

Создание помощника «Заметки» на языке программирования C# в Visual Studio

Саяпин Никита Анатольевич

Приамурский государственный университет им.Шолом-Алейхема

Студент

Аннотация

Цель исследования — создание приложения «Заметки» на C# с графическим интерфейсом с использованием Windows Forms. Реализация позволяет пользователю создавать, редактировать, сохранять и удалять текстовые заметки, с отображением списка заметок и их содержимого. Проект выполнен в Visual Studio, демонстрируя работу с .NET Framework, элементами управления, обработкой событий и файловым вводом/выводом. Приложение подходит для обучения программированию и повседневного использования.

Ключевые слова: Заметки, C#, Windows Forms, Visual Studio, .NET Framework, приложение.

Creating the Notes Assistant in the C# programming language in Visual Studio

Sayapin Nikita Anatolyevich

Sholom-Aleichem Priamursky State University

Student

Abstract

The purpose of the article is to create a Notes application in C# with a graphical interface using Windows Forms. The implementation allows the user to create, edit, save, and delete text notes, displaying a list of notes and their contents. The project is executed in Visual Studio, demonstrating the work with .NET Framework, controls, event handling, and file I/O. The application is suitable for learning programming and everyday use.

Keywords: Notes, C#, Windows Forms, Visual Studio, .NET Framework, an application.

1. Введение

1.1 Актуальность

Разработка приложения «Заметки» на языке программирования C# с использованием Windows Forms в среде Visual Studio обусловлена необходимостью создания удобного инструмента для организации личных записей. Проект позволяет изучить ключевые аспекты создания пользовательских интерфейсов, включая работу с элементами управления, обработку пользовательских действий и взаимодействие с файловой системой.

Реализация демонстрирует практическое применение C# для построения функциональных приложений, способствуя развитию навыков программирования и созданию полезных решений для повседневных задач.

1.2 Обзор исследований

В статье Н.М. Мельникова и др. рассматриваются инструменты разработки VR-приложений, включая Unity, C# и Blender, и их роль в создании интерактивного досуга [1]. Е.А. Ульянов показал и описал процесс создания игры «Угадай слово на c#» [2]. С. Горнаков в своей книге с помощью студии XNA Game Studio Express рассматривает полный цикл создания компьютерных игр для операционной системы Windows. Рассказывает, как освоить основы работ с инструментариями Visual C# Express и XNA Game Studio Express. Научиться работать с двухмерной и трехмерной графикой, анимацией, техникой создания игровых классов и формированием механизма игровых состояний [3]. В книге Х. Ферроне «Изучаем C# через разработку игр на Unity» дается возможность с нуля изучить программирование на C# без терминов и непонятной логики программирования, причем процесс изучения сопровождается созданием простой игры на Unity [4]. В.Подбельский рассказывает про базовый курс программирования для начинающих [5].

1.3 Цель исследования

Целью исследования является разработка приложения «Заметки» на языке C# с графическим интерфейсом на базе Windows Forms, обеспечивающего создание, редактирование, сохранение и удаление текстовых заметок. Реализация включит в себя интуитивный интерфейс с элементами управления, взаимодействие с файловой системой для хранения данных и обработку пользовательских действий, с корректной логикой управления списком заметок.

2 Материалы и методы

В данном исследовании используется среда Visual Studio и язык программирования C# с фреймворком Windows Forms для создания графического интерфейса. Применяются библиотеки .NET Framework для обработки событий, управления графикой и реализации игровой логики.

2.2 Подготовка

В рамках подготовки к разработке настольного приложения «Заметки» на платформе Windows Forms, реализованного с использованием языка C#, первым этапом выступает создание проекта в среде Visual Studio. Пользователь инициирует процесс через интерфейс создания нового проекта, выбирая шаблон «Windows Forms App (.NET Framework)» с версией .NET Framework 4.8. Далее задаётся имя проекта и выбирается директория для хранения файлов. После этого автоматически формируются базовые файлы: Program.cs, Form1.cs и Form1.Designer.cs, содержащие, соответственно, точку входа, основной код формы и её визуальное описание (Рис.1).

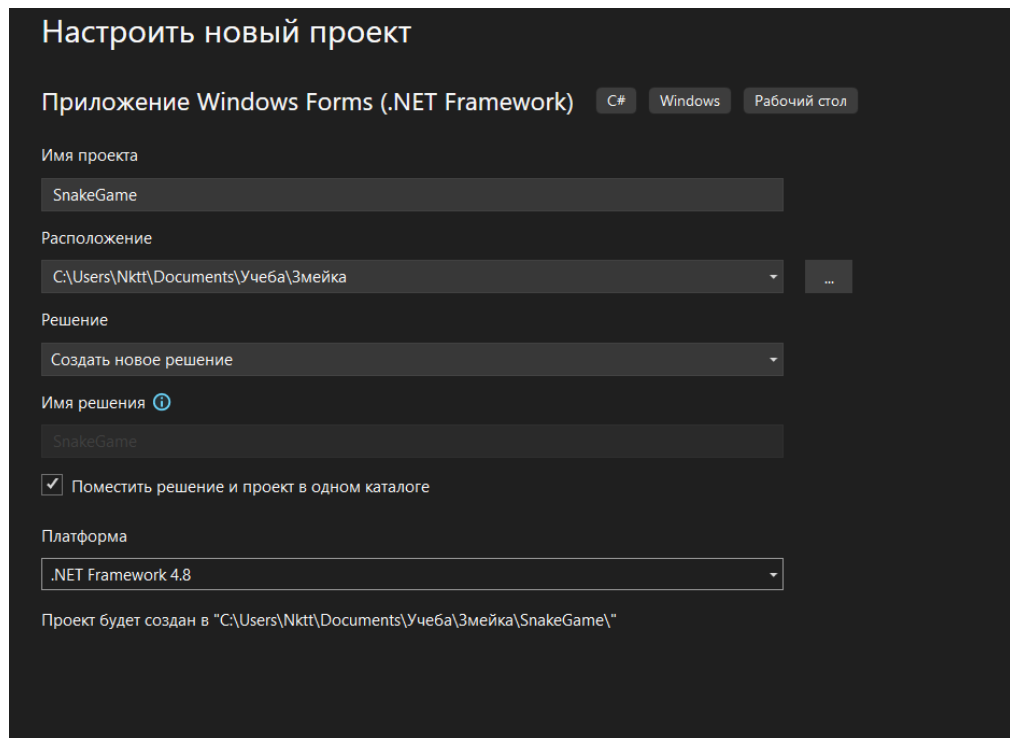


Рисунок 1 – Окно создания проекта

3. Результаты

На следующем этапе осуществляется объявление и инициализация ключевых переменных, необходимых для функционирования интерфейса: графических элементов управления (таких как `ListBox`, `TextBox`, `Button`), а также переменной для хранения пути к файлу, используемому при сохранении данных заметок. Все эти элементы размещаются и конфигурируются в методе инициализации компонентов формы, который вызывается в конструкторе (Рис.2).

```
using System;
using System.IO;
using System.Linq;
using System.Windows.Forms;

namespace NoteTakingApp
{
    Ссылок: 2
    class NoteTakingApp : Form
    {
        private ListBox noteList;
        private TextBox noteText;
        private Button addButton;
        private Button saveButton;
        private Button deleteButton;
        private string notesFilePath = "notes.txt";
    }
}
```

Рисунок 2 - Объявление переменных класса

Основная логика приложения реализуется через обработчиков событий, обеспечивающих взаимодействие пользователя с системой. Так, метод `AddButton_Click` отвечает за добавление новой заметки в список при условии, что поле ввода не пусто; `SaveButton_Click` обновляет выбранную заметку; `DeleteButton_Click` — удаляет её из списка; метод `NoteList_SelectedIndexChanged` обеспечивает отображение текста выбранной заметки в соответствующем текстовом поле. Эти методы обеспечивают базовую CRUD-функциональность (создание, чтение, обновление, удаление). (Рис. 3).

```
private void InitializeComponents()
{
    // Список заметок
    noteList = new ListBox
    {
        Location = new System.Drawing.Point(10, 10),
        Size = new System.Drawing.Size(200, 300),
    };
    noteList.SelectedIndexChanged += NoteList_SelectedIndexChanged;

    // Поле для ввода заметки
    noteText = new TextBox
    {
        Location = new System.Drawing.Point(220, 10),
        Size = new System.Drawing.Size(350, 300),
        Multiline = true,
        ScrollBars = ScrollBars.Vertical
    };

    // Кнопка "Добавить"
    addButton = new Button
    {
        Text = "Добавить",
        Location = new System.Drawing.Point(220, 320),
        Size = new System.Drawing.Size(100, 30)
    };
    addButton.Click += AddButton_Click;

    // Кнопка "Сохранить"
    saveButton = new Button
    {
        Text = "Сохранить",
        Location = new System.Drawing.Point(330, 320),
        Size = new System.Drawing.Size(100, 30)
    };
    saveButton.Click += SaveButton_Click;
}
```

Рисунок 3 – Конструктор класса `NoteTakingApp`

Логика приложения управляется несколькими обработчиками событий: `AddButton_Click` для добавления новой заметки (Рис.4), `SaveButton_Click` для сохранения изменений, `DeleteButton_Click` для удаления заметки (Рис.5) и `NoteList_SelectedIndexChanged` для отображения выбранной заметки в текстовом поле. Метод `AddButton_Click` проверяет, что введенный текст не пуст, добавляет его в список и сохраняет в файл. Метод `SaveButton_Click` обновляет выбранную заметку, а `DeleteButton_Click` удаляет её из списка. Метод `NoteList_SelectedIndexChanged` отображает текст выбранной заметки в текстовом поле (Рис. 6).

```
Ссылка: 1
private void AddButton_Click(object sender, EventArgs e)
{
    if (!string.IsNullOrEmpty(noteText.Text))
    {
        noteList.Items.Add(noteText.Text);
        noteText.Clear();
        SaveNotes();
    }
    else
        MessageBox.Show("Введите текст заметки!", "Ошибка");
}
```

Рисунок 4 – Обработчик события AddButton_Click

```
Ссылка: 1
private void SaveButton_Click(object sender, EventArgs e)
{
    if (noteList.SelectedIndex >= 0 && !string.IsNullOrEmpty(noteText.Text))
    {
        noteList.Items[noteList.SelectedIndex] = noteText.Text;
        SaveNotes();
    }
    else
        MessageBox.Show("Выберите заметку или введите текст!", "Ошибка");
}

Ссылка: 1
private void DeleteButton_Click(object sender, EventArgs e)
{
    if (noteList.SelectedIndex >= 0)
    {
        noteList.Items.RemoveAt(noteList.SelectedIndex);
        noteText.Clear();
        SaveNotes();
    }
    else
        MessageBox.Show("Выберите заметку для удаления!", "Ошибка");
}
```

Рисунок 5 – Обработчик событий SaveButton_Click и DeleteButton_Click

```
Ссылка: 1
private void NoteList_SelectedIndexChanged(object sender, EventArgs e)
{
    if (noteList.SelectedIndex >= 0)
        noteText.Text = noteList.SelectedItem.ToString();
}
```

Рисунок 6 – Метод NoteList_SelectedIndexChanged

Для обеспечения сохранности данных между сессиями разработаны методы LoadNotes (Рис.7) и SaveNotes (Рис.8). Первый считывает данные из текстового файла при запуске программы, второй — сохраняет изменения при каждом действии пользователя, изменяющем список заметок. Такое решение позволяет реализовать простую, но эффективную систему хранения данных на основе файловой структуры без подключения базы данных.

```
Ссылка: 1
private void LoadNotes()
{
    if (File.Exists(notesFilePath))
    {
        var notes = File.ReadAllLines(notesFilePath);
        foreach (var note in notes)
            if (!string.IsNullOrEmpty(note))
                noteList.Items.Add(note);
    }
}
```

Рисунок 7 - Метод LoadNotes

```
Ссылка: 3
private void SaveNotes()
{
    File.WriteAllLines(notesFilePath, noteList.Items.Cast<string>());
}
```

Рисунок 8 - Метод SaveNotes

После компиляции и запуска программы пользователь получает доступ к удобному графическому интерфейсу: список заметок расположен в левой части окна, текстовое поле — справа, а кнопки управления размещены снизу. Интерфейс интуитивно понятен, обеспечивает быструю ориентацию в действиях и содержит базовые проверки на корректность ввода (например, защита от добавления пустой заметки). По мере работы пользователь может создавать, редактировать и удалять записи, взаимодействуя с приложением в интерактивном режиме. Таким образом, реализуется полноценный функционал для управления заметками в настольной среде (Рис.9).

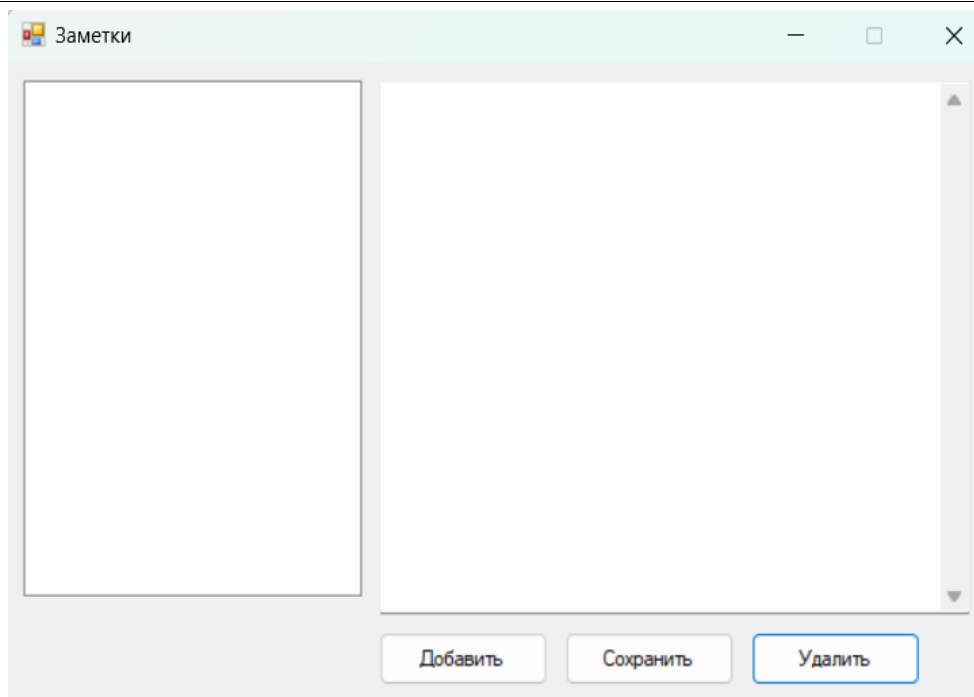


Рисунок 9 – Интерфейс приложения в начальном состоянии

Во время работы пользователь может добавлять новые заметки, редактировать существующие и удалять ненужные. Интерфейс остаётся интуитивным, с чётким разделением элементов и сообщениями об ошибках при некорректных действиях (например, попытке добавить пустую заметку). Интерфейс приложения с несколькими добавленными заметками изображён на рисунке 10.

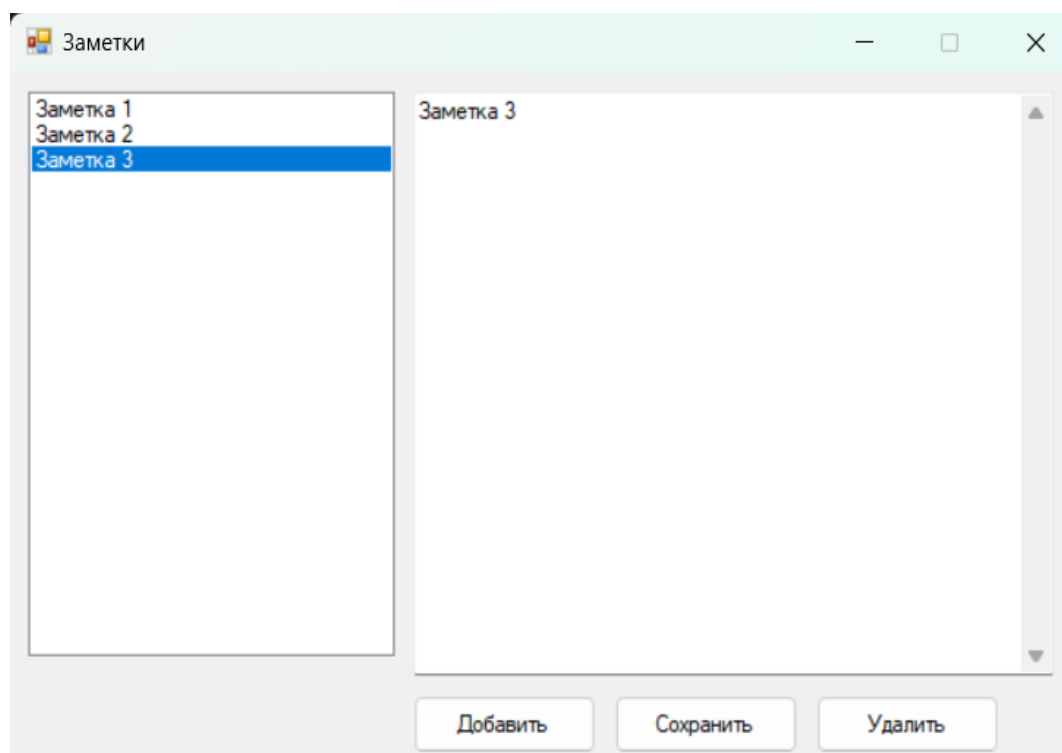


Рисунок 10 – Интерфейс приложения с несколькими заметками

4 Выводы

Разработка приложения «Заметки» на языке программирования C# с использованием фреймворка Windows Forms в среде Visual Studio успешно реализовала функциональное приложение с графическим интерфейсом для управления текстовыми записями. Проект позволил изучить и применить ключевые принципы программирования, включая создание пользовательских интерфейсов, обработку событий, работу с элементами управления и взаимодействие с файловой системой. Использование C# и .NET Framework обеспечило надёжность, удобство разработки и возможность дальнейшего расширения функционала приложения.

Библиографический список

1. Мельников М.Н. и др. Приложение виртуальной реальности «Тур по музеям» // BIG DATA и анализ высокого уровня: сборник научных статей X Международной научно-практической конференции, Минск, 13 марта 2024 г.: в 2 ч. Ч. 2. Минск, 2024. С. 383–390.
2. Ульянов Е. А. Создание простой игры «Угадай слово» на C // Постулат. 2024. №. 1.
3. Горнаков С. Программирование компьютерных игр под Windows в XNA Game Studio Express. Litres, 2022.
4. Ферроне Х. Изучаем C# через разработку игр на Unity. СПб.: Питер, 2022. 400 с.
5. Подбельский В. Программирование. Базовый курс C#. Учебник для СПО. Litres, 2020.