

Создание игры «Крестики-нолики» на языке программирования C# в Visual Studio

Анишкова Анастасия Сергеевна

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

Целью данного исследования является разработка компьютерной версии классической игры «Крестики-нолики», позволяющей двум игрокам или одному игроку против системы соревноваться на стандартном поле 3x3, с возможностью выбора режима игры и корректной реализацией логики выигрыша. Работа выполнялась в среде Visual Studio с использованием языка C#, что обеспечило высокий уровень функциональности, стабильности и удобства разработки благодаря мощным инструментам отладки, поддержке объектно-ориентированного программирования и интеграции с библиотеками .NET Framework. C# — это современный, строго типизированный, объектно-ориентированный язык программирования, разработанный корпорацией Microsoft в рамках инициативы .NET. Он сочетает в себе мощь и гибкость низкоуровневых языков с удобством и безопасностью высокого уровня. Результат может быть применён в обучающих или развлекательных целях.

Ключевые слова: Крестики-нолики, C#, NET Framework, Visual Studio, игра.

Creating a Tic-Tac-Toe game in the C# programming language in Visual Studio

Anishkova Anastasia Sergeevna

Sholom-Aleichem Priamurskiy State University

Student

Abstract

The purpose of this research is to develop a computer version of the classic Tic-Tac-Toe game, which allows two players or one player to compete against the system on a standard 3x3 field, with the ability to select the game mode and correctly implement the winning logic. The work was performed in the Visual Studio environment using the C# language, which provided a high level of functionality, stability and ease of development thanks to powerful debugging tools, support for object-oriented programming and integration with libraries.NET Framework. C# (pronounced "c-sharp") is a modern, strongly typed, object-oriented programming language developed by Microsoft as part of the .NET initiative. It combines the power and flexibility of low-level languages with the convenience and security of high-level languages. The result can be used for educational or entertainment purposes.

Key words: Tic-Tac-Toe, C#, NET Framework, Visual Studio, game.

1 Введение

1.1 Актуальность

Актуальность создания игры «Крестики-нолики» на языке программирования C# в среде Visual Studio заключается в том, что этот проект позволяет наглядно продемонстрировать применение базовых принципов программирования, таких как алгоритмизация, работа с событиями, управление пользовательским интерфейсом и реализация логики взаимодействия между элементами приложения.

1.2 Обзор исследований

В статье Н. А. Базеевой и Д. С. Лебедева описано исследование о начале игровой индустрии, а также рассмотрены особенности языков программирования для разработки игр [1]. R. Bourbia и др. в своей научной работе рассказывают об игре, цель которой улучшение навыков сборки компьютера студентами в виртуальной лаборатории [2]. Об основных вопросах разработки однопользовательских двумерных игр на языке JavaScript написано в учебном пособии С. А. Беляева [3]. Н. В. Бужинская и Д. Ю. Тазабеков в статье рассматривают процесс разработки компьютерной игры «Winduza» на игровом движке Unity с использованием языка программирования C# [4]. Обширный обзор литературы по подходу, ориентированному на разработку видеоигр представлен в исследовании у Н. Engström и др. [5]. И. С. Голубь и Д. В. Лучанинов описывают алгоритм создания развивающей игры «Парные картинки» с помощью языка программирования JavaScript [6]. В статье Э. Э. Сейдаметова, А. Э. Шабанова представлены наиболее популярные средства разработки игр и проанализированы их характерные особенности [7].

1.3 Цель исследования

Разработка компьютерной версии классической игры «Крестики-нолики», позволяющей двум игрокам или одному игроку против системы соревноваться на стандартном поле 3x3, с возможностью выбора режима игры и корректной реализацией логики выигрыша.

2 Материалы и методы

В данном исследовании используется среда Visual Studio и язык программирования C#.

3 Результаты

Чтобы создать игру, откройте Visual Studio, создайте новый проект Windows Forms App (.NET Framework) с названием «TicTacToe», затем замените содержимое файла «Form1.cs» на код игры.

Игру можно скачать по ссылке:
<https://drive.google.com/drive/folders/1SwZ4EabuZdd-PmPUvSfgOFF8128MxTo8?usp=sharing>.

Начнем с объявления и инициализацию основных переменных, необходимых для реализации логики игры в крестики-нолики на поле 3x3, включая массив кнопок для игрового поля, флаг очередности хода игрока, счётчик ходов, элементы интерфейса для отображения информации и статистики, а также переменные для хранения результатов игр и режима против ИИ или двух игроков (рис.1).

```
using System;
using System.Drawing;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace TicTacToe1
{
    Ссылка:3
    public partial class Form1 : Form
    {
        private Button[,] buttons = new Button[3, 3];
        private bool isPlayerTurn = true;
        private int turns = 0;
        private Label labelInfo, labelScore;
        private Button buttonRestart, buttonMode;
        private int playerWins = 0, computerWins = 0, draws = 0;
        private bool playingAgainstAI = false;
    }
}
```

Рисунок 1. Объявление переменных класса

Далее реализуем конструктор формы и метод инициализации игры в крестики-нолики, в котором создаётся графический интерфейс с игровым полем 3x3, настраивается внешний вид формы, добавляются элементы управления и подписываются обработчики событий для кнопок (рис.2, 3).

```
Содерж: 1
private void InitializeGame()
{
    this.Text = "Крестики-Нолики";
    this.Size = new Size(320, 460);
    this.StartPosition = FormStartPosition.CenterScreen;
    this.BackColor = Color.FromArgb(40, 40, 40);

    TableLayoutPanel panel = new TableLayoutPanel
    {
        Dock = DockStyle.Fill,
        ColumnCount = 3,
        RowCount = 3,
        Padding = new Padding(5),
        CellBorderStyle = TableLayoutPanelCellBorderStyle.Single
    };

    for (int i = 0; i < 3; i++)
    {
        panel.ColumnStyles.Add(new ColumnStyle(SizeType.Percent, 33F));
        panel.RowStyles.Add(new RowStyle(SizeType.Percent, 33F));
    }

    for (int row = 0; row < 3; row++)
    {
        for (int col = 0; col < 3; col++)
        {
            Button btn = new Button
            {
                Font = new Font("Arial", 24F, FontStyle.Bold),
                Text = "",
                Dock = DockStyle.Fill,
                BackColor = Color.WhiteSmoke,
                FlatStyle = FlatStyle.Flat,
                FlatAppearance = { BorderSize = 0 }
            };
            btn.Click += ButtonClick;
            btn.MouseEnter += (s, e) => btn.BackColor = Color.LightBlue;
            btn.MouseLeave += (s, e) => btn.BackColor = Color.WhiteSmoke;
            buttons[row, col] = btn;
            panel.Controls.Add(btn, col, row);
        }
    }
}
```

Рисунок 2. Конструктор формы и инициализация игры

```
Controls.Add(panel);

labelScore = new Label
{
    Text = "Вы: 0 | Компьютер: 0 | Ничья: 0",
    Dock = DockStyle.Top,
    TextAlign = ContentAlignment.MiddleCenter,
    Height = 30,
    Font = new Font("Arial", 18F, FontStyle.Regular),
    ForeColor = Color.LightGray,
    BackColor = Color.FromArgb(50, 50, 50)
};
Controls.Add(labelScore);

labelInfo = new Label
{
    Text = "Ходит: Игрок",
    Dock = DockStyle.Top,
    TextAlign = ContentAlignment.MiddleCenter,
    Height = 40,
    Font = new Font("Arial", 14F, FontStyle.Bold),
    ForeColor = Color.White,
    BackColor = Color.FromArgb(60, 60, 60)
};
Controls.Add(labelInfo);

buttonRestart = new Button
{
    Text = "[R] Перезапуск",
    Dock = DockStyle.Bottom,
    Height = 40,
    Font = new Font("Arial", 12F, FontStyle.Bold),
    BackColor = Color.OrangeRed,
    ForeColor = Color.White,
    FlatStyle = FlatStyle.Flat
};
buttonRestart.Click += (s, e) => RestartGame();
Controls.Add(buttonRestart);

buttonMode = new Button
{
    Text = "[C] Против компьютера",
    Dock = DockStyle.Bottom,
    Height = 40,
    Font = new Font("Arial", 12F, FontStyle.Bold),
    BackColor = Color.Goldenrod,
    ForeColor = Color.White,
    FlatStyle = FlatStyle.Flat
};
buttonMode.Click += (s, e) =>
{
    playingAgainstAI = !playingAgainstAI;
    buttonMode.Text = playingAgainstAI ? "[P] Два игрока" : "[C] Против компьютера";
    RestartGame();
};
Controls.Add(buttonMode);
}
```

Рисунок 3. Конструктор формы и инициализация игры

Этот фрагмент кода реализует обработчик события нажатия на кнопку игрового поля, который управляет логикой хода игрока или компьютера, проверяет результаты игры после каждого хода и обеспечивает смену очередности ходов в зависимости от режима игры (рис.4).

```
private async void ButtonClick(object sender, EventArgs e)
{
    if (!isPlayerTurn || playingAgainstAI && turns >= 9) return;

    Button btn = sender as Button;

    if (btn.Text != "") return;

    btn.Text = "X";
    btn.ForeColor = Color.Red;
    btn.Font = new Font("Arial", 24F, FontStyle.Bold | FontStyle.Italic);
    HighlightButton(btn);

    turns++;
    if (CheckWinner("X"))
    {
        playerWins++;
        UpdateScore();
        MessageBox.Show("Вы выиграли!");
        RestartGame();
        return;
    }
    else if (turns == 9)
    {
        draws++;
        UpdateScore();
        MessageBox.Show("Ничья!");
        RestartGame();
        return;
    }

    isPlayerTurn = false;
    labelInfo.Text = "Ходит: Компьютер";

    await Task.Delay(500);
    ComputerMove();
}
```

Рисунок 4. Обработка ходов игроков

Далее реализуем логику хода компьютера в игре «Крестики-нолики» с использованием алгоритма Minimax для выбора оптимального хода, включая рекурсивную оценку всех возможных вариантов и определение наилучшего результата (рис.5).

```

Слайд 1
private void ComputerMove()
{
    var move = GetBestMove(); // Minimax
    if (move != null)
    {
        Button btn = buttons[move.Value.Row, move.Value.Col];

        btn.Text = "O";
        btn.ForeColor = Color.Blue;
        btn.Font = new Font("Arial", 24F, FontStyle.Bold | FontStyle.Italic);

        HighlightButton(btn);

        turns++;
        if (CheckWinner("O"))
        {
            computerWins++;
            UpdateScore();
            MessageBox.Show("Компьютер победил!");
            RestartGame();
            return;
        }
        else if (turns == 9)
        {
            draws++;
            UpdateScore();
            MessageBox.Show("Ничья!");
            RestartGame();
            return;
        }
    }

    isPlayerTurn = true;
    labelInfo.Text = "Ходит: Игрок";
}

```

Рисунок 5. Логика ИИ (компьютерный игрок)

Напишем фрагмент кода, который включает в себя метод обновления отображения счёта и реализацию функции выбора наилучшего хода для компьютера с использованием алгоритма Minimax, который перебирает все возможные ходы и оценивает их эффективность (рис.6).

```

Слайд 4
private void UpdateScore()
{
    labelScore.Text = $"Вы: {playerWins} | Компьютер: {computerWins} | Ничьи: {draws}";
}

Слайд 1
private (int Row, int Col)? GetBestMove()
{
    int bestScore = int.MinValue;
    (int Row, int Col)? bestMove = null;

    for (int r = 0; r < 3; r++)
    {
        for (int c = 0; c < 3; c++)
        {
            if (buttons[r, c].Text == "")
            {
                buttons[r, c].Text = "O";
                int score = Minimax(0, false);
                buttons[r, c].Text = "";

                if (score > bestScore)
                {
                    bestScore = score;
                    bestMove = (r, c);
                }
            }
        }
    }

    return bestMove;
}

```

Рисунок 6. Обновление счёта и выбор наилучшего хода

Реализуем (рис.7) рекурсивный алгоритм Minimax для оценки всех возможных исходов игры с текущей позиции, где компьютер выбирает наиболее выгодный ход, максимизируя свой выигрыш, а игрок — минимизируя его, на основе анализа побед, поражений и ничьих на определённую глубину.

```

Ссылка: 3
private int Minimax(int depth, bool isMaximizing)
{
    string result = "";
    if (CheckWinner("X")) result = "X";
    else if (CheckWinner("O")) result = "O";
    else if (turns + depth >= 9) result = "Draw";

    if (result == "X") return -1;
    if (result == "O") return 1;
    if (result == "Draw") return 0;

    int bestScore;
    if (isMaximizing)
    {
        bestScore = int.MinValue;
        for (int r = 0; r < 3; r++)
        {
            for (int c = 0; c < 3; c++)
            {
                if (buttons[r, c].Text == "")
                {
                    buttons[r, c].Text = "O";
                    int score = Minimax(depth + 1, false);
                    buttons[r, c].Text = "";
                    bestScore = Math.Max(score, bestScore);
                }
            }
        }
    }
    else
    {
        bestScore = int.MaxValue;
        for (int r = 0; r < 3; r++)
        {
            for (int c = 0; c < 3; c++)
            {
                if (buttons[r, c].Text == "")
                {
                    buttons[r, c].Text = "X";
                    int score = Minimax(depth + 1, true);
                    buttons[r, c].Text = "";
                    bestScore = Math.Min(score, bestScore);
                }
            }
        }
    }

    return bestScore;
}

```

Рисунок 7. Алгоритм Minimax

Этот фрагмент кода реализует метод визуального выделения нажатой кнопки с последующим восстановлением её цвета и функцию проверки победителя в игре, которая анализирует все возможные выигрышные комбинации для указанного игрока (рис.8).

```

Ссылка 2
private void HighlightButton(Button btn)
{
    var originalBackColor = btn.BackColor;
    btn.BackColor = isPlayerTurn ? Color.MistyRose : Color.Lavender;

    Task.Delay(100).ContinueWith(t => this.Invoke((MethodInvoker)delegate {
        btn.BackColor = originalBackColor;
    })), TaskScheduler.FromCurrentSynchronizationContext());
}

Ссылка 4
private bool CheckWinner(string player)
{
    for (int i = 0; i < 3; i++)
    {
        if ((buttons[i, 0].Text == player && buttons[i, 1].Text == player && buttons[i, 2].Text == player) ||
            (buttons[0, i].Text == player && buttons[1, i].Text == player && buttons[2, i].Text == player))
        {
            return true;
        }
    }

    if ((buttons[0, 0].Text == player && buttons[1, 1].Text == player && buttons[2, 2].Text == player) ||
        (buttons[0, 2].Text == player && buttons[1, 1].Text == player && buttons[2, 0].Text == player))
    {
        return true;
    }

    return false;
}

```

Рисунок 8. Проверки победителя в игре

Реализует метод перезапуска игры, который сбрасывает текст и стиль всех кнопок игрового поля, а также обнуляет счётчики и устанавливает начальное состояние игры (рис.9).

```

Ссылка 6
private void RestartGame()
{
    foreach (Button btn in buttons)
    {
        btn.Text = "";
        btn.BackColor = Color.WhiteSmoke;
        btn.Font = new Font("Arial", 24F, FontStyle.Bold);
    }

    turns = 0;
    isPlayerTurn = true;
    LabelInfo.Text = "Ходит: Игрок";
}

```

Рисунок 9. Перезапуск игры

Код программы успешно готов к запуску — нажмите Ctrl + F5, чтобы начать игру. После открытия игрового поля 3x3 выберите предпочтительный режим: игра против человека или против компьютера. Теперь вы сможете приступить к игре (Рис.10).

На рисунке 10 изображен интерфейс игры с человеком.



Рисунок 10. Игра с человеком

Интерфейс игры против компьютера изображен на рисунке 11.

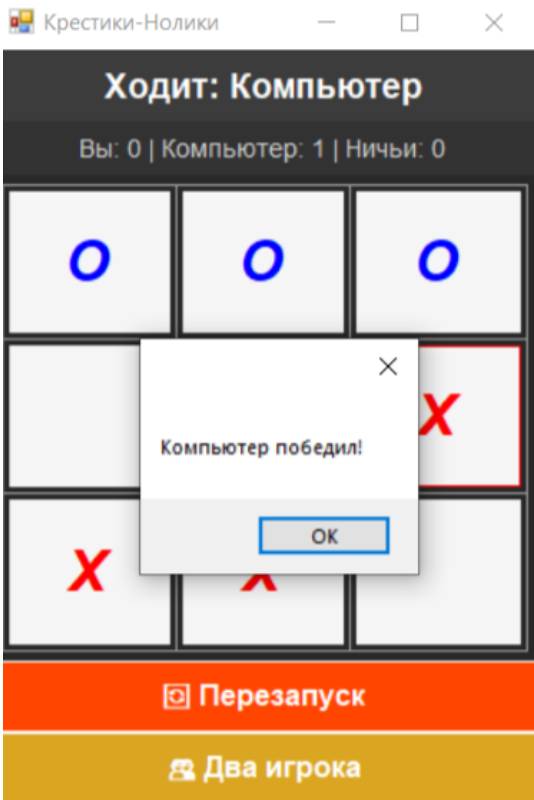


Рисунок 11. Игра с компьютером

Библиографический список

1. Базеева Н. А., Лебедев Д. С. Языки программирования для создания игр //E-Scio. 2019. №. 4 (31).
2. Bourbia R. et al. Development of serious game to improve computer assembly skills //Procedia-Social and Behavioral Sciences. 2014. Т. 141. С. 96-100.
3. Беляев С.А. Разработка игр на языке javascript. СПб.: Издательство Лань, 2016. 128 с.
4. Бужинская Н. В., Тазабеков Д. Ю. Разработка компьютерной обучающей игры "WINDUZA" //Заметки ученого. 2019. №. 2. С. 16-21.
5. Engström H. et al. Game development from a software and creative product perspective: A quantitative literature review approach //Entertainment Computing. 2018. Т. 27. С. 10-22.
6. Голубь И. С., Лучанинов Д. В. Разработка обучающей игры «Парные картинки» с помощью JavaScript //Постулат. 2016. №. 12.
7. Сейдаметов Э. Э., Шабанов А. Э. Современные средства разработки игр //Информационно-компьютерные технологии в экономике, образовании и социальной сфере. 2020. №. 1. С. 54-62.