Windows PowerShell Get-Help on Cmdlet 'Write-Output'

PS:\>Get-HELP Write-Output -Full

NAME

Write-Output

SYNOPSIS

Writes the specified objects to the pipeline.

SYNTAX

Write-Output [-InputObject] <System.Management.Automation.PSObject[]> [-NoEnumerate] [<CommonParameters>]

DESCRIPTION

Writes the specified objects to the pipeline. If `Write-Output` is the last command in the pipeline, the objects are displayed in the console.

`Write-Output` sends objects to the primary pipeline, also known as the success stream . To send error objects to the error stream, use `Write-Error`.

This cmdlet is typically used in scripts to display strings and other objects on the console. One of the built-in aliases for `Write-Output` is `echo` and similar to

other shells that use `echo`. The default behavior is to display the output at the end of a pipeline. In PowerShell, it is generally not necessary to use the cmdlet

in instances where the output is displayed by default. For example, `Get-Process | Write-Output` is equivalent to `Get-Process`. Or, `echo "Home directory: $HOME"`

can be written, `"Home directory: $HOME"`.

By default, `Write-Output` enumerates objects in a collection. However, `Write-Output` can also pass collections down the pipeline as a single object with the

NoEnumerate parameter.

PARAMETERS

-InputObject <System.Management.Automation.PSObject[]>

Specifies the objects to send down the pipeline. Enter a variable that contains the objects, or type a command or expression that gets the objects.

Required? true

Position? 0

Default value None

Accept pipeline input? True (ByValue)

Accept wildcard characters? false

-NoEnumerate <System.Management.Automation.SwitchParameter>

By default, the `Write-Output` cmdlet always enumerates its output. The NoEnumerate parameter suppresses the default behavior, and prevents `Write-Output` from

enumerating output. The NoEnumerate parameter has no effect if the command is wrapped in parentheses, because the parentheses force enumeration. For example,

`(Write-Output 1,2,3)` still enumerates the array.

The NoEnumerate parameter is only useful within a pipeline. Trying to see the effects of NoEnumerate in the console is problematic because PowerShell adds

`Out-Default` to the end of every command line, which results in enumeration. But if you pipe `Write-Output -NoEnumerate` to another cmdlet, the downstream cmdlet

receives the collection object, not the enumerated items of the collection.

> [!IMPORTANT] > There is an issue with this switch in Windows PowerShell that is fixed in PowerShell 6.2 and > above. When using NoEnumerate and explicitly using

the InputObject parameter, the command > still enumerates. To work around this, pass the InputObject argument(s) positionally.

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.Management.Automation.PSObject

You can pipe objects to this cmdlet.

OUTPUTS

System.Management.Automation.PSObject

This cmdlet returns the objects that are submitted as input.

NOTES

Windows PowerShell includes the following aliases for `Write-Output`:

- `echo`

- `write`

----- Example 1: Get objects and write them to the console -----

```
$P = Get-Process
Write-Output $P
```

---------- Example 2: Pass output to another cmdlet ----------

```
Write-Output "test output" | Get-Member
```

---------- Example 3: Suppress enumeration in output ----------

```
Write-Output 1,2,3 | Measure-Object

Count    : 3
...

Write-Output 1,2,3 -NoEnumerate | Measure-Object

Count    : 1
...
```

RELATED LINKS

gethelp

about_Output_Streams

about_Redirection

Tee-Object

Write-Debug

Write-Error

Write-Host

Write-Information

Write-Progress

Write-Verbose

Write-Warning