«Работа с файловой системой»

РАБОТА С ДИСКАМИ

Для представления диска в пространстве **имен System.IO** имеется класс **DriveInfo**

РАБОТА С ДИСКАМИ

AvailableFreeSpace: указывает на объем доступного свободного места на диске в байтах

DriveFormat: получает имя файловой системы

DriveType: представляет тип диска

РАБОТА С ДИСКАМИ

Name: получает имя диска

RootDirectory: возвращает корневой каталог диска

TotalFreeSpace: получает общий объем свободного места на диске в байтах

TotalSize: общий размер диска в байтах

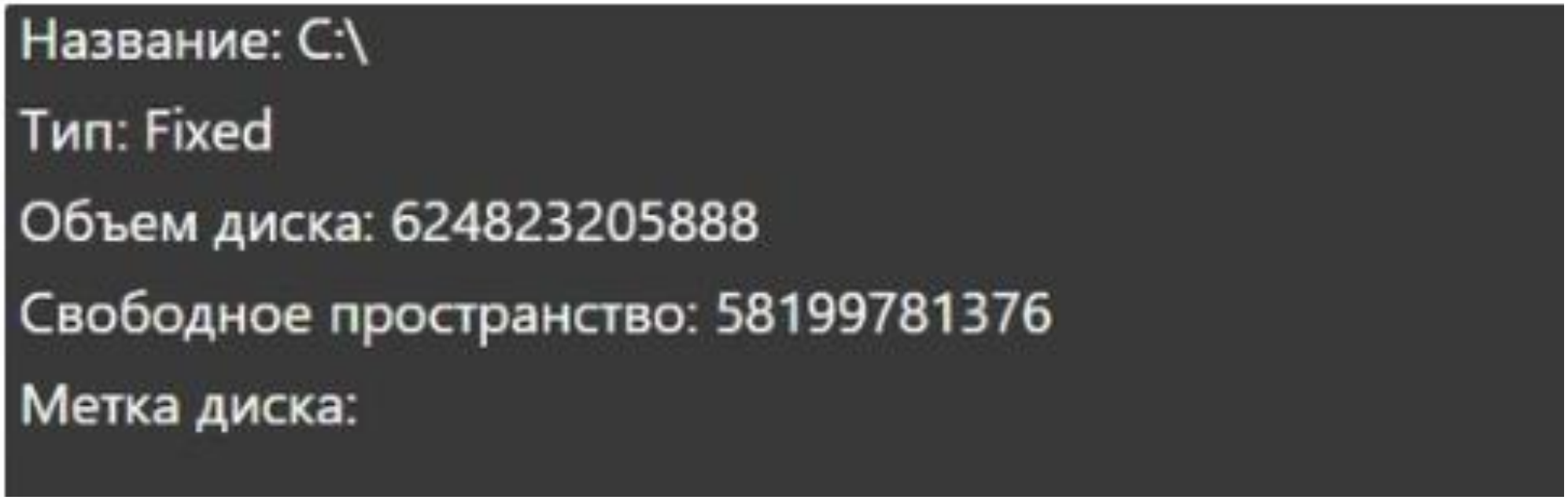
VolumeLabel: получает или устанавливает метку тома

РАБОТА С ДИСКАМИ

```
DriveInfo[] drives = DriveInfo.GetDrives();

foreach (DriveInfo drive in drives)
{
    Console.WriteLine($"Название: {drive.Name}");
    Console.WriteLine($"Тип: {drive.DriveType}");
    if (drive.IsReady)
    {
        Console.WriteLine($"Объем диска: {drive.TotalSize}");
        Console.WriteLine($"Свободное пространство: {drive.TotalFreeSpace}");
        Console.WriteLine($"Метка диска: {drive.VolumeLabel}");
    }
    Console.WriteLine();
}
```

РАБОТА С ДИСКАМИ



РАБОТА С КАТАЛОГАМИ

Для работы с каталогами в пространстве имен **System.IO** предназначены сразу два класса: **Directory** и **DirectoryInfo**.

РАБОТА С КАТАЛОГАМИ

CreateDirectory(path): создает каталог по указанному пути path

Delete(path): удаляет каталог по указанному пути path

Exists(path): определяет, существует ли каталог по указанному пути path. Если существует, возвращается true, если не существует, то false

GetCurrentDirectory(): получает путь к текущей папке

РАБОТА С КАТАЛОГАМИ

CreateDirectory(path): создает каталог по указанному пути path

Delete(path): удаляет каталог по указанному пути path

Exists(path): определяет, существует ли каталог по указанному пути path. Если существует, возвращается true, если не существует, то false

GetCurrentDirectory(): получает путь к текущей папке

РАБОТА С КАТАЛОГАМИ

GetFiles(path): получает список файлов в каталоге path

GetFileSystemEntries(path): получает список подкаталогов и файлов в каталоге path

Move(sourceDirName, destDirName): перемещает каталог

GetParent(path): получение родительского каталога

GetLastWriteTime(path): возвращает время последнего изменения каталога

GetLastAccessTime(path): возвращает время последнего обращения к каталогу

GetCreationTime(path): возвращает время создания каталога

ФИЛЬТРАЦИЯ ПАПОК И ФАЙЛОВ

Методы получения папок и файлов позволяют выполнять фильтрацию.
Фильтр может содержать два **плейсхолдера**: * и ?

ФИЛЬТРАЦИЯ ПАПОК И ФАЙЛОВ

```
// класс Directory
string[] dirs = Directory.GetDirectories(dirName, "books*.");

// класс DirectoryInfo
var directory = new DirectoryInfo(dirName);
DirectoryInfo[] dirs = directory.GetDirectories("books*.");
```

Или получим все файлы с расширением ".exe":

```
// класс Directory
string[] files = Directory.GetFiles(dirName, "*.exe");

// класс DirectoryInfo
var directory = new DirectoryInfo(dirName);
FileInfo[] files = directory.GetFiles("*.exe");
```

СОЗДАНИЕ КАТАЛОГА

```
string path = @"C:\SomeDir";
string subpath = @"program\avalon";
DirectoryInfo dirInfo = new DirectoryInfo(path);
if (!dirInfo.Exists)
{
    dirInfo.Create();
}
dirInfo.CreateSubdirectory(subpath);
```

ПОЛУЧЕНИЕ ИНФОРМАЦИИ О КАТАЛОГЕ

```
string dirName = "C:\\Program Files";

DirectoryInfo dirInfo = new DirectoryInfo(dirName);

Console.WriteLine($"Название каталога: {dirInfo.Name}");
Console.WriteLine($"Полное название каталога: {dirInfo.FullName}");
Console.WriteLine($"Время создания каталога: {dirInfo.CreationTime}");
Console.WriteLine($"Корневой каталог: {dirInfo.Root}");
```

УДАЛЕНИЕ КАТАЛОГА

```
string dirName = @"C:\SomeDir";

DirectoryInfo dirInfo = new DirectoryInfo(dirName);
if (dirInfo.Exists)
{
    dirInfo.Delete(true);
    Console.WriteLine("Каталог удален");
}
else
{
    Console.WriteLine("Каталог не существует");
}
```

ПЕРЕМЕЩЕНИЕ КАТАЛОГА

```
string oldPath = @"C:\SomeFolder";
string newPath = @"C:\SomeDir";
DirectoryInfo dirInfo = new DirectoryInfo(oldPath);
if (dirInfo.Exists && !Directory.Exists(newPath))
{
    dirInfo.MoveTo(newPath);
    // или так
    // Directory.Move(oldPath, newPath);
}
```

РАБОТА С ФАЙЛАМИ. КЛАССЫ FILE И FILEINFO

CopyTo(path): копирует файл в новое место по указанному пути path

Create(): создает файл

Delete(): удаляет файл

MoveTo(destFileName): перемещает файл в новое место

РАБОТА С ФАЙЛАМИ. КЛАССЫ FILE И FILEINFO

File

Класс **File** реализует похожую функциональность с помощью статических методов:

Copy(): копирует файл в новое место

Create(): создает файл

Delete(): удаляет файл

Move: перемещает файл в новое место

Exists(file): определяет, существует ли файл

ПУТИ К ФАЙЛАМ

```
// абсолютные пути
string path1 = @"C:\Users\alex\Documents\content.txt";  // для Windows
string path2 = "C:\\Users\\alex\\Documents\\content.txt";  // для Windows
string path3 = "/Users/alex/Documents/content.txt";  // для MacOS/Linux

// относительные пути
string path4 = "MyDir\\content.txt";  // для Windows
string path5 = "MyDir/content.txt";  // для MacOS/Linux
```

ПУТИ К ФАЙЛАМ

Получение информации о файле

```
string path = @"C:\Users\eugene\Documents\content.txt";
// string path = "/Users/eugene/Documents/content.txt";  // для MacOS/Linux
FileInfo fileInfo = new FileInfo(path);
if (fileInfo.Exists)
{
    Console.WriteLine($"Имя файла: {fileInfo.Name}");
    Console.WriteLine($"Время создания: {fileInfo.CreationTime}");
    Console.WriteLine($"Размер: {fileInfo.Length}");
}
```

ПУТИ К ФАЙЛАМ

Удаление файла

```
string path = @"C:\app\content.txt";
FileInfo fileInf = new FileInfo(path);
if (fileInf.Exists)
{
    fileInf.Delete();
    // альтернатива с помощью класса File
    // File.Delete(path);
}
```