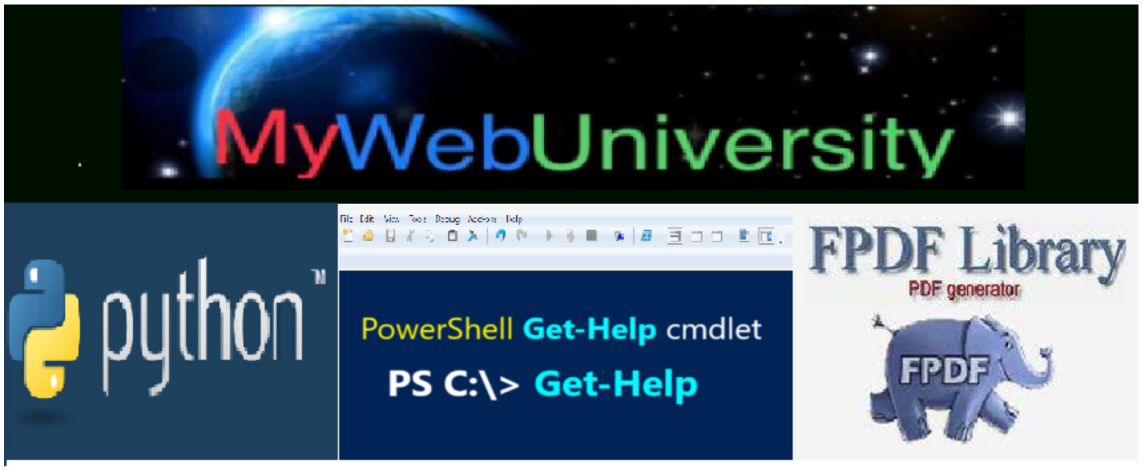

Full credit is given to all the above companies including the Operating System that this PDF file was generated!

Windows PowerShell Get-Help on Cmdlet 'Copy-Item'

PS:\>Get-HELP Copy-Item -Full

NAME

Copy-Item

SYNOPSIS

Copies an item from one location to another.

SYNTAX

Copy-Item [[-Destination] <System.String>] [-Container] [-Credential <System.Management.Automation.PSCredential>] [-Exclude <System.String[]>] [-Filter <System.String>] [-Force] [-FromSession <System.Management.Automation.Runspaces.PSSession>] [-Include <System.String[]>] -LiteralPath <System.String[]> [-PassThru] [-Recurse] [-ToSession <System.Management.Automation.Runspaces.PSSession>] [-UseTransaction] [-Confirm] [-WhatIf] [<CommonParameters>]

Copy-Item [-Path] <System.String[]> [[-Destination] <System.String>] [-Container] [-Credential <System.Management.Automation.PSCredential>] [-Exclude <System.String[]>] [-Filter <System.String>] [-Force] [-FromSession <System.Management.Automation.Runspaces.PSSession>] [-Include <System.String[]>] [-PassThru] [-Recurse] [-ToSession <System.Management.Automation.Runspaces.PSSession>] [-UseTransaction] [-Confirm]

[-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Copy-Item` cmdlet copies an item from one location to another location in the same namespace. For instance, it can copy a file to a folder, but it can't copy a
file to a certificate drive.

This cmdlet doesn't cut or delete the items being copied. The particular items that the cmdlet can copy depend on the PowerShell provider that exposes the item. For
instance, it can copy files and directories in a file system drive and registry keys and entries in the registry drive.

This cmdlet can copy and rename items in the same command. To rename an item, enter the new name in the value of the Destination parameter. To rename an item and not
copy it, use the `Rename-Item` cmdlet.

PARAMETERS

-Container <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet preserves container objects during the copy operation. By default, the Container parameter is set to True .

```
Required?                    false
Position?                    named
Default value                True
Accept pipeline input?       False
Accept wildcard characters?  false
```

-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).

Required?   false

Position?   named

Default value   Current user

Accept pipeline input?   True (ByPropertyName)

Accept wildcard characters? false

-Destination <System.String>

Specifies the path to the new location. The default is the current directory.

To rename the item being copied, specify a new name in the value of the Destination parameter.

Required?   false

Position?   1

Default value   Current directory

Accept pipeline input?   True (ByPropertyName)

Accept wildcard characters? false

-Exclude <System.String[]>

Specifies one or more path elements or patterns, such as `"*.txt"`, to limit this cmdlet's operation. The value of this parameter filters against the

wildcard-matching result of the Path parameter, not the final results. This parameter is only effective when the Path is specified with one or more wildcards.

Since this parameter only filters on the paths resolved for the Path parameter, it doesn't filter any items discovered when recursing through child folders with

the Recurse parameter.

Required?   false

Position?   named

Default value   None

Accept pipeline input?   False

Accept wildcard characters? true

-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.

Required?	false
Position?	named
Default value	None
Accept pipeline input?	False
Accept wildcard characters?	true

-Force <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet copies items that can't otherwise be changed, such as copying over a read-only file or alias.

Required?	false
Position?	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

-FromSession <System.Management.Automation.Runspaces.PSSession>

This is a dynamic parameter made available by the FileSystem provider.

Specify the PSSession object from which a remote file is being copied. When you use this parameter, the Path and LiteralPath parameters refer to the local path on

the remote machine.

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

```
        Required?                    false

        Position?                    named

        Default value                None

        Accept pipeline input?       False

        Accept wildcard characters?  false
```

-Include <System.String[]>

Specifies one or more path elements or patterns, such as `"*.txt"`, to limit this cmdlet's operation. The value of this parameter filters against the

wildcard-matching result of the Path parameter, not the final results. This parameter is only effective when the Path is specified with one or more wildcards.

Since this parameter only filters on the paths resolved for the Path parameter, it doesn't filter any items discovered when recursing through child folders with

the Recurse parameter.

```
        Required?                    false

        Position?                    named

        Default value                None

        Accept pipeline input?       False

        Accept wildcard characters?  true
```

-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).

```
        Required?                    true

        Position?                    named

        Default value                None

        Accept pipeline input?       True (ByPropertyName)
```

Accept wildcard characters? false

-PassThru <System.Management.Automation.SwitchParameter>

Returns an object that represents the item with which you're working. By default, this cmdlet doesn't generate any output.

Required?   false

Position?   named

Default value   False

Accept pipeline input?   False

Accept wildcard characters? false

-Path <System.String[]>

Specifies, as a string array, the path to the items to copy. Wildcard characters are permitted.

Required?   true

Position?   0

Default value   None

Accept pipeline input?   True (ByPropertyName, ByValue)

Accept wildcard characters? true

-Recurse <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet does a recursive copy.

Required?   false

Position?   named

Default value   False

Accept pipeline input?   False

Accept wildcard characters? false

-ToSession <System.Management.Automation.Runspaces.PSSession>

This is a dynamic parameter made available by the FileSystem provider.

Specify the PSSession object to which a remote file is being copied. When you use this parameter, the Destination parameter refers to the local path on the remote
machine.

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

```
Required?                    false
Position?                    named
Default value                None
Accept pipeline input?       False
Accept wildcard characters?  false
```

-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).

```
Required?                    false
Position?                    named
Default value                False
Accept pipeline input?       False
Accept wildcard characters?  false
```

-Confirm <System.Management.Automation.SwitchParameter>

Prompts you for confirmation before running the cmdlet.

```
Required?                    false
Position?                    named
Default value                False
Accept pipeline input?       False
Accept wildcard characters?  false
```

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


        Required?                    false
        Position?                    named
        Default value                False
        Accept pipeline input?       False
        Accept wildcard characters?  false


    <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).

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



OUTPUTS
    None
        By default, this cmdlet returns no output.


    System.Management.Automation.PSObject
        When you use the PassThru parameter, this cmdlet returns an object representing the copied item.



NOTES



        Windows PowerShell includes the following aliases for `Copy-Item`:
```

- `copy`

- `cp`

- `cpi`

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 (../Microsoft.PowerShell.Core/About/about_Providers.md).

------ Example 1: Copy a file to the specified directory ------

```
Copy-Item "C:\Wabash\Logfiles\mar1604.log.txt" -Destination "C:\Presentation"
```

- Example 2: Copy directory contents to an existing directory -

```
Copy-Item -Path "C:\Logfiles\*" -Destination "C:\Drawings" -Recurse
```

> [!NOTE] > If the path `C:\Drawings` doesn't exist the cmdlet copies all the files from the `Logfiles` > folder into a single file `C:\Drawings`.

-- Example 3: Copy directory and contents to a new directory --

```
Copy-Item -Path "C:\Logfiles" -Destination "C:\Drawings\Logs" -Recurse
```

> [!NOTE] > If the Path includes `*`, all the directory's file contents, including the subdirectory > trees, are copied to the new destination directory. For example:
> > `Copy-Item -Path "C:\Logfiles*" -Destination "C:\Drawings\Logs" -Recurse`

Example 4: Copy a file to the specified directory and rename the file

```
Copy-Item "\\Server01\Share\Get-Widget.ps1" -Destination "\\Server12\ScriptArchive\Get-Widget.ps1.txt"
```

--------- Example 5: Copy a file to a remote computer ---------

```
$Session = New-PSSession -ComputerName "Server01" -Credential "Contoso\User01"
Copy-Item "D:\Folder001\test.log" -Destination "C:\Folder001_Copy\" -ToSession $Session
```

-------- Example 6: Copy a folder to a remote computer --------

```
$Session = New-PSSession -ComputerName "Server02" -Credential "Contoso\User01"
Copy-Item "D:\Folder002\" -Destination "C:\Folder002_Copy\" -ToSession $Session
```

Example 7: Recursively copy the entire contents of a folder to a remote computer

```
$Session = New-PSSession -ComputerName "Server04" -Credential "Contoso\User01"
Copy-Item "D:\Folder003\" -Destination "C:\Folder003_Copy\" -ToSession $Session -Recurse
```

Example 8: Copy a file to a remote computer and then rename the file

```
$Session = New-PSSession -ComputerName "Server04" -Credential "Contoso\User01"
Copy-Item "D:\Folder004\scriptingexample.ps1" -Destination "C:\Folder004_Copy\scriptingexample_copy.ps1"
-ToSession $Session
```

----- Example 9: Copy a remote file to the local computer -----

```
$Session = New-PSSession -ComputerName "Server01" -Credential "Contoso\User01"
Copy-Item "C:\MyRemoteData\test.log" -Destination "D:\MyLocalData\" -FromSession $Session
```

Example 10: Copy the entire contents of a remote folder to the local computer

```
$Session = New-PSSession -ComputerName "Server01" -Credential "Contoso\User01"

Copy-Item "C:\MyRemoteData\scripts" -Destination "D:\MyLocalData\" -FromSession $Session
```

Example 11: Recursively copy the entire contents of a remote folder to the local computer

```
$Session = New-PSSession -ComputerName "Server01" -Credential "Contoso\User01"

Copy-Item "C:\MyRemoteData\scripts" -Destination "D:\MyLocalData\scripts" -FromSession $Session -Recurse
```

Example 12: Recursively copy files from a folder tree into the current folder

```
PS C:\temp\test> (Get-ChildItem C:\temp\tree -Recurse).FullName
C:\temp\tree\subfolder
C:\temp\tree\file1.txt
C:\temp\tree\file2.txt
C:\temp\tree\file3.txt
C:\temp\tree\subfolder\file3.txt
C:\temp\tree\subfolder\file4.txt
C:\temp\tree\subfolder\file5.txt

PS C:\temp\test> Get-Content C:\temp\tree\file3.txt
This is file3.txt in the root folder

PS C:\temp\test> Get-Content C:\temp\tree\subfolder\file3.txt
This is file3.txt in the subfolder

PS C:\temp\test> Copy-Item -Path C:\temp\tree -Filter *.txt -Recurse -Container:$false
PS C:\temp\test> (Get-ChildItem . -Recurse).FullName
C:\temp\test\subfolder
C:\temp\test\file1.txt
```

```
    C:\temp\test\file2.txt
    C:\temp\test\file3.txt
    C:\temp\test\file4.txt
    C:\temp\test\file5.txt

    PS C:\temp\test> Get-Content .\file3.txt
    This is file3.txt in the subfolder
```

The `Copy-Item` cmdlet has the Container parameter set to `$false`. This causes the contents of the source folder to be copied but doesn't preserve the folder

structure. Notice that files with the same name are overwritten in the destination folder.

-- Example 13: Using filters to copy items without recursion --

```
    PS D:\temp\test\out> Copy-Item -Path D:\temp\tree\* -Include ex*
    PS D:\temp\test\out> (Get-ChildItem -Recurse).FullName
    D:\temp\out\examples
    D:\temp\out\example.ps1
    D:\temp\out\example.txt
```

The Include parameter is applied to the contents of `D:\temp\tree` folder to copy all items that match `ex*`. Notice that, without recursion, the

`D:\temp\out\examples` folder is copied, but none of its contents are copied.

---- Example 14: Using filters to copy items with recursion ----

```
    D:\temp\out> Copy-Item -Path D:\temp\tree\* -Include ex* -Recurse
    D:\temp\out> (Get-ChildItem -Recurse).FullName
    D:\temp\out\examples
    D:\temp\out\example.ps1
    D:\temp\out\example.txt
    D:\temp\out\examples\subfolder
    D:\temp\out\examples\example_1.txt
    D:\temp\out\examples\example_2.txt
    D:\temp\out\examples\subfolder\test.txt
```

The Include parameter is applied to the contents of `D:\temp\tree` folder to copy all items that match `ex*`. Notice that, with recursion, the `D:\temp\out\examples`

folder is copied along with all the files and subfolders. The copy includes files that do not match the include filter. When using `Copy-Item`, the filters only apply

to the top-level specified by the Path parameter. Then recursion is applied to those matching items.

> [!NOTE] > The behavior of the Exclude parameter is the same as described in this example, except that > it limits the operation to only those paths that don't match

the pattern.

Example 15: Limit the files to recursively copy from a wildcard-specified path

```
D:\temp\out> Get-ChildItem -Path D:\temp\tree -Recurse -Filter ex* | Copy-Item
D:\temp\out> (Get-ChildItem -Recurse).FullName
D:\temp\out\examples
D:\temp\out\example_1.txt
D:\temp\out\example_2.txt
D:\temp\out\example.ps1
D:\temp\out\example.txt
```

Unlike the `Copy-Item`, the Filter parameter for `Get-ChildItem` applies to the items discovered during recursion. This enables you to find, filter, and then copy

items recursively.

RELATED LINKS

about_Providers

Clear-Item

Get-Item

Get-PSProvider

Invoke-Item

Move-Item

New-Item

Remove-Item

Rename-Item

Set-Item