

#### МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

## «МИРЭА – Российский технологический университет»

#### РТУ МИРЭА

Институт информационных технологий (ИИТ) Кафедра инструментального и прикладного программного обеспечения (ИиППО)

# ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ

по дисциплине «Моделирование сред и разработка приложений виртуальной и дополненной реальности»

# Итоговый проект

Студент группы	ИКБО-20-21, Хитров H.C.	
		(подпись)
Преподаватель	Егоркин Н.Е.	
		(подпись)
Отчет представлен	« » 2024 г.	

# СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
ПРАКТИЧЕСКАЯ ЧАСТЬ	4
ЗАКЛЮЧЕНИЕ	. 13

## **ВВЕДЕНИЕ**

Unity — это популярная платформа для разработки 2D и 3D игр, а также виртуальной и дополненной реальности. Она поддерживает множество платформ, включая мобильные устройства, ПК, консоли и VR/AR. Unity предлагает интуитивно понятный интерфейс, мощные инструменты и скриптинг на языке С#. Благодаря большому сообществу, обширной документации и Unity Asset Store, разработчики могут быстро создавать высококачественные игры и интерактивные приложения. Unity используется не только в игровой индустрии, но и в архитектуре, кино, медицине и образовании.

В отчете рассматриваются свледующие ключевые аспекты разработки

- Проектирование игры
- Реализация сцены
- Реализация действий игрока
- Реализация NPC
- Пользовательский интерфейс

#### ПРАКТИЧЕСКАЯ ЧАСТЬ

Целью данного проекта является реализация динамической однопользовательской мини-игры «BlowEmUp», в которой перед игроком стоит задача разобраться с пиратами, захватившими трактир, в качестве вооружения у пользователя есть бесконечный запас бомб. Чтобы игра казалась не слишком легкой и более реалистичной, когда пираты замечают игрока, они начинают сближаться с ним для нанесения ударов.

На рисунке 1 представлен игровой уровень, построенный при помощи tilemap. Есть два типа поверхностей, ground и wall. Также для расстановки окружения (свечи, бочки, окна и т.д.) были созданы соответствующие префабы. В сцену был добавлен звуковой эффект, громкость которого можно регулировать в меню при нажатии на клавишу "Esc".

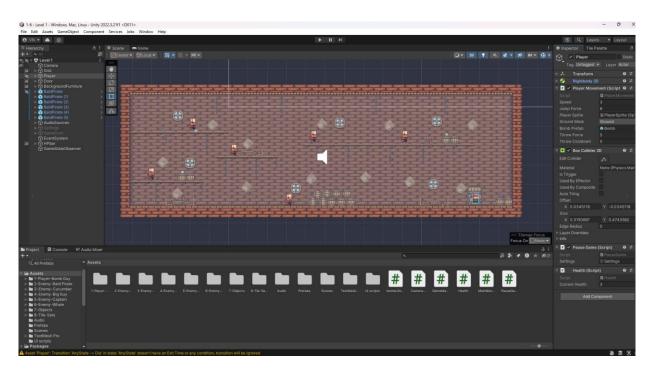


Рисунок 1 — Игровой уровень

Далее был создан объект игрока, ему были присвоены компоненты для задания физики и столкновений с другими предметами, а также скрипт, реализующий действия (Рисунки 2-3)

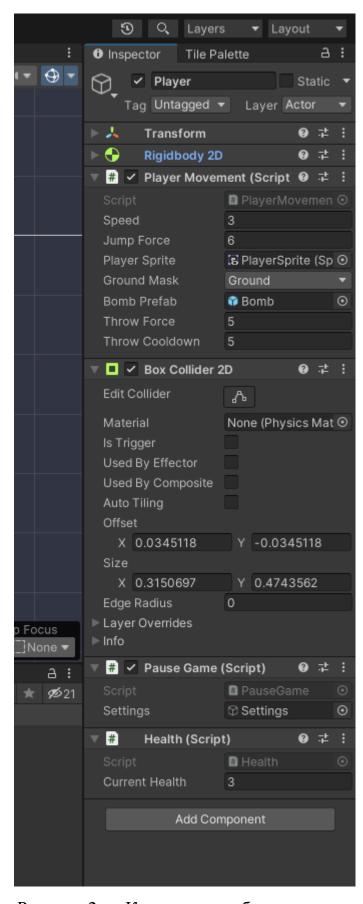


Рисунок 2 — Компоненты объекта игрока

```
₽> □ ··
 EXPLORER
                        PlayerMovement.cs X
∨ OPEN EDITO... 🖺 🖺 🗟
                        Assets > 1-Player-Bomb Guy > C PlayerMovement.cs
  X 😅 PlayerMovement...
                                    ривіїє тібає спромеобійомі = эт; // кулдаўн ороска
∨ 1-6
         回の哲却
                                   private Vector3 direction = Vector3.left;
 > .vscode

✓ Assets

                                   void Start()

✓ 1-Player-Bomb Guy

                                        gameObject.tag = "Player";
   > 1-Idle
                                       _rigidbody = GetComponent<Rigidbody2D>();
   > 2-Run
                                        animations = GetComponentInChildren<PlayerAnimation>();
   > 3-Fall
                                        _boxCollider = GetComponent<BoxCollider2D>();
   > 4-Jump
   > 7-Hit
   > 8-Dead Hit
                                   private void Update() {
                                        if (throwCooldown > 0) {
   > 9-Dead Ground
                                            throwCooldown -= Time.deltaTime;
   > 10-Door In
   > 11-Door Out
                                       Move();
   HealthDisplay.cs
                                        CheckGround();
   ■ Player.controller
                                        if (Input.GetKeyDown(KeyCode.Space)) {
                                            if (_isGrounded) {
   C PlayerAnimation.cs
                                                Jump();
   C PlayerMovement.cs
                                                _animations.Jump();
  > 2-Enemy-Bald Pirate
  > 3-Enemy-Cucumber
  > 4-Enemy-Big Guy
                                        if (Input.GetKeyDown(KeyCode.E) && throwCooldown <= 0)</pre>
  > 5-Enemy-Captain
  > 6-Enemy-Whale
                                            ThrowBomb();
                                            throwCooldown = 3f;
  > 7-Objects
  > 8-Tile-Sets
  > Audio
                                        _animations.isMoving = _isMoving;
  > Prefabs
                                        _animations.isFlying = !_isGrounded;
   > Scenes
   > TextMesh Pro
                                   private void CheckGround() {
  > UI scripts
                                         isGrounded = (bool)Physics2D.BoxCast(_boxCollider.bounds
```

Рисунок 3 — Пример кода реализующего действия игрока

После был создан объект врага, ему также, как и игроку были присвоены компоненты для задания физики и столкновений с другими предметами и также скрипт, реализующий действия (Рисунки 4-5). После для удобства добавления врагов на сцену из объекта был составлен префаб врага.

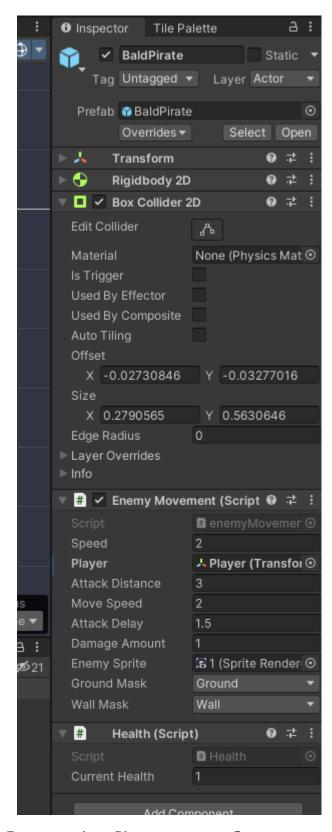


Рисунок 4 — Компоненты объекта врага

```
File Edit Selection View \cdots \leftarrow \Rightarrow
                                                                                              .. □ ~
    EXPLORER
                          c enemyMovement.cs ×

√ OPEN EDITORS

                          Assets > 2-Enemy-Bald Pirate > C enemyMovement.cs
    X 😻 enemyMovemen...
          中の甘む
                                          ChooseAction();
   > .vscode
                                          Move();
   Assets

✓ 1-Player-Bomb Guy

                                     private void ChooseAction() {
      > 1-Idle
                                          _actionTime -= Time.deltaTime;
      > 2-Run
                                          if (!CheckMoveAllowed()) {
      > 3-Fall
                                           _action = Mathf.Floor(Random.Range(0, 3));
      > 4-Jump
                                             _direction = _direction == 0 ? 1 : 0;
_actionTime = Mathf.Floor(Random.Range(1, 3));;
      > 7-Hit
      > 8-Dead Hit
                                         else if (_actionTime <= 0) {
      > 9-Dead Ground
                                             _action = Mathf.Floor(Random.Range(0, 3));
      > 10-Door In
                                              _direction = Mathf.Floor(Random.Range(0, 2));
      > 11-Door Out
                                              _actionTime = 1.5f;
      C HealthDisplay.cs

■ Player.controller

     C PlayerAnimation.cs
     C PlaverMovement.cs
                                    private bool CheckMoveAllowed() {

✓ 2-Enemy-Bald Pirate

                                          float rayside = _direction == 0 ? -1 : 1;
      > 1-Idle
                                          float distance = _boxCollider.bounds.size.x/2 + .1f;
      > 2-Run
                                          float height = _boxCollider.bounds.size.y;
      > 4-Jump
                                         Vector2 vertstart = new Vector2(transform.position.x + distance * rayside, tra
      > 5-Fall
                                        RaycastHit2D downhit = Physics2D.Raycast(vertstart, Vector2.down, height, grou
      > 6-Ground
      > 7-Attack
                                          Vector2 horzstart = new Vector2(transform.position.x, transform.position.y);
                                          Vector2 horzdir = _direction == 0 ? Vector2.left : Vector2.right;
      > 8-Hit
                                          RaycastHit2D sidehit = Physics2D.Raycast(horzstart, horzdir, distance, wallMas
      > 9-Dead Hit
      > 10-Dead Ground
                                          Debug.DrawLine(horzstart, sidehit.point, Color.red);

    ■ Attack.anim

      ■ Enemy.controller
                                          if (!downhit || sidehit) {
                                              return false;
      C EnemyAnimation.cs
```

Рисунок 5 — Пример кода реализующего действия врага

Для реализации пользовательского интерфейса было разработано меню игры, и также внутриигровое меню для регулировки громкости звука и возможности приостановки игры (Рисунки 6-7).

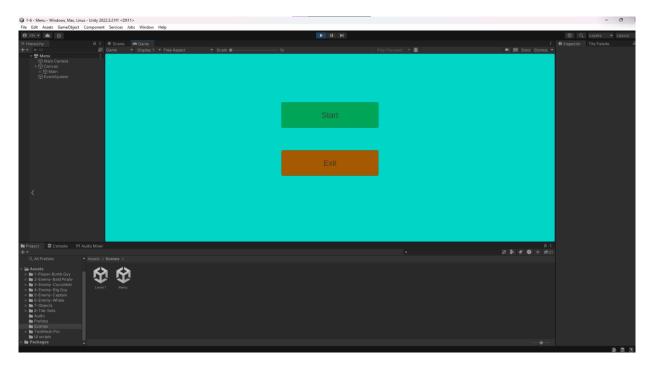


Рисунок 6 — Стартовое меню игры

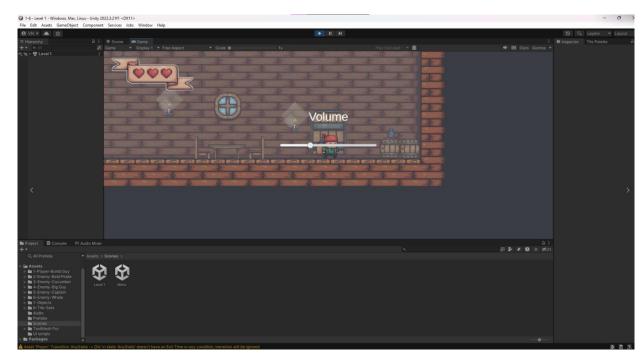


Рисунок 7 — Внутриигровое меню

Также был добавлен индикатор жизней игрока в левом верхнем углу, регулируемый специальным скриптом (Рисунки 8-9).

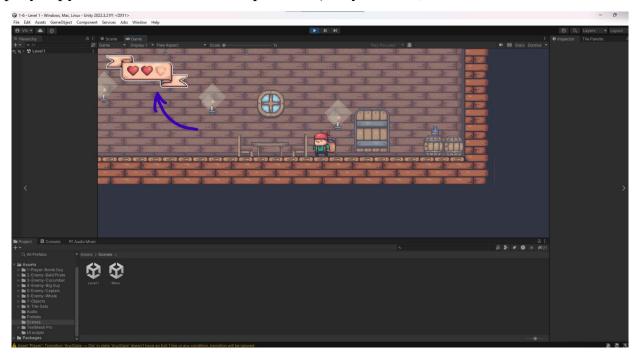


Рисунок 8 — Индикатор жизней игрока

```
	imes File Edit Selection View \cdots \leftarrow \rightarrow
                                                                                                      . □ ·
                        ... C HealthDisplay.cs X
       EXPLORER
                               Assets > 1-Player-Bomb Guy > ♥ HealthDisplay.cs > ...

∨ OPEN EDITORS

                                 using System.Collections;
using System.Collections.Generic;
        X ♥ HealthDisplay.cs...
             ច្ចេជ្ជ
                                  3 using UnityEngine;
وړ
       Assets

✓ 1-Player-Bomb Guy

                                      public class HealthDisplay : MonoBehaviour
         > 1-Idle
                                           1 reference public GameObject player; // Ссылка на компонент Health игрока
         > 2-Run
œP
          > 3-Fall
          > 4-Jump
                                           public GameObject[] hearts; // Массив объектов-сердечек
          > 7-Hit
                                           private Health playerHealth;
          > 8-Dead Hit
Д
          > 9-Dead Ground
                                           0 references
void Start() {
          > 10-Door In
         > 11-Door Out
                                               playerHealth = player.GetComponent<Health>();
         C HealthDisplay.cs
          ■ Player.controller
         C PlayerAnimation.cs
                                           void Update()
         C PlayerMovement.cs

✓ 2-Enemy-Bald Pirate

                                               int heartsToShow = Mathf.Clamp(playerHealth.currentHealth, 0, hearts.Length);
         > 1-Idle
         > 2-Run
         > 4-Jump
                                               for (int i = 0; i < hearts.Length; i++)</pre>
         > 5-Fall
         > 6-Ground
                                                    hearts[i].SetActive(i < heartsToShow);</pre>
         > 7-Attack
         > 8-Hit
         > 9-Dead Hit
          > 10-Dead Ground

    ■ Attack.anim

          ≡ Enemy.controller
         C EnemyAnimation.cs
```

Рисунок 9 — Скрипт реализующий отображение жизней игрока

# ЗАКЛЮЧЕНИЕ

В ходе разработки игры-платформера «BlowEmUp» на платформе Unity были рассмотрены и успешно реализованы ключевые аспекты, необходимые для создания увлекательного и стабильного игрового опыта. Использование Unity предоставило нам мощные инструменты для создания и редактирования игровых объектов, а также возможности кроссплатформенной разработки, что значительно упростило процесс адаптации игры под различные устройства.