Windows PowerShell Get-Help on Cmdlet 'Set-TraceSource'

PS:\>Get-HELP Set-TraceSource -Full

NAME

Set-TraceSource

SYNOPSIS

Configures, starts, and stops a trace of PowerShell components.

SYNTAX

Set-TraceSource [-Name] <System.String[]> [[-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] [-ListenerOption {None |

LogicalOperationStack | DateTime | Timestamp | ProcessId | ThreadId | Callstack}] [-PassThru] [-PSHost] [<CommonParameters>]

Set-TraceSource [-Name] <System.String[]> [-RemoveFileListener <System.String[]>] [<CommonParameters>]

Set-TraceSource [-Name] <System.String[]> [-RemoveListener <System.String[]>] [<CommonParameters>]

DESCRIPTION

The `Set-TraceSource` cmdlet configures, starts, and stops a trace of a PowerShell component. You can use it to specify which components will be traced and where the

tracing output is sent.

PARAMETERS

-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 Microsoft 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

-FilePath <System.String>

Specifies a file that this cmdlet sends the trace output to. This parameter also selects the file trace listener. If you use this parameter to start the trace,

use the RemoveFileListener parameter to stop the trace.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Force <System.Management.Automation.SwitchParameter>

Indicates that the cmdlet overwrites a read-only file. Use with the FilePath parameter.

Required? false

Position? named

Default value False

Accept pipeline input? False

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 which components are traced. Enter the name of the trace source of each component. Wildcards are permitted.

Required? true

Position? 0

Default value None

Accept pipeline input? True (ByPropertyName, ByValue)

Accept wildcard characters? true

-Option <System.Management.Automation.PSTraceSourceOptions>

Specifies 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? 1

Default value None

Accept pipeline input? True (ByPropertyName)

Accept wildcard characters? false

-PassThru <System.Management.Automation.SwitchParameter>

Returns an object representing the item with which you are working. By default, this cmdlet does not generate any output.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-PSHost <System.Management.Automation.SwitchParameter>

Indicates that this 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

-RemoveFileListener <System.String[]>

Stops the trace by removing the file trace listener associated with the specified file. Enter the path and file name of the trace output file.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-RemoveListener <System.String[]>

Stops the trace by removing the trace listener.

Use the following values with RemoveListener :

- To remove PSHost (console), type `Host`.

- To remove Debugger, type `Debug`.

- To remove all trace listeners, type `*`.

To remove the file trace listener, use the RemoveFileListener parameter.

Required? false

Position? named

Default value None

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 name to this cmdlet.

OUTPUTS

None

By default, this cmdlet returns no output.

System.Management.Automation.PSTraceSource

When you use the PassThru parameter, this cmdlet returns a PSTraceSource object representing the trace session.

NOTES

- 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 are available to all users. They let you monitor nearly every aspect of the

functionality of PowerShell.

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.

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

- To start a trace, use the Name parameter to specify a trace source and the FilePath , Debugger , or PSHost parameters to specify a listener (a destination for the output). Use the Options parameter to determine the types of events that are traced and the ListenerOption parameter to configure the trace output. - To change the configuration of a trace, enter a `Set-TraceSource` command as you would to start a trace. PowerShell recognizes that the trace source is already being traced. It stops the trace, adds the new configuration, and starts or restarts the trace. - To stop a trace, use the RemoveListener parameter. To stop a trace that uses the file listener (a trace started by using the FilePath parameter), use the RemoveFileListener parameter. When you remove the listener, the trace stops. - To determine which components can be traced, use Get-TraceSource. The trace sources for each module are loaded automatically when the component is in use, and they appear in the output of `Get-TraceSource`.

------- Example 1: Trace the ParameterBinding component -------

```
Set-TraceSource -Name "ParameterBinding" -Option ExecutionFlow -PSHost -ListenerOption "ProcessId,TimeStamp"
```

This command starts tracing for the ParameterBinding component of PowerShell. It uses the Name parameter to specify the trace source, the Option parameter to select the `ExecutionFlow` trace events, and the PSHost parameter to select the PowerShell host listener, which sends the output to the console. The ListenerOption parameter adds the `ProcessID` and `TimeStamp` values to the trace message prefix.

---------- Example 2: Stop a trace ----------

```
Set-TraceSource -Name "ParameterBinding" -RemoveListener "Host"
```

This command stops the trace of the ParameterBinding component of PowerShell. It uses the Name parameter to identify

the component that was being traced and the

RemoveListener parameter to identify the trace listener.

RELATED LINKS

Online Version:

https://learn.microsoft.com/powershell/module/microsoft.powershell.utility/set-tracesource?view=powershell-5.1&WT.mc_id=ps-gethelp

Get-TraceSource

Trace-Command