Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра прикладной математики и кибернетики

Расчетно-графическое задание
по дисциплине «Программирование для мобильный устройств»

Выполнил:

студент группы ИП-811
Разумов Д.Б.

Работу проверил:

Павлова У.В.

Новосибирск 2021

Содержание

Задание

В рамках выполнения РГР необходимо написать игровое приложение. Сюжет игры выбирается согласно варианту. Сюжет может быть произвольным (по согласованию с преподавателем).

Я выбрал вариант игры в жанре платформер. Игрок может передвигаться по 2D-уровням, прыгать, собирать монетки, сражаться с противниками.

Описание программы

В начале разработки игры я скачал несколько бесплатных ассетов и создал и создал первый уровень с помощью Tilemap.

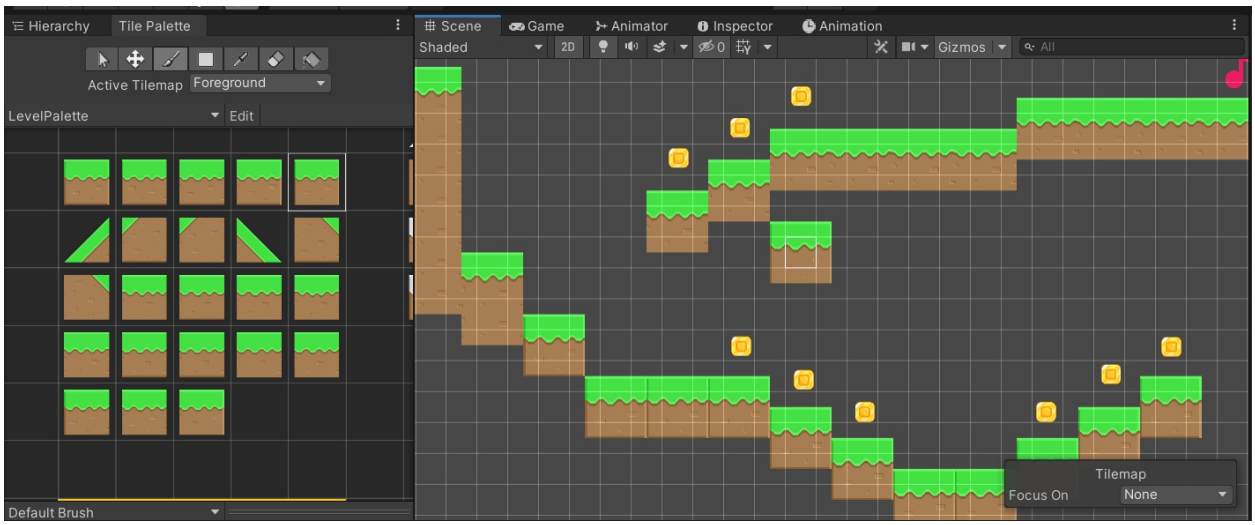

Рисунок 1 - окна с Tile Palette и Tilemap

Далее я добавил главного персонажа, добавил анимации (бега, прыжка и т.д.), Rigidbody 2D (для физики), Box Collider 2D (для «столкновений» с другими объектами) и создал скрипт **MainCharacter.cs**, в котором я задал скорость персонажа, изменение положения модели при перемещении, изменение анимаций. Также я вложил MainCamera и задний фон в объект персонажа, чтобы камера и задний фон всегда «следовали» за ним.

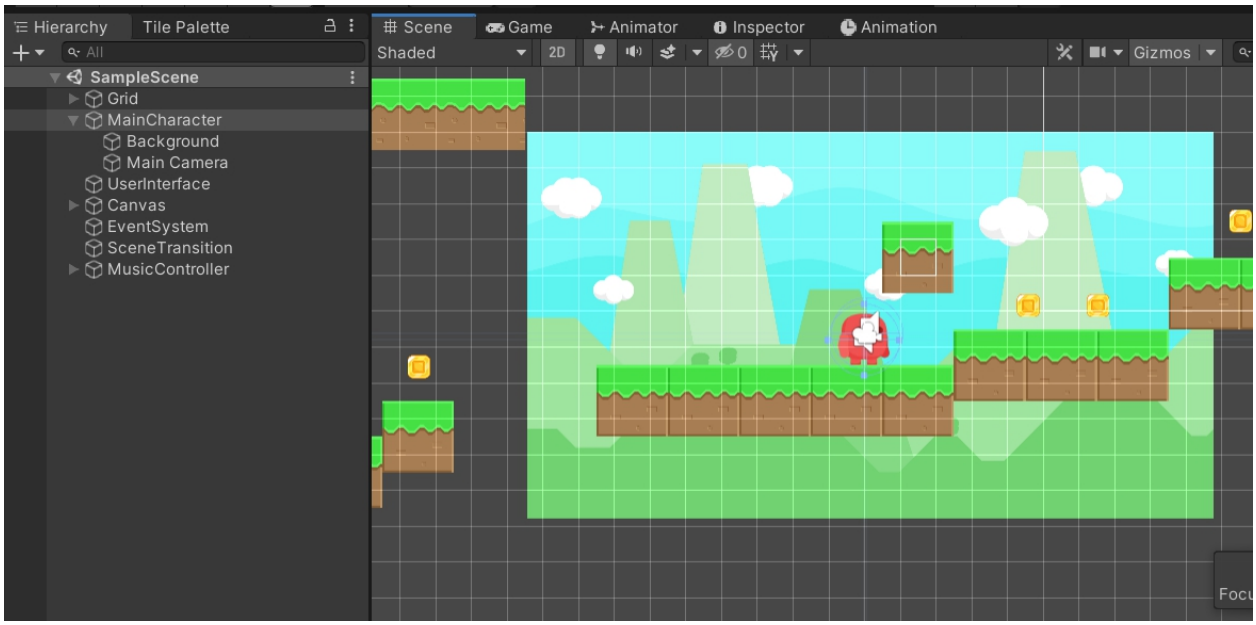

Рисунок 2

Затем я добавил монетки: поместил изображение монетки на игровую сцену, добавил скрипт **Coin.cs**, в котором определил метод OnTriggerEnter2D() так, что при столкновении с главным персонажем объект монетки уничтожается, а счетчик монеток из класса **PlayerMoney.cs** увеличивается на 1. Последний скрипт я создал для того, чтобы на экране отображался счетчик монет в виде текста (через ссылку на объект текста и метод OnGUI()).

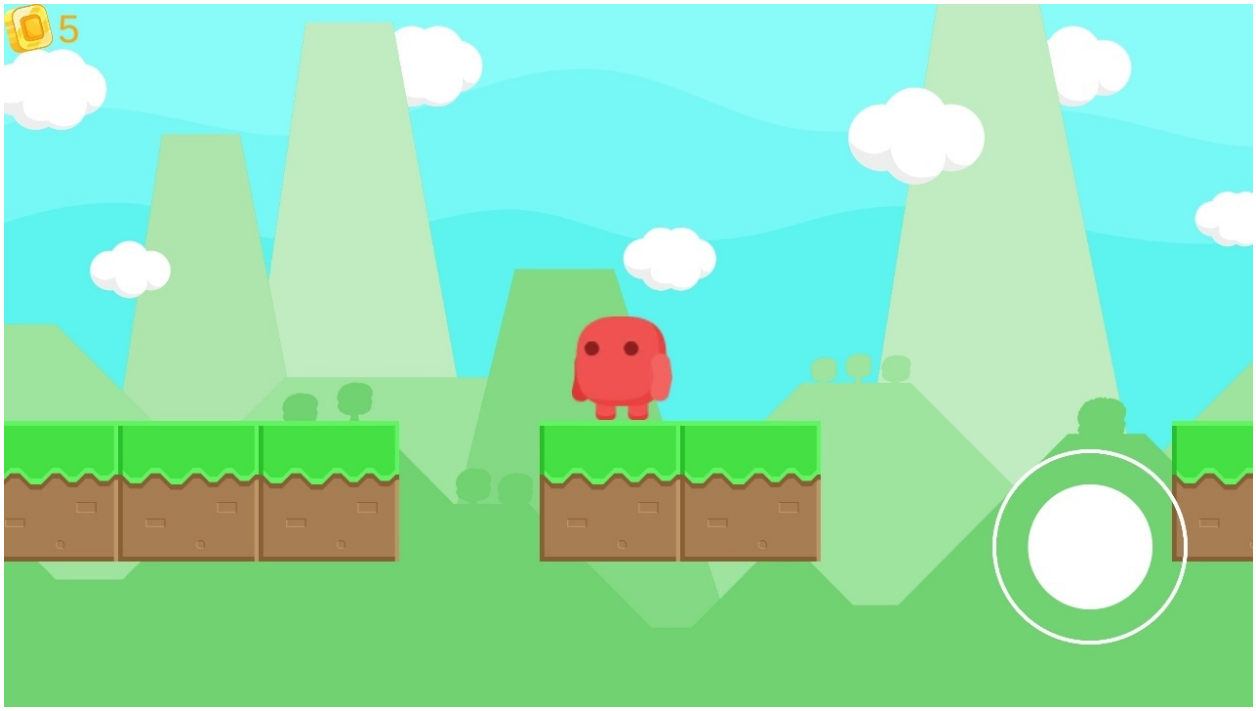

Рисунок 3 - счетчик монет сверху-слева

Я добавил музыку и звуковой эффект прыжка. Для этого я создал пустой объект MusicController, в который добавил два компонента AudioSource и скрипт **MusicController.cs**, в котором 1-ый AudioSource проигрывает звуки, 2-ой - музыку (чтобы они могли звучать одновременно). В этом скрипте я определил два публичных метода для проигрывания звуков и музыки. С помощью них можно проигрывать музыку через другие объекты в игре. Например, я создал ноту, при столкновении с которой меняется музыка в игре (скрипт **ChangeMusic.cs**).

Я скачал из UnityStore джойстик, который добавил в игру. Отслеживать движения можно с помощью переменных класса Joystick.Horizontal и Joystick.Vertical.

Объекты на каждой сцене в игре имеют разные Sorting Layer (чтобы, например, отображать задний фон сзади всех других объектов) и Layer. Слои используются для разных задач. На слое «Ground» персонаж может стоять, а на слое «InvisibleWater» (который не видно и размещен в низу игровых уровней) игровая сцена загружается заново.

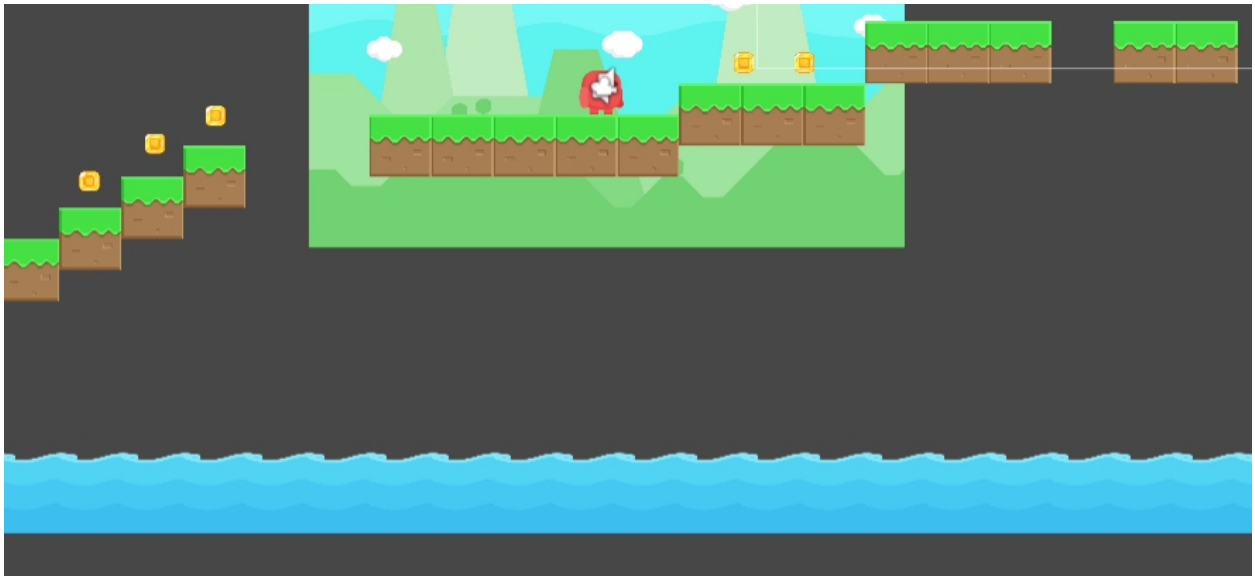

Рисунок 4 - невидимая вода в низу уровня

Я сделал 3 игровых уровня, перемещения через которые осуществляется с помощью объектов в виде порталов, к которым прикреплен скрипт **SceneTransition.cs**. Каждый игровой уровень я сделал как отдельную сцену в Unity.

На втором уровне я добавил неподвижных монстров (скрипт **Enemy.cs**) и hitpoints для персонажа. На третьем уровне я добавил лед, на котором скорость персонажа увеличивается. По мерей добавления нового функционала также дополнялся и скрипт **MainCharacter.cs**.

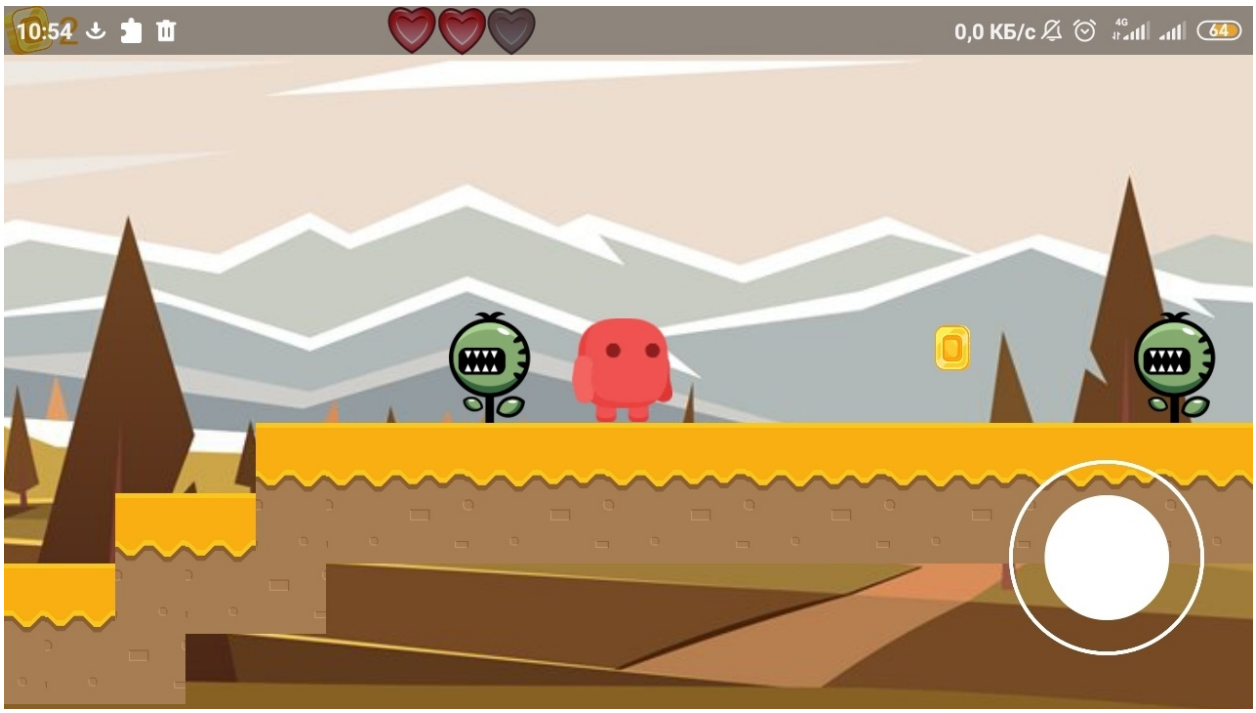

Рисунок 5 - второй уровень

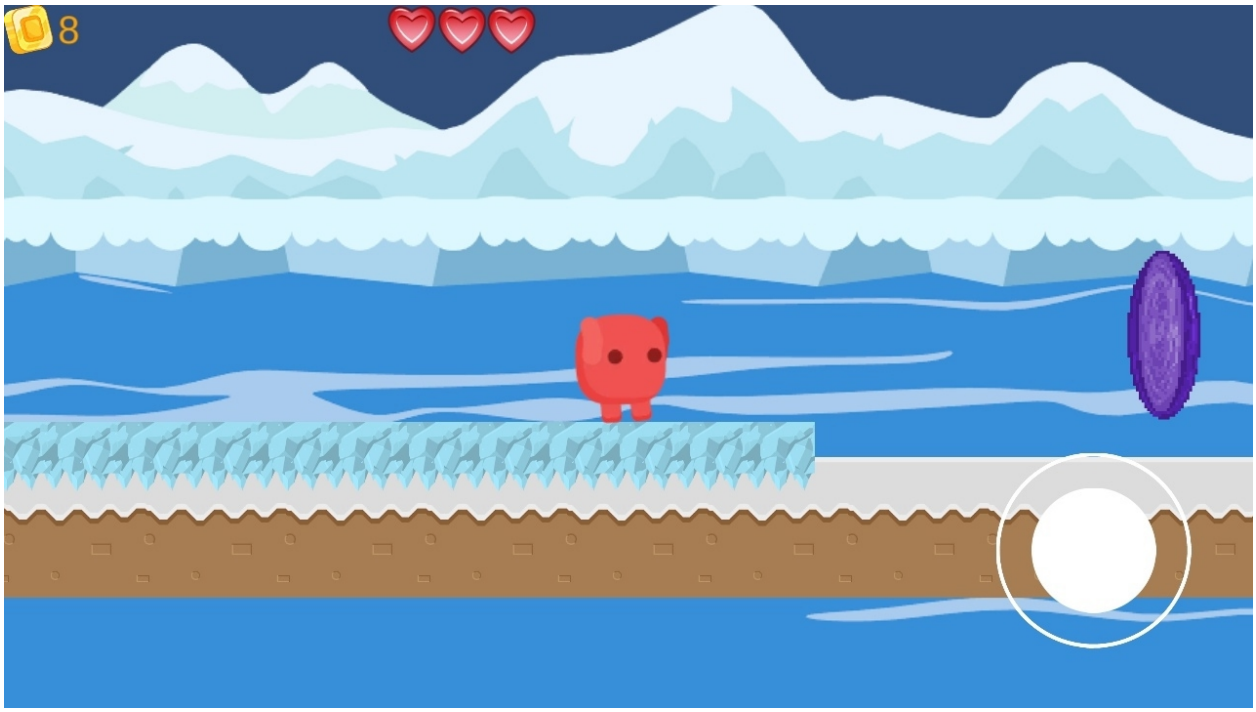

Рисунок 6 - третий уровень

Листинг скриптов

MainCharacter.cs

```
using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;


public class MainCharacter : MonoBehaviour

{

    /**

    ** Ускорение игрока

    **/

    /*[Header("Player velocity")]

    // Ось Ox

    public int xVelocity = 7;

    // Ось Oy

    public int yVelocity = 10;*/

    public Vector2 playerVelocity = new Vector2(7, 7);


    // слой земли с foreground

    [SerializeField] private LayerMask ground;

    // Физическое поведение (тело) объекта

    private Rigidbody2D rigidBody2dComponent;

    // Коллайдер, проверка на столкновения

    private Collider2D collider2dComponent;

    // доступ к спрайту

    private SpriteRenderer spriteRendererComponent;

    // доступ к анимации

    private Animator animatorComponent;

    // джойстик для управления движением
```

```
public Joystick joystick;
//
[SerializeField] public AudioClip audioClip; // звук прыжка
private AudioSource audioSource; // объявляем компонент аудио источника
// hitpoints:
public int hitpoints = 3;
[SerializeField] private Image[] hearts;
[SerializeField] private Sprite aliveHeart;
[SerializeField] private Sprite deadHeart;



//                              0      1        2        3
private enum AnimationState { idle, running, jumping, falling };
// Значением по-умолчанию ставим idle (0), как мы и задавали в аниматоре
private AnimationState currentAnimationState = AnimationState.idle;



// Start (https://docs.unity3d.com/ScriptReference/MonoBehaviour.Start.html)
private void Start()
{
    // Получаем доступ к Rigidbody2D объекта Player
    rigidBody2dComponent = gameObject.GetComponent<Rigidbody2D>();
    collider2dComponent = gameObject.GetComponent<Collider2D>();
    spriteRendererComponent = gameObject.GetComponent<SpriteRenderer>();
    animatorComponent = gameObject.GetComponent<Animator>();
    audioSource = GetComponent<AudioSource>();
}

private void Update()
{
    updatePlayerPosition();
```

```
    }


    //FIXED UPDATE??


    bool inAir;
    // Обновляем местоположение игрока
    private void updatePlayerPosition()
    {
        float horizontaMoveInput = joystick.Horizontal;//Input.GetAxis("Horizontal");
        float jumpInput = joystick.Vertical;//Input.GetAxis("Jump");


        // Движение влево
        if (horizontaMoveInput < 0)
        {
            rigidBody2dComponent.velocity = new Vector2(-playerVelocity.x,
rigidBody2dComponent.velocity.y);
            spriteRendererComponent.flipX = true;
            // Движение вправо
        }
        else if (horizontaMoveInput > 0)
        {
            rigidBody2dComponent.velocity = new Vector2(playerVelocity.x,
rigidBody2dComponent.velocity.y);
            spriteRendererComponent.flipX = false;
            // Если персонаж стоит на земле и не двигается, отключаем инерцию
        }
        else if (collider2dComponent.IsTouchingLayers(ground))
        {
            rigidBody2dComponent.velocity = Vector2.zero;
        }


        // Если нажата клавиша прыжка и персонаж касается земли - прыгаем
```

```
        if (jumpInput > 0.5 && !inAir)//collider2dComponent.IsTouchingLayers(ground))
        {
            inAir = true;
            rigidBody2dComponent.velocity = new Vector2(rigidBody2dComponent.velocity.x,
playerVelocity.y);
            audioSource.Play();//PlayOneShot(audioClip, 0.4f);
        }

        // Вызываем менеджер анимаций
        SetAnimationState();
    }

    // Выбираем текущую анимацию
    private void SetAnimationState()
    {
        // Персонаж касается земли
        if (collider2dComponent.IsTouchingLayers(ground))
        {
            // При помощи Mathf.Abs получаем модуль значения ускорения (если бежим влево,
оно отрицательное)
            // Если оно больше 0.1 (не стоим на месте), то персонаж бежит
            if (Mathf.Abs(rigidBody2dComponent.velocity.x) > 0.1f)
            {
                currentAnimationState = AnimationState.running;
            }
            else
            {
                // Если нет - стоим на месте
                currentAnimationState = AnimationState.idle;
            }
            // Персонаж не касается земли
        }
```

```
        else
        {
            // Ставим текущей анимацией прыжок
            currentAnimationState = AnimationState.jumping;

            if (currentAnimationState == AnimationState.jumping)
            {
                // Если усорение уходит в отрицательное значение, значит персонаж падает
вниз
                if (rigidBody2dComponent.velocity.y < .1f)
                {
                    currentAnimationState = AnimationState.falling;
                }
            }
            else if (currentAnimationState == AnimationState.falling)
            {
                // Если он коснулся земли, то персонаж переходит в состояние спокойствия
                if (collider2dComponent.IsTouchingLayers(ground))
                {
                    currentAnimationState = AnimationState.idle;
                }
            }
        }

        // Измениям значение state в аниматоре
        animatorComponent.SetInteger("state", (int)currentAnimationState);
    }

    void Hurt()
    {
        hitpoints--;
        if (hitpoints <= 0)
```

```
            SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
        for (int i = 1; i <= hitpoints; i++)
        {
            hearts[i].sprite = aliveHeart;
        }
        for (int i = hitpoints; i <= 3; i++)
        {
            hearts[i].sprite = deadHeart;
            //hearts[i].enabled = false;
        }
    }

    void OnCollisionEnter2D(Collision2D collision)
    {
        if (collision.gameObject.layer == LayerMask.NameToLayer("Ground"))
        {
            inAir = false;
            playerVelocity.x = 7;
            return;
        }
        if (collision.gameObject.layer == LayerMask.NameToLayer("InvisibleWater"))
        { //если столкнулся с невидимой водой на "дне" уровня, то запустить уровень
заново
            hitpoints = 3;
            SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
            return;
        }
        if (collision.gameObject.layer == LayerMask.NameToLayer("IceLayer"))
        { //если столкнулся с невидимой водой на "дне" уровня, то запустить уровень
заново
            playerVelocity.x = 17;
            return;
```

```
        }

        Enemy enemy = collision.collider.GetComponent<Enemy>();
        if (enemy != null)
        {
            foreach (ContactPoint2D point in collision.contacts)
            {
                if (point.normal.y >= 0.6f)
                {
                    enemy.EnemyHurt();
                }
                else
                {
                    Hurt();
                }
            }
        }

    }
}
```

Coin.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class Coin : MonoBehaviour
{
    public GameObject mainCharacter; //компонент, который отвечает за главного героя
    void Start()
```

```
    {
        mainCharacter = GameObject.Find("MainCharacter"); //найти компонент главного
героя
    }

    // Активация триггера при попадании в него объекта
    void OnTriggerEnter2D(Collider2D collision)
    {
        /**
        * Проверяем, тэг объекта, активировавшего триггер
        * Если его тэг "Player", то условия выполнено
        **/
        if (collision.gameObject.tag == "MainCharacter")
        {
            Destroy(gameObject); //очевидно, уничтожить монетку
            mainCharacter.GetComponent<PlayerMoney>().money_sum += 1; //получить скрипт
главного героя и изменить в ней money_sum
        }
    }
}
```

PlayerMoney.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class PlayerMoney : MonoBehaviour
{
       public int money_sum; // переменная для монет

       [SerializeField]
```

```
	public Text TextMoney; //в данную ссылку будем переносить текстовую информацию о монетах



	void Start()
	{
		money_sum = 0; // по умолчанию, при запуске сцены количество монет = 0
	}


	void OnGUI()
	{
		//GUI.Label(new Rect(10, 10, 100, 100), "Score: " + money_sum.ToString());
		TextMoney.text = money_sum.ToString();
	}
}
```

MusicController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;


public class MusicController : MonoBehaviour
{
	// Audio players components.
	public AudioSource EffectsSource;
	public AudioSource MusicSource;
	public AudioClip standartMusic;


	// Random pitch adjustment range.
	public float LowPitchRange = .95f;
	public float HighPitchRange = 1.05f;
```

```
        // Singleton instance.
        public static MusicController Instance = null;


        // Initialize the singleton instance.
        private void Awake()
        {
                // If there is not already an instance of SoundManager, set it to this.
                if (Instance == null)
                {
                        Instance = this;
                }
                //If an instance already exists, destroy whatever this object is to enforce
the singleton.
                else if (Instance != this)
                {
                        Destroy(gameObject);
                }


                //Set SoundManager to DontDestroyOnLoad so that it won't be destroyed when
reloading our scene.
                DontDestroyOnLoad(gameObject);


                PlayMusic(standartMusic);
        }


        // Play a single clip through the sound effects source.
        public void Play(AudioClip clip)
        {
                EffectsSource.clip = clip;
                EffectsSource.Play();
        }
```

```
        // Play a single clip through the music source.
        public void PlayMusic(AudioClip clip)
        {
                MusicSource.clip = clip;
                MusicSource.loop = true; //зациклить музыку
                MusicSource.Play();
        }

        // Play a random clip from an array, and randomize the pitch slightly.
        public void RandomSoundEffect(params AudioClip[] clips)
        {
                int randomIndex = Random.Range(0, clips.Length);
                float randomPitch = Random.Range(LowPitchRange, HighPitchRange);

                EffectsSource.pitch = randomPitch;
                EffectsSource.clip = clips[randomIndex];
                EffectsSource.Play();
        }
}
```

ChangeMusic.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ChangeMusic : MonoBehaviour
{
    public AudioClip battleMusic;

    // Активация триггера при попадании в него объекта
```

```
    void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "MainCharacter")
        {
            MusicController.Instance.PlayMusic(battleMusic);
            Destroy(gameObject);
        }
    }
}
```

SceneTransition.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class SceneTransition : MonoBehaviour
{
    void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "MainCharacter")
            //SceneManager.LoadScene(1);
            SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
    }
}
```

Enemy.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

```
public class Enemy : MonoBehaviour
{
    public AudioClip soundHurt;


    public void EnemyHurt()
    {
        MusicController.Instance.Play(soundHurt);
        Destroy(this.gameObject);
    }
}
```