

## Создание компьютерной игры «Математический кроссворд» на Unity

*Голубева Евгения Павловна*

*Приамурский государственный университет имени Шолом-Алейхема*

*Студент*

### Аннотация

Цель статьи - разработка компьютерной игры «Математический кроссворд» на базе платформы Unity. В процессе создания использовалась кроссплатформенная среда Unity, предоставляющая широкие возможности для разработки интерактивных 2D- и 3D-приложений, включая мощный физический движок и набор визуальных инструментов, что позволяет эффективно реализовывать двухмерные проекты. Итогом работы стала готовая компьютерная игра «Математический кроссворд».

**Ключевые слова:** Unity, игра, кроссворд, математика.

## Creating a Mathematical Crossword Puzzle Game in Unity

*Golubeva Evgeniya Pavlovna*

*Sholom-Aleichem Priamursky State University*

*Student*

### Abstract

The purpose of the article is to develop a computer game "Mathematical Crossword" based on the Unity platform. During the creation process, the cross-platform environment Unity was used, which provides extensive opportunities for developing interactive 2D and 3D applications, including a powerful physics engine and a set of visual tools, allowing for the effective implementation of two-dimensional projects. The result of the work was the completed computer game "Mathematical Crossword".

**Keywords:** Unity, game, crossword, mathematics.

## 1 Введение

### 1.1 Актуальность

Современный этап развития образования характеризуется глубокой цифровой трансформацией, которая охватывает все уровни обучения - от школы до высшего образования. В контексте преподавания математики цифровизация открывает практически неограниченные возможности для визуализации абстрактных понятий и делает образовательный процесс более интересным и доступным. Однако, несмотря на наличие широкого спектра цифровых инструментов - электронных курсов, игровых платформ, образовательных квестов - сохраняется необходимость в разработке новых средств обучения, которые одновременно способствовали бы развитию

математических навыков учащихся и соответствовали запросам цифрового поколения к форме подачи информации.

Одной из ключевых проблем современного математического образования является снижение познавательной мотивации учащихся. Традиционные формы обучения зачастую не способны удержать внимание школьников и студентов, что приводит к формальному усвоению материала и отсутствию глубокого понимания математических концепций. В этой связи особую актуальность приобретает поиск новых инструментов обучения, которые помогут сделать процесс овладения знаниями более продуктивным и значительно повысят мотивацию.

Игровые технологии и геймификация образования рассматриваются сегодня как один из наиболее активно развивающихся подходов в мировой педагогической практике. Многочисленные исследования подтверждают положительное влияние игровых приемов на учебную мотивацию и вовлеченность учащихся в процесс обучения. Цифровые дидактические игры обладают значительным потенциалом в формировании у школьников математических и логических умений, а также познавательной мотивации. Игровой мотив выступает действенным подкреплением познавательного мотива, способствует активности мыслительной деятельности, повышает концентрацию внимания и работоспособность.

В качестве инструмента реализации такой игры обоснованно выбрать игровой движок Unity. Unity является одним из наиболее популярных и функциональных игровых движков, позволяющих создавать образовательные игры для различных платформ. Его преимущества включают кроссплатформенность, поддержку как 2D-, так и 3D-графики, широкий набор инструментов для разработки интерактивного контента и обширное сообщество разработчиков. Современные исследования демонстрируют успешное применение Unity для создания образовательных игр по математике, подтверждая их валидность, практичность и эффективность в повышении учебной мотивации.

Таким образом, актуальность настоящего исследования обусловлена цифровой трансформацией образования, требующей разработки новых, соответствующих запросам цифрового поколения средств обучения математике, необходимостью повышения познавательной мотивации учащихся при изучении математики, что может быть достигнуто за счёт внедрения игровых технологий в образовательный процесс, недостаточной разработанностью цифровых интерактивных аналогов математических кроссвордов как эффективного дидактического средства, а также потенциалом игрового движка Unity как универсальной платформы для создания качественных образовательных игр, сочетающих педагогическую ценность с высоким уровнем вовлечённости; разработка компьютерной игры «Математический кроссворд» на Unity направлена на решение обозначенных проблем и представляет собой своевременный вклад в развитие теории и практики игровых технологий в математическом образовании.

## 1.2 Обзор исследований

А. Е. Унарова представила разработку мобильной игры «Кустук» на Unity для дошкольников и младших школьников на якутском языке [1]. Посвятил статью обзору ключевых этапов, необходимых для разработки компьютерных игр с двухмерной графикой на платформе Unity в жанре приключения с элементами головоломок Я. А. Фаст [2]. Н.В. Максимов рассматривает метод разработки мобильной 2D игры «Bootur Jump» на платформе Unity [3]. Показал процесс разработки двухмерной платформенной игры на игровом движке Unity с использованием языка программирования C# А.А. Любич [4]. Д.В. Морозова в рамках работы разработала развивающую 2D игра-платформер для детей, написанная на Unity [5].

## 1.3 Цель исследования

Цель исследования – создание компьютерной игры «Математический кроссворд» при помощи кроссплатформенной среды для создания интерактивных 2D и 3D приложений Unity.

## 2 Материалы и методы

В качестве базового инструмента для разработки игры «Математический кросс» применяется платформа Unity. Выбор обоснован простотой освоения данной среды, эргономичностью интерфейса и встроенной поддержкой языка C#. Непосредственное программирование и отладка кода выполняются в среде Visual Studio Community.

## 3 Результаты

Первым шагом в разработке является создание проекта в Unity. В Unity Hub необходимо выбрать шаблон «Universal 2D» предусматривающий универсальный конвейер рендеринга, после чего нажать кнопку «Создать проект» (см. рис. 1).

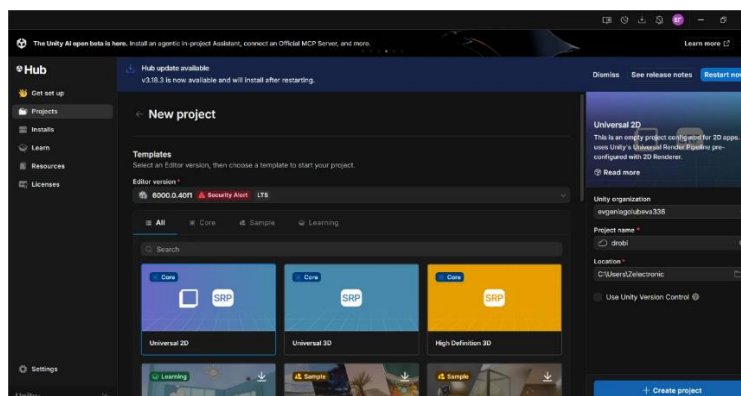


Рисунок 1 – Создание проекта

После создания проекта на базе шаблона Universal 2D необходимо выполнить его начальную настройку для целевой ПК-платформы. Первым

делом следует открыть меню File -> Build Profiles и установить в качестве платформы Windows (см. рис. 2).

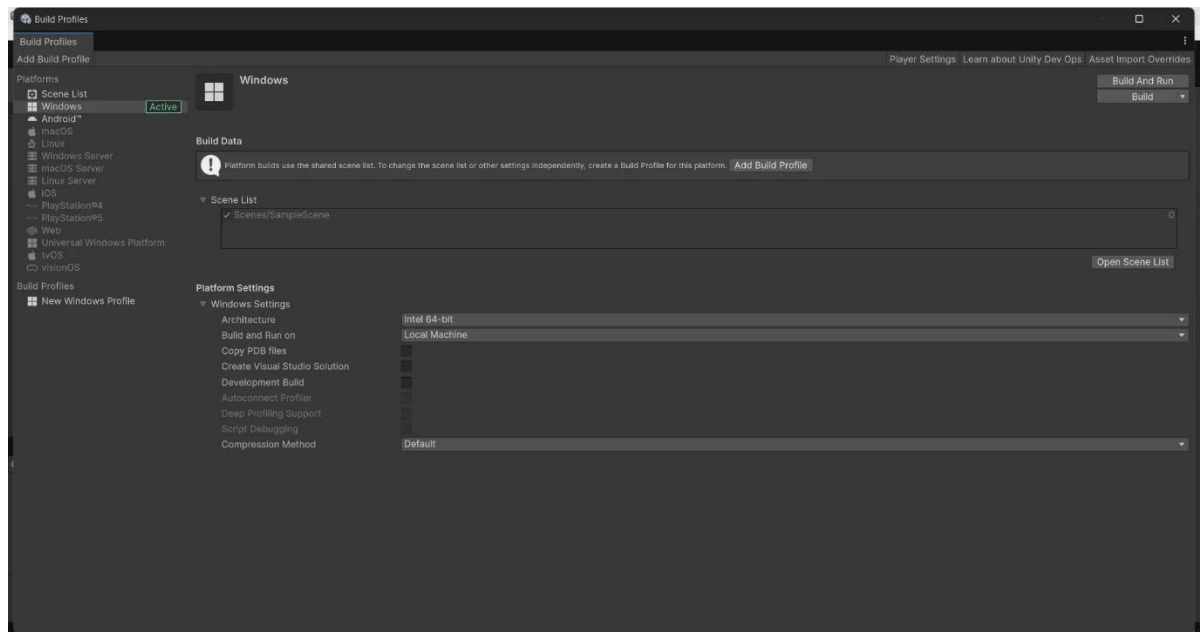


Рисунок 2 – Настройка проекта

Все заранее подготовленные графические материалы, задействованные в игре, необходимо скопировать в папку «image» (см. рис. 3).

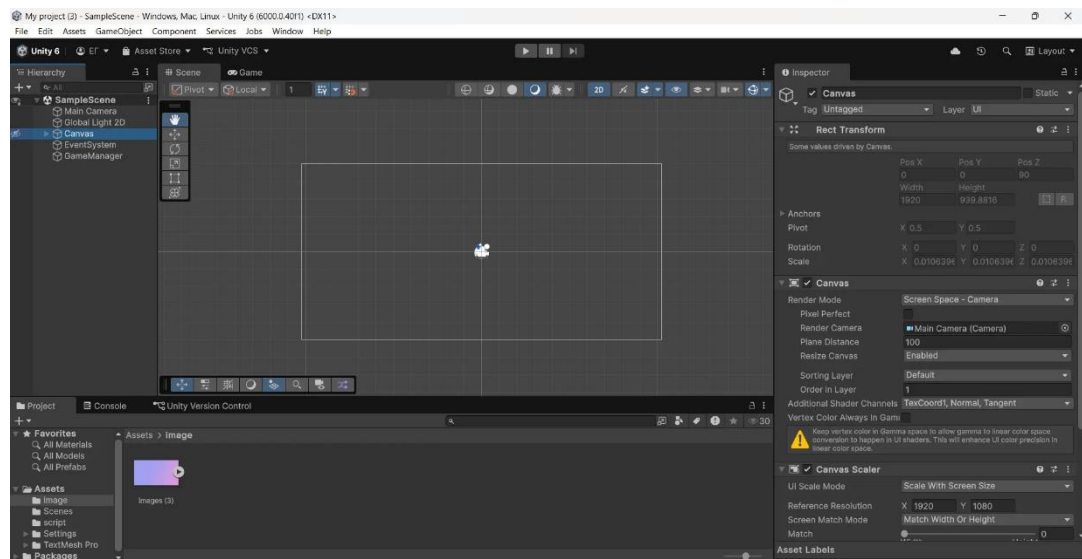


Рисунок 3 – Добавление изображений

На данном этапе в проекте доступна одна сцена. Первым шагом становится добавление объекта Canvas - базовой площадки для всей пользовательской интерфейсной графики. Создать его можно через контекстное меню в Hierarchy, выбрав пункт UI -> Canvas. Чтобы интерфейс отображался правильно, свеже созданный Canvas нужно донастроить. В окне Inspector указывается режим Screen Space - Camera, в соответствующее поле

подтягивается основная камера сцены, параметр порядка слоя устанавливается в 1, а эталонное разрешение фиксируется на значении 1920×1080 (см. рис. 4).

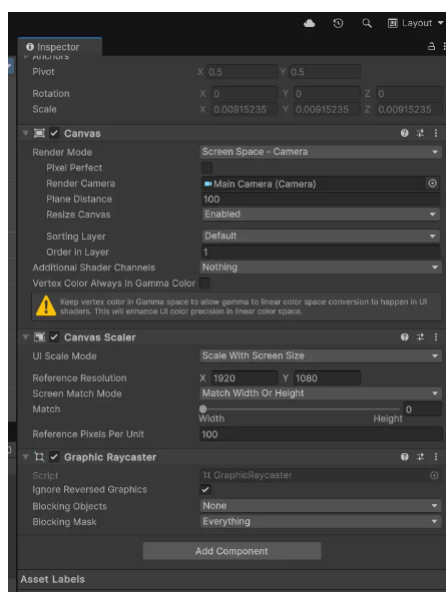


Рисунок 4 – Настройка Canvas

Следующим шагом на созданном Canvas появляется объект Image, который берёт на себя роль фоновой заливки игрового пространства. Его необходимо правильно расположить: через инспектор корректируются координаты, размеры и якорные привязки так, чтобы фон заполнял всю видимую область камеры без пробелов. Завершается этот этап назначением в соответствующем разделе инспектора заранее заготовленного файла с фоновой графикой (см. рис. 5).

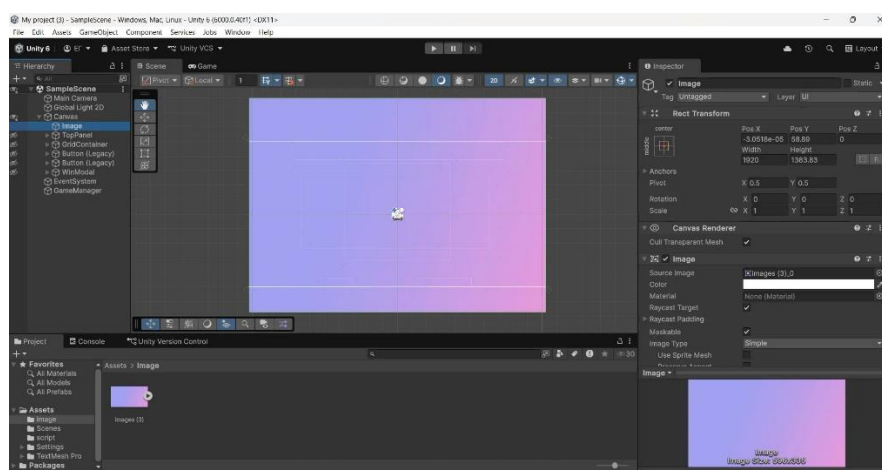


Рисунок 5 – Добавление фона

Далее необходимо создать панель для демонстрации сколько решено кроссвордов, время и среднее количество времени на прохождения кроссворда. Для этого создается объект Panel в объекте Canvas, и переименовывается в TopPanel. В панели создается шесть объектов Text. Три объекта необходимы для текстов: решено, время и среднее, остальные три

объекта Text будут показывать результаты решено, время и среднее (см. рис. 6).

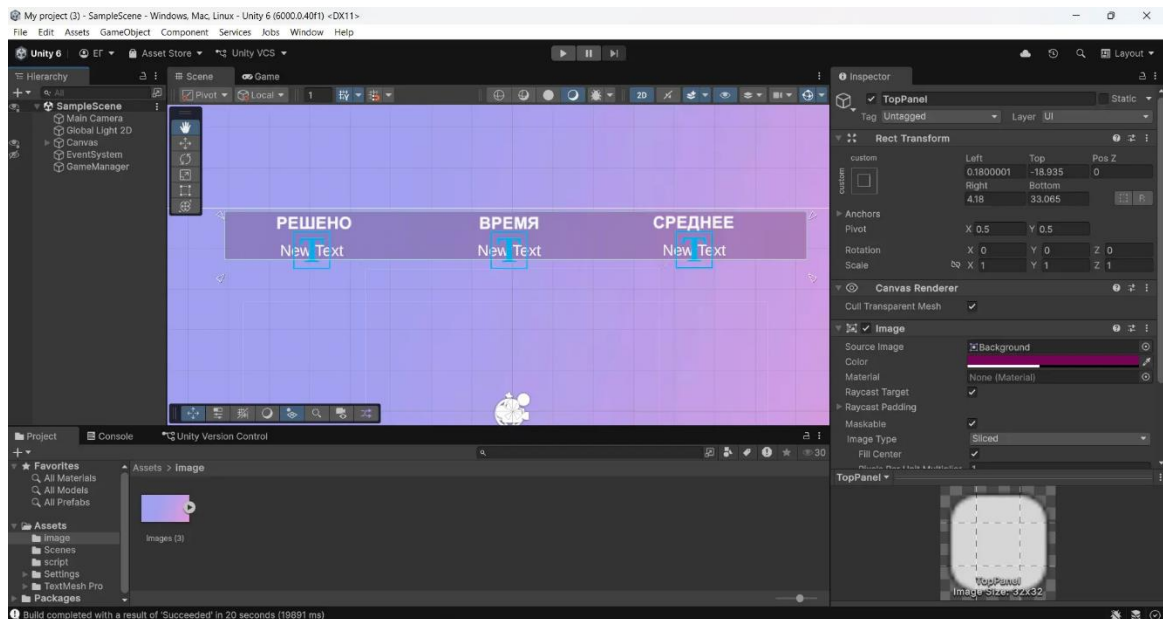


Рисунок 6 – Настройка панели

Следующим шагом создается панель для кроссворда. Необходимо создать объект Panel в объекте Canvas, и переименовать в GridContainer. К панели необходимо добавить компонент Grid Layout Group, а также надо настроить Cell Size = 120, 120, Spacing = 5, 5, Child Alignment = Middle Center, Constraint Count = 5. В панели GridContainer надо создать объект ячейки, для этого объект необходимо добавить объект image и переименовать в Cell\_0\_0, и в объекте image добавить объект Text – TextMeshPro, и создать еще 24 ячейки (см. рис. 7).

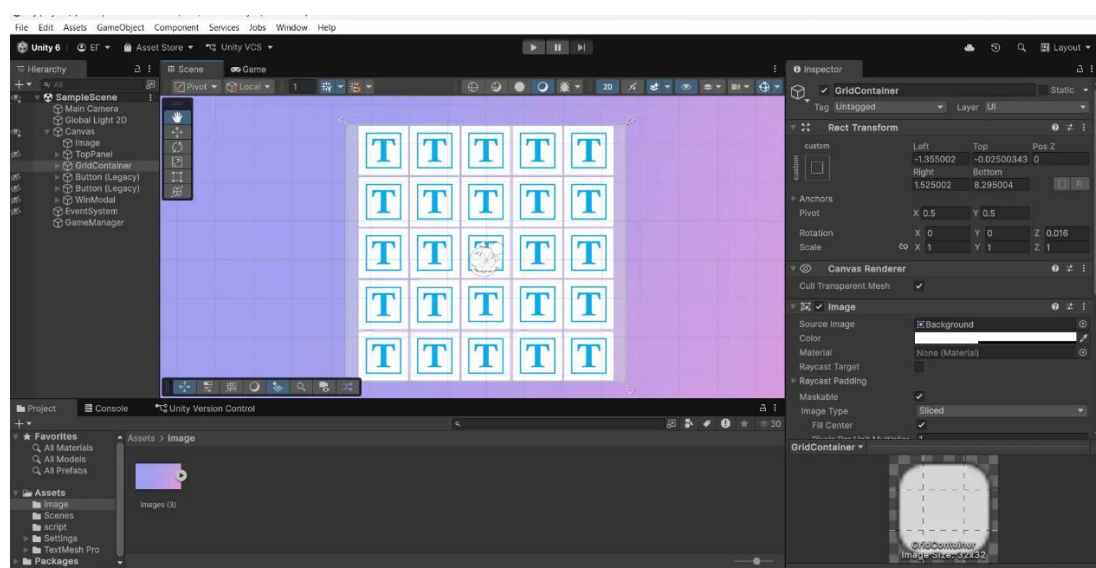


Рисунок 7 – Настройка панели кроссворда

Для механики ячеек необходимо создать скрипт GridCell. Данный скрипт отвечает за конкретную ячейку игрового поля. Хранит ссылки на её графические элементы текст и фон, управляет состояниями скрыта, число, оператор, заблокирована, решена, ошибка и обрабатывает клик мышью.

### Скрипт GridCell:

```
using TMPro;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;
public class GridCell : MonoBehaviour, IPointerClickHandler
{
    [SerializeField] private TextMeshProUGUI displayText;
    [SerializeField] private Image backgroundImage;
    private string cellId = "";
    private string inputValue = "";
    private bool isSolved = false;
    private bool isInteractable = false;
    private bool isError = false;
    public event System.Action<GridCell> OnSelected;
    public void SetNumber(int value)
    {
        displayText.text = value.ToString();
        displayText.color = Color.black
;
        backgroundImage.color = new Color(0.9f, 0.9f, 0.95f);
        isInteractable = false;
        isSolved = false;
        isError = false;
        inputValue = "";
    }
    public void SetOperator(char op)
    {
        displayText.text = op.ToString();
        if (op == '+') displayText.color = new Color(0.2f, 0.9f, 0.4f);
        else if (op == '-') displayText.color = new Color(0.95f, 0.3f, 0.2f);
        else if (op == '=') displayText.color = new Color(0.3f, 0.6f, 1f);
        backgroundImage.color = new Color(0.8f, 0.8f, 0.9f);
        isInteractable = false;
        isSolved = false;
        isError = false;
        inputValue = "";
    }
    public void SetBlocked()
    {
        displayText.text = "";
        backgroundImage.color = new Color(0.5f, 0.5f, 0.55f, 0.3f);
        isInteractable = false;
        isSolved = false;
        isError = false;
        inputValue = "";
    }
    public void SetHidden(string id)
    {
        cellId = id;
        displayText.text = "";
        displayText.color = Color.black
;
        backgroundImage.color = new Color(0.9f, 0.9f, 0.95f);
        isInteractable = true;
        isSolved = false;
        isError = false;
        inputValue = "";
    }
    public void SetInteractable(bool value) => isInteractable = value;
    public void SetSelected(bool selected)
    {

```

```

        if (selected && isInteractable && !isSolved)
            backgroundImage.color = new Color(0.7f, 0.85f, 1f);
        else if (!isSolved && !isError)
            backgroundImage.color = new Color(0.9f, 0.9f, 0.95f);
    }
    public void SetSolved(bool solved)
    {
        isSolved = solved;
        if (solved)
        {
            backgroundImage.color = new Color(0.7f, 0.95f, 0.75f);
            isInteractable = false;
        }
    }
    public void SetError(bool error = true)
    {
        isError = error;
        backgroundImage.color = error ? new Color(1f, 0.7f, 0.7f) : new Color(0.9f,
0.9f, 0.95f);
    }
    public void SetInputValue(string value)
    {
        inputValue = value;
        displayText.text = value;
    }
    public string GetInputValue() => inputValue;
    public string GetId() => cellId;
    public bool IsSolved() => isSolved;
    public void PlayCelebration()
    {
        StartCoroutine(CelebrationRoutine());
    }
    private System.Collections.IEnumerator CelebrationRoutine()
    {
        float duration = 0.6f;
        float elapsed = 0f;
        Vector3 startScale = transform.localScale;
        while (elapsed < duration)
        {
            elapsed += Time.deltaTime;
            float t = elapsed / duration;
            float scale = 1f + 0.15f * Mathf.Sin(t * Mathf.PI * 4f) * (1f - t);
            transform.localScale = startScale * scale;
            yield return null;
        }
        transform.localScale = startScale;
    }
    public void OnPointerClick(PointerEventData eventData)
    {
        if (isInteractable && !isSolved)
        {
            OnSelected?.Invoke(this);
        }
    }
}

```

Скрипт GridCell необходимо прикрепить к каждой ячейки и в инспекторе каждой ячейки произвести настройку (см. рис. 8).

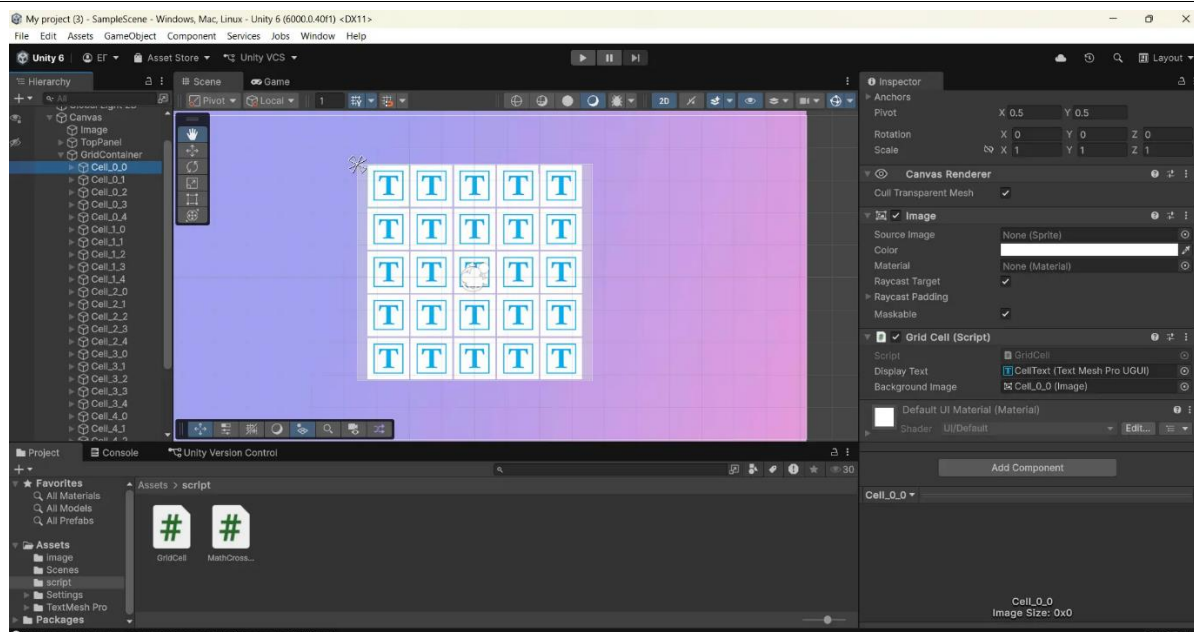


Рисунок 8 – Настройка механики кроссворда

После создания панели кроссворда, необходимо создать кнопки подсказка, и новый кроссворд. Для этого в Canvas необходимо создать два объекта Button и их настроить (см. рис. 9)

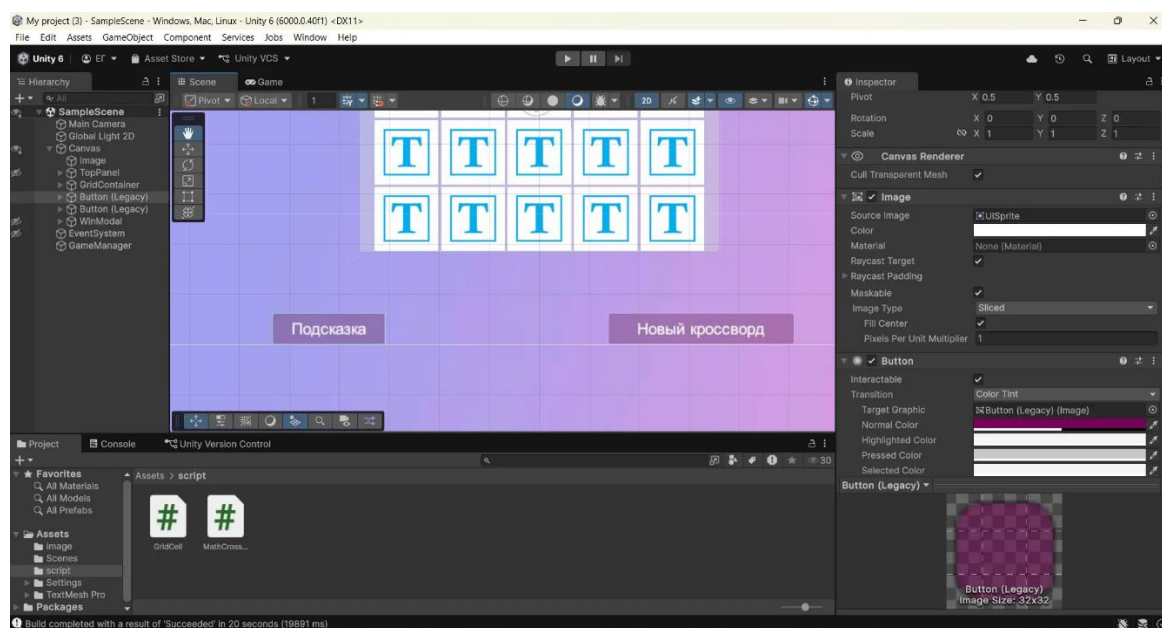


Рисунок 9 – Создание кнопок

Также необходимо создать финальную панель, которая должна появиться после прохождения кроссворда. Для этого надо создать объект Panel, и переименовать в WinModal. В панель необходимо добавить два объекта image, и к ним добавить объект компонент Text – TextMeshPro и их настроить в Font Style выбрать Normal, размер текста 80, цвет текста белый (см. рис. 10).

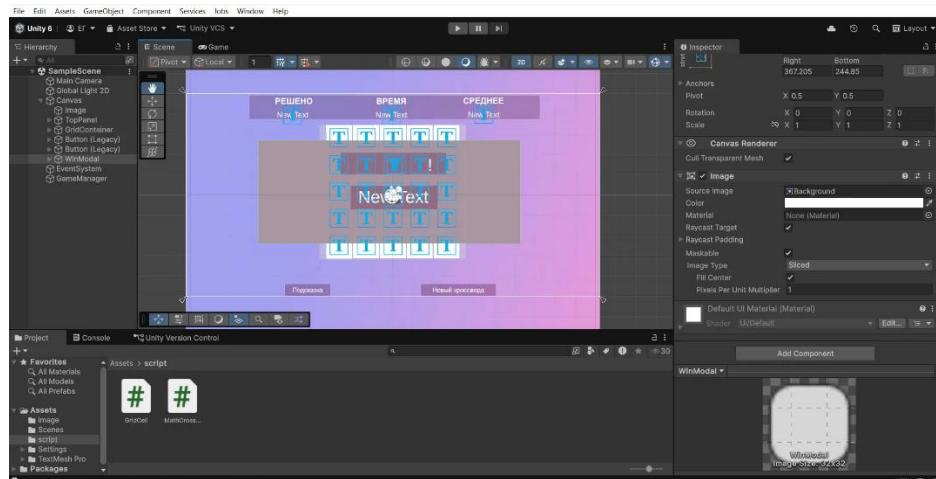


Рисунок 10 – Финальная панель

Далее необходимо создать главный управляющий скрипт MathCrosswordGame. Он генерирует корректные математические головоломки, строит сетку, обрабатывает ввод с клавиатуры, проверяет ответы, управляет таймером, сохраняет статистику, даёт подсказки и запускает новую игру.

#### Скрипт MathCrosswordGame:

```
using System;
using System.Collections.Generic;
using System.Linq;
using TMPro;
using UnityEngine;
using UnityEngine.UI;
using Random = UnityEngine.Random;

public class MathCrosswordGame : MonoBehaviour
{
    [Header("Статистика")]
    [SerializeField] private TextMeshProUGUI scoreText;
    [SerializeField] private TextMeshProUGUI timerText;
    [SerializeField] private TextMeshProUGUI avgText;

    [Header("Сетка")]
    [SerializeField] private GridLayoutGroup gridContainer;
    [SerializeField] private GridCell[] gridCells;

    [Header("Кнопки (опционально)")]
    [SerializeField] private Button backspaceButton;
    [SerializeField] private Button enterButton;

    [Header("Кнопки управления")]
    [SerializeField] private Button hintButton;
    [SerializeField] private Button newGameButton;

    [Header("Модальное окно победы")]
    [SerializeField] private GameObject winModal;
    [SerializeField] private TextMeshProUGUI modalTimeText;

    private Dictionary<string, int> solution = new Dictionary<string, int>();
    private List<GridCell> hiddenCells = new List<GridCell>();
    private GridCell selectedCell;

    private float timerSeconds = 0f;
    private bool timerRunning = false;
    private Coroutine timerCoroutine;

    private int score = 0;
    private int totalGames = 0;
```

```
private float totalTime = 0f;

private void Awake()
{
    if (backspaceButton != null) backspaceButton.onClick.AddListener(OnBackspace);
    if (enterButton != null) enterButton.onClick.AddListener(OnEnter);
    hintButton.onClick.AddListener(UseHint);
    newGameButton.onClick.AddListener(NewGame);

    foreach (var cell in gridCells)
    {
        cell.OnSelected += OnCellSelected;
    }

    LoadStats();
    UpdateUI();
    NewGame();
}

private void OnCellSelected(GridCell cell) => SelectCell(cell);

private void SelectCell(GridCell cell)
{
    if (selectedCell != null) selectedCell.SetSelected(false);
    selectedCell = cell;
    if (selectedCell != null && !selectedCell.IsSolved())
        selectedCell.SetSelected(true);
}

// ----- Генерация кроссворда -----
private class PuzzleData
{
    public int N00, N04, N22, N44;
    public int H02, H20, H24, H40, H42;
    public char op1, op2, op3, op4, op5, op6;
    public int GetValue(string id)
    {
        switch (id)
        {
            case "0-2": return H02;
            case "2-0": return H20;
            case "2-4": return H24;
            case "4-0": return H40;
            case "4-2": return H42;
            default: return 0;
        }
    }
}

private PuzzleData GeneratePuzzle()
{
    for (int attempt = 0; attempt < 5000; attempt++)
    {
        int N00 = Random.Range(1, 21);
        int N04 = Random.Range(1, 21);
        int N22 = Random.Range(1, 21);
        int N44 = Random.Range(1, 21);
        char[] ops = { '+', '-' };
        char op1 = ops[Random.Range(0, 2)];
        char op2 = ops[Random.Range(0, 2)];
        char op3 = ops[Random.Range(0, 2)];
        char op4 = ops[Random.Range(0, 2)];
        char op5 = ops[Random.Range(0, 2)];
        char op6 = ops[Random.Range(0, 2)];

        int H02 = GetMissing(N00, op1, N04);
        if (!IsValid(H02)) continue;
        int H42 = GetResult(H02, op4, N22);
        if (!IsValid(H42)) continue;
        int H24 = GetMissing(N04, op6, N44);
        if (!IsValid(H24)) continue;
```

```

        int H20 = GetFirst(op3, N22, H24);
        if (!IsValid(H20)) continue;
        int H40 = GetResult(N00, op2, H20);
        if (!IsValid(H40)) continue;
        if (GetResult(H40, op5, H42) == N44)
            return new PuzzleData
            {
                N00 = N00, N04 = N04, N22 = N22, N44 = N44,
                H02 = H02, H20 = H20, H24 = H24, H40 = H40, H42 = H42,
                op1 = op1, op2 = op2, op3 = op3, op4 = op4, op5 = op5, op6 = op6
            };
    }
    throw new Exception("Не удалось сгенерировать кроссворд");
}

private int GetMissing(int A, char op, int C)
{
    if (op == '+') return C - A;
    if (op == '-') return A - C;
    return 0;
}

private int GetResult(int A, char op, int B)
{
    if (op == '+') return A + B;
    if (op == '-') return A - B;
    return 0;
}

private int GetFirst(char op, int B, int C)
{
    if (op == '+') return C - B;
    if (op == '-') return C + B;
    return 0;
}

private bool IsValid(int val) => val >= 1 && val <= 20;

// ----- Построение сетки -----
private void BuildGrid(PuzzleData data)
{
    solution.Clear();
    hiddenCells.Clear();

    solution["0-2"] = data.H02;
    solution["2-0"] = data.H20;
    solution["2-4"] = data.H24;
    solution["4-0"] = data.H40;
    solution["4-2"] = data.H42;

    var layout = new (string type, object val)[5, 5]
    {
        { ("number", data.N00), ("op", data.op1), ("hidden", "0-2"), ("op", '='),
("number", data.N04) },
        { ("op", data.op2), ("blocked", null), ("op", data.op4), ("blocked", null),
("op", data.op6) },
        { ("hidden", "2-0"), ("op", data.op3), ("number", data.N22), ("op", '='),
("hidden", "2-4") },
        { ("op", '='), ("blocked", null), ("op", '='), ("blocked", null), ("op",
'=') },
        { ("hidden", "4-0"), ("op", data.op5), ("hidden", "4-2"), ("op", '='),
("number", data.N44) }
    };

    for (int r = 0; r < 5; r++)
    {
        for (int c = 0; c < 5; c++)
        {
            int index = r * 5 + c;
            GridCell cell = gridCells[index];
            var (type, val) = layout[r, c];

            switch (type)
            {

```

```

        case "number":
            cell.SetNumber((int)val);
            cell.SetInteractable(false);
            break;
        case "op":
            cell.SetOperator((char)val);
            cell.SetInteractable(false);
            break;
        case "blocked":
            cell.SetBlocked();
            cell.SetInteractable(false);
            break;
        case "hidden":
            string id = (string)val;
            cell.SetHidden(id);
            cell.SetInteractable(true);
            hiddenCells.Add(cell);
            break;
    }
}

if (selectedCell != null) selectedCell.SetSelected(false);
selectedCell = null;

if (hiddenCells.Count > 0) SelectCell(hiddenCells[0]);
StartTimer();
}

// ----- Ввод и проверка -----
private void OnNumberPadDigit(int digit)
{
    if (selectedCell == null || selectedCell.IsSolved()) return;

    string current = selectedCell.GetInputValue();
    if (current.Length >= 2) return;

    string newValue = current + digit;
    selectedCell.SetInputValue(newValue);

    string correctStr = solution[selectedCell.GetId()].ToString();

    if (newValue == correctStr)
    {
        CheckCell(selectedCell);
    }
    else if (newValue.Length == 2)
    {
        CheckCell(selectedCell);
    }
}

private void OnBackspace()
{
    if (selectedCell == null || selectedCell.IsSolved()) return;
    string current = selectedCell.GetInputValue();
    if (current.Length > 0)
        selectedCell.SetInputValue(current.Substring(0, current.Length - 1));
}

private void OnEnter()
{
    if (selectedCell != null && !selectedCell.IsSolved())
        CheckCell(selectedCell);
}

private void CheckCell(GridCell cell)
{
    string valStr = cell.GetInputValue();
    if (string.IsNullOrEmpty(valStr)) return;
    if (!int.TryParse(valStr, out int val)) return;

```

```

        int correct = solution[cell.GetId()];
        if (val == correct)
        {
            cell.SetSolved(true);
            cell.SetSelected(false);
            GridCell next = hiddenCells.FirstOrDefault(c => !c.IsSolved());
            if (next != null) SelectCell(next);
            else OnWin();
        }
        else
        {
            cell.SetError(true);
            cell.SetInteractable(false);
            StartCoroutine(ClearErrorDelayed(cell));
        }
    }

private System.Collections.IEnumerator ClearErrorDelayed(GridCell cell)
{
    yield return new WaitForSeconds(0.4f);
    if (cell != null && !cell.IsSolved())
    {
        cell.SetInputValue("");
        cell.SetError(false);
        cell.SetInteractable(true);
    }
}

// ----- Победа -----
private void OnWin()
{
    StopTimer();
    score++;
    totalGames++;
    totalTime += timerSeconds;
    SaveStats();
    UpdateUI();

    winModal.SetActive(true);
    modalTimeText.text = FormatTime(timerSeconds);
    foreach (var c in hiddenCells) c.PlayCelebration();

    StartCoroutine(HideWinModalDelayed(2.5f));
}

private System.Collections.IEnumerator HideWinModalDelayed(float delay)
{
    yield return new WaitForSeconds(delay);
    winModal.SetActive(false);
    NewGame();
}

// ----- Таймер -----
private void StartTimer()
{
    timerSeconds = 0f;
    timerRunning = true;
    if (timerCoroutine != null) StopCoroutine(timerCoroutine);
    timerCoroutine = StartCoroutine(TimerRoutine());
}

private void StopTimer()
{
    timerRunning = false;
    if (timerCoroutine != null)
    {
        StopCoroutine(timerCoroutine);
        timerCoroutine = null;
    }
}

```

```
private System.Collections.IEnumerator TimerRoutine()
{
    while (timerRunning)
    {
        timerSeconds += Time.deltaTime;
        timerText.text = FormatTime(timerSeconds);
        yield return null;
    }
}

private string FormatTime(float seconds)
{
    int mins = Mathf.FloorToInt(seconds / 60);
    int secs = Mathf.FloorToInt(seconds % 60);
    return $"{mins:00}:{secs:00}";
}

// ----- Статистика -----
private void LoadStats()
{
    score = PlayerPrefs.GetInt("MC_Score", 0);
    totalGames = PlayerPrefs.GetInt("MC_Total", 0);
    totalTime = PlayerPrefs.GetFloat("MC_Time", 0);
}

private void SaveStats()
{
    PlayerPrefs.SetInt("MC_Score", score);
    PlayerPrefs.SetInt("MC_Total", totalGames);
    PlayerPrefs.SetFloat("MC_Time", totalTime);
    PlayerPrefs.Save();
}

private void UpdateUI()
{
    scoreText.text = score.ToString();
    float avg = totalGames > 0 ? totalTime / totalGames : 0;
    avgText.text = FormatTime(avg);
}

// ----- Подсказка -----
private void UseHint()
{
    GridCell target = hiddenCells.FirstOrDefault(c => !c.IsSolved());
    if (target == null) return;

    string id = target.GetId();
    int correct = solution[id];
    target.SetInputValue(correct.ToString());
    target.SetSolved(true);
    target.SetSelected(false);

    score = Mathf.Max(0, score - 1);
    SaveStats();
    UpdateUI();

    GridCell next = hiddenCells.FirstOrDefault(c => !c.IsSolved());
    if (next != null) SelectCell(next);
    else OnWin();
}

// ----- Новая игра -----
public void NewGame()
{
    winModal.SetActive(false);
    try
    {
        var data = GeneratePuzzle();
        BuildGrid(data);
        UpdateUI();
    }
}
```

```

    }
    catch
    {
        NewGame ();
    }
}

// ----- Клавиатурный ввод -----
private void Update()
{
    if (Input.GetKeyDown(KeyCode.Backspace))
        OnBackspace ();
    else if (Input.GetKeyDown(KeyCode.Return) || Input.GetKeyDown(KeyCode.KeypadEnter))
        OnEnter ();
    else if (Input.inputString.Length > 0 && char.IsDigit(Input.inputString[0]))
    {
        int digit = Input.inputString[0] - '0';
        OnNumberPadDigit(digit);
    }
}

private void OnDestroy()
{
    StopTimer ();
    SaveStats ();
}
}

```

Необходимо создать пустой объект, и добавить скрипт MathCrosswordGame. В инспекторе необходимо добавить текстовые объекты TopPanel для отображения результатов, панель кроссворда и каждую ее ячейку, также добавить кнопки управления подсказка и новый кроссворд, и объекты модального окна победы (см. рис. 11).

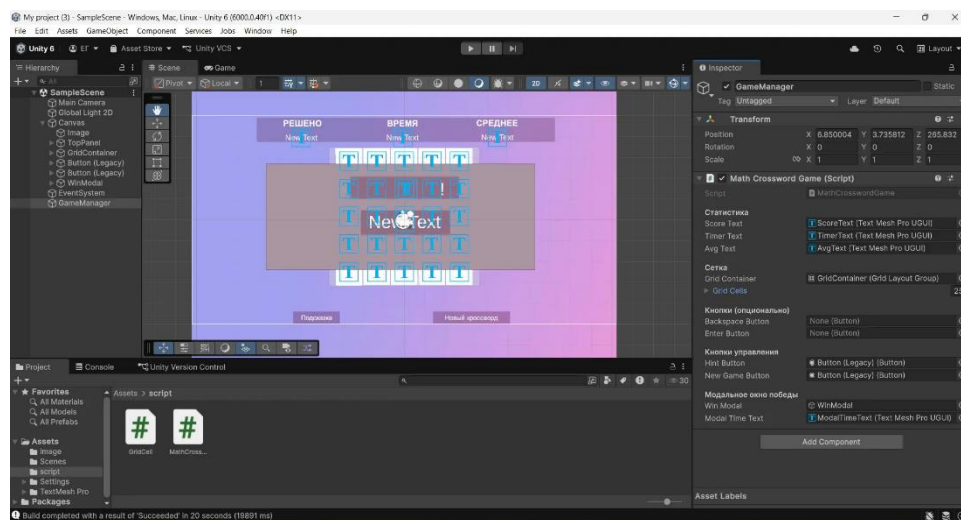


Рисунок 11 – Механика игры

Все настройки игры завершены. Готовая игра (см. рис. 11. 1, 11. 2, 11. 3).

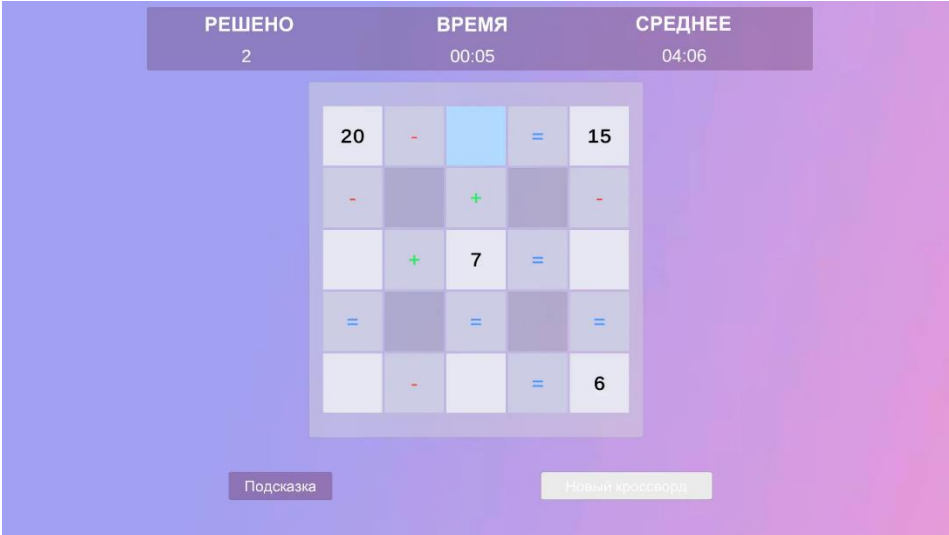


Рисунок 11.1 – Готовая игра

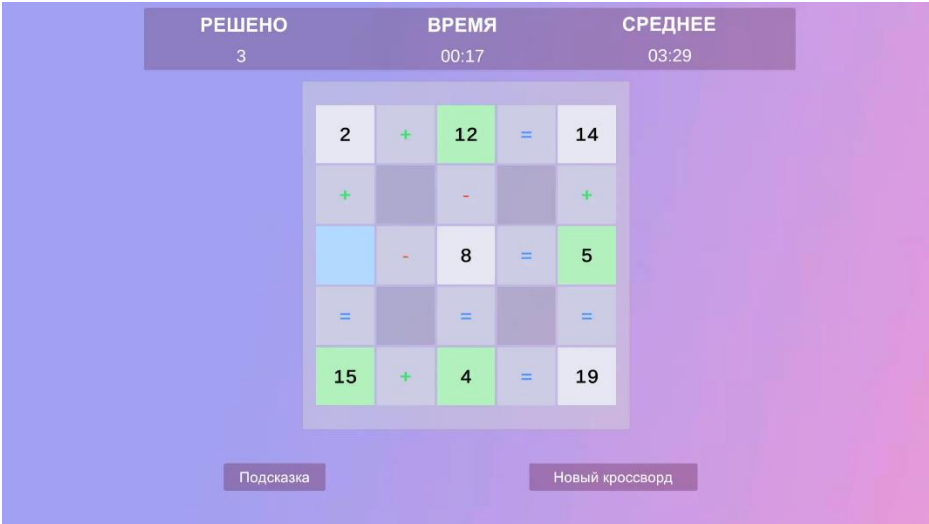


Рисунок 11.2 – Готовая игра

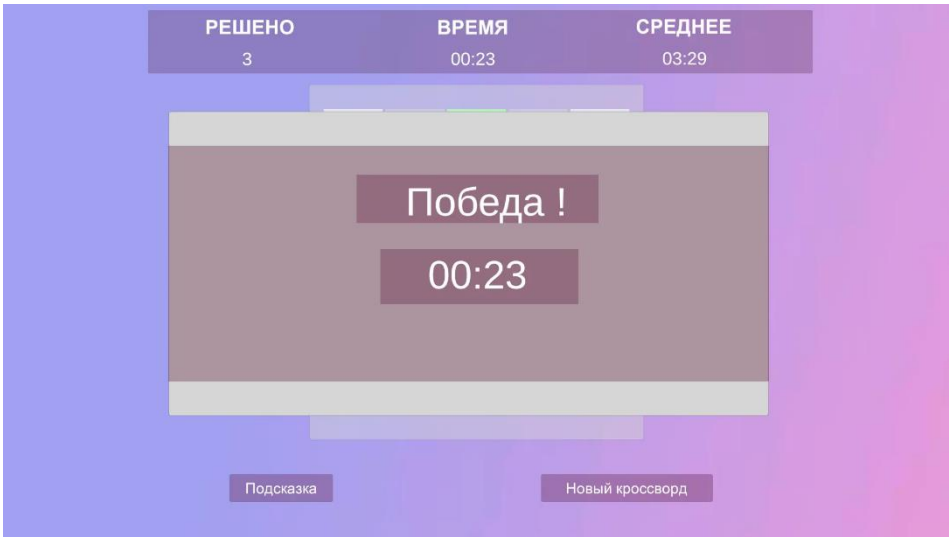


Рисунок 11.3 – Готовая игра

4 Выводы

В данной работе был показан процесс создания компьютерной игры «Пазл знаний» с помощью кроссплатформенной среды для создания интерактивных 2D и 3D приложений. Научная статья может служить методическим пособием в учебной деятельности.

### **Библиографический список**

1. Унарова А. Е. Разработка игры на платформе Unity // Технические и естественные науки: вчера, сегодня, завтра. Волгоград: Общество с ограниченной ответственностью "Сфера", 2024. С. 22-25. URL: <https://www.elibrary.ru/item.asp?id=68624300>
2. Фаст Я. А. Разработка 2D игр с элементами головоломок на платформе Unity // V Международная научная конференция по междисциплинарным исследованиям. Екатеринбург: Общество с ограниченной ответственностью "Институт Цифровой Экономики и Права", 2024. С. 266-269. URL: <https://www.elibrary.ru/item.asp?id=67143462>
3. Максимов Н. В. Разработка мобильной 2D игры "Bootur Jump" на Unity // Фундаментальные и прикладные аспекты развития современной науки. Уфа: Общество с ограниченной ответственностью "Научно-издательский центр "Вестник науки", 2023. С. 306-308. URL: <https://www.elibrary.ru/item.asp?id=59762421>
4. Любич А. А. Создание 2D-игры «б-у-у-м» на движке Unity // Фундаментальные и прикладные исследования молодых ученых. Новокузнецк: Кемеровский государственный университет, 2025. С. 167-169. URL: <https://www.elibrary.ru/item.asp?id=82653081>