Full credit is given to the above companies including the OS that this PDF file was generated!

PowerShell Get-Help on command 'Remove-Item'

PS C:\Users\wahid> Get-Help Remove-Item

NAME

Remove-Item

SYNOPSIS

Deletes the specified items.

SYNTAX

Remove-Item [-Credential <System.Management.Automation.PSCredential>]

[-DeleteKey] [-Exclude <System.String[]>] [-Filter <System.String>] [-Force]

[-Include <System.String[]>] -LiteralPath <System.String[]> [-Recurse]

[-Stream <System.String[]>] [-UseTransaction] [-Confirm] [-WhatIf]

[<CommonParameters>]

Remove-Item [-Path] <System.String[]> [-Credential

<System.Management.Automation.PSCredential>] [-DeleteKey] [-Exclude

<System.String[]>] [-Filter <System.String>] [-Force] [-Include

<System.String[]>] [-Recurse] [-Stream <System.String[]>] [-UseTransaction]

[-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Remove-Item` cmdlet deletes one or more items. Because it's supported by many providers, it can delete many different types of items, including files, folders, registry keys, variables, aliases, and functions.

PARAMETERS

-Credential <System.Management.Automation.PSCredential>

> [!NOTE] > This parameter isn't supported by any providers installed with PowerShell. > To impersonate another user, or elevate your credentials when running this cmdlet, > use Invoke-Command (../Microsoft.PowerShell.Core/Invoke-Command.md).

-DeleteKey <System.Management.Automation.SwitchParameter>

This is a dynamic parameter made available by the Certificate provider. The Certificate provider and this parameter are only available on Windows platforms.

When provided, the cmdlet deletes the private key when the certificate is deleted.

For more information, see about_Certificate_Provider (../Microsoft.PowerShell.Security/About/about_Certificate_Provider.md).

-Exclude <System.String[]>

Specifies, as a string array, an item or items that this cmdlet excludes in the operation. The value of this parameter qualifies the Path parameter. Enter a path element or pattern, such as ` .txt`. Wildcard characters are permitted. The Exclude * parameter is effective only when the command includes the contents of an item, such as `C:\Windows*`, where the wildcard character specifies the contents of the `C:\Windows` directory.

When using Recurse with Exclude , Exclude only filters results of the current directory. If there are files that match the Exclude pattern in subfolders, those files are removed along with its parent directory.

-Filter <System.String>

Specifies a filter to qualify the Path parameter. The FileSystem (../Microsoft.PowerShell.Core/About/about_FileSystem_Provider.md)provider is the only installed PowerShell provider that supports the use of filters. You can find the syntax for the FileSystem filter language in about_Wildcards (../Microsoft.PowerShell.Core/About/about_Wildcards.md). Filters are more efficient than other parameters, because the provider applies them when the cmdlet gets the objects rather than having PowerShell filter the objects after they're retrieved.

-Force <System.Management.Automation.SwitchParameter>

Forces the cmdlet to remove items that can't otherwise be changed, such as hidden or read-only files or read-only aliases or variables. The cmdlet can't remove constant aliases or variables. Implementation varies from provider to provider. For more information, see about_Providers (../Microsoft.PowerShell.Core/About/about_Providers.md). Even using the Force parameter, the cmdlet can't override security restrictions.

-Include <System.String[]>

Specifies, as a string array, an item or items that this cmdlet includes in the operation. The value of this parameter qualifies the Path parameter. Enter a path element or pattern, such as `" .txt"`. Wildcard characters are permitted. The Include * parameter is effective only when the command includes the contents of an item, such as `C:\Windows*`, where the wildcard character specifies the contents of the `C:\Windows` directory.

-LiteralPath <System.String[]>

Specifies a path to one or more locations. The value of LiteralPath is

used exactly as it's typed. No characters are interpreted as wildcards. If the path includes escape characters, enclose it in single quotation marks. Single quotation marks tell PowerShell not to interpret any characters as escape sequences.

For more information, see about_Quoting_Rules (../Microsoft.Powershell.Core/About/about_Quoting_Rules.md).

-Path <System.String[]>

Specifies a path of the items being removed. Wildcard characters are permitted.

-Recurse <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet deletes the items in the specified locations and in all child items of the locations.

The Recurse parameter might not delete all subfolders or all child items. This is a known issue.

> [!NOTE] > This behavior was fixed in Windows versions 1909 and newer.

-Stream <System.String[]>

This is a dynamic parameter made available by the FileSystem provider. This parameter is only available on Windows. This parameter can't be used in combination with the Recurse parameter.

You can use `Remove-Item` to delete an alternative data stream, such as `Zone.Identifier`. However, it isn't the recommended way to eliminate security checks that block files that are downloaded from the Internet. If you verify that a downloaded file is safe, use the `Unblock-File` cmdlet.

This parameter was introduced in Windows PowerShell 3.0.

For more information, see about_FileSystem_Provider
(../Microsoft.PowerShell.Core/About/about_FileSystem_Provider.md).

-UseTransaction <System.Management.Automation.SwitchParameter>
Includes the command in the active transaction. This parameter is valid
only when a transaction is in progress. For more information, see
about_Transactions
(../Microsoft.PowerShell.Core/About/about_Transactions.md)

-Confirm <System.Management.Automation.SwitchParameter>
Prompts you for confirmation before running the cmdlet. For more
information, see the following articles:

- about_Preference_Variables (../microsoft.powershell.core/about/about_pref
erence_variables.md#confirmpreference)-
about_Functions_CmdletBindingAttribute (../microsoft.powershell.core/about/
about_functions_cmdletbindingattribute.md?#confirmimpact)

-WhatIf <System.Management.Automation.SwitchParameter>
Shows what would happen if the cmdlet runs. The cmdlet isn't run.

<CommonParameters>
This cmdlet supports the common parameters: Verbose, Debug,
ErrorAction, ErrorVariable, WarningAction, WarningVariable,
OutBuffer, PipelineVariable, and OutVariable. For more information, see
about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).

----- Example 1: Delete files that have any file extension -----

Remove-Item C:\Test*.*

--------- Example 2: Delete document files in a folder ---------

```
Remove-Item * -Include *.doc -Exclude *1*
```

It uses the wildcard character (`*`) to specify the contents of the current folder. It uses the Include and Exclude parameters to specify the files to delete.

---------- Example 3: Delete hidden, read-only files ----------

```
Remove-Item -Path C:\Test\hidden-RO-file.txt -Force
```

It uses the Path parameter to specify the file. It uses the Force parameter to delete it. Without Force , you can't delete read-only or hidden files.

------ Example 4: Delete files in subfolders recursively ------

```
Get-ChildItem * -Include *.csv -Recurse | Remove-Item
```

In the `Get-ChildItem` command, Path has a value of (`*`), which represents the contents of the current folder. It uses Include to specify the CSV file type, and it uses Recurse to make the retrieval recursive. If you try to specify the file type in the path, such as `-Path *.csv`, the cmdlet interprets the subject of the search to be a file that has no child items, and Recurse fails.

> [!NOTE] > This behavior was fixed in Windows versions 1909 and up.

---------- Example 5: Delete subkeys recursively ----------

```
Remove-Item HKLM:\Software\MyCompany\OldApp -Recurse
```

------ Example 6: Deleting files with special characters ------

```
Get-ChildItem
```

```
Directory: C:\temp\Downloads


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---          6/1/2018  12:19 PM           1362 myFile.txt
-a---          6/1/2018  12:30 PM           1132 myFile[1].txt
-a---          6/1/2018  12:19 PM           1283 myFile[2].txt
-a---          6/1/2018  12:19 PM           1432 myFile[3].txt


Get-ChildItem | Where-Object Name -Like '*`[*'


Directory: C:\temp\Downloads


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---          6/1/2018  12:30 PM           1132 myFile[1].txt
-a---          6/1/2018  12:19 PM           1283 myFile[2].txt
-a---          6/1/2018  12:19 PM           1432 myFile[3].txt


Get-ChildItem | Where-Object Name -Like '*`[*' | ForEach-Object { Remove-Item
-LiteralPath $_.Name }
Get-ChildItem


Directory: C:\temp\Downloads


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---          6/1/2018  12:19 PM           1362 myFile.txt



---------- Example 7: Remove an alternate data stream ----------


Get-Item C:\Test\Copy-Script.ps1 -Stream Zone.Identifier
```

```
    FileName: \\C:\Test\Copy-Script.ps1

    Stream                 Length
    ------                 ------
    Zone.Identifier            26


    Remove-Item C:\Test\Copy-Script.ps1 -Stream Zone.Identifier
    Get-Item C:\Test\Copy-Script.ps1 -Stream Zone.Identifier

    Get-Item : Could not open alternate data stream 'Zone.Identifier' of file
    'C:\Test\Copy-Script.ps1'.
    At line:1 char:1
    + Get-Item 'C:\Test\Copy-Script.ps1' -Stream Zone.Identifier
    + ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        + CategoryInfo          : ObjectNotFound: (C:\Test\Copy-Script.ps1:String)
    [Get-Item], FileNotFoundException
        + FullyQualifiedErrorId :
    AlternateDataStreamNotFound,Microsoft.PowerShell.Commands.GetItemCommand

    The Stream parameter `Get-Item` gets the `Zone.Identifier` stream of the
    `Copy-Script.ps1` file. `Remove-Item` uses the Stream parameter to remove the
    `Zone.Identifier` stream of the file. Finally, the `Get-Item` cmdlet shows
    that the `Zone.Identifier` stream was deleted.
REMARKS
    To see the examples, type: "get-help Remove-Item -examples".
    For more information, type: "get-help Remove-Item -detailed".
    For technical information, type: "get-help Remove-Item -full".
    For online help, type: "get-help Remove-Item -online"
```