МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ «МИСИС»**



*ИНСТИТУТ КОМПЮТЕРНЫХ НАУК*

*НАПРАВЛЕНИЕ ИНФОРМАТИКА и ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА*

**Семестровая**

**Домашняя**

**Работа**

**на тему:** Сервис для размещения текстовых файлов и обработки текста с помощью регулярных выражений.

*Студент: Ерицян Давид Овикович  БИВТ-23-13*

*Руководитель* *работы:* Куреноква Татьяна Васильевна

Оглавление

Цель работы, техническое задание..................................................................3

Описание проекта..............................................................................................4

Техническое описание.......................................................................................6

Приложения........................................................................................................7

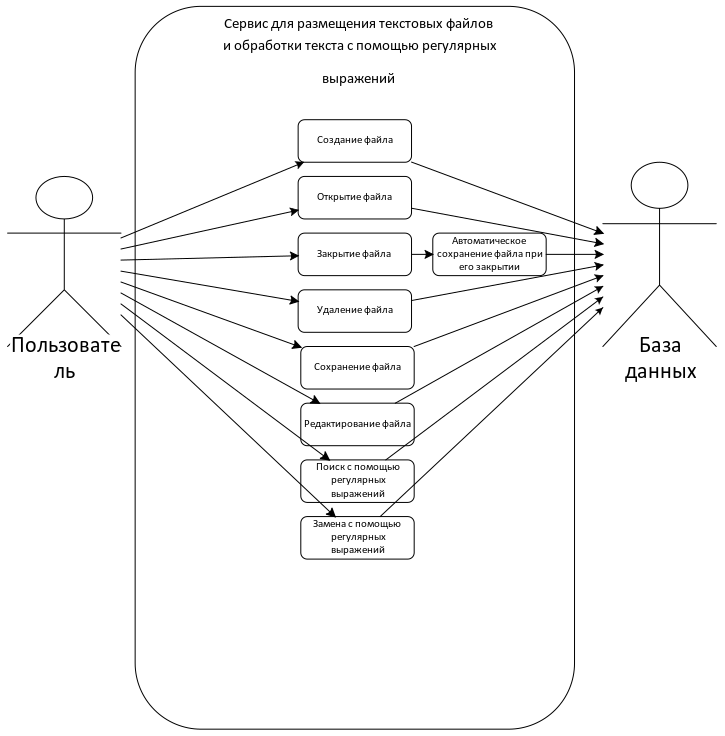
1. Цель работы:

1. Создать приложение Windows Form в Visual Studio на языке C#.
2. Разработать сервис для размещения файлов.
3. Добавить возможность обработки файлов с помощью регулярных выражений

2. Техническое задание:

Разработать сервис для работы с файлами. Он должен обеспечить работу следующих функций:

1. Возможность открывать и изменять файлы из базы данных.
2. Создание новых файлов и удаление.
3. Переключение между открытыми файлами.
4. Поиск с помощью регулярных выражений.
5. Замена участков текста с помощью регулярных выражений

Рис. 1

4. Описание проекта:

При запуске программы открывается Windows Form (рис. 2). В окне, которое находится левее кнопки “Открыть” пользователь пишет имя файла которое он хочет открыть. Если файла с таким именем не существует, то программа сама его создаст. В окне под номером 1 будет текст файла, где его можно редактировать. С помощью вкладок под номером 3 можно переключаться между открытыми файлами. Кнопки “Закрыть” и “Удалить” будут действовать на выбранный файл.

В окне слева от кнопки “Найти” записывается либо слово, которое нужно найти в тексте, либо шаблон для использования регулярных запросов.

Рассмотрим вкратце некоторые элементы синтаксиса регулярных выражений:

1. ^: соответствие должно начинаться в начале строки (например, выражение @"^пр\w\*" соответствует слову "привет" в строке "привет мир")
2. $: конец строки (например, выражение @"\w\*ир$" соответствует слову "мир" в строке "привет мир", так как часть "ир" находится в самом конце)
3. .: знак точки определяет любой одиночный символ (например, выражение "м.р" соответствует слову "мир" или "мор")
4. \*: предыдущий символ повторяется 0 и более раз
5. +: предыдущий символ повторяется 1 и более раз
6. ?: предыдущий символ повторяется 0 или 1 раз
7. \s: соответствует любому пробельному символу
8. \S: соответствует любому символу, не являющемуся пробелом
9. \w: соответствует любому алфавитно-цифровому символу
10. \W: соответствует любому не алфавитно-цифровому символу
11. \d: соответствует любой десятичной цифре
12. \D: соответствует любому символу, не являющемуся десятичной цифрой

При нажатии кнопки “Найти” в окне под номером 2 выведется сначала число найденных слов, потом сами слова в том порядке в каком они в тексте. Затем они же, но сгруппированные по первой букве. Далее в окне слева от кнопки “Заменить” можно написать слово и при нажатии все найденные значения заменятся на это.

В левом нижнем углу есть кнопка “Сохранить” при его нажатии выбранный файл сохраняется. Также файл автоматически сохраняется при его закрытии.

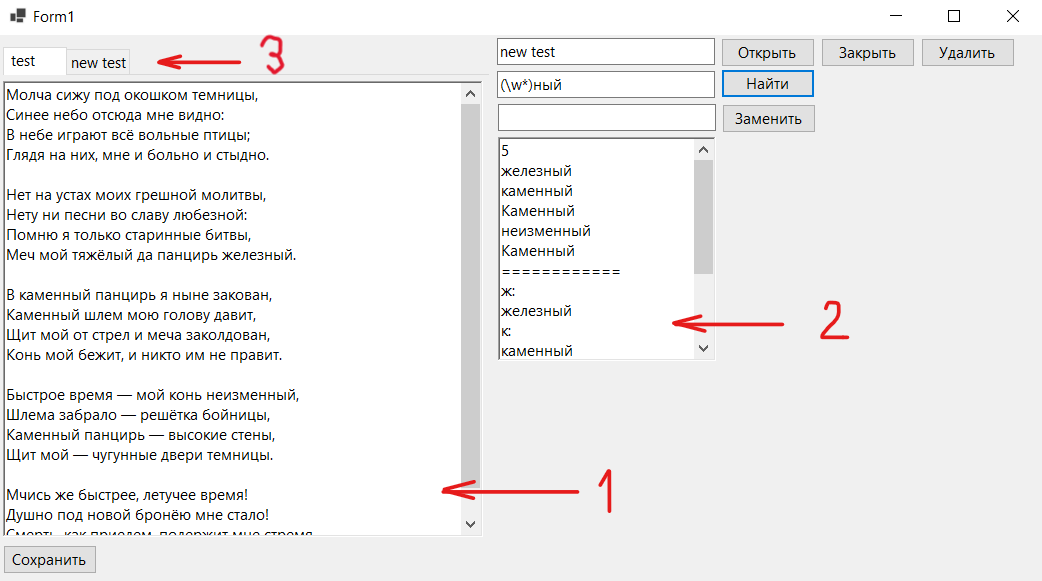


Рис. 2

5. Техническое описание:

Я использовал среду разработки Visual Studio 2022 и создал следующие классы:

1. NorFile (см. Приложение 1) - класс, где объявлены поля text, path, name.Text хранит текст файла. Path это путь к файлу. Все файлы хранятся в отдельной папке Files. Name это имя файла. Метод Find находит все слова по заданному шаблону, используя подключенную библиотеку System.Text.RegularExpressions для регулярных выражений. Метод возвращает массив строк всех найденных слов, либо, если совпадений не было, одномерный массив со строкой “Совпадений не найдено”. Метод Replace заменяет все найденные слова по заданному шаблону на то, что указал пользователь при работе в программе. Метод Rewrite переписывает текст файла. Delete удаляет выбранный файл из папки Files.
2. FileList (см. Приложение 2) - ссылочный класс, массив экземпляров класса NorFile (см. Приложение 1). В нём есть методы Add, Remove и CurrentFile. Add добавляет переменную NorFile в массив. Remove удаляет переменную из массива. CurrentFile возвращает индекс переменной массива по заданному имени файла, если его нет в массиве, то возвращает значение -1.
3. Form1 (см. Приложение 3) - класс, для формы, где реализован весь интерфейс программы.

6. Приложения

1. NorFile.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using System.Text.RegularExpressions;

namespace FileService

{

public class NorFile

{

private string text;

private string path;

private string name;

public string Text { get { return text; } }

public string Path { get { return path; } }

public string Name { get { return name; } }

public NorFile()

{

this.text = null;

this.path = null;

this.name = null;

}

public NorFile(string text, string name)

{

this.text = text;

this.name= name;

this.path = "";

}

public NorFile(string path)

{

this.path = "Files\\" + path + ".txt";

if (!File.Exists(this.path))

{

File.Create(this.path);

this.text = "";

}

else

this.text = File.ReadAllText(this.path);

this.name = path;

}

public string[] Find(string pattern)

{

Regex reg = new Regex(@pattern);

List<string> allMatches = new List<string>();

MatchCollection matches = reg.Matches(text);

if (matches.Count > 0)

{

foreach (Match match in matches)

allMatches.Add(match.Value);

}

else

allMatches.Add("Совпадений не найдено");

return allMatches.ToArray();

}

public void Replace(string pattern, string target)

{

Regex reg = new Regex(pattern);

this.text = reg.Replace(text, target);

}

public async void Rewrite(string NewText)

{

await File.WriteAllTextAsync(this.path, NewText);

}

public void Delete()

{

File.Delete(this.path);

}

}

}

2. FileList.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace FileService

{

public class FileList

{

private NorFile[] fs;

private int size;

public int Size { get { return fs.Length; } }

public FileList()

{

this.fs = new NorFile[0];

this.size = 0;

}

public FileList(NorFile[] fs)

{

this.fs = fs;

this.size = fs.Length;

}

public NorFile this[int index]

{

get => fs[index];

set => fs[index] = value;

}

public System.Collections.IEnumerator GetEnumerator()

{

foreach (NorFile file in this.fs)

yield return file;

}

public void Add(NorFile file)

{

NorFile[] temp = this.fs;

this.fs = new NorFile[++this.size];

for (int i = 0; i < temp.Length; i++)

this.fs[i] = temp[i];

this.fs[this.size - 1] = file;

}

public void Remove(int indx)

{

NorFile[] temp = new NorFile[--this.size];

int j = 0;

for (int i = 0; i <= this.size; i++)

{

if (i == indx)

continue;

temp[j++] = this.fs[i];

}

this.fs = temp;

}

public void Remove(NorFile file)

{

NorFile[] temp = new NorFile[--this.size];

int j = 0;

foreach (NorFile f in fs)

{

if (f == file)

continue;

temp[j++] = f;

}

this.fs = temp;

}

public int CurrentFile(string fileName)

{

for (int i = 0; i < this.size; i++)

if (fs[i].Path == fileName)

return i;

return -1;

}

}

}

3.Form1.cs:

namespace FileService

{

public partial class Form1 : Form

{

static int i = -1;

static int MainTextChangeCount = 0;

FileList files = new FileList();

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void MainText\_TextChanged(object sender, EventArgs e)

{

//if (MainTextChangeCount == 5)

//{

// fs[i].Rewrite(MainText.Text);

// MainTextChangeCount = 0;

//}

//else

// MainTextChangeCount++;

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void buttonOpen\_Click(object sender, EventArgs e)

{

if (textBox1.Text != "" && files.CurrentFile("Files\\" + textBox1.Text + ".txt") == -1)

{

TabPage tabPage = new TabPage(textBox1.Text);

tabControl1.TabPages.Add(tabPage);

files.Add(new NorFile(textBox1.Text));

MainText.Text = files[files.Size - 1].Text;

i = files.Size - 1;

tabControl1.SelectedTab = tabControl1.TabPages[i];

}

}

private void richTextBox2\_TextChanged(object sender, EventArgs e)

{

}

private void buttonSave\_Click(object sender, EventArgs e)

{

if (tabControl1.TabPages.Count > 0)

files[i].Rewrite(MainText.Text);

}

private void buttonClose\_Click(object sender, EventArgs e)

{

if (tabControl1.TabPages.Count > 0)

{

files[i].Rewrite(MainText.Text);

files.Remove(i);

tabControl1.TabPages.RemoveAt(i);

i = 0;

if (files.Size != 0)

MainText.Text = files[0].Text;

else

MainText.Text = "";

}

}

private void buttonDelete\_Click(object sender, EventArgs e)

{

if (tabControl1.TabPages.Count > 0)

{

files[i].Delete();

files.Remove(i);

tabControl1.TabPages.RemoveAt(i);

i = 0;

if (files.Size != 0)

MainText.Text = files[0].Text;

else

MainText.Text = "";

}

}

private void FindBox\_TextChanged(object sender, EventArgs e)

{

}

private void buttonFind\_Click(object sender, EventArgs e)

{

if (tabControl1.TabPages.Count > 0)

{

richTextBox2.Text = "";

string[] list = files[i].Find(@FindBox.Text);

if (list.Length != 1)

{

richTextBox2.Text += list.Length.ToString() + '\n';

foreach (string line in list)

richTextBox2.Text += line + '\n';

richTextBox2.Text += "============\n";

var groupList = from line in list

group line by line[0];

foreach (var key in groupList)

{

richTextBox2.Text += key.Key + ":\n";

foreach (var line in key)

richTextBox2.Text += line + '\n';

Console.WriteLine();

}

}

else

richTextBox2.Text = list[0] + "\n";

}

}

private void tabControl1\_SelectedIndexChanged(object sender, EventArgs e)

{

if (files.Size != 0)

{

MainText.Text = File.ReadAllText("Files\\" + tabControl1.SelectedTab.Text + ".txt");

i = files.CurrentFile("Files\\" + tabControl1.SelectedTab.Text + ".txt");

}

}

private void ReplaceBox\_TextChanged(object sender, EventArgs e)

{

}

private void buttonReplace\_Click(object sender, EventArgs e)

{

if (tabControl1.TabPages.Count > 0)

{

files[i].Replace(FindBox.Text, ReplaceBox.Text);

MainText.Text = files[i].Text;

}

}

}

}