**Работа с файловой системой**

Path.DirectorySeparatorChar – для работы с файловыми путями

Path.Combine – позволяет объединить\составить полное имя

var path = Path.Combine(Environment.CurrentDirectory, "asd", "123");

Console.WriteLine(path);

Path.DirectorySeparatorChar

Path.VolumeSeparatorChar

Path.AltDirectorySeparatorChar

Path.PathSeparator

Path.GetDirectoryName(path) – не пишет конечную папку

Path.GetRandomFileName()

Path.Exists(path) – проверяет наличие папки

Path.GetPathRoot(path) – показывает корневую папку (текущий диск)

Path.GetRelativePath(Environment.CurrentDirectory, path) – переход из одной папки в другую

System.IO.MemoryStream – чтобы картинку загружать там

System.IO.Binary - работа с двоичным кодом

DriveInfo - предоставлет информацию о дисках

DriveInfo drive = new DriveInfo(path);

var drives = DriveInfo.GetDrives(); - получение всех устройств

foreach (var drive in drives)

{ Console.WriteLine(drive.Name); - имя

Console.WriteLine(drive.VolumeLabel);

Console.WriteLine(drive.IsReady); - готов ли к использованию

Console.WriteLine(drive.DriveFormat);

Console.WriteLine(drive.DriveType); - тип

Console.WriteLine(drive.TotalSize); - различные размеры и пространства

Console.WriteLine(drive.AvailableFreeSpace);

Console.WriteLine(drive.TotalFreeSpace); - то что доступно в пределах }

**Работа с каталогами или папками**

Directory содержит набор статических методов для управления каталогом и получения в нем информации.

*Directory.ИмяМетода(path, параметры)*

*path* – имя папки

Примеры пути:

Var path = Directory.CurrentDirectory; – получение текущего пути

var path = Path.Combine(…);

var path = “C:\\Temp\\ispp21”; или var path = @“C:\Temp\ispp21”;

DirectoryInfo предоставляет набор методов к экземплярам для управления каталогом и свойств для получения информации о каталоге.

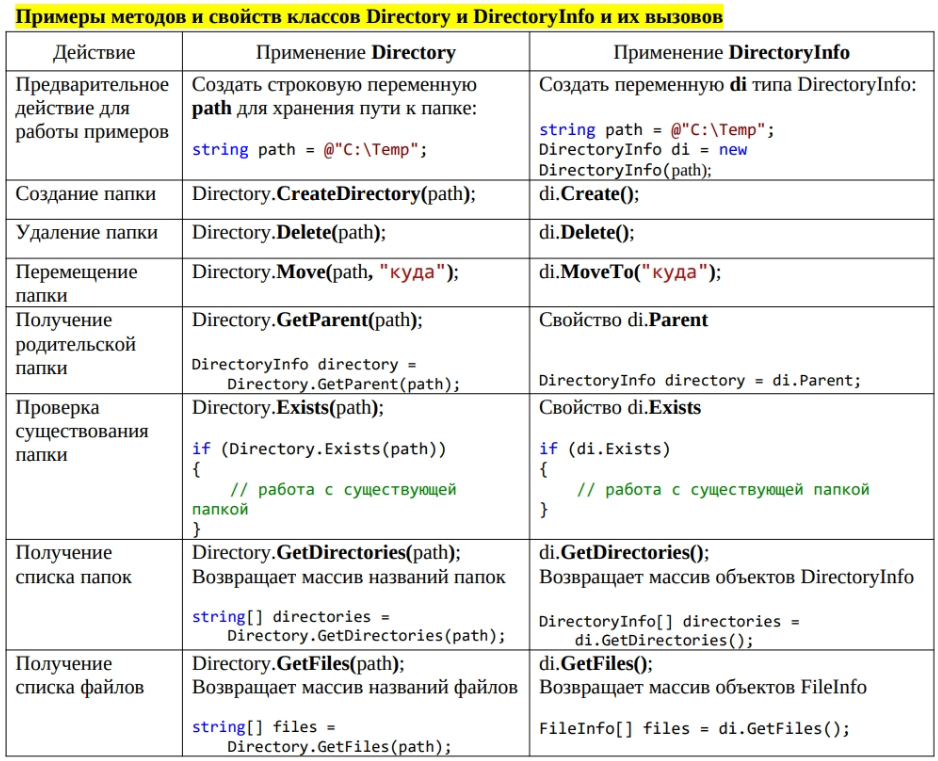
*DirectoryInfo объект = new(path);*

*объект.ИмяМетода(параметры)*

*объект.ИмяСвойства* // подробная информация

или

*DirectoryInfo объект = new(path, параметры);* // менее подробная



У метода GetDirectories и GetFiles есть дополнительные параметры:

1. searchPattern – это шаблон для поиска

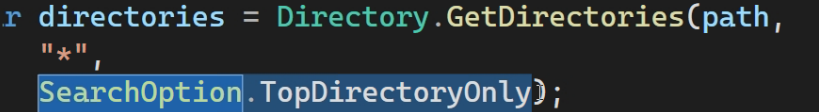
\* в шаблоне – это любой символ

“Lab\*”

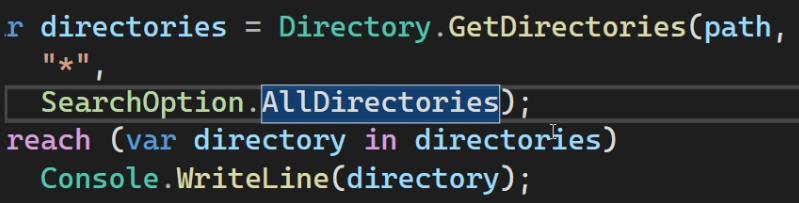
“\*.pdf”

1. SearchOption – это перечисление

По умолчанию:

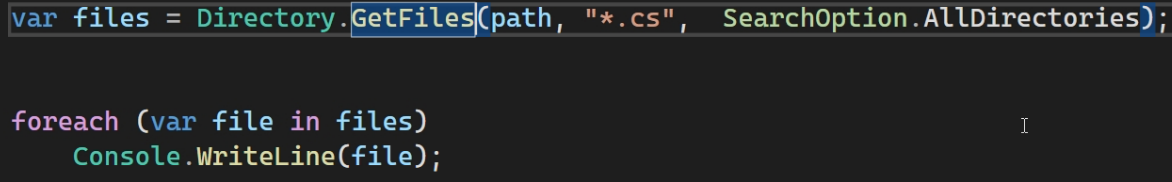


TopDirectoryOnly - поиск без учета вложенных папок.

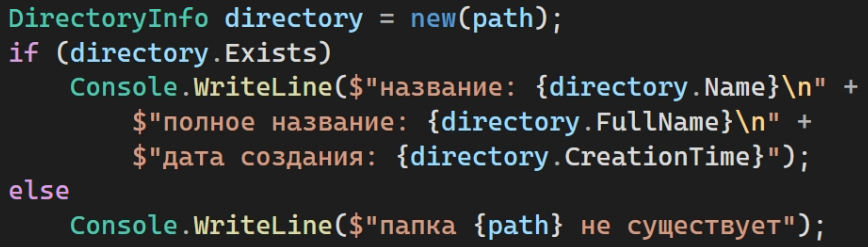


AllDirectories - поиск по папке и вложенным.

Пример для получения списка файлов:

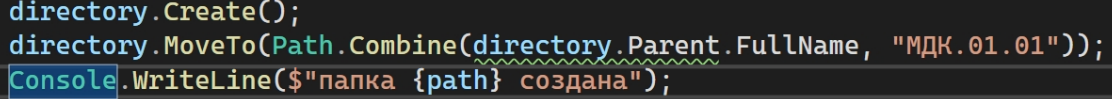


Пример получение информации о папке:



Exists для проверки существует ли файл.

Создание папки и перенос её:



**Работа с файлами**

**File** предоставляет набор методов для работы с файлами и получения информации. Работает аналогично Directory.

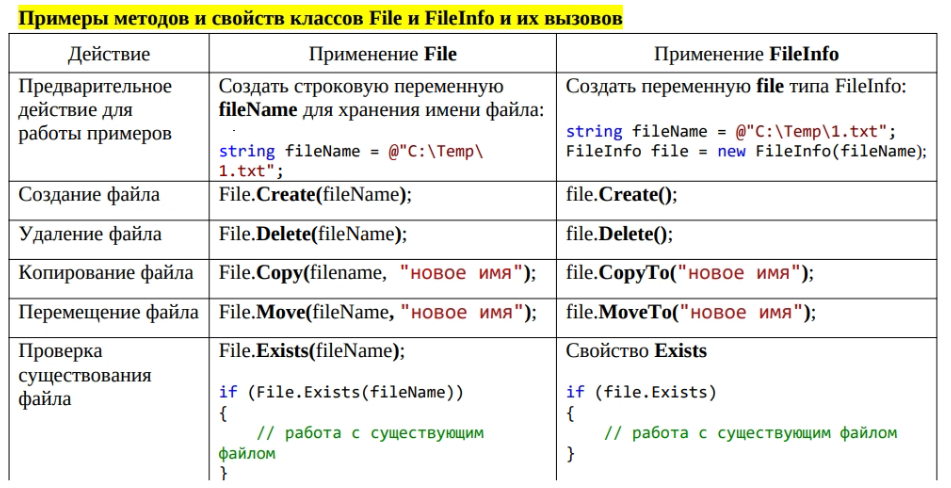
*File.ИмяМетода(filename, параметры)*

**FileInfo** предоставляет методы экземпляра для работы с файлами и свойства для полученимя информации о файле. Работает аналогично DirectoryInfo.

*FileInfo объект = new(filename, параметры);*

*объект.ИмяМетода(параметры)*

*объект.ИмяСвойства*



Свойства переменных FileInfo:

*Name* – имя без пути

*FullName* –полное имя папки

*Extension* – расширение файла

*DirectoryName* – полное имя папки

*Length* – размер файла байтах

Методы класса File для чтения и записи данных:

*File.ReadДанные(имя файла)* – чтение данных с файла

*File.WriteДанные(имя файла, записываемые данные);* - запись из приложения в файл. Если файл не существует, то он будет создан

*File.AppendДанные(имя файла, дозаписываемые данные)* – дозапись данный в конец файла.

Последний параметр методов – *кодировка.*

Encoding.ASCII

Виды данных:

* *AllBytes* – массив байтов.

Пример записи массива байтов в файл:

File.WriteAllBytes(filename, new byte[] {65, 66, 67}); // ABC

65,66,67 – коды символов.

* *AllLines* – массив строк.

Пример дозаписи массива строк в файл:

File.AppendAllLines(filename, [“hello”, “world”]};

* *AllText* – весь файл в виде одной строки.

Пример чтения всего файла:

string text = File.ReadAllText(filename);

Пример построчного чтения:

