

РАЗДЕЛ 6. РАБОТА С ФАЙЛАМИ

C#: ФАЙЛОВЫЙ ВВОД/ВЫВОД

ПОНЯТИЕ ПОТОКА. ВИДЫ ПОТОКОВ

Файл (file) это коллекция данных, сохраненных на диске с определенным именем и по определенному пути на этом диске.

Когда файл открывается для чтения или записи, он становится потоком (stream).

Поток - это последовательность байт, проходящих через канал обмена данными.

Существует 2 основных потока: поток ввода (input stream) и поток вывода (output stream).

Поток ввода используется для чтения данных из файла (read operation) и поток вывода используется для записи данных в файл (write operation).

КЛАССЫ ВВОДА/ВЫВОДА C#

В Си-# в пространстве имен `System.IO`, реализованы все необходимые классы для работы с файлами (15 классов).

Чтобы подключить это пространство имен, необходимо в самом начале программы добавить строку `using System.IO`.

Для использования кодировок используется пространство `using System.Text`;

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.IO;
```

ОПЕРАЦИИ С ФАЙЛАМИ:

- создание файлов;
- удаление файлов;
- чтение данных;
- запись данных;
- изменение параметров файла (имя, расширение);

КЛАССЫ ВВОДА/ВЫВОДА C#

Имя класса	Описание
BinaryReader	Читает примитивные данные из двоичного потока.
BinaryWriter	Записывает примитивные данные в двоичном формате.
BufferedStream	Временное хранилище байтового потока данных.
Directory	Помогает работать со структурой каталогов файловой системы.
DirectoryInfo	Используется для выполнения операций над каталогами.
DriveInfo	Предоставляет информацию о приводах (дисках).
File	Помогает в манипуляциях с файлами.
FileInfo	Используется для выполнения операций с файлами.
FileStream	Используется для чтения и записи любого места в файле.
MemoryStream	Используется для произвольного доступа к данным потока, сохраненного в памяти.
Path	Операции над информацией пути файлов и каталогов.
StreamReader	Используется для чтения символов в байтовом потоке.
StreamWriter	Используется для записи символов в поток.
StringReader	Используется для чтения из буфера строки.
StringWriter	Используется для записи в буфер строки.

КЛАСС **STREAM** - БАЗОВЫЙ КЛАСС ДЛЯ ВСЕХ ПОТОКОВЫХ КЛАССОВ В C

класс **FileStream**

- представляет поток, который позволяет выполнять операции чтения/записи в файл

класс **StreamReader**

- предоставляет множество методов для удобного считывания данных

класс **StreamWriter**

- предоставляет методы записи данных

**Чтобы создать новый файл или открыть существующий файл,
нужно создать объект FileStream.**

Синтаксис создания объекта FileStream:

```
FileStream < object_name> = new FileStream( < имя файла>,  
    < FileMode Enumerator>,  
    < FileAccess Enumerator>,  
    < FileShare Enumerator>);
```

ПРИМЕР N°1:

```
FileStream F = new FileStream("sample.txt", FileMode.Open,  
FileAccess.Read, FileShare.Read);
```

НАЗНАЧЕНИЕ ПАРАМЕТРОВ КОНСТРУКТОРА КЛАССА FILESTREAM:

Параметр	Описание
	<p>Перечисление FileMode определяет различные методы, которые применяются для работы с файлами:</p> <p>Append - открывает существующий файл и перемещает позицию ввода (так называемый курсор файла) в конец файла, или создает новый файл, если указанный по имени файл не существует.</p> <p>Create - создает новый файл.</p> <p>CreateNew - указывает операционной системе, что она должна создать новый файл.</p> <p>Open - открывает существующий файл.</p> <p>OpenOrCreate - указывает операционной системе, что нужно открыть файл, если он существует, и если он не существует, то нужно создать новый файл.</p> <p>Truncate - откроет существующий файл и обрежет его, в результате файл станет пустым (его размер составит 0 байт).</p>
FileAccess	<p>Перечислитель, определяющий доступ к файлу: Read (только чтение), ReadWrite (чтение и запись) и Write (только запись).</p>
	<p>Перечисление, определяющее совместное использование файла:</p>
FileShare	<p>Inheritable - позволяет наследовать доступ к дескриптору файла для дочерних процессов.</p> <p>None - запрещает совместный доступ к текущему файлу.</p> <p>Read - позволяет другим потокам открыть файл на чтение.</p> <p>ReadWrite - позволяет совместную работу с файлом на чтение и запись.</p> <p>Write - позволяет другим потокам открыть файл на запись.</p>

ПРИМЕР №2: ОТКРЫВАЕТ ФАЙЛ ТОЛЬКО ДЛЯ ЧТЕНИЯ

```
static void Main(string[] args)
{
    FileStream file = new FileStream("d:\\test.txt", FileMode.Open,
    FileAccess.Read);
}
```



Режим доступа



Режим открытия

<i>Read</i>	<i>Open</i>
<i>ReadWrite</i>	<i>Append</i>
<i>Write</i>	<i>Create</i>

КЛАССЫ STREAMREADER И STREAMWRITER

Эти классы помогают в доступе к информации текстового файла на чтение и запись.

Они наследуются из абстрактного базового класса `Stream`, который поддерживает чтение и запись байт файлового потока.



МЕТОДЫ ЧТЕНИЯ ИЗ ФАЙЛА КЛАССА STREAMREADER

Метод `ReadToEnd()`

- считывает все данные из файла

Метод `ReadLine()`

- считывает одну строку

Свойство `EndOfStream`

- указывает, находится ли текущая позиция в потоке в конце потока (достигнут ли конец файла). Возвращает *true* или *false*

Класс `StreamReader` также наследуется из абстрактного базового класса `TextReader`, который представляет средство для последовательного чтения символов.

МЕТОДЫ ЗАПИСИ В ФАЙЛ КЛАССА STREAMWRITER

Close()

- *закрывает текущий объект StreamWriter и связанный с ним поток*

Write(string value)

- *записывает строку в поток*

WriteLine()

- *записывает терминатор строки в строку текста или поток*

Класс StreamWriter наследуется из абстрактного класса TextWriter, который предоставляет способ записи последовательности символов.

ДЛЯ РАБОТЫ С ФАЙЛАМИ ИСПОЛЬЗУЕТСЯ КЛАСС **FILE** (ПРОСТРАНСТВО ИМЕН: *SYSTEM.IO*), КОТОРЫЙ ИМЕЕТ СЛЕДУЮЩИЕ МЕТОДЫ:

Create()

- создает файл . Он принимает один аргумент – путь. Если файл с таким именем уже существует, он будет переписан на новый пустой файл.

WriteAllText()

- создает новый файл (если такого нет), либо открывает существующий и записывает текст, заменяя всё, что было в файле.

AppendAllText()

- новый текст дописывается в конец файла

Delete()

- удаляет файл по указанному пути

ПРИМЕР СОЗДАНИЯ ПУСТОГО ТЕКСТОВОГО
ФАЙЛА NEW_FILE.TXT НА ДИСКЕ D:

```
static void Main(string[] args)  
{  
    File.Create("D:\\new_file.txt");  
}
```

МАТЕРИАЛЫ

1. [HTTP://MICROSIN.NET/PROGRAMMING/PC/CSHARP-FILE-IO.HTML](http://MICROSIN.NET/PROGRAMMING/PC/CSHARP-FILE-IO.HTML) C#: ФАЙЛОВЫЙ ВВОД/ВЫВОД
2. [HTTP://MYCSHARP.RU/POST/21/2013_06_12_RABOTA_S_FAJLAMI_V_SI-SHARP_KLASSY_STREAMREADER_I_STREAMWRITER.HTML](http://MYCSHARP.RU/POST/21/2013_06_12_RABOTA_S_FAJLAMI_V_SI-SHARP_KLASSY_STREAMREADER_I_STREAMWRITER.HTML) РАБОТА С ФАЙЛАМИ В СИ-ШАРП. КЛАССЫ STREAMREADER И STREAMWRITER
3. [HTTPS://YOUTU.BE/HBYT7CTDJF0C#](https://YOUTU.BE/HBYT7CTDJF0C#). КЛАССЫ FILE И FILEINFO