**Лабораторная работа № 5**

**Работа с файлами**

**Цель работы:**

Приобретение навыков практического применения, закрепление знаний при создания простейших программ для работы с файлами.

**Задачи:**

1. Реализовать решение задания своего варианта;
2. Исходный текст считывать из текстового файла, результат работы программы записывать в новый текстовый файл;
3. Для работы со строками можно воспользоваться классом StringBuilder;
4. Должны быть обработаны ошибки, которые могут возникнуть при работе с файлами (см. класс IOException);
5. Выполнить требования к лабораторным работам (см. файл «Требования к лабораторным»);
6. Ознакомиться с литературой по теме лабораторной работы;
7. Ответить на контрольные вопросы.

*Дополнительно:*

1. Сделать возможность работы с файлами, папками. Например, создание/удаление папки, вывод информации на экран о файлах, создание текстового файла и запись в него введенных пользователем данных (использовать классы DirectoryInfo и FileInfo, Троелсен Э. - C# и платформа .NET. Библиотека программиста. Типы Directory(Info) и File(Info) 521- 529 стр.)

**Контрольные вопросы:**

1. Для чего используются классы StreamWriter и StreamReader?
2. Для чего используется класс IOException?
3. Для чего используется класс StringBuilder? В чем его отличия от класса String?

**Пояснения и примеры:**

**Класс IOException** является базовым классом для исключений, возникающих при доступе к данным с помощью потоков, файлов и каталогов.

Библиотека базовых классов включает следующие типы, каждый из которых является производным по отношению к классу IOException.

* [*DirectoryNotFoundException*](http://msdn.microsoft.com/ru-ru/library/system.io.directorynotfoundexception.aspx) - Исключение, возникающее при невозможности найти часть файла или каталога.
* [*FileNotFoundException*](http://msdn.microsoft.com/ru-ru/library/system.io.filenotfoundexception.aspx) - Это исключение создается, когда попытка доступа к файлу, не существующему на диске, заканчивается неудачей.
* [*PathTooLongException*](http://msdn.microsoft.com/ru-ru/library/system.io.pathtoolongexception.aspx) - Исключение, создаваемое, когда путь или имя файла превышает максимальную длину, определенную системой.
* [*DriveNotFoundException*](http://msdn.microsoft.com/ru-ru/library/system.io.drivenotfoundexception.aspx) - Исключение вызывается при попытке доступа к недоступному диску или данным совместного использования.

В следующем примере показан способ выполнения и восстановления из DirectoryNotFoundException.

using System;

using System.IO;

class Program

{

static void Main(string[] args)

{

try

{

// Specify a directory name that does not exist for this demo.

string dir = @"c:\78fe9lk";

// If this directory does not exist, a DirectoryNotFoundException is thrown

// when attempting to set the current directory.

Directory.SetCurrentDirectory(dir);

}

catch (DirectoryNotFoundException dirEx)

{

// Let the user know that the directory did not exist.

Console.WriteLine("Directory not found: " + dirEx.Message);

}

}

}

**Классы StreamWriter и StreamReader**

Классы StreamReader и StreamWriter пригодятся в тех ситуациях, когда необходимо считать или записать символьные данные (данные в формате string).

В следующем примере показано, как с помощью объекта StreamWriter выполнить запись файла, в котором перечисляются каталоги на диске C, а затем с помощью объекта[StreamReader](http://msdn.microsoft.com/ru-ru/library/system.io.streamreader.aspx) считать и отобразить имя каждого каталога.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

namespace StreamReadWrite

{

class Program

{

static void Main(string[] args)

{

// Get the directories currently on the C drive.

DirectoryInfo[] cDirs = new DirectoryInfo(@"c:\").GetDirectories();

// Write each directory name to a file.

using (StreamWriter sw = new StreamWriter("CDriveDirs.txt"))

{

foreach (DirectoryInfo dir in cDirs)

{

sw.WriteLine(dir.Name);

}

}

// Read and show each line from the file.

string line = "";

using (StreamReader sr = new StreamReader("CDriveDirs.txt"))

{

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

{

Console.WriteLine(line);

}

}

}

}

}

**StringBuilder**

Объект [String](http://msdn.microsoft.com/ru-ru/library/system.string.aspx) является неизменяемым. При каждом использовании методов класса [System.String](http://msdn.microsoft.com/ru-ru/library/system.string.aspx) в памяти создается новый объект строки, для которого необходимо выделение нового места. В ситуациях, когда необходимо выполнять многократные изменения строки, дополнительные издержки, связанные с созданием новых объектов [String](http://msdn.microsoft.com/ru-ru/library/system.string.aspx), могут быть весьма значительными. Класс [System.Text.StringBuilder](http://msdn.microsoft.com/ru-ru/library/system.text.stringbuilder.aspx) может использоваться при необходимости изменения строки без создания нового объекта.

В следующем примере показывается, как вызывать многие из методов, определенных классом StringBuilder.

using System;

using System.Text;

public sealed class App

{

static void Main()

{

// Create a StringBuilder that expects to hold 50 characters.

// Initialize the StringBuilder with "ABC".

StringBuilder sb = new StringBuilder("ABC", 50);

// Append three characters (D, E, and F) to the end of the StringBuilder.

sb.Append(new char[] { 'D', 'E', 'F' });

// Append a format string to the end of the StringBuilder.

sb.AppendFormat("GHI{0}{1}", 'J', 'k');

// Display the number of characters in the StringBuilder and its string.

Console.WriteLine("{0} chars: {1}", sb.Length, sb.ToString());

// Insert a string at the beginning of the StringBuilder.

sb.Insert(0, "Alphabet: ");

// Replace all lowercase k's with uppercase K's.

sb.Replace('k', 'K');

// Display the number of characters in the StringBuilder and its string.

Console.WriteLine("{0} chars: {1}", sb.Length, sb.ToString());

}

}

// This code produces the following output.

//

// 11 chars: ABCDEFGHIJk

// 21 chars: Alphabet: ABCDEFGHIJK

**Литература:**

1. Троелсен Э. - C# и платформа .NET. Библиотека программиста. Классы StreamWriter и StreamReader 535 – 541 стр.;
2. Павловская ПЯВУ C#. Потоки символов 255 – 259 стр.;
3. msdn – IOException;
4. Рихтер C# via .Net. Эффективное создание строки динамически 251 – 254 стр.