**Занятие № 15**

**Дата выполнения работы:** 06.05.2023

# **Тема работы:** ««Работа с файлами. Потоки символов. Работа с каталогами. Сохранение объектов»

**Ход работы**

**Задание 1**

Создать класс XXXLog. Он должен отвечать за работу с текстовым файлом

xxxlogfile.txt. в который записываются все действия пользователя и

соответственно методами записи в текстовый файл, чтения, поиска нужной

информации.

a. Используя данный класс выполните запись всех

последующих действиях пользователя с указанием действия,

детальной информации (имя файла, путь) и времени

(дата/время)

**Листинг программы:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace App1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

string fileName = "D:\\Учёба\\Практика КПиЯП\\день 15\\myfile.txt";

string filePath = "D:\\Учёба\\Практика КПиЯП\\день 15\\";

string action = "Открытие файла";

XXXLog log = new XXXLog("D:\\Учёба\\Практика КПиЯП\\день 15\\xxxlogfile.txt");

log.WriteLog(action, fileName, filePath);

}

}

public class XXXLog

{

private readonly string logFilePath;

public XXXLog(string logFilePath)

{

this.logFilePath = logFilePath;

}

public void WriteLog(string action, string fileName, string filePath)

{

try

{

// Добавляем запись в конец файла xxxlogfile.txt

using (StreamWriter writer = File.AppendText(logFilePath))

{

writer.WriteLine($"{DateTime.Now}: {action} - {fileName} ({filePath})");

}

}

catch (Exception ex)

{

Console.WriteLine($"Ошибка записи лог-файла: {ex.Message}");

}

}

public string ReadLog()

{

try

{

// Считываем все содержимое файла xxxlogfile.txt

using (StreamReader reader = new StreamReader(logFilePath))

{

return reader.ReadToEnd();

}

}

catch (Exception ex)

{

Console.WriteLine($"Ошибка чтения лог-файла: {ex.Message}");

return string.Empty;

}

}

public string SearchLog(string searchString)

{

try

{

string result = string.Empty;

// Считываем построчно файл xxxlogfile.txt и ищем заданную строку

using (StreamReader reader = new StreamReader(logFilePath))

{

string line;

while ((line = reader.ReadLine()) != null)

{

if (line.Contains(searchString))

{

result += line + "\n";

}

}

}

return result;

}

catch (Exception ex)

{

Console.WriteLine($"Ошибка поиска в лог-файле: {ex.Message}");

return string.Empty;

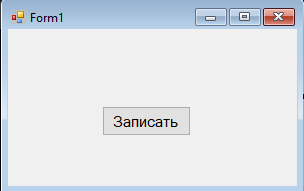
}

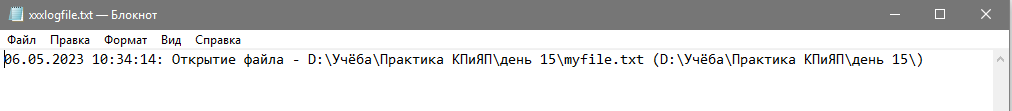
}

}

}

**Результат:**

****

****

**Задание 2**

Создать класс XXXDiskInfo c методами для вывода информации о

a. свободном месте на диске

b. Файловой системе

c. Для каждого существующего диска - имя, объем,

доступный объем, метка тома.

d. Продемонстрируйте работу класса

**Листинг программы:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace App2

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

// Вывод информации о свободном месте на диске C:

long freeSpace = XXXDiskInfo.GetFreeSpace("C:");

textBox1.Text = ("Free space on drive C: " + freeSpace + " bytes");

// Вывод файловой системы диска C:

string fileSystem = XXXDiskInfo.GetFileSystem("C:");

textBox2.Text = ("File system of drive C: " + fileSystem);

// Вывод информации о каждом диске

string allDrivesInfo = XXXDiskInfo.GetAllDrivesInfo();

textBox3.Text = (allDrivesInfo);

}

}

public class XXXDiskInfo

{

// Возвращает информацию о свободном месте на диске

public static long GetFreeSpace(string driveName)

{

DriveInfo drive = new DriveInfo(driveName);

return drive.AvailableFreeSpace;

}

// Возвращает файловую систему диска

public static string GetFileSystem(string driveName)

{

DriveInfo drive = new DriveInfo(driveName);

return drive.DriveFormat;

}

// Возвращает информацию о каждом диске, включая имя, объем, доступный объем и метку тома

public static string GetAllDrivesInfo()

{

string result = "";

DriveInfo[] drives = DriveInfo.GetDrives();

foreach (DriveInfo drive in drives)

{

if (drive.IsReady)

{

result += "Drive name: " + drive.Name + "\n" +

"Drive volume label: " + drive.VolumeLabel + "\n" +

"Drive file system: " + drive.DriveFormat + "\n" +

"Drive total size: " + drive.TotalSize + " bytes\n" +

"Drive available space: " + drive.AvailableFreeSpace + " bytes\n\n";

}

}

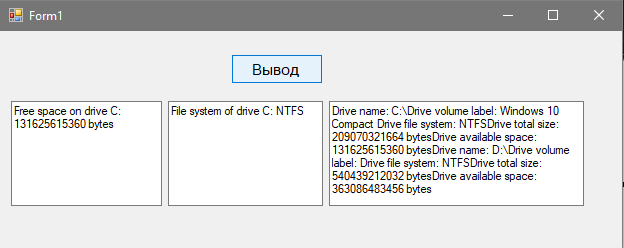
return result;

}

}

}

**Результат:**

****

**Задание 3**

Создать класс XXXFileInfo c методами для вывода информации о

конкретном файле

a. Полный путь

b. Размер, расширение, имя

c. Время создания

d. Продемонстрируйте работу класса

**Листинг программы:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace App3

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

string filePath = "D:\\Учёба\\Практика КПиЯП\\день 15\\example.txt";

// Получение информации о файле и вывод ее на экран

XXXFileInfo.GetFileInfo(filePath, pathLabel, sizeLabel, extensionLabel, createdLabel);

}

}

class XXXFileInfo

{

public static void GetFileInfo(string filePath, Label pathLabel, Label sizeLabel, Label extensionLabel, Label createdLabel)

{

try

{

FileInfo fileInfo = new FileInfo(filePath);

// Полный путь к файлу

pathLabel.Text = $"Полный путь: {fileInfo.FullName}";

// Размер файла

long fileSize = fileInfo.Length;

sizeLabel.Text = $"Размер файла: {fileSize} байт";

// Расширение и имя файла

string extension = fileInfo.Extension;

string fileName = fileInfo.Name;

extensionLabel.Text = $"Расширение файла: {extension}";

extensionLabel.Text = $"Имя файла: {fileName}";

// Время создания файла

DateTime createdTime = fileInfo.CreationTime;

createdLabel.Text = $"Дата создания: {createdTime}";

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}");

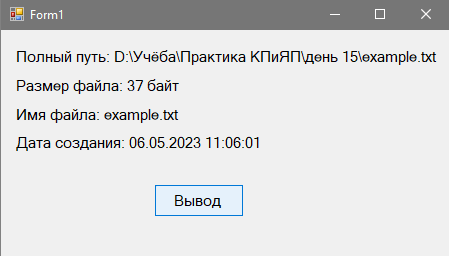
}

}

}

}

**Результат:**

****

**Задание 4**

Создать класс XXXDirInfo c методами для вывода информации о конкретном

директории

a. Количестве файлов

b. Время создания

c. Количестве поддиректориев

d. Список родительских директориев

e. Продемонстрируйте работу класса

**Листинг программы:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Задание\_4

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click\_1(object sender, EventArgs e)

{

var dirPath = "C:\\Users\\Admin\\Documents";

var dirInfo = new XXXDirInfo(dirPath);

textBox1.AppendText($"Files count: {dirInfo.GetFilesCount()}{Environment.NewLine}");

textBox1.AppendText($"Creation time: {dirInfo.GetCreationTime()}{Environment.NewLine}");

textBox1.AppendText($"Subdirectories count: {dirInfo.GetSubdirectoriesCount()}{Environment.NewLine}");

var parents = dirInfo.GetParentDirectories();

textBox1.AppendText("Parent directories:");

foreach (var parent in parents)

{

textBox1.AppendText($" {parent}");

}

}

}

class XXXDirInfo

{

private readonly string \_dirPath;

public XXXDirInfo(string dirPath)

{

\_dirPath = dirPath;

}

public int GetFilesCount()

{

return Directory.GetFiles(\_dirPath).Length;

}

public DateTime GetCreationTime()

{

return Directory.GetCreationTime(\_dirPath);

}

public int GetSubdirectoriesCount()

{

return Directory.GetDirectories(\_dirPath).Length;

}

public string[] GetParentDirectories()

{

var parents = new List<string>();

var parent = Directory.GetParent(\_dirPath);

while (parent != null)

{

parents.Add(parent.FullName);

parent = parent.Parent;

}

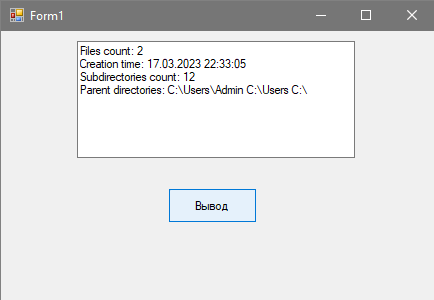
return parents.ToArray();

}

}

}

**Результат:**

****

**Задание 5**

Создать класс XXXFileManager. Набор методов определите

самостоятельно. С его помощью выполнить следующие действия:

a. Прочитать список файлов и папок заданного диска.

Создать директорий XXXInspect, создать текстовый файл

xxxdirinfo.txt и сохранить туда информацию. Создать

копию файла и переименовать его. Удалить

первоначальный файл.

b. Создать еще один директорий XXXFiles. Скопировать в

него все файлы с заданным расширением из заданного

пользователем директория. Переместить XXXFiles в

XXXInspect.

c. Сделайте архив из файлов директория XXXFiles.

Разархивируйте его в другой директорий.

**Листинг программы:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.IO.Compression;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace WinFormsApp1

{

public partial class Form1 : Form

{

private XXXFileManager fileManager;

public Form1()

{

InitializeComponent();

fileManager = new XXXFileManager("D:\\", "D:\\Test");

}

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

{

}

private void btnCreateDirectory\_Click\_1(object sender, EventArgs e)

{

fileManager.CreateDirectoryAndFile();

textBox1.AppendText("Директория XXXInspect и файл xxxdirinfo.txt созданы." + Environment.NewLine);

}

private void btnGetFiles\_Click\_1(object sender, EventArgs e)

{

string[] filesAndFolders = fileManager.GetFilesAndFolders();

foreach (string fileOrFolder in filesAndFolders)

{

textBox1.AppendText(fileOrFolder + Environment.NewLine);

}

}

private void btnCopyFiles\_Click\_1(object sender, EventArgs e)

{

string sourceDirPath = txtSourceDir.Text;

string fileExtension = txtFileExtension.Text;

fileManager.CopyFilesToXXXFiles(sourceDirPath, fileExtension);

textBox1.AppendText($"Файлы с расширением {fileExtension} скопированы в директорию XXXFiles." + Environment.NewLine);

}

private void btnCreateArchive\_Click\_1(object sender, EventArgs e)

{

fileManager.CreateArchive();

textBox1.AppendText("Создан архив из файлов директории XXXFiles." + Environment.NewLine);

}

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

{

fileManager.ExtractArchive();

textBox1.AppendText("Архив разархивирован в директорию Test." + Environment.NewLine);

}

}

public class XXXFileManager

{

private readonly string diskName;

private readonly string xxxInspectPath;

private readonly string xxxFilesPath;

public XXXFileManager(string diskName, string xxxInspectPath)

{

this.diskName = diskName;

this.xxxInspectPath = Path.Combine(xxxInspectPath, "XXXInspect");

this.xxxFilesPath = Path.Combine(this.xxxInspectPath, "XXXFiles");

}

public string[] GetFilesAndFolders()

{

return Directory.GetFileSystemEntries(diskName);

}

public void CreateDirectoryAndFile()

{

Directory.CreateDirectory(xxxInspectPath);

using (StreamWriter sw = File.CreateText(Path.Combine(xxxInspectPath, "xxxdirinfo.txt")))

{

sw.WriteLine("Список файлов и папок на диске:");

string[] filesAndFolders = GetFilesAndFolders();

foreach (string fileOrFolder in filesAndFolders)

{

sw.WriteLine(fileOrFolder);

}

}

}

public void CopyFilesToXXXFiles(string sourceDirPath, string fileExtension)

{

Directory.CreateDirectory(xxxFilesPath);

string[] sourceFiles = Directory.GetFiles(sourceDirPath, $"\*.{fileExtension}");

foreach (string sourceFile in sourceFiles)

{

string fileName = Path.GetFileName(sourceFile);

string destFile = Path.Combine(xxxFilesPath, fileName);

File.Copy(sourceFile, destFile);

}

}

public void CreateArchive()

{

string archivePath = Path.Combine(xxxInspectPath, "XXXFiles.zip");

ZipFile.CreateFromDirectory(xxxFilesPath, archivePath);

File.Delete(Path.Combine(xxxFilesPath, "file1.txt"));

File.Delete(Path.Combine(xxxFilesPath, "file2.txt"));

Directory.Delete(xxxFilesPath);

}

public void ExtractArchive()

{

string archivePath = Path.Combine(xxxInspectPath, "XXXFiles.zip");

string extractPath = Path.Combine("D:\\", "Test", "XXXFilesExtracted");

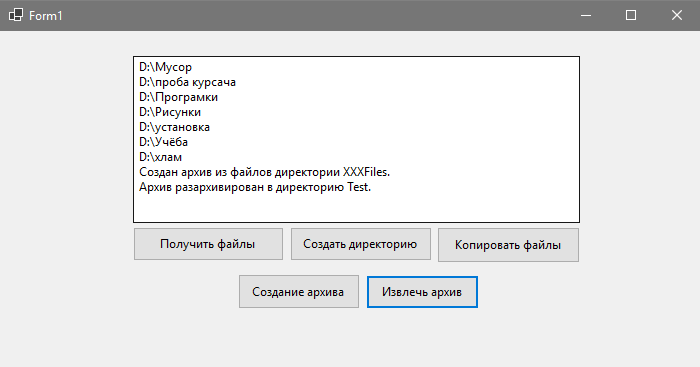
ZipFile.ExtractToDirectory(archivePath, extractPath);

}

}

}

**Результат:**

****

**Задание 6**

Найдите и выведите сохраненную информацию в файле xxxlogfile.txt о

действиях пользователя за определенный день/ диапазон времени/по

ключевому слову. Посчитайте количество записей в нем. Удалите часть

информации, оставьте только записи за текущий час.

**Листинг программы:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

using System.IO;

using System.Windows.Forms.VisualStyles;

namespace Задание\_6

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load\_1(object sender, EventArgs e)

{

}

private void button1\_Click\_1(object sender, EventArgs e)

{

string filePath = "xxxlogfile.txt";

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

int count = 0;

string line;

using (StreamReader reader = new StreamReader(filePath))

{

while (!reader.EndOfStream)

{

line = reader.ReadLine();

if (line.Contains(textBox1.Text) && line.Contains(textBox2.Text))

{

results.Add(line);

count++;

}

}

}

using (StreamWriter writer = new StreamWriter(filePath))

{

foreach (string result in results)

{

writer.WriteLine(result);

}

}

textBox3.Text = string.Join(Environment.NewLine, results);

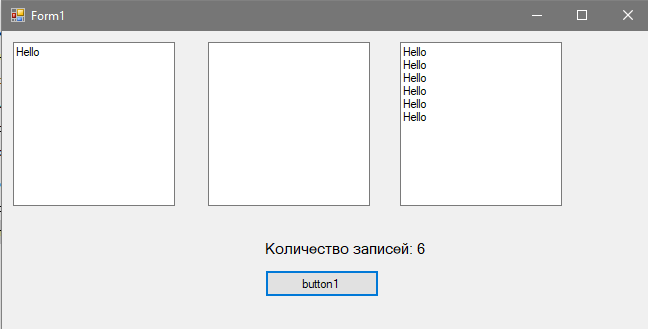
label1.Text = $"Количество записей: {count}";

}

}

}

**Результат:**

****

**Вывод:** В C# для работы с файлами используются классы File, Directory, Path, Stream и другие. Чтение и запись файлов можно выполнять с помощью потоков символов, таких как StreamReader и StreamWriter. Для работы с каталогами используется класс Directory, который позволяет получить информацию о содержимом каталога и создавать новые каталоги. Для сохранения объектов используются сериализация и десериализация с помощью классов BinaryFormatter, XmlSerializer и других.