Path.GetPathRoot Method

Reference

Definition

Namespace: System.IO

Assembly: System.Runtime.dll

Overloads

Expand table

GetPathRoot(String)	Gets the root directory information from the path contained in the specified string.
GetPathRoot(ReadOnlySpan<Char>)	Gets the root directory information from the path contained in the specified character span.

GetPathRoot(String)

Source: Path.Unix.cs

Gets the root directory information from the path contained in the specified string.

C#

```
public static string? GetPathRoot (string? path);
```

Parameters

path String

A string containing the path from which to obtain root directory information.

Returns

String

The root directory of `path` if it is rooted.

-or-

Empty if `path` does not contain root directory information.

-or-

`null` if `path` is `null` or is effectively empty.

Exceptions

ArgumentException

.NET Framework and .NET Core versions older than 2.1: `path` contains one or more of the invalid characters defined in GetInvalidPathChars().

-or-

.NET Framework only: Empty was passed to `path`.

Examples

The following example demonstrates a use of the `GetPathRoot` method.

C#

```
string path = @"\mydir\";
string fileName = "myfile.ext";
string fullPath = @"C:\mydir\myfile.ext";
string pathRoot;

pathRoot = Path.GetPathRoot(path);
Console.WriteLine("GetPathRoot('{0}') returns '{1}'",
    path, pathRoot);

pathRoot = Path.GetPathRoot(fileName);
Console.WriteLine("GetPathRoot('{0}') returns '{1}'",
    fileName, pathRoot);

pathRoot = Path.GetPathRoot(fullPath);
Console.WriteLine("GetPathRoot('{0}') returns '{1}'",
    fullPath, pathRoot);

// This code produces output similar to the following:
//
// GetPathRoot('\mydir\') returns '\'
// GetPathRoot('myfile.ext') returns ''
// GetPathRoot('C:\mydir\myfile.ext') returns 'C:\'
```

Remarks

This method does not verify that the path or file exists.

This method will normalize directory separators.

A string is "effectively empty" if:

- In Windows, calling `IsEmpty` on this string returns `true`, or all its characters are spaces (' ').
- In Unix, calling IsNullOrEmpty on this string returns `true`.

Possible patterns for the string returned by this method are as follows:

- `null` (`path` was null or an empty string).
- An empty string (`path` specified a relative path on the current drive or volume).
- "/" (Unix: `path` specified an absolute path on the current drive).
- "X:" (Windows: `path` specified a relative path on a drive, where *X* represents a drive or volume letter).
- "X:\" (Windows: `path` specified an absolute path on a given drive).
- "\ComputerName\SharedFolder" (Windows: a UNC path).
- "\?\C:" (Windows: a DOS device path, supported in .NET Core 1.1 and later versions, and in .NET Framework 4.6.2 and later versions).

For more information on file paths on Windows, see File path formats on Windows systems. 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

GetPathRoot(ReadOnlySpan<Char>)

Source: Path.Unix.cs

Gets the root directory information from the path contained in the specified character span.

C#

```
public static ReadOnlySpan<char> GetPathRoot (ReadOnlySpan<char>
path);
```

Parameters

`path` ReadOnlySpan<Char>

A read-only span of characters containing the path from which to obtain root directory information.

Returns

ReadOnlySpan<Char>

A read-only span of characters containing the root directory of `path`.

Remarks

This method does not verify that the path or file exists.

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

A `ReadOnlySpan<System.Char>` is "effectively empty" if:

- In Windows, calling ReadOnlySpan<T>.IsEmpty on this span of characters returns `true`, or all its characters are spaces (' ').
- In Unix, calling ReadOnlySpan<T>.IsEmpty on this span of characters returns `true`.

Possible patterns for the read-only character span returned by this method are as follows:

- ReadOnlySpan<T>.Empty (`path` was ReadOnlySpan<T>.Empty.
- ReadOnlySpan<T>.Empty (`path` specified a relative path on the current drive or volume).
- "/" (Unix: `path` specified an absolute path on the current drive).
- "X:" (Windows: `path` specified a relative path on a drive, where *X* represents a drive or volume letter).
- "X:\" (Windows: `path` specified an absolute path on a given drive).
- "\ComputerName\SharedFolder" (Windows: a UNC path).
- "\?\C:" (Windows: a DOS device path, supported in .NET Core 1.1 and later versions, and in .NET Framework 4.6.2 and later versions).

For more information on file paths on Windows, see File path formats on Windows systems. For a list of common I/O tasks, see Common I/O Tasks.

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

The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see our contributor guide.

.NET feedback

.NET is an open source project. Select a link to provide feedback:

Open a documentation issue

Provide product feedback