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 'Trace-Command'

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

NAME

Trace-Command

SYNOPSIS

Configures and starts a trace of the specified expression or command.

SYNTAX

Trace-Command [-Name] <System.String[]> [-Command] <System.String> [[-Option] {None | Constructor | Dispose | Finalizer | Method | Property | Delegates | Events |

Exception | Lock | Error | Errors | Warning | Verbose | WriteLine | Data | Scope | ExecutionFlow | Assert | All}] [-ArgumentList <System.Object[]>] [-Debugger]

[-FilePath <System.String>] [-Force] [-InputObject <System.Management.Automation.PSObject>] [-ListenerOption {None | LogicalOperationStack | DateTime | Timestamp |

ProcessId | ThreadId | Callstack}] [-PSHost] [<CommonParameters>]

Trace-Command [-Name] <System.String[]> [-Expression] <System.Management.Automation.ScriptBlock> [[-Option] {None | Constructor | Dispose | Finalizer | Method |

Property | Delegates | Events | Exception | Lock | Error | Errors | Warning | Verbose | WriteLine | Data | Scope | ExecutionFlow | Assert | All}] [-Debugger]

[-FilePath <System.String>] [-Force] [-InputObject <System.Management.Automation.PSObject>] [-ListenerOption {None | LogicalOperationStack | DateTime | Timestamp |
ProcessId | ThreadId | Callstack}] [-PSHost] [<CommonParameters>]

DESCRIPTION

The `Trace-Command` cmdlet configures and starts a trace of the specified expression or command. It works like Set-TraceSource, except that it applies only to the
specified command.

PARAMETERS

-ArgumentList <System.Object[]>

Specifies the parameters and parameter values for the command being traced. The alias for ArgumentList is Args . This feature is useful for debugging dynamic
parameters.

For more information about the behavior of ArgumentList , see about_Splatting (../Microsoft.PowerShell.Core/About/about_Splatting.md#splatting-with-arrays).

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

-Command <System.String>

Specifies a command that's being processed during the trace.

When you use this parameter, PowerShell processes the command just as it would be processed in a pipeline. For example, command discovery isn't repeated for each
incoming object.

Required?   true

Position?   1

Default value   None

Accept pipeline input?   False

Accept wildcard characters? false

-Debugger <System.Management.Automation.SwitchParameter>

Indicates that the cmdlet sends the trace output to the debugger. You can view the output in any user-mode or kernel mode debugger or in Visual Studio. This

parameter also selects the default trace listener.

Required?   false

Position?   named

Default value   False

Accept pipeline input?   False

Accept wildcard characters? false

-Expression <System.Management.Automation.ScriptBlock>

Specifies the expression that's being processed during the trace. Enclose the expression in braces (`{}`).

Required?   true

Position?   1

Default value   None

Accept pipeline input?   False

Accept wildcard characters? false

-FilePath <System.String>

Specifies a file that the cmdlet sends the trace output to. This parameter also selects the file trace listener.

Required?   false

Position?   named

Default value   None

Accept pipeline input?   False

        Accept wildcard characters? false

    -Force <System.Management.Automation.SwitchParameter>

        Forces the command to run without asking for user confirmation. Used with the FilePath parameter. Even using the Force parameter, the cmdlet can't override

        security restrictions.

        Required?   false

        Position?   named

        Default value   False

        Accept pipeline input?   False

        Accept wildcard characters? false

    -InputObject <System.Management.Automation.PSObject>

        Specifies input to the expression that's being processed during the trace. You can enter a variable that represents the input that the expression accepts, or pass

        an object through the pipeline.

        Required?   false

        Position?   named

        Default value   None

        Accept pipeline input?   True (ByValue)

        Accept wildcard characters? false

    -ListenerOption <System.Diagnostics.TraceOptions>

        Specifies optional data to the prefix of each trace message in the output. The acceptable values for this parameter are:

        - `None`

        - `LogicalOperationStack`

        - `DateTime`

- `Timestamp`

- `ProcessId`

- `ThreadId`

- `Callstack`

`None` is the default.

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 ListenerOption 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?       False
Accept wildcard characters?  false
```

-Name <System.String[]>

Specifies an array of PowerShell components that are traced. Enter the name of the trace source of each component. Wildcards are permitted. To find the trace sources on your computer, type `Get-TraceSource`.

```
Required?                    true
Position?                    0
Default value                None
Accept pipeline input?       False
Accept wildcard characters?  false
```

-Option <System.Management.Automation.PSTraceSourceOptions>

Determines the type of events that are traced. The acceptable values for this parameter are:

- `None`

- `Constructor`

- `Dispose`

- `Finalizer`

- `Method`

- `Property`

- `Delegates`

- `Events`

- `Exception`

- `Lock`

- `Error`

- `Errors`

- `Warning`

- `Verbose`

- `WriteLine`

- `Data`

- `Scope`

- `ExecutionFlow`

- `Assert`

- `All`

`All` is the default.

The following values are combinations of other values:

- `ExecutionFlow`: `Constructor`, `Dispose`, `Finalizer`, `Method`, `Delegates`, `Events`, `Scope`

- `Data`: `Constructor`, `Dispose`, `Finalizer`, `Property`, `Verbose`, `WriteLine`

- `Errors`: `Error`, `Exception`

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 Option 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?	2
Default value	None
Accept pipeline input?	False

Accept wildcard characters? false

-PSHost <System.Management.Automation.SwitchParameter>

Indicates that the cmdlet sends the trace output to the PowerShell host. This parameter also selects the PSHost trace listener.

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 that represent input to the expression to this cmdlet.

OUTPUTS

System.Management.Automation.PSObject

This cmdlet returns no output of its own. The traced command may return output. This cmdlet writes the command trace to the debug stream.

NOTES

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

- `trcm`

Tracing is a method that developers use to debug and refine programs. When tracing, the program generates detailed messages about each step in its internal

processing. The PowerShell tracing cmdlets are designed to help PowerShell developers, but they're available to all users. They let you monitor nearly every

aspect of the functionality of the shell.

A trace source is the part of each PowerShell component that manages tracing and generates trace messages for the component. To trace a component, you identify

its trace source.

Use `Get-TraceSource` to see a list of PowerShell components that are enabled for tracing.

A trace listener receives the output of the trace and displays it to the user. You can elect to send the trace data to a user-mode or kernel-mode debugger, to the

host or console, to a file, or to a custom listener derived from the System.Diagnostics.TraceListener class.

Example 1: Trace metadata processing, parameter binding, and an expression

```
Trace-Command -Name metadata,parameterbinding,cmdlet -Expression {Get-Process Notepad} -PSHost
```

It uses the Name parameter to specify the trace sources, the Expression parameter to specify the command, and the PSHost parameter to send the output to the console.

Because it doesn't specify any tracing options or listener options, the command uses the defaults:

- All for the tracing options

- None for the listener options

- Example 2: Trace the actions of ParameterBinding operations -

```
$A = "i*"
```

```
Trace-Command ParameterBinding {Get-Alias $Input} -PSHost -InputObject $A
```

In `Trace-Command`, the InputObject parameter passes an object to the expression that's being processed during the trace.

The first command stores the string `i*` in the `$A` variable. The second command uses the `Trace-Command` cmdlet with the ParameterBinding trace source. The PSHost
parameter sends the output to the console.

The expression being processed is `Get-Alias $Input`, where the `$Input` variable is associated with the InputObject parameter. The InputObject parameter passes the
variable `$A` to the expression. In effect, the command being processed during the trace is `Get-Alias -InputObject $A" or "$A | Get-Alias`.

RELATED LINKS

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

Get-TraceSource

Measure-Command

Set-TraceSource

Show-Command