МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  
им. Н.Э. Баумана

Факультет “Информатика и системы управления”  
Кафедра “Системы обработки информации и управления”

A logo with a gold star and blue ribbon

Description automatically generated

Дисциплина “Парадигмы и конструкции языков программирования”

Отчет по домашнему заданию  
“Создание игры на юнити”

**Выполнил:**Студент группы ИУ5-34Б  
Изотов Е.А.  
**Преподаватель:**Нардид А.Н.

Москва 2025

Задание

1. Выберите язык программирования (который Вы ранее не изучали) и (1) напишите по нему реферат с примерами кода или (2) реализуйте на нем небольшой проект (с детальным текстовым описанием).

2. Реферат (проект) может быть посвящен отдельному аспекту (аспектам) языка или содержать решение какой-либо задачи на этом языке.

3. Необходимо установить на свой компьютер компилятор (интерпретатор, транспилятор) этого языка и произвольную среду разработки.

4. В случае написания реферата необходимо разработать и откомпилировать примеры кода (или модифицировать стандартные примеры).

5. В случае создания проекта необходимо детально комментировать код.

6. При написании реферата (создании проекта) необходимо изучить и корректно использовать особенности парадигмы языка и основных конструкций данного языка.

7. Приветствуется написание черновика статьи по результатам выполнения ДЗ. Черновик статьи может быть подготовлен группой студентов, которые исследовали один и тот же аспект в нескольких языках или решили одинаковую задачу на нескольких языках.

Код скрипта PlayerController

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerController : MonoBehaviour

{

    public float speed = 5f;

    public float jumpForce = 7f;

    private float moveInput;

    private bool facingRight = true;

    private Rigidbody2D rb;

    private bool isGrounded;

    public Transform feetPos;

    public float checkRadius = 0.2f;

    public LayerMask whatIsGround;

    private Animator anim;

    private void Start()

    {

        anim = GetComponent<Animator>();

        rb = GetComponent<Rigidbody2D>();

    }

    private void FixedUpdate()

    {

        moveInput = Input.GetAxis("Horizontal");   // A/D или стрелки

        rb.linearVelocity = new Vector2(moveInput \* speed, rb.linearVelocity.y);

        if (!facingRight && moveInput > 0)

        {

            Flip();

        }

        else if (facingRight && moveInput < 0)

        {

            Flip();

        }

        if(moveInput ==0)

        {

            anim.SetBool("isRunning", false);

        }

        else

        {

            anim.SetBool("isRunning", true);

        }

    }

    private void Update()

{

    isGrounded = Physics2D.OverlapCircle(feetPos.position, checkRadius, whatIsGround);

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

    {

        rb.linearVelocity = Vector2.up \* jumpForce;

        anim.SetTrigger("takeOF");

    }

    if(isGrounded == true)

    {

        anim.SetBool("isJumping", false);

    }

    else

    {

        anim.SetBool("isJumping", true);

    }

}

    void Flip()

    {

        facingRight = !facingRight;

        Vector3 scaler = transform.localScale;

        scaler.x \*= -1;

        transform.localScale = scaler;

    }

}

Код скрипта ParallaxLayer

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[ExecuteInEditMode]

public class ParallaxLayer : MonoBehaviour

{

    public float parallaxFactor;

    public void Move(float delta)

    {

        Vector3 newPos = transform.localPosition;

        newPos.x -= delta \* parallaxFactor;

        transform.localPosition = newPos;

    }

}

Код скрипта ParallaxCamera

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[ExecuteInEditMode]

public class ParallaxCamera : MonoBehaviour

{

    public delegate void ParallaxCameraDelegate(float deltaMovement);

    public ParallaxCameraDelegate onCameraTranslate;

    private float oldPosition;

    void Start()

    {

        oldPosition = transform.position.x;

    }

    void Update()

    {

        if (transform.position.x != oldPosition)

        {

            if (onCameraTranslate != null)

            {

                float delta = oldPosition - transform.position.x;

                onCameraTranslate(delta);

            }

            oldPosition = transform.position.x;

        }

    }

}

Код скрипта ParallaxCamera

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[ExecuteInEditMode]

public class ParallaxBackground : MonoBehaviour

{

    public ParallaxCamera parallaxCamera;

    List<ParallaxLayer> parallaxLayers = new List<ParallaxLayer>();

    void Start()

    {

        if (parallaxCamera == null && Camera.main != null)

            parallaxCamera = Camera.main.GetComponent<ParallaxCamera>();

        if (parallaxCamera != null)

            parallaxCamera.onCameraTranslate += Move;

        SetLayers();

    }

    void SetLayers()

    {

        parallaxLayers.Clear();

        for (int i = 0; i < transform.childCount; i++)

        {

            ParallaxLayer layer = transform.GetChild(i).GetComponent<ParallaxLayer>();

            if (layer != null)

            {

                layer.name = "Layer-" + i;

                parallaxLayers.Add(layer);

            }

        }

    }

    void Move(float delta)

    {

        foreach (ParallaxLayer layer in parallaxLayers)

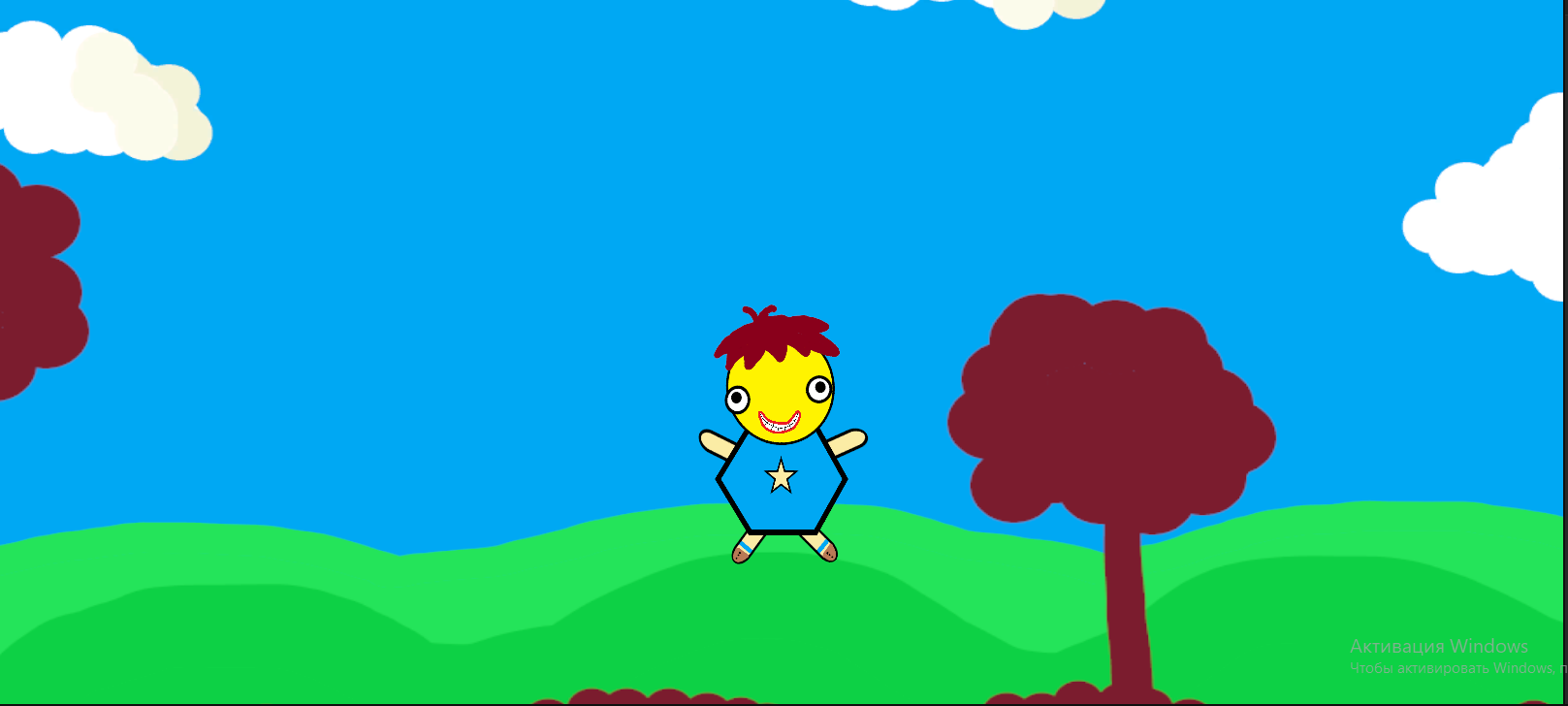
        {

            layer.Move(delta);

        }

    }

}



Работа приложения

