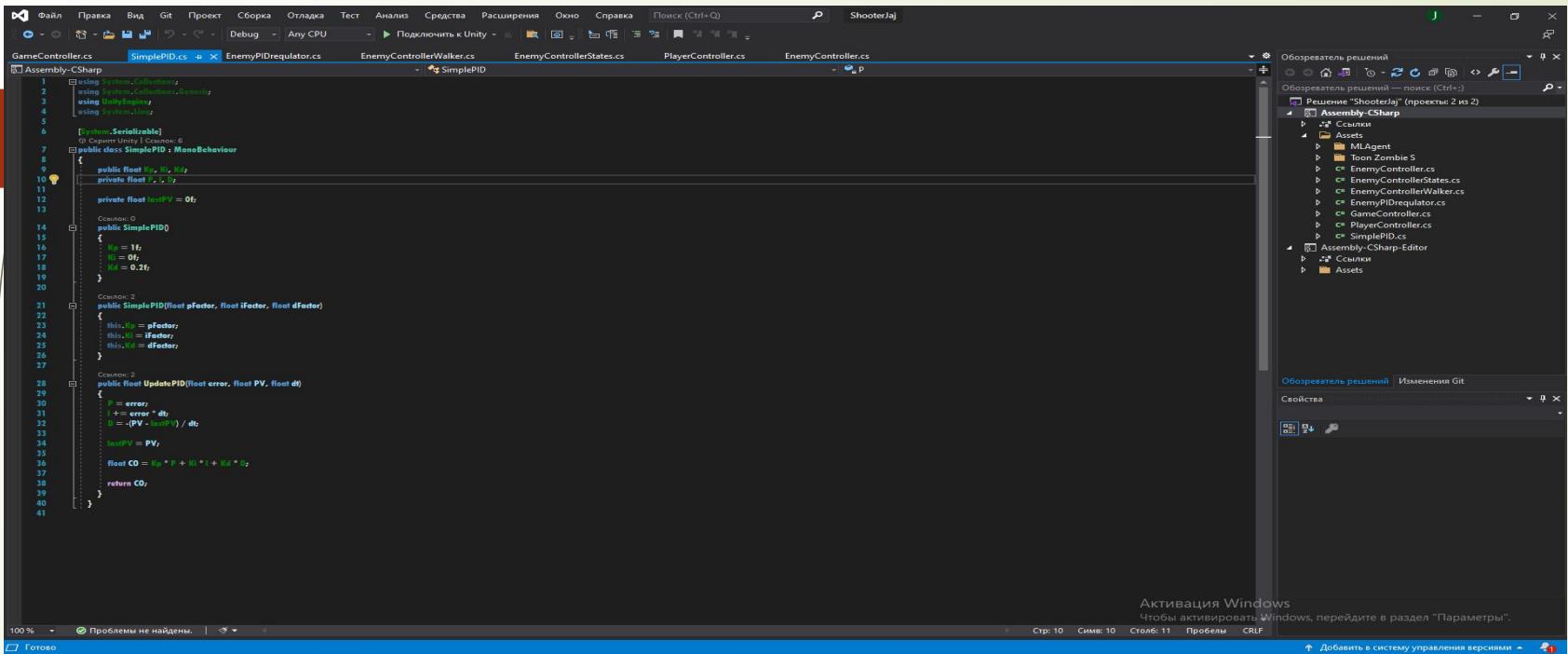



7) Реализовать метод «машина состояний» пытаемся без задействования сторонних программ, так как это занимает неоправданно много времени.

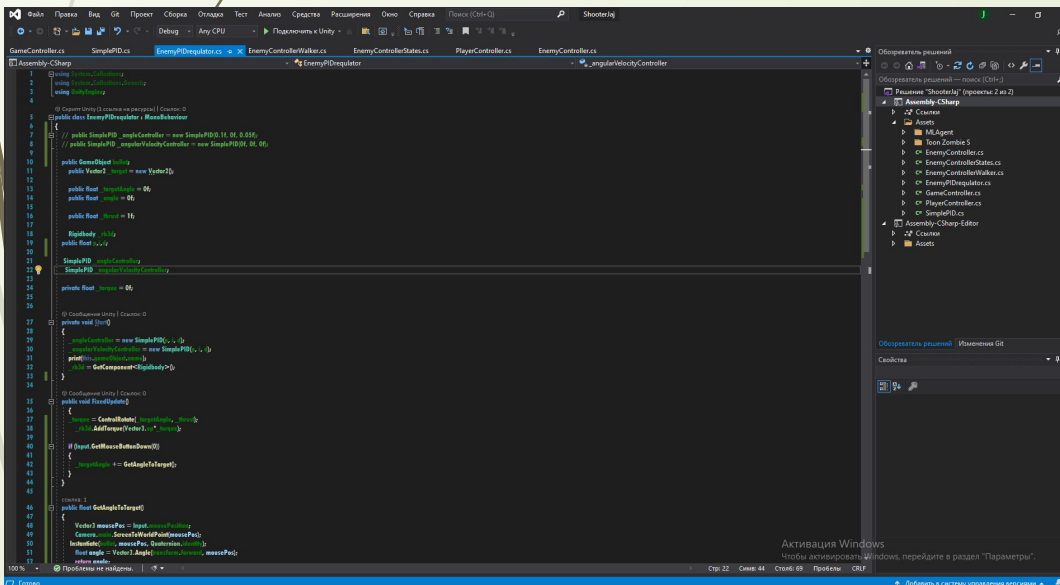
- Производим объявление перечисления контроля кода.
- Делаем небольшие упрощения в сокращении кода

8) С использованием ТАУ реализуем в игровой программе настройку поведения для одного из объектов («дрона»). Программируем его повороты в установленном мышью направлении по так называемому методу пид-регуляции, который стабилизирует и корректирует поворот по собственной формуле:





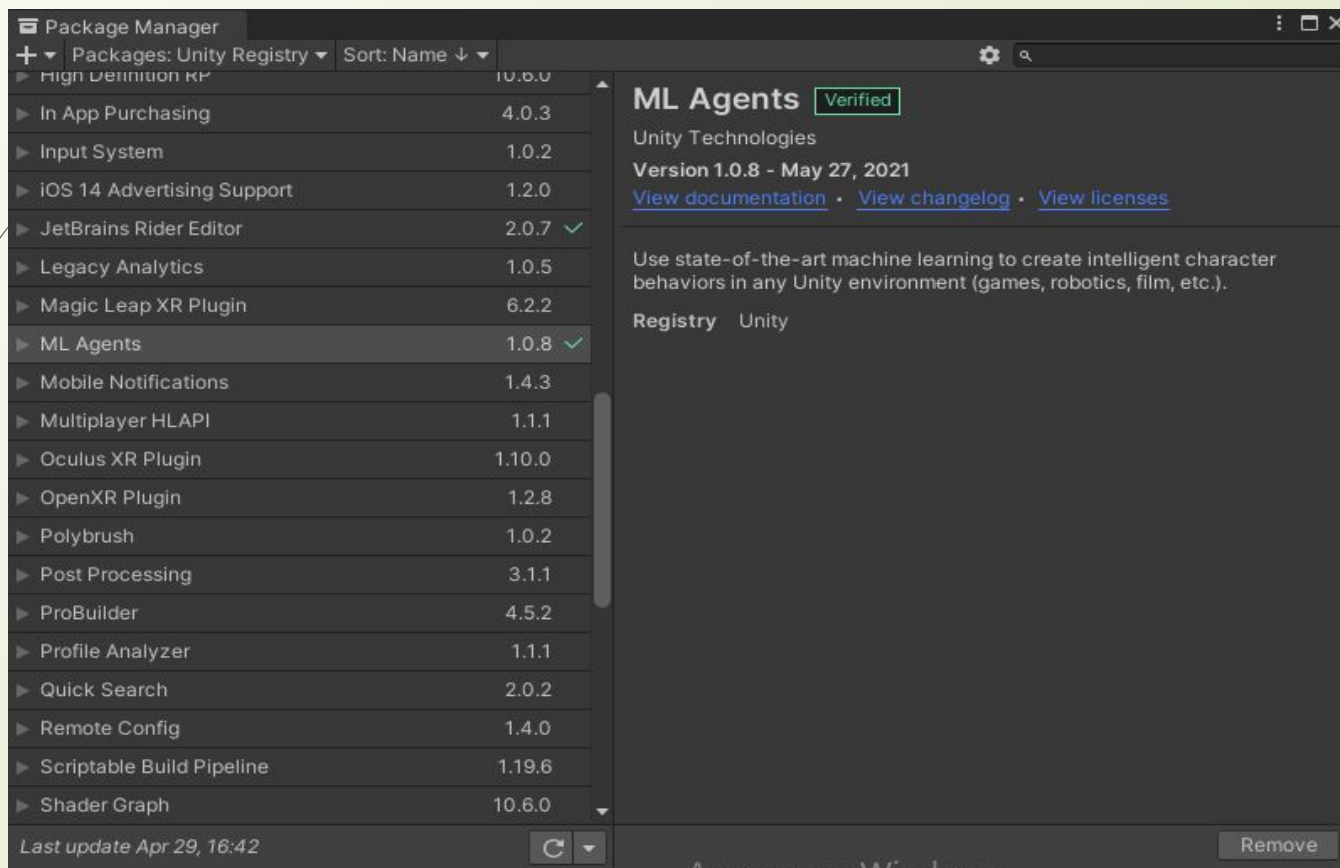
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".



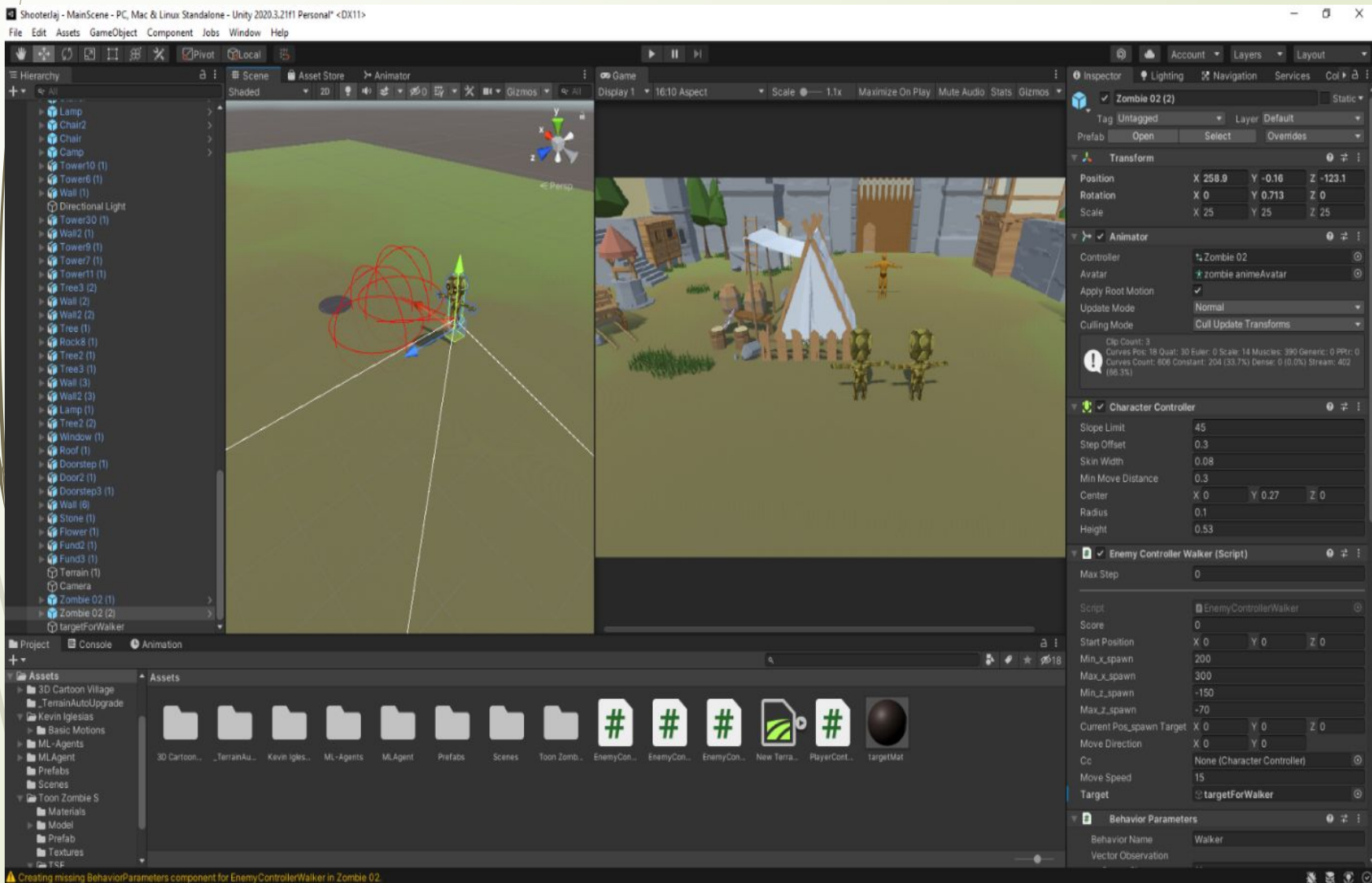
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".

Программируем поведение «дрона» с применением PID-регуляции

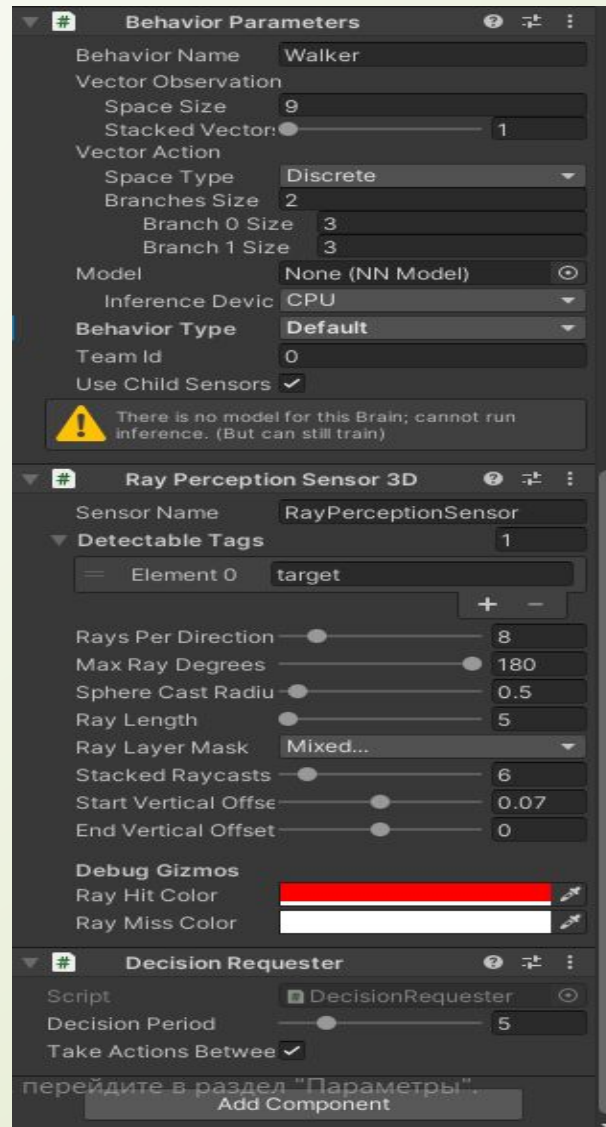
9) Нейросети. Обеспечиваем разнообразное и более сложное человекоподобное поведение с помощью нейросетей. Производим настройку в окне менеджера пакетов с данными пакета ML Agents.



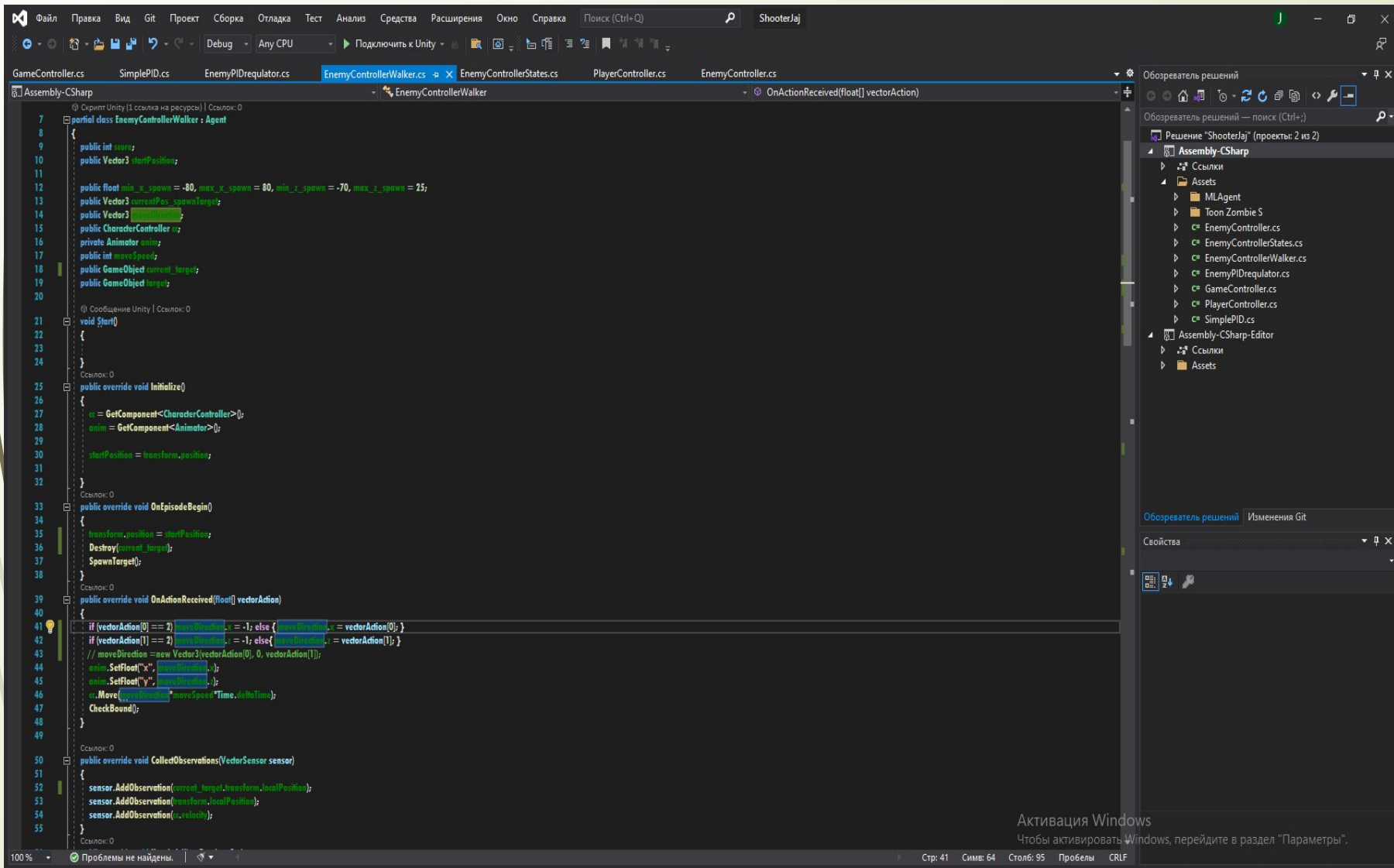
Выставляем параметры в окне инспектора «врага Агента»



Используем окно инспектора «врага» с
встроенными функциями из пакета:



Окно поведения. Настраиваем поведение «Агента»:



```
7 partial class EnemyControllerWalker : Agent
8 {
9     public int score;
10    public Vector3 startPosition;
11
12    public float min_x_spawn = -80, max_x_spawn = 80, min_z_spawn = -70, max_z_spawn = 25;
13    public Vector3 currentPos, spawnTarget;
14    public Vector3 moveDirection;
15    public CharacterController cc;
16    private Animator anim;
17    public int moveSpeed;
18    public GameObject current_target;
19    public GameObject target;
20
21    void Start()
22    {
23    }
24
25    public override void Initialize()
26    {
27        cc = GetComponent<CharacterController>();
28        anim = GetComponent<Animator>();
29
30        startPosition = transform.position;
31    }
32
33    public override void OnEpisodeBegin()
34    {
35        transform.position = startPosition;
36        Destroy(current_target);
37        SpawnTarget();
38    }
39
40    public override void OnActionReceived(float[] vectorAction)
41    {
42        if (vectorAction[0] == 2) { moveDirection.x = -1; } else { moveDirection.x = vectorAction[0]; }
43        if (vectorAction[1] == 2) { moveDirection.z = -1; } else { moveDirection.z = vectorAction[1]; }
44        // moveDirection = new Vector3(vectorAction[0], 0, vectorAction[1]);
45        anim.SetFloat("x", moveDirection.x);
46        anim.SetFloat("y", moveDirection.z);
47        cc.Move(moveDirection * moveSpeed * Time.deltaTime);
48        CheckBound();
49    }
50
51    public override void CollectObservations(VectorSensor sensor)
52    {
53        sensor.AddObservation(current_target.transform.localPosition);
54        sensor.AddObservation(transform.localPosition);
55        sensor.AddObservation(cc.velocity);
56    }
57 }
```

Обозреватель решений

Обозреватель решений — поиск (Ctrl+)

Решение "ShooterLaj" (проекты: 2 из 2)

- Assembly-CSharp
 - Ссылки
 - Assets
 - MLAgent
 - Toon Zombie S
 - EnemyController.cs
 - EnemyControllerStates.cs
 - EnemyControllerWalker.cs
 - EnemyPIDregulator.cs
 - GameController.cs
 - PlayerController.cs
 - SimplePID.cs
- Assembly-CSharp-Editor
 - Ссылки
 - Assets

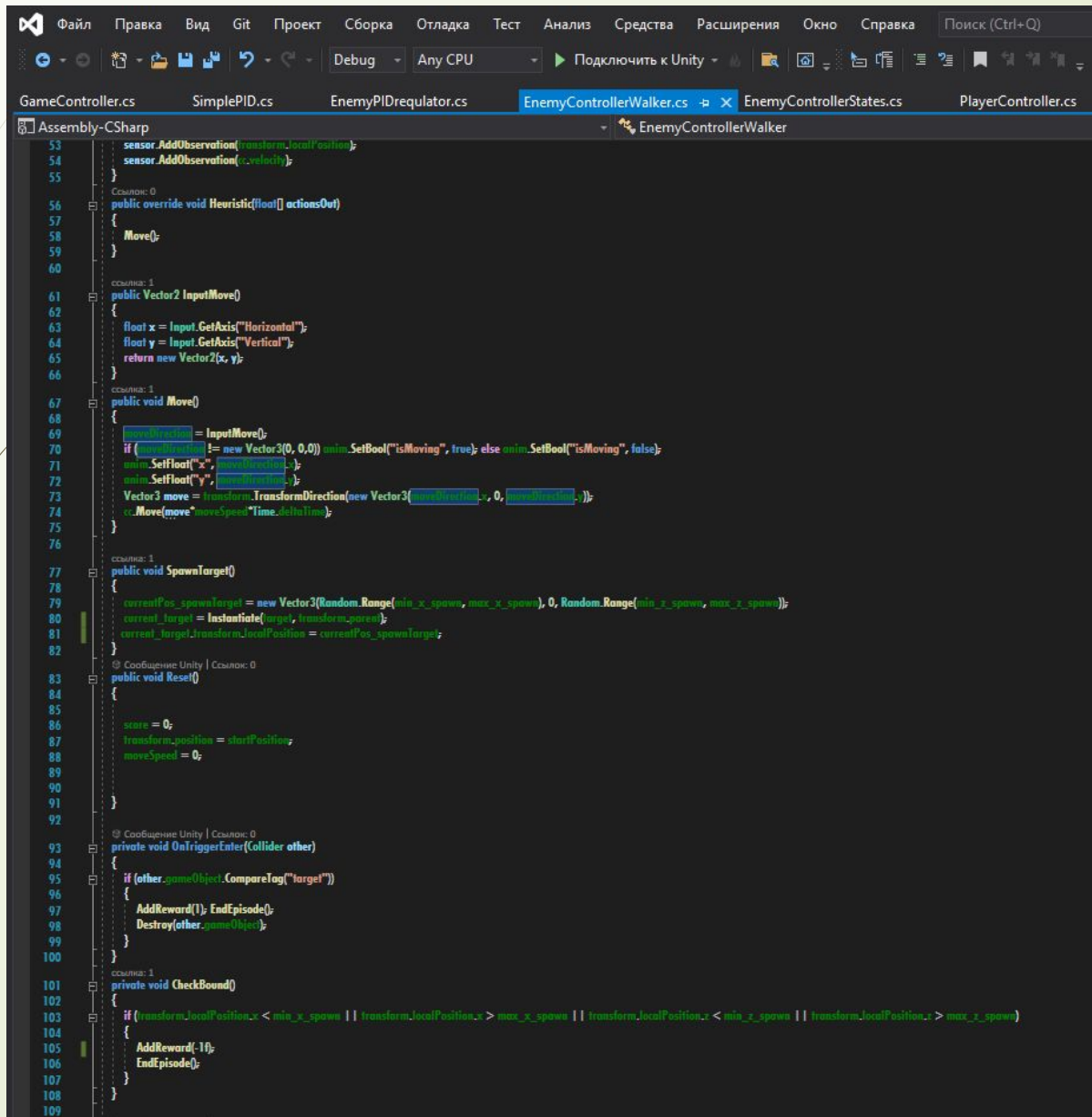
Обозреватель решений | Изменения Git

Свойства

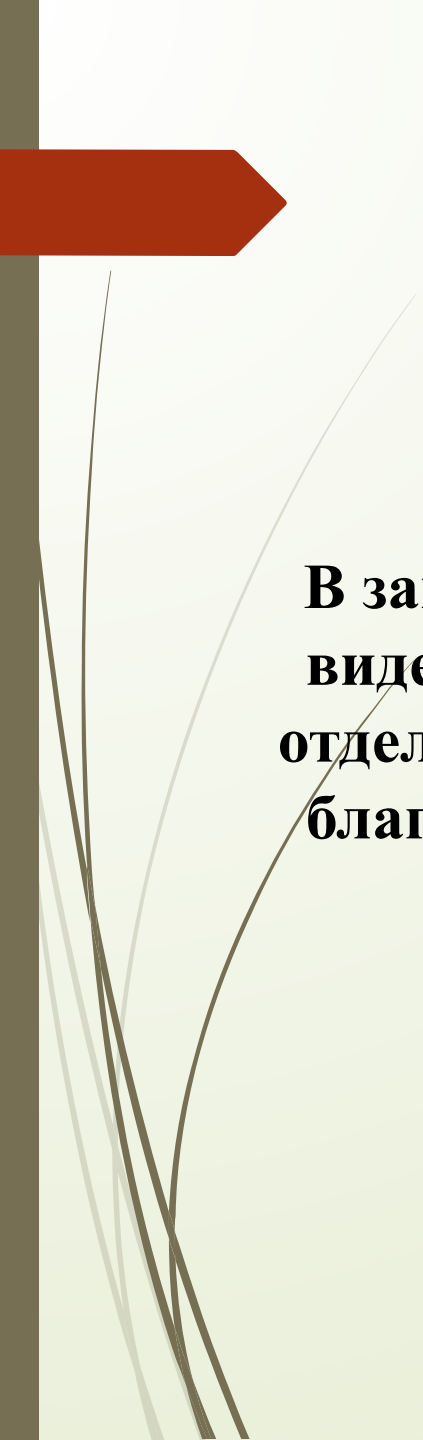
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".

100% Проблемы не найдены. Стр: 41 Симв: 64 Столб: 95 Пробелы CRLF

Окно поведения - настраиваем поведение «Агента»:



```
53 sensor.AddObservation(transform.localPosition);
54 sensor.AddObservation(v.velocity);
55 }
56
57 Ссылка: 0
58 public override void Heuristic(float[] actionsOut)
59 {
60     Move();
61 }
62
63 Ссылка: 1
64 public Vector2 InputMove()
65 {
66     float x = Input.GetAxis("Horizontal");
67     float y = Input.GetAxis("Vertical");
68     return new Vector2(x, y);
69 }
70
71 Ссылка: 1
72 public void Move()
73 {
74     moveDirection = InputMove();
75     if (isMoving != new Vector3(0, 0, 0)) anim.SetBool("isMoving", true); else anim.SetBool("isMoving", false);
76     anim.SetFloat("x", moveDirection.x);
77     anim.SetFloat("y", moveDirection.y);
78     Vector3 move = transform.TransformDirection(new Vector3(moveDirection.x, 0, moveDirection.y));
79     v.Move(move * moveSpeed * Time.deltaTime);
80 }
81
82 Ссылка: 1
83 public void SpawnTarget()
84 {
85     currentPos_spawnTarget = new Vector3(Random.Range(min_x_spawn, max_x_spawn), 0, Random.Range(min_z_spawn, max_z_spawn));
86     current_target = Instantiate(target, transform.position);
87     current_target.transform.localPosition = currentPos_spawnTarget;
88 }
89
90 Сообщение Unity | Ссылка: 0
91 public void Reset()
92 {
93     score = 0;
94     transform.position = startPosition;
95     moveSpeed = 0;
96 }
97
98 Сообщение Unity | Ссылка: 0
99 private void OnTriggerEnter(Collider other)
100 {
101     if (other.gameObject.CompareTag("target"))
102     {
103         AddReward(1); EndEpisode();
104         Destroy(other.gameObject);
105     }
106 }
107
108 Ссылка: 1
109 private void CheckBound()
110 {
111     if (transform.localPosition.x < min_x_spawn || transform.localPosition.x > max_x_spawn || transform.localPosition.z < min_z_spawn || transform.localPosition.z > max_z_spawn)
112     {
113         AddReward(-1);
114         EndEpisode();
115     }
116 }
```

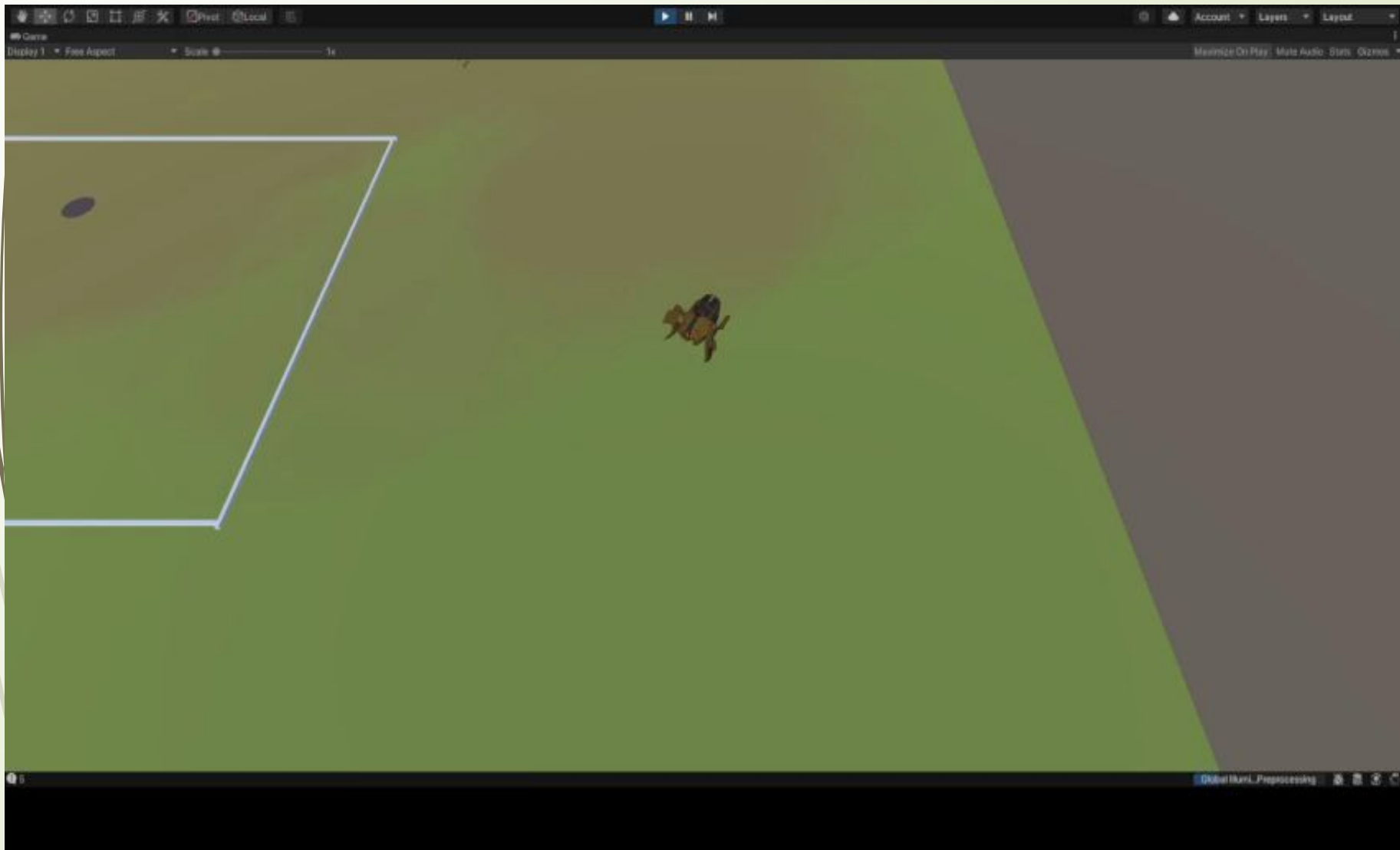


В завершение выступления представляю ряд видео созданного искусственного интеллекта отдельных персонажей, который был настроен благодаря использованию четырех основных методов:

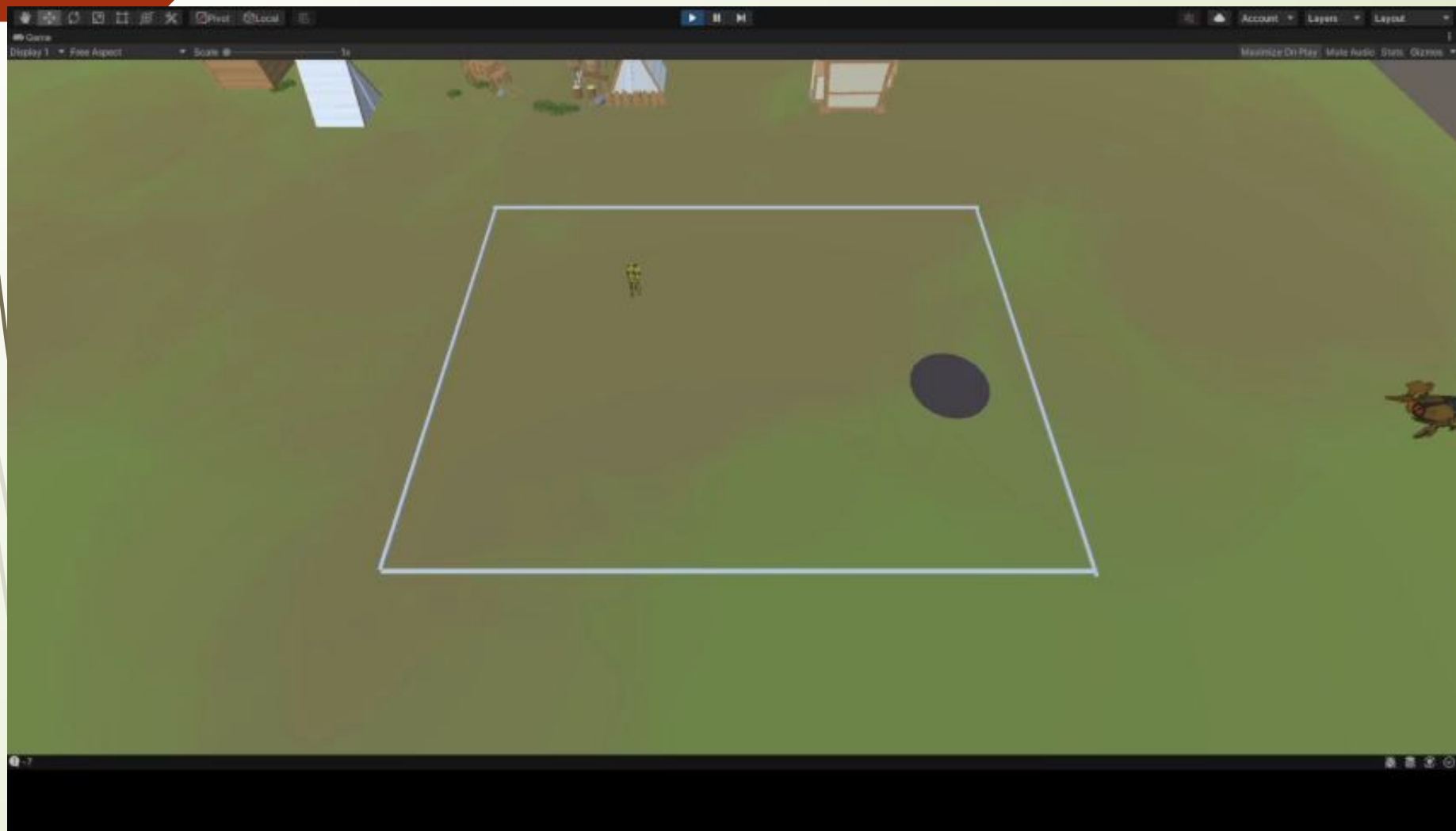
ГЕРОЙ



ДРОН



ВРАГ





Спасибо за внимание!

mydiv [Новости и Статьи](#) [Программы](#) [Игры](#) [Драйверы](#) [Техника](#) [Форум](#)

Windows [Разработка](#) > [Игровые движки и конструкторы игр](#) > [Unity 3D](#) > [Описание](#)

Обзор

Unity 3D

 [Скачать](#) Версия для  Windows

[? Как скачивать и устанавливать](#) [? Задать вопрос о программе](#)

Специальные требования

- графическая карта с поддержкой [DirectX 10](#).

Системы:	Windows
Размер:	2,3 Гб
Автор:	Unity Technologies (Сайт)
Языки:	Английский
Лицензия:	Freeware (Бесплатная)
Обновление:	13.05.2022
Версия:	2022.1
Имя файла:	UnitySetup64.exe