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

PowerShell Get-Help on command 'Write-Information'

PS C:\Users\wahid> Get-Help Write-Information

NAME

Write-Information

SYNOPSIS

Specifies how PowerShell handles information stream data for a command.

SYNTAX

Write-Information [-MessageData] <System.Object> [[-Tags] <System.String[]>]

[<CommonParameters>]

DESCRIPTION

The `Write-Information` cmdlet specifies how PowerShell handles information stream data for a command.

Windows PowerShell 5.0 introduces a new, structured information stream. You can use this stream to transmit structured data between a script and its callers or the host application. `Write-Information` lets you add an informational message to the stream, and specify how PowerShell handles information stream data for a command. Information streams also work for

`PowerShell.Streams`, jobs, and scheduled tasks.

> [!NOTE] > The information stream does not follow the standard convention of prefixing its messages with > "[Stream Name]:". This was intended for brevity and visual cleanliness.

The `$InformationPreference` preference variable value determines whether the message you provide to `Write-Information` is displayed at the expected point in a script's operation. Because the default value of this variable is SilentlyContinue , by default, informational messages are not shown. If you don't want to change the value of `$InformationPreference`, you can override its value by adding the InformationAction common parameter to your command. For more information, see about_Preference_Variables (../Microsoft.PowerShell.Core/About/about_Preference_Variables.md)and about_CommonParameters (../Microsoft.PowerShell.Core/About/about_CommonParameters.md).

> [!NOTE] > Starting in Windows PowerShell 5.0, `Write-Host` is a wrapper for `Write-Information` This allows > you to use `Write-Host` to emit output to the information stream. This enables the capture or > suppression of data written using `Write-Host` while preserving backwards compatibility. For more > information see Write-Host (Write-Host.md)`Write-Information` is also a supported workflow activity in Windows PowerShell 5.1.

PARAMETERS

-MessageData <System.Object>

Specifies an informational message that you want to display to users as they run a script or command. For best results, enclose the informational message in quotation marks.

-Tags <System.String[]>

Specifies a simple string that you can use to sort and filter messages

that you have added to the information stream with `Write-Information`.
This parameter works similarly to the Tags parameter in
`New-ModuleManifest`.

<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: Write information for Get- results --------

```
Write-Information -MessageData "Processes starting with 'P'"
-InformationAction Continue
Get-Process -Name p*
```

Processes starting with 'P'

```
    18    19.76      15.16       0.00    6232   0 PFERemediation
    20     8.92      25.15       0.00   24944   0 policyHost
     9     1.77       7.64       0.00    1780   0 powercfg
    10    26.67      32.18       0.00    7028   0 powercfg
     8    26.55      31.59       0.00   13600   0 powercfg
     9     1.66       7.55       0.00   22620   0 powercfg
    21     6.17       4.54     202.20   12536   1 PowerMgr
    42    84.26      12.71   2,488.84   20588   1 powershell
    27    47.07      45.38       2.05   25988   1 powershell
    27    24.45       5.31       0.00   12364   0 PresentationFontCache
    92   112.04      13.36      82.30   13176   1 pwsh
   106   163.73      93.21     302.25   14620   1 pwsh
   227   764.01      92.16   1,757.22   25328   1 pwsh
```

---------- Example 2: Write information and tag it ----------

```
$message = "To filter your results for PowerShell, pipe your results to the
Where-Object cmdlet."
Get-Process -Name p*
Write-Information -MessageData $message -Tags "Instructions"
-InformationAction Continue
```

```
NPM(K)    PM(M)      WS(M)     CPU(s)      Id  SI ProcessName
 ------    -----      -----     ------      --  -- -----------
     18    19.76      15.16       0.00    6232   0 PFERemediation
     20     8.92      25.15       0.00   24944   0 policyHost
      9     1.77       7.64       0.00    1780   0 powercfg
     10    26.67      32.18       0.00    7028   0 powercfg
      8    26.55      31.59       0.00   13600   0 powercfg
      9     1.66       7.55       0.00   22620   0 powercfg
     21     6.17       4.54     202.20   12536   1 PowerMgr
     42    84.26      12.71   2,488.84   20588   1 powershell
     27    47.07      45.38       2.05   25988   1 powershell
     27    24.45       5.31       0.00   12364   0 PresentationFontCache
     92   112.04      13.36      82.30   13176   1 pwsh
    106   163.73      93.21     302.25   14620   1 pwsh
    227   764.01      92.16   1,757.22   25328   1 pwsh
```

To filter your results for PowerShell, pipe your results to the Where-Object cmdlet.

---------- Example 3: Write information to a file ----------

```
function Test-Info
{
    Get-Process P*
```

```
    Write-Information "Here you go"
}
Test-Info 6> Info.txt
```

----- Example 4: Saving information records to a variable -----

```
$psproc = Get-Process -Id $PID | Select-Object ProcessName, CPU, Path
Write-Information -MessageData $psproc -Tags 'PowerShell' -InformationVariable
'InfoMsg'
$InfoMsg | Select-Object *

MessageData     : @{ProcessName=powershell; CPU=1.609375;

Path=C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe}
Source          : Write-Information
TimeGenerated   : 10/19/2023 11:28:15
Tags            : {PowerShell}
User            : sdwheeler
Computer        : circumflex
ProcessId       : 237
NativeThreadId  : 261
ManagedThreadId : 10
```

REMARKS

To see the examples, type: "get-help Write-Information -examples".

For more information, type: "get-help Write-Information -detailed".

For technical information, type: "get-help Write-Information -full".

For online help, type: "get-help Write-Information -online"