Path.GetDirectoryName Method

Reference

Definition

Namespace: System.IO

Assembly: System.Runtime.dll

Overloads

GetDirectoryName(String)	Returns the directory information for the specified path.
GetDirectoryName(ReadOnlySpan<Char>)	Returns the directory information for the specified path represented by a character span.

GetDirectoryName(String)

Source: Path.cs ⬀

Returns the directory information for the specified path.

```C#
public static string? GetDirectoryName (string? path);
```

Parameters

`path`   String

The path of a file or directory.

Returns

String

Directory information for `path`, or `null` if `path` denotes a root directory or is null. Returns Empty if `path` does not contain directory information.

Exceptions

ArgumentException

.NET Framework and .NET Core versions older than 2.1: The `path` parameter contains invalid characters, is empty, or contains only white spaces.

PathTooLongException

The `path` parameter is longer than the system-defined maximum length.

Note: In .NET for Windows Store apps or the Portable Class Library, catch the base class exception, IOException, instead.

Examples

The following example demonstrates using the `GetDirectoryName` method on a Windows-based desktop platform.

C#

```csharp
string filePath = @"C:\MyDir\MySubDir\myfile.ext";
string directoryName;
int i = 0;

while (filePath != null)
{
    directoryName = Path.GetDirectoryName(filePath);
    Console.WriteLine("GetDirectoryName('{0}') returns '{1}'",
        filePath, directoryName);
    filePath = directoryName;
    if (i == 1)
    {
        filePath = directoryName + @"\";  // this will preserve
the previous path
    }
    i++;
}
/*
This code produces the following output:

GetDirectoryName('C:\MyDir\MySubDir\myfile.ext') returns
'C:\MyDir\MySubDir'
GetDirectoryName('C:\MyDir\MySubDir') returns 'C:\MyDir'
GetDirectoryName('C:\MyDir\') returns 'C:\MyDir'
GetDirectoryName('C:\MyDir') returns 'C:\'
GetDirectoryName('C:\') returns ''
*/
```

Remarks

In most cases, the string returned by this method consists of all characters in the path up to, but not including, the last directory separator character(s). A directory separator character can be either DirectorySeparatorChar or AltDirectorySeparatorChar. If the path consists of a root directory, such as "c:\", `null` is returned.

This method does not support paths using "file:".

Because the returned path does not include the last directory separator character(s), passing the returned path back into the GetDirectoryName method truncates one folder level per subsequent call on the result path. For example, passing the path "C:\Directory\SubDirectory\test.txt" into GetDirectoryName returns "C:\Directory\SubDirectory". Passing that path, "C:\Directory\SubDirectory", into GetDirectoryName returns "C:\Directory".

For a list of common I/O tasks, see Common I/O tasks.

See also

- File path formats on Windows systems
- File and Stream I/O
- How to: Read Text from a File
- How to: Write Text to a File

Applies to

▼ .NET 9 and other versions

Product	Versions
.NET	Core 1.0, Core 1.1, Core 2.0, Core 2.1, Core 2.2, Core 3.0, Core 3.1, 5, 6, 7, 8, 9
.NET Framework	1.1, 2.0, 3.0, 3.5, 4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1
.NET Standard	1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 2.0, 2.1
UWP	10.0

GetDirectoryName(ReadOnlySpan<Char>)

Source: Path.cs

Returns the directory information for the specified path represented by a character span.

```csharp
public static ReadOnlySpan<char> GetDirectoryName
(ReadOnlySpan<char> path);
```

Parameters

path   ReadOnlySpan<Char>

The path to retrieve the directory information from.

Returns

ReadOnlySpan<Char>

Directory information for `path`, or an empty span if `path` is `null`, an empty span, or a root (such as \, C:, or \server\share).

Remarks

Unlike the string overload, this method doesn't normalize directory separators.

See also

- File path formats on Windows systems

Applies to

▼ .NET 9 and other versions

Product	Versions
.NET	Core 2.1, Core 2.2, Core 3.0, Core 3.1, 5, 6, 7, 8, 9
.NET Standard	2.1

 Collaborate with us on GitHub

.NET   .NET feedback