Join-Path (Microsoft.PowerShell.Management) - PowerShell

sdwheeler ⋮ 9-11 minutes

Join-Path

- Reference

In this article

Combines a path and a child path into a single path.

Syntax

```
Join-Path
    [-Path] <String[]>
    [-ChildPath] <String>
    [[-AdditionalChildPath] <String[]>]
    [-Resolve]
    [-Credential <PSCredential>]
    [<CommonParameters>]
```

Description

The `Join-Path` cmdlet combines a path and child-path into a single path. The provider supplies the path delimiters.

Examples

Example 1: Combine a path with a child path

```
PS C:\> Join-Path -Path "path" -ChildPath "childpath"

path\childpath
```

This command uses `Join-Path` to combine a path with a childpath.

Since the command is executed from the `FileSystem` provider, it provides the `\` delimiter to join the paths.

Example 2: Combine paths that already contain directory separators

```
PS C:\> Join-Path -Path "path\" -ChildPath "\childpath"

path\childpath
```

Existing directory separators `\` are handled so there is only one separator between `Path` and `ChildPath`

Example 3: Display files and folders by joining a path with a child path

```
Join-Path "C:\win*" "System*" -Resolve
```

This command displays the files and folders that are referenced by joining the `C:\Win\*` path and the `System\*` child path. It displays the same files and folders as `Get-ChildItem`, but it displays the fully qualified path to each item. In this command, the `Path` and `ChildPath` optional parameter names are omitted.

Example 4: Use Join-Path with the PowerShell registry provider

```
PS HKLM:\> Join-Path -Path System -ChildPath *ControlSet* -Resolve

HKLM:\System\ControlSet001
HKLM:\System\CurrentControlSet
```

This command displays the registry keys in the `HKLM\System` registry subkey that include `ControlSet`.

The `Resolve` parameter, attempts to resolve the joined path, including wildcards from the current provider path `HKLM:\`

Example 5: Combine multiple path roots with a child path

```
Join-Path -Path C:, D:, E:, F: -ChildPath New

C:\New
D:\New
E:\New
F:\New
```

This command uses `Join-Path` to combine multiple path roots with a child path.

Note

The Drives specified by `Path` must exist or the join of that entry will fail.

Example 6: Combine the roots of a file system drive with a child path

```
Get-PSDrive -PSProvider filesystem | ForEach-Object {$_.root} | Join-
Path -ChildPath "Subdir"

C:\Subdir
D:\Subdir
```

This command combines the roots of each PowerShell file system drive in the console with the `Subdir` child path.

The command uses the `Get-PSDrive` cmdlet to get the PowerShell drives supported by the FileSystem provider. The `ForEach-Object` statement selects only the **Root** property of the **PSDriveInfo** objects and combines it with the specified child path.

The output shows that the PowerShell drives on the computer included a drive mapped to the `C:\Program Files` directory.

Example 7: Combine an indefinite number of paths

```
Join-Path a b c d e f g

a\b\c\d\e\f\g
```

The `AdditionalChildPath` parameter allows the joining of an unlimited number of paths.

In this example, no parameter names are used, thus "a" binds to `Path`, "b" to `ChildPath` and "c-g" to `AdditionalChildPath`

Parameters

-AdditionalChildPath

Specifies additional elements to append to the value of the *Path* parameter. The `ChildPath` parameter is still mandatory and must be specified as well.

This parameter is specified with the `ValueFromRemainingArguments` property which enables joining an indefinite number of paths.

This parameter was added in PowerShell 6.0.

Type:	String[]
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True
Accept wildcard characters:	False

-ChildPath

Specifies the elements to append to the value of the `Path` parameter. Wildcards are permitted. The `ChildPath` parameter is required, although the parameter name ("ChildPath") is optional.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True
Accept wildcard characters:	True

-Credential

Note

This parameter is not supported by any providers installed with PowerShell. To impersonate another user, or elevate your credentials when running this cmdlet, use Invoke-Command.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True
Accept wildcard characters:	False

-Path

Specifies the main path (or paths) to which the child-path is appended. Wildcards are permitted.

The value of `Path` determines which provider joins the paths and adds the path delimiters. The `Path` parameter is required, although the parameter name ("Path") is optional.

Type:	String[]
Aliases:	PSPath
Position:	0
Default value:	None
Required:	True
Accept pipeline input:	True
Accept wildcard characters:	True

-Resolve

Indicates that this cmdlet should attempt to resolve the joined path from the current provider.

- If wildcards are used, the cmdlet returns all paths that match the joined path.
- If **no** wildcards are used, the cmdlet will error if the path does not exist.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

Inputs

String

You can pipe a string that contains a path to this cmdlet.

Outputs

String

This cmdlet returns a string that contains the resulting path.

Notes

The cmdlets that contain the Path noun (the Path cmdlets) manipulate path names and return the names in a concise format that all PowerShell providers can interpret. They are designed for use in programs and scripts where you want to display all or part of a path name in a particular format. Use them like you would use `Dirname`, `Normpath`, `Realpath`, `Join`, or other path manipulators.

You can use the path cmdlets with several providers, including the `FileSystem`, `Registry`, and `Certificate` providers.

This cmdlet is designed to work with the data exposed by any provider. To list the providers available in your session, type `Get-PSProvider`. For more information, see about_Providers.

- Convert-Path
- Resolve-Path
- Split-Path
- Test-Path
- Get-PSProvider
- Get-ChildItem
- Get-PSDrive

In this article

Previous Chapter Next Chapter