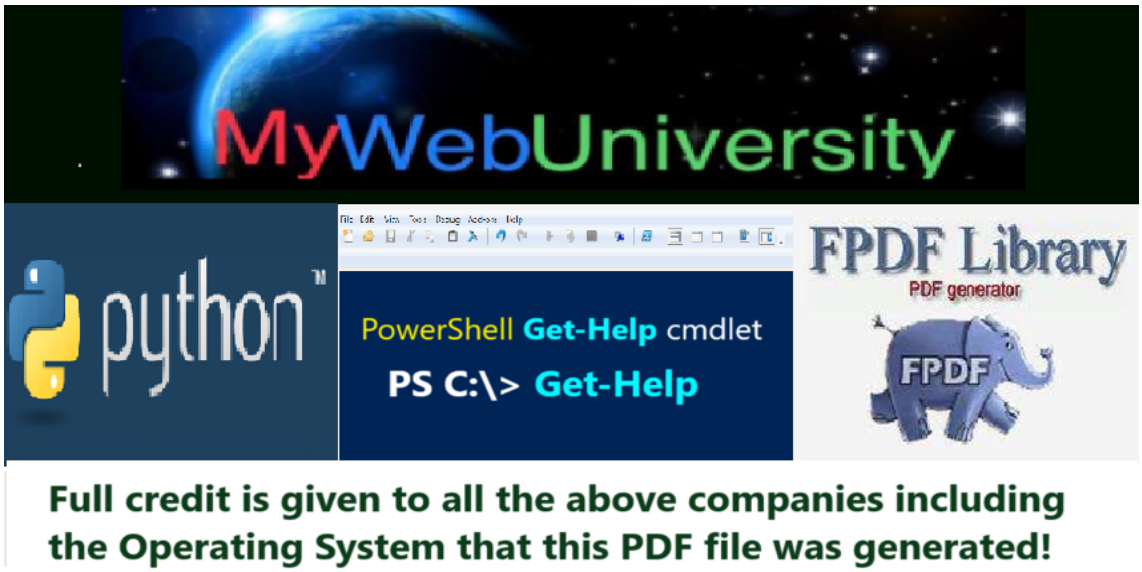

Windows PowerShell Get-Help on Cmdlet 'Get-Command'

PS:\>Get-HELP Get-Command -Full

NAME

Get-Command

SYNOPSIS

Gets all commands.

SYNTAX

Get-Command [[-Name] <System.String[]>] [[-ArgumentList] <System.Object[]>] [-All] [-CommandType {Alias | Function | Filter | Cmdlet | ExternalScript | Application |

Script | Workflow | Configuration | All}] [-FullyQualifiedModule <Microsoft.PowerShell.Commands.ModuleSpecification[]>] [-ListImported] [-Module <System.String[]>]

[-ParameterName <System.String[]>] [-ParameterType <System.Management.Automation.PSTypeName[]>] [-ShowCommandInfo] [-Syntax] [-TotalCount <System.Int32>]

[<CommonParameters>]

Get-Command [[-ArgumentList] <System.Object[]>] [-All] [-FullyQualifiedModule <Microsoft.PowerShell.Commands.ModuleSpecification[]>] [-ListImported] [-Module

<System.String[]>] [-Noun <System.String[]>] [-ParameterName <System.String[]>] [-ParameterType <System.Management.Automation.PSTypeName[]>] [-ShowCommandInfo]

[-Syntax] [-TotalCount <System.Int32>] [-Verb <System.String[]>] [<CommonParameters>]

DESCRIPTION

The `Get-Command` cmdlet gets all commands that are installed on the computer, including cmdlets, aliases, functions, filters, scripts, and applications.

`Get-Command` gets the commands from PowerShell modules and commands that were imported from other sessions. To get only commands that have been imported into the current session, use the ListImported parameter.

Without parameters, `Get-Command` gets all of the cmdlets, functions, and aliases installed on the computer. `Get-Command *` gets all types of commands, including all of the non-PowerShell files in the Path environment variable (`$env:Path`), which it lists in the Application command type.

`Get-Command` that uses the exact name of the command, without wildcard characters, automatically imports the module that contains the command so that you can use the command immediately. To enable, disable, and configure automatic importing of modules, use the `$PSModuleAutoLoadingPreference` preference variable. For more information, see about_Preference_Variables (About/about_Preference_Variables.md).

`Get-Command` gets its data directly from the command code, unlike `Get-Help`, which gets its information from help topics.

Starting in Windows PowerShell 5.0, results of the `Get-Command` cmdlet display a Version column by default. A new Version property has been added to the CommandInfo class.

PARAMETERS

-All <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet gets all commands, including commands of the same type that have the same name. By default, `Get-Command` gets only the commands that run when you type the command name.

For more information about the method that PowerShell uses to select the command to run when multiple commands have the same name, see about_Command_Precedence

(About/about_Command_Precedence.md). For information about module-qualified command names and running commands that do not run by default because of a name

conflict, see about_Modules (About/about_Modules.md).

This parameter was introduced in Windows PowerShell 3.0.

In Windows PowerShell 2.0, `Get-Command` gets all commands by default.

Required? false

Position? named

Default value False

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? false

-ArgumentList <System.Object[]>

Specifies an array of arguments. This cmdlet gets information about a cmdlet or function when it is used with the specified parameters ("arguments"). The alias

for ArgumentList is Args .

To detect dynamic parameters that are available only when certain other parameters are used, set the value of ArgumentList to the parameters that trigger the

dynamic parameters.

To detect the dynamic parameters that a provider adds to a cmdlet, set the value of the ArgumentList parameter to a path in the provider drive, such as WSMan:,

HKLM:, or Cert:. When the command is a PowerShell provider cmdlet, enter only one path in each command. The provider cmdlets return only the dynamic parameters

for the first path the value of ArgumentList . For information about the provider cmdlets, see about_Providers (About/about_Providers.md).

Required? false

Position? 1

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-CommandType <System.Management.Automation.CommandTypes>

Specifies the types of commands that this cmdlet gets. Enter one or more command types. Use CommandType or its alias, Type . By default, `Get-Command` gets all

cmdlets, functions, and aliases.

The acceptable values for this parameter are:

- `Alias`: Gets the aliases of all PowerShell commands. For more information, see about_Aliases (About/about_Aliases.md).

- `All`: Gets all command types. This parameter value is the equivalent of `Get-Command *`.

- `Application`: Gets non-PowerShell files in paths listed in the Path environment variable (`$env:path`), including `.txt`, `.exe`, and `.dll` files. For more

information about the Path environment variable, see about_Environment_Variables (About/about_Environment_Variables.md).

- `Cmdlet`: Gets all cmdlets.

- `ExternalScript`: Gets all `.ps1` files in the paths listed in the Path environment variable (`$env:path`).

- `Filter` and `Function`: Gets all PowerShell advanced and simple functions and filters.

- `Script`: Gets all script blocks. To get PowerShell scripts (`.ps1 `files), use the `ExternalScript` value.

- `Workflow`: Gets all workflows. For more information about workflows, see Introducing Windows PowerShell Workflow.

These values are defined as a flag-based enumeration. You can combine multiple values together to set multiple flags using this parameter. The values can be

passed to the CommandType parameter as an array of values or as a comma-separated string of those values. The cmdlet will combine the values using a binary-OR

operation. Passing values as an array is the simplest option and also allows you to use tab-completion on the values.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? false

-FullyQualifiedModule <Microsoft.PowerShell.Commands.ModuleSpecification[]>

The value can be a module name, a full module specification, or a path to a module file.

When the value is a path, the path can be fully qualified or relative. A relative path is resolved relative to the script that contains the using statement.

When the value is a name or module specification, PowerShell searches the PSModulePath for the specified module.

A module specification is a hashtable that has the following keys.

- `ModuleName` - Required Specifies the module name.

- `GUID` - Optional Specifies the GUID of the module.

- It's also Required to specify at least one of the three below keys.

- `ModuleVersion` - Specifies a minimum acceptable version of the module.

- `MaximumVersion` - Specifies the maximum acceptable version of the module.

- `RequiredVersion` - Specifies an exact, required version of the module. This can't be used with the other Version keys.

You cannot specify the FullyQualifiedModule parameter in the same command as a Module parameter. The two parameters are mutually exclusive.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? false

-ListImported <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet gets only commands in the current session.

Starting in PowerShell 3.0, by default, `Get-Command` gets all installed commands, including, but not limited to, the commands in the current session. In

PowerShell 2.0, it gets only commands in the current session.

This parameter was introduced in Windows PowerShell 3.0.

Required? false

Position? named

Default value False

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? false

-Module <System.String[]>

Specifies an array of modules. This cmdlet gets the commands that came from the specified modules or snap-ins. Enter the names of modules or snap-ins.

This parameter takes string values, but the value of this parameter can also be a PSModuleInfo or PSSnapinInfo object, such as the objects that the `Get-Module`,

`Get-PSSnapin`, and `Import-PSSession` cmdlets return.

You can refer to this parameter by its name, Module , or by its alias, PSSnapin . The parameter name that you choose has no effect on the command output.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? true

-Name <System.String[]>

Specifies an array of names. This cmdlet gets only commands that have the specified name. Enter a name or name pattern. Wildcard characters are permitted.

To get commands that have the same name, use the All parameter. When two commands have the same name, by default, `Get-Command` gets the command that runs when

you type the command name.

Required? false

Position? 0

Default value None

Accept pipeline input? True (ByPropertyName, ByValue)

Accept wildcard characters? true

-Noun <System.String[]>

Specifies an array of command nouns. This cmdlet gets commands, which include cmdlets, functions, and aliases, that have names that include the specified noun.

Enter one or more nouns or noun patterns. Wildcard characters are permitted.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? true

-ParameterName <System.String[]>

Specifies an array of parameter names. This cmdlet gets commands in the session that have the specified parameters. Enter parameter names or parameter aliases.

Wildcard characters are supported.

The ParameterName and ParameterType parameters search only commands in the current session.

This parameter was introduced in Windows PowerShell 3.0.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? true

-ParameterType <System.Management.Automation.PSTypeName[]>

Specifies an array of parameter names. This cmdlet gets commands in the session that have parameters of the specified type. Enter the full name or partial name of

a parameter type. Wildcard characters are supported.

The ParameterName and ParameterType parameters search only commands in the current session.

This parameter was introduced in Windows PowerShell 3.0.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? true

-ShowCommandInfo <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet displays command information.

This parameter was introduced in Windows PowerShell 5.0.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-Syntax <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet gets only the following specified data about the command:

- Aliases Gets the standard name.

- Cmdlets. Gets the syntax.

- Functions and filters. Gets the function definition.

- Scripts and applications or files. Gets the path and filename.

Required? false

Position? named

Default value False

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? false

-TotalCount <System.Int32>

Specifies the number of commands to get. You can use this parameter to limit the output of a command.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? false

-Verb <System.String[]>

Specifies an array of command verbs. This cmdlet gets commands, which include cmdlets, functions, and aliases, that have names that include the specified verb.

Enter one or more verbs or verb patterns. Wildcard characters are permitted.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? true

<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 command names to this cmdlet.

OUTPUTS

System.Management.Automation.CommandInfo

This cmdlet returns objects derived from the CommandInfo class. The type of object that is returned depends on the type of command that `Get-Command` gets.

System.Management.Automation.AliasInfo

Represents aliases.

System.Management.Automation.ApplicationInfo

Represents applications and files.

System.Management.Automation.CmdletInfo

Represents cmdlets.

System.Management.Automation.FunctionInfo

Represents functions and filters.

System.Management.Automation.WorkflowInfo

Represents workflows.

NOTES

Windows PowerShell includes the following aliases for `Get-Command`:

- `gcm`

- When more than one command that has the same name is available to the session, `Get-Command` returns the command that runs when you type the command name. To get commands that have the same name, listed in run order, use the All parameter. For more information, see about_Command_Precedence (../Microsoft.PowerShell.Core/About/about_Command_Precedence.md). - When a module is imported automatically, the effect is the same as using the `Import-Module` cmdlet. The module can add commands, types and formatting files, and run scripts in the session. To enable, disable, and configuration automatic importing of modules, use the `$PSModuleAutoLoadingPreference` preference variable. For more information, see about_Preference_Variables (../Microsoft.PowerShell.Core/About/about_Preference_Variables.md).

-------- Example 1: Get cmdlets, functions, and aliases --------

Get-Command

-------- Example 2: Get commands in the current session --------

Get-Command -ListImported

------- Example 3: Get cmdlets and display them in order -------

Get-Command -Type Cmdlet | Sort-Object -Property Noun | Format-Table -GroupBy Noun

---------- Example 4: Get commands in a module ----------

Get-Command -Module Microsoft.PowerShell.Security, Microsoft.PowerShell.Utility

---------- Example 5: Get information about a cmdlet ----------

Get-Command Get-AppLockerPolicy

When a module is imported automatically, the effect is the same as using the Import-Module cmdlet. The module can add commands, types and formatting files, and run

scripts in the session. To enable, disable, and configuration automatic importing of modules, use the `$PSModuleAutoLoadingPreference` preference variable. For more

information, see about_Preference_Variables (../Microsoft.PowerShell.Core/About/about_Preference_Variables.md).

---------- Example 6: Get the syntax of a cmdlet ----------

Get-Command -Name Get-Childitem -Args Cert: -Syntax

When you compare the syntax displayed in the output with the syntax that is displayed when you omit the Args (ArgumentList) parameter, you'll see that the

Certificate provider adds a dynamic parameter, CodeSigningCert , to the `Get-ChildItem` cmdlet.

For more information about the Certificate provider, see about_Certificate_Provider (../Microsoft.PowerShell.Security/About/about_Certificate_Provider.md).

---------- Example 7: Get dynamic parameters ----------

```
function Get-DynamicParameters
{
    param ($Cmdlet, $PSDrive)
    (Get-Command -Name $Cmdlet -ArgumentList $PSDrive).ParameterSets |
      ForEach-Object {$_.Parameters} |
        Where-Object { $_.IsDynamic } |
          Select-Object -Property Name -Unique
}
Get-DynamicParameters -Cmdlet Get-ChildItem -PSDrive Cert:

Name
----
CodeSigningCert
```

The `Get-DynamicParameters` function in this example gets the dynamic parameters of a cmdlet. This is an alternative to the method used in the previous example.

Dynamic parameter can be added to a cmdlet by another cmdlet or a provider.

---------- Example 8: Get all commands of all types ----------

```
Get-Command *
```

It returns an ApplicationInfo object (System.Management.Automation.ApplicationInfo) for each file, not a FileInfo object (System.IO.FileInfo).

-- Example 9: Get cmdlets by using a parameter name and type --

Get-Command -ParameterName *Auth* -ParameterType AuthenticationMechanism

You can use a command like this one to find cmdlets that let you specify the method that is used to authenticate the user.

The ParameterType parameter distinguishes parameters that take an AuthenticationMechanism value from those that take an AuthenticationLevel parameter, even when they

have similar names.

---------- Example 10: Get an alias ----------

Get-Command -Name dir

CommandType Name ModuleName

---------- ---- ----------

Alias dir -> Get-ChildItem

Although it is typically used on cmdlets and functions, `Get-Command` also gets scripts, functions, aliases, and executable files.

The output of the command shows the special view of the Name property value for aliases. The view shows the alias and the full command name.

----- Example 11: Get all instances of the Notepad command -----

Get-Command Notepad -All | Format-Table CommandType, Name, Definition

CommandType Name Definition

---------- ---- ----------

Application notepad.exe C:\WINDOWS\system32\notepad.exe

Application NOTEPAD.EXE C:\WINDOWS\NOTEPAD.EXE

The All parameter is useful when there is more than one command with the same name in the session.

Beginning in Windows PowerShell 3.0, by default, when the session includes multiple commands with the same name, `Get-Command` gets only the command that runs when

you type the command name. With the All parameter, `Get-Command` gets all commands with the specified name and returns them in execution precedence order. To run a

command other than the first one in the list, type the fully qualified path to the command.

For more information about command precedence, see about_Command_Precedence (About/about_Command_Precedence.md).

- Example 12: Get the name of a module that contains a cmdlet -

```
(Get-Command Get-Date).ModuleName
```

```
Microsoft.PowerShell.Utility
```

This command format works on commands in PowerShell modules, even if they are not imported into the session.

Example 13: Get cmdlets and functions that have an output type

```
Get-Command -Type Cmdlet | Where-Object OutputType | Format-List -Property Name, OutputType
```

This command gets the cmdlets and functions that have an output type and the type of objects that they return.

The first part of the command gets all cmdlets. A pipeline operator (`|`) sends the cmdlets to the `Where-Object` cmdlet, which selects only the ones in which the

OutputType property is populated. Another pipeline operator sends the selected cmdlet objects to the `Format-List` cmdlet, which displays the name and output type of

each cmdlet in a list.

The OutputType property of a CommandInfo object has a non-null value only when the cmdlet code defines the OutputType attribute for the cmdlet.

Example 14: Get cmdlets that take a specific object type as input

```
Get-Command -ParameterType (((Get-NetAdapter)[0]).PSTypeNames)
```

```
CommandType     Name                                ModuleName
-----------     ----                                ----------
```

Function	Disable-NetAdapter	NetAdapter
Function	Enable-NetAdapter	NetAdapter
Function	Rename-NetAdapter	NetAdapter
Function	Restart-NetAdapter	NetAdapter
Function	Set-NetAdapter	NetAdapter

This command finds cmdlets that take net adapter objects as input. You can use this command format to find the cmdlets that accept the type of objects that any

command returns.

The command uses the PSTypeNames intrinsic property of all objects, which gets the types that describe the object. To get the PSTypeNames property of a net adapter,

and not the PSTypeNames property of a collection of net adapters, the command uses array notation to get the first net adapter that the cmdlet returns. To get the

PSTypeNames property of a net adapter, and not the PSTypeNames property of a collection of net adapters, the command uses array notation to get the first net adapter

that the cmdlet returns.

RELATED LINKS

Online Version: https://learn.microsoft.com/powershell/module/microsoft.powershell.core/get-command?view=powershell-5.1&WT.mc_id=ps-gethelp

Export-PSSession

Get-Help

Get-Member

Get-PSDrive

Import-PSSession

about_Command_Precedence