ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

Харківський національний університет
радіоелектроніки
Кафедра ЕОМ

Кваліфікаційна робота
Перший (бакалаврський) рівень

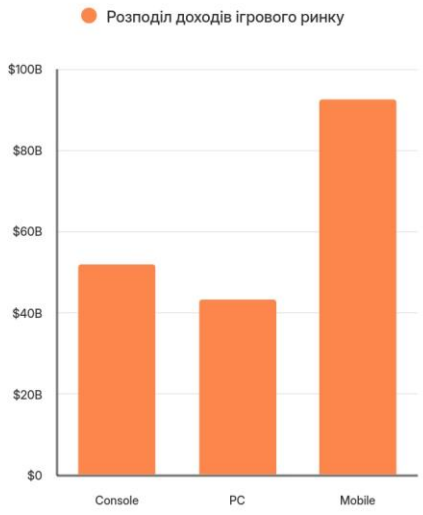
Мобільний застосунок 3D-гри на основі рушія Unity

Автор:
Закіпний Костянтин
студ. гр. КІУКІ-21-6

Керівник:
Андрусенко Юлія
ас. кафедри ЕОМ

1

Актуальність теми



2

Постановка задачі

Розробити мобільну 3D-гру у жанрі головоломки

Забезпечити технічну оптимізацію ігрового процесу

Забезпечити можливість масштабування проєкту в майбутньому

Постановка задачі є важливим етапом, який допомагає зосередитися на ключових цілях проєкту.

3

Інструменти розробки

Unity
Основний ігровий рушій

Blender
Створення 3D-моделей

Marmoset Toolbag
Попередній перегляд моделей

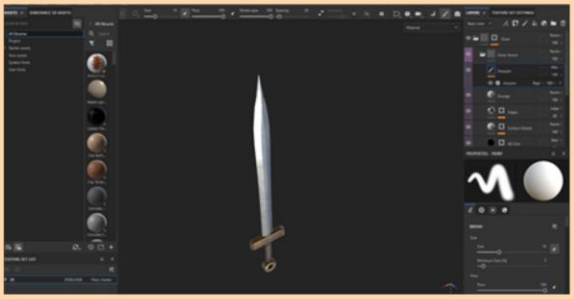
Substance Painter
Створення та нанесення PBR-текстур на моделі

Photoshop
Фінальна обробка текстур

Visual Studio Code
Написання C# скриптів

4

Використані технології



Графіка та моделі

Unity 3D

Ігровий рушій та програмування

5

Цілі ігрового дизайну

Плавна крива складності

Логічна послідовність дій

Зручний інтерфейс

Занурення в ігровий світ

Ігровий дизайн формує загальне сприйняття гри гравцем і визначають характер його взаємодії з ігровим середовищем.

6

Ігрова концепція

Ігрова концепція — це загальна ідея побудови гри: як гравець взаємодіє з середовищем, які правила, механіки та цілі визначають геймплей.

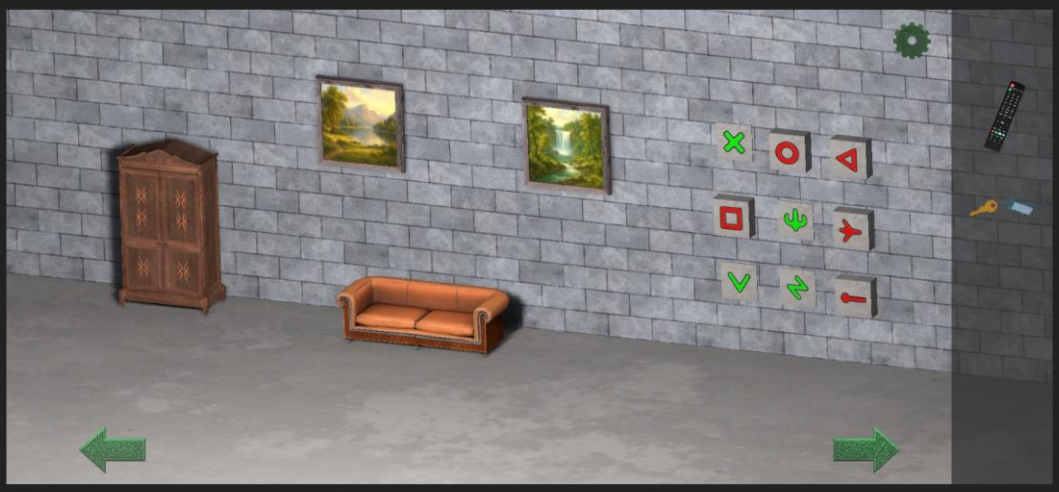
7

Технології оптимізації

Технолія	Опис
Baked Lighting	Заздалегідь обчислена інформація про освітлення зберігається у світлових картах, щоб скоротити розрахунки освітлення в реальному часі.
Texture Optimization	Зменшено роздільну здатність текстур і розміри файлів для мінімізації використання пам'яті та часу завантаження.
Polygon Reduction	Спрощення 3D-моделей за рахунок зменшення кількості полігонів без втрати візуальної точності.
Shader Simplification	Використовував базові, ефективні шейдери з меншою кількістю функцій для покращення продуктивності рендерингу.

8

Демонстрація роботи застосунку



9

Висновки

1. Було реалізовано повноцінну мобільну 3D-гру у жанрі головоломки, яка містить в собі 10 унікальних рівнів.
2. На тему кваліфікаційної роботи було написано та опубліковано тези доповіді для XV МНТК “Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління”.
3. Гра має масштабовану структуру, що дозволяє легко додавати нові рівні, головоломки та функціональні елементи.
4. Гра була успішно випущена в Google Play під назвою **Mystical Puzzleum** та на даний момент перебуває на етапі модерації перед публічним розміщенням на платформі.

10

ДОДАТОК Б

Код застосунку

Б.1 Скрипт ButtonController.cs

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

public class ButtonController : MonoBehaviour
{
    public bool isPressed = false;
    private bool isOpen = false;
    private Animation anim;
    private Renderer childRenderer;
    private List<string> animations = new List<string>();

    private void Start()
    {
        Transform secondChild = transform.GetChild(1);
        childRenderer = secondChild.GetComponent<Renderer>();

        UpdateColor();

        anim = GetComponent<Animation>();
        if (anim != null)
        {
            foreach (AnimationState animState in anim)
            {
                animations.Add(animState.name);
            }
        }
    }

    void Update()
    {
        if (RaycastController.Instance.isPressed &&
RaycastController.Instance.hit.collider.gameObject ==
gameObject)
        {
            if (!anim.isPlaying && animations.Count >= 2)
            {
                anim.Play(isOpen ? animations[1] :
animations[0]);
                isOpen = !isOpen;

                if (isOpen)
                    OnOpen();
```

```
                else
                    OnClose();
            }
        }
    }

    public void OnOpen()
    {
        isPressed = true;
        UpdateColor();
        ButtonPuzzleController.Instance.Check();
    }

    public void OnClose()
    {
        isPressed = false;
        UpdateColor();
        ButtonPuzzleController.Instance.Check();
    }

    private void UpdateColor()
    {
        if (childRenderer != null)
        {
            Color color = isPressed ? Color.green : Color.red;
            childRenderer.material.color = color;
        }
    }
}
```

Б.2 Скрипт ButtonPuzzleController.cs

```
using UnityEngine;


public class ButtonPuzzleController : MonoBehaviour,
IItemHasClue
{
    public ButtonController[] buttonControllers;
    public GameObject exitKey;
    public bool IsDone { get; set; } = false;
    public static ButtonPuzzleController Instance;

    void Awake()
    {
        Instance = this;
    }
    public void Check()
    {
        bool[] targetState = new bool[9] { true, false, false,
false, true, false, true, true, false };
```

```
        for (int i = 0; i < buttonControllers.Length; i++)
        {
            if (buttonControllers[i].isPressed !=
targetState[i])
                return;
        }

        exitKey.GetComponent<Rigidbody>().isKinematic = false;
        IsDone = true;
        for (int i = 0; i < 9; i++)
        {
            Destroy(buttonControllers[i]);
        }
    }
}
```

Б.3 Скрипт DoorController.cs

```
using UnityEngine;

public class DoorController : MonoBehaviour, IItemUsable
{
    public AnimationController animationController;
    public GameObject finishLevel;
    public float animationDuration = 1f;

    public void OnItemUsed()
    {
        animationController.PlayAnimation(0);
        StartCoroutine(ShowPanelAfterAnimation());
    }

    private System.Collections.IEnumerator
ShowPanelAfterAnimation()
    {
        yield return new WaitForSeconds(animationDuration);
        finishLevel.SetActive(true);
    }
}
```

Б.4 Скрипт LampEmissionController.cs

```
using UnityEngine;

public class LampEmissionController : MonoBehaviour
{
    public int id;
```

```
    public WardrobeController wardrobeController;

    private Renderer rend;
    private Material mat;

    void Start()
    {
        rend = GetComponent<Renderer>();
        mat = rend.material;
        DisableEmission();
    }

    void Update()
    {
        if
(RaycastController.Instance.inputContinueHoldingFirstObj &&
RaycastController.Instance.hit.collider.gameObject ==
gameObject)
            EnableEmission();
        else
            DisableEmission();

        if ((RaycastController.Instance.isReleased &&
RaycastController.Instance.hit.collider.gameObject == gameObject
&& RaycastController.Instance.firstAfterSwitchHolding)
        || (RaycastController.Instance.isEndHoldingFirstObj &&
RaycastController.Instance.lastHoldingObject == gameObject))
        {
            wardrobeController.CheckPassword(id);
            AudioManager.Instance.Play("LampaSFX");
        }


        if (wardrobeController.isPasswordCorrect)
        {
            DisableEmission();
            Destroy(this);
        }


    }


    void EnableEmission()
    {
        mat.EnableKeyword("_EMISSION");
    }

    void DisableEmission()
    {
        mat.DisableKeyword("_EMISSION");
    }
}
```

Б.5 Скрипт TorchColorChanger.cs

```
using UnityEngine;

public class TorchColorChanger : MonoBehaviour
{
    private ParticleSystem particleSys;

    [SerializeField] public int currentColorIndex = 0;
    [SerializeField]
    private Color[] colorSequence;

    private void Start()
    {
        particleSys = GetComponent<ParticleSystem>();
        ApplyColor(colorSequence[currentColorIndex], 6f);
    }



    private void Update()
    {
        if (RaycastController.Instance.isPressed &&
RaycastController.Instance.hit.collider != null &&
RaycastController.Instance.hit.collider.gameObject ==
gameObject)
        {
            currentColorIndex = (currentColorIndex + 1) %
colorSequence.Length;
            ApplyColor(colorSequence[currentColorIndex], 8f);
            TorchesController.Instance?.CheckPassword();
        }
    }



    void ApplyColor(Color baseColor, float intensity)
    {
        var psRenderer =
particleSys.GetComponent<ParticleSystemRenderer>();

        MaterialPropertyBlock block = new
MaterialPropertyBlock();
        psRenderer.GetPropertyBlock(block);

        block.SetColor("_EmissionColor", baseColor * intensity);

        psRenderer.SetPropertyBlock(block);
    }
}
```

Б.7 Скрипт TorchesController.cs

```
using UnityEngine;

public class TorchesController : MonoBehaviour, IItemHasClue
{
    public static TorchesController Instance { get; private set;
}
    public GameObject passworQuad;
    public bool IsDone { get; set; } = false;

    public TorchColorChanger[] torchColorChangers = new
TorchColorChanger[5];
    private readonly int[] correctPassword = new int[5] { 1, 0,
0, 1, 2 };


    private void Awake()
    {
        if (Instance != null && Instance != this)
        {
            Destroy(this);
            return;
        }
        Instance = this;
    }




    public void CheckPassword()
    {
        for (int i = 0; i < torchColorChangers.Length; i++)
        {
            if (torchColorChangers[i].currentColorIndex !=
correctPassword[i])
            {
                return;
            }
        }
        IsDone = true;


passworQuad.GetComponent<AnimationController>().PlayAnimation(0)
;
        for (int i = 0; i < torchColorChangers.Length; i++)
        {
            Destroy(torchColorChangers[i]);
        }
    }
}
```

Б.8 Скрипт TVcontroller.cs

```
using UnityEngine;

public class TVcontroller : MonoBehaviour, IItemUsable
{

    public VideoPlayerController videoPlayerController;
    public GameObject TVvideo;

    public void OnItemUsed()
    {
        videoPlayerController.PlayNextVideo();
    }
}
```

Б.9 Скрипт WardrobeController.cs

```
using UnityEngine;

public class WardrobeController : MonoBehaviour, IItemHasClue
{
    public AnimationController animationControllerSingle;
    public int[] correctPassword = { 0, 1, 1, 0 };
    public int[] userPassword = { 2, 2, 2, 2, };
    public bool isPasswordCorrect = false;
    public GameObject password;
    public bool IsDone { get; set; } = false;
    private bool hasStartedAnimation = false;

    void Start()
    {
        animationControllerSingle =
GetComponent<AnimationController>();
    }

    public void CheckPassword(int newNumber)
    {
        for (int i = 0; i < userPassword.Length - 1; i++)
        {
            userPassword[i] = userPassword[i + 1];
        }

        userPassword[userPassword.Length - 1] = newNumber;

        for (int i = 0; i < correctPassword.Length; i++)
        {
            if (userPassword[i] != correctPassword[i])
                return;
```

```
        }
        isPasswordCorrect = true;
    }

    void Update()
    {
        if (isPasswordCorrect &&
!animationControllerSingle.isPlaying)
        {
            if (!hasStartedAnimation)
            {
                password.SetActive(true);

animationControllerSingle.animat.Play(animationControllerSingle.
animations[1]);
                hasStartedAnimation = true;
                return;
            }

            if (!animationControllerSingle.isPlaying)
            {
                Destroy(animationControllerSingle);
                IsDone = true;
            }
        }
    }
}
```