MyWebUniversity
python
PowerShell Get-Help cmdlet
PS C:\> Get-Help
FPDF Library
PDF generator
FPDF
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 'Connect-PSSession'

PS:\>Get-HELP Connect-PSSession -Full

NAME

Connect-PSSession

SYNOPSIS

Reconnects to disconnected sessions.

SYNTAX

Connect-PSSession [-ConnectionUri] <System.Uri[]> [-AllowRedirection] [-Authentication {Default | Basic | Negotiate | NegotiateWithImplicitCredential | Credssp |

Digest | Kerberos}] [-CertificateThumbprint <System.String>] [-ConfigurationName <System.String>] [-Credential <System.Management.Automation.PSCredential>] [-Name

<System.String[]>] [-SessionOption <System.Management.Automation.Remoting.PSSessionOption>] [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Connect-PSSession [-ConnectionUri] <System.Uri[]> [-AllowRedirection] [-Authentication {Default | Basic | Negotiate | NegotiateWithImplicitCredential | Credssp |

Digest | Kerberos}] [-CertificateThumbprint <System.String>] [-ConfigurationName <System.String>] [-Credential <System.Management.Automation.PSCredential>]

-InstanceId <System.Guid[]> [-SessionOption <System.Management.Automation.Remoting.PSSessionOption>]

[-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf]

[<CommonParameters>]

Connect-PSSession [-ComputerName] <System.String[]> [-ApplicationName <System.String>] [-Authentication {Default | Basic | Negotiate | NegotiateWithImplicitCredential

| Credssp | Digest | Kerberos}] [-CertificateThumbprint <System.String>] [-ConfigurationName <System.String>] [-Credential

<System.Management.Automation.PSCredential>] [-Name <System.String[]>] [-Port <System.Int32>] [-SessionOption <System.Management.Automation.Remoting.PSSessionOption>]

[-ThrottleLimit <System.Int32>] [-UseSSL] [-Confirm] [-WhatIf] [<CommonParameters>]

Connect-PSSession [-ComputerName] <System.String[]> [-ApplicationName <System.String>] [-Authentication {Default | Basic | Negotiate | NegotiateWithImplicitCredential

| Credssp | Digest | Kerberos}] [-CertificateThumbprint <System.String>] [-ConfigurationName <System.String>] [-Credential

<System.Management.Automation.PSCredential>] -InstanceId <System.Guid[]> [-Port <System.Int32>] [-SessionOption

<System.Management.Automation.Remoting.PSSessionOption>] [-ThrottleLimit <System.Int32>] [-UseSSL] [-Confirm] [-WhatIf] [<CommonParameters>]

Connect-PSSession [-Id] <System.Int32[]> [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Connect-PSSession -InstanceId <System.Guid[]> [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Connect-PSSession [-Name <System.String[]>] [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Connect-PSSession [-Session] <System.Management.Automation.Runspaces.PSSession[]> [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Connect-PSSession` cmdlet reconnects to user-managed PowerShell sessions (PSSessions) that were disconnected. It works on sessions that are disconnected

intentionally, such as by using the `Disconnect-PSSession` cmdlet or the InDisconnectedSession parameter of the `Invoke-Command` cmdlet, and those that were

disconnected unintentionally, such as by a temporary network outage.

`Connect-PSSession` can connect to any disconnected session that was started by the same user. These include those that were started by or disconnected from other

sessions on other computers.

However, `Connect-PSSession` cannot connect to broken or closed sessions, or interactive sessions started by using the `Enter-PSSession` cmdlet. Also you cannot

connect sessions to sessions started by other users, unless you can provide the credentials of the user who created the session.

For more information about the Disconnected Sessions feature, see about_Remote_Disconnected_Sessions (about/about_Remote_Disconnected_Sessions.md).

This cmdlet was introduced in Windows PowerShell 3.0.

PARAMETERS

-AllowRedirection <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet allows redirection of this connection to an alternate URI.

When you use the ConnectionURI parameter, the remote destination can return an instruction to redirect to a different URI. By default, PowerShell does not

redirect connections, but you can use this parameter to allow it to redirect the connection.

You can also limit the number of times the connection is redirected by changing the MaximumConnectionRedirectionCount session option value. Use the

MaximumRedirection parameter of the `New-PSSessionOption` cmdlet or set the MaximumConnectionRedirectionCount property of the $PSSessionOption preference

variable. The default value is `5`.

Required?   false

Position?   named

Default value   False

Accept pipeline input?   False

Accept wildcard characters? false

-ApplicationName <System.String>

Specifies the name of an application. This cmdlet connects only to sessions that use the specified application.

Enter the application name segment of the connection URI. For example, in the following connection URI, the application name is WSMan:

`http://localhost:5985/WSMAN`. The application name of a session is stored in the Runspace.ConnectionInfo.AppName property of the session.

The value of this parameter is used to select and filter sessions. It does not change the application that the session uses.

Required?   false

Position?   named

Default value   None

Accept pipeline input?   True (ByPropertyName)

Accept wildcard characters? false

-Authentication <System.Management.Automation.Runspaces.AuthenticationMechanism>

Specifies the mechanism that is used to authenticate user credentials in the command to reconnect to the disconnected session. The acceptable values for this

parameter are:

- `Default`

- `Basic`

- `Credssp`

- `Digest`

- `Kerberos`

- `Negotiate`

- `NegotiateWithImplicitCredential`

The default value is `Default`.

For more information about the values of this parameter, see AuthenticationMechanism Enumeration (/dotnet/api/system.management.automation.runspaces.authenticationmechanism).

> [!CAUTION] > Credential Security Support Provider (CredSSP) authentication, in which the user's credentials are > passed to a remote computer to be
authenticated, is designed for commands that require > authentication on more than one resource, such as accessing a remote network share. This mechanism >
increases the security risk of the remote operation. If the remote computer is compromised, the > credentials that are passed to it can be used to control the
network session.

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

-CertificateThumbprint <System.String>

Specifies the digital public key certificate (X509) of a user account that has permission to connect to the disconnected session. Enter the certificate thumbprint of the certificate.

Certificates are used in client certificate-based authentication. They can be mapped only to local user accounts. They do not work with domain accounts.

To get a certificate thumbprint, use a `Get-Item` or `Get-ChildItem` command in the PowerShell `Cert:` drive.

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

-ComputerName <System.String[]>

Specifies the computers on which the disconnected sessions are stored. Sessions are stored on the computer that is at the server-side or receiving end of a

connection. The default is the local computer.

Type the NetBIOS name, an IP address, or a fully qualified domain name of one computer. Wildcard characters are not permitted. To specify the local computer, type

the computer name, `localhost`, or a dot (`.`)

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

-ConfigurationName <System.String>

Connects only to sessions that use the specified session configuration.

Enter a configuration name or the fully qualified resource URI for a session configuration. If you specify only the configuration name, the following schema URI is prepended: `http://schemas.microsoft.com/powershell`. The configuration name of a session is stored in the ConfigurationName property of the session.

The value of this parameter is used to select and filter sessions. It does not change the session configuration that the session uses.

For more information about session configurations, see about_Session_Configurations (About/about_Session_Configurations.md).

Required?	false
Position?	named
Default value	None
Accept pipeline input?	True (ByPropertyName)
Accept wildcard characters?	false

-ConnectionUri <System.Uri[]>

Specifies the URIs of the connection endpoints for the disconnected sessions.

The URI must be fully qualified. The format of this string is as follows:

`<Transport>://<ComputerName>:<Port>/<ApplicationName>`

The default value is as follows:

`http://localhost:5985/WSMAN`

If you do not specify a connection URI, you can use the UseSSL and Port parameters to specify the connection URI values.

Valid values for the Transport segment of the URI are HTTP and HTTPS. If you specify a connection URI with a Transport segment, but do not specify a port, the

session is created with standards ports: `80` for HTTP and `443` for HTTPS. To use the default ports for PowerShell remoting, specify port `5985` for HTTP or

`5986` for HTTPS.

If the destination computer redirects the connection to a different URI, PowerShell prevents the redirection unless you use the AllowRedirection parameter in the
command.

```
Required?                    true
Position?                    0
Default value                None
Accept pipeline input?       True (ByPropertyName)
Accept wildcard characters?  false
```

-Credential <System.Management.Automation.PSCredential>

Specifies a user account that has permission to connect to the disconnected session. The default is the current user.

Type a username, such as `User01` or `Domain01\User01`, or enter a PSCredential object generated by the `Get-Credential` cmdlet. If you type a user name, you're
prompted to enter the password.

Credentials are stored in a PSCredential (/dotnet/api/system.management.automation.pscredential)object and the password is stored as a SecureString
(/dotnet/api/system.security.securestring).

> [!NOTE] > For more information about SecureString data protection, see > How secure is SecureString?
(/dotnet/api/system.security.securestring#how-secure-is-securestring).

```
Required?                    false
Position?                    named
Default value                Current user
Accept pipeline input?       False
Accept wildcard characters?  false
```

-Id <System.Int32[]>

Specifies the IDs of the disconnected sessions. The Id parameter works only when the disconnected session was previously connected to the current session.

This parameter is valid, but not effective, when the session is stored on the local computer, but was not connected to the current session.

```
Required?                    true
Position?                    0
Default value                None
Accept pipeline input?       True (ByPropertyName)
Accept wildcard characters?  false
```

-InstanceId <System.Guid[]>

Specifies the instance IDs of the disconnected sessions.

The instance ID is a GUID that uniquely identifies a PSSession on a local or remote computer.

The instance ID is stored in the InstanceID property of the PSSession .

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

-Name <System.String[]>

Specifies the friendly names of the disconnected sessions.

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

-Port <System.Int32>

Specifies the network port on the remote computer that is used to reconnect to the session. To connect to a remote computer, the remote computer must be listening

on the port that the connection uses. The default ports are `5985`, which is the WinRM port for HTTP, and `5986`, which is the WinRM port for HTTPS.

Before using an alternate port, you must configure the WinRM listener on the remote computer to listen at that port. To configure the listener, type the following

two commands at the PowerShell prompt:

`Remove-Item -Path WSMan:\Localhost\listener\listener* -Recurse`

`New-Item -Path WSMan:\Localhost\listener -Transport http -Address * -Port <port-number>`

Do not use the Port parameter unless you must. The port that is set in the command applies to all computers or sessions on which the command runs. An alternate

port setting might prevent the command from running on all computers.

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

-Session <System.Management.Automation.Runspaces.PSSession[]>

Specifies the disconnected sessions. Enter a variable that contains the PSSession objects or a command that creates or gets the PSSession objects, such as a

`Get-PSSession` command.

Required?	true
Position?	0
Default value	None

Accept pipeline input?   True (ByPropertyName, ByValue)

Accept wildcard characters? false

-SessionOption <System.Management.Automation.Remoting.PSSessionOption>

Specifies advanced options for the session. Enter a SessionOption object, such as one that you create by using the `New-PSSessionOption` cmdlet, or a hash table

in which the keys are session option names and the values are session option values.

The default values for the options are determined by the value of the `$PSSessionOption` preference variable, if it is set. Otherwise, the default values are

established by options set in the session configuration.

The session option values take precedence over default values for sessions set in the `$PSSessionOption` preference variable and in the session configuration.

However, they do not take precedence over maximum values, quotas or limits set in the session configuration.

For a description of the session options that includes the default values, see `New-PSSessionOption`. For information about the $PSSessionOption preference

variable, see about_Preference_Variables (About/about_Preference_Variables.md). For more information about session configurations, see

about_Session_Configurations (About/about_Session_Configurations.md).

Required?   false

Position?   named

Default value   None

Accept pipeline input?   False

Accept wildcard characters? false

-ThrottleLimit <System.Int32>

Specifies the maximum number of concurrent connections that can be established to run this command. If you omit this parameter or enter a value of `0`, the

default value, `32`, is used.

The throttle limit applies only to the current command, not to the session or to the computer.

Required?   false

Position?   named

Default value   None

Accept pipeline input?   False

Accept wildcard characters? false

-UseSSL <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet uses the Secure Sockets Layer (SSL) protocol to connect to the disconnected session. By default, SSL is not used.

WS-Management encrypts all PowerShell content transmitted over the network. The UseSSL parameter is an additional protection that sends the data across an HTTPS

connection instead of an HTTP connection.

If you use this parameter, but SSL is not available on the port that is used for the command, the command fails.

Required?   false

Position?   named

Default value   False

Accept pipeline input?   False

Accept wildcard characters? false

-Confirm <System.Management.Automation.SwitchParameter>

Prompts you for confirmation before running the cmdlet.

Required?   false

Position?   named

Default value   False

Accept pipeline input?   False

Accept wildcard characters? false

-WhatIf <System.Management.Automation.SwitchParameter>

Shows what would happen if the cmdlet runs. The cmdlet is not run.

```
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.Runspaces.PSSession

You can pipe a session (PSSession) to this cmdlet.

OUTPUTS

System.Management.Automation.Runspaces.PSSession

This cmdlet returns an object that represents the session to which it reconnected.

NOTES

Windows PowerShell includes the following aliases for `Connect-PSSession`:

- `cnsn`

- This cmdlet is only available on Windows platforms.

- `Connect-PSSession` reconnects only to sessions that are disconnected, that is, sessions that have a value of Disconnected for the State property. Only
sessions that are connected to, or end at, computers that run Windows PowerShell 3.0 or later versions can be disconnected and reconnected.

- If you use `Connect-PSSession` on a session that is not disconnected, the command does not affect the session and it does not generate errors.

- Disconnected loopback sessions with interactive tokens, which are created by using the EnableNetworkAccess parameter, can be reconnected only from the computer
on which the session was created. This restriction protects the computer from malicious access.

- The value of the State property of a PSSession is relative to the current session. Therefore, a value of Disconnected means that the PSSession is not
connected to the current session. However, it does not mean that the PSSession is disconnected from all sessions. It might be connected to a different
session. To determine whether you can connect or reconnect to the session, use the Availability property.

An Availability value of None indicates that you can connect to the session. A value of Busy indicates that you cannot connect to the PSSession because it is
connected to another session.

For more information about the values of the State property of sessions, see RunspaceState Enumeration
(/dotnet/api/system.management.automation.runspaces.runspacestate).

For more information about the values of the Availability property of sessions, see RunspaceAvailability Enumeration
(/dotnet/api/system.management.automation.runspaces.runspaceavailability).

- You cannot change the idle time-out value of a PSSession when you connect to the PSSession . The SessionOption parameter of `Connect-PSSession` takes a
SessionOption object that has an IdleTimeout value. However, the IdleTimeout value of the SessionOption object and the IdleTimeout value of the `$PSSessionOption`

variable are ignored when connecting to a PSSession .

You can set and change the idle time-out of a PSSession when you create the PSSession , by using the `New-PSSession` or `Invoke-Command` cmdlets, and when you disconnect from the PSSession .

The IdleTimeout property of a PSSession is critical to disconnected sessions, because it determines how long a disconnected session is maintained on the remote computer. Disconnected sessions are considered to be idle from the moment that they are disconnected, even if commands are running in the disconnected session.

------------- Example 1: Reconnect to a session -------------

```
Connect-PSSession -ComputerName Server01 -Name ITTask

Id Name          ComputerName    State        ConfigurationName     Availability
-- ----          ------------    -----        -----------------     ------------
 4 ITTask        Server01        Opened       ITTasks               Available
```

This command reconnects to the `ITTask` session on the Server01 computer.

The output shows that the command was successful. The State of the session is `Opened` and the Availability is `Available`, which indicates that you can run commands in the session.

----- Example 2: Effect of disconnecting and reconnecting -----

```
Get-PSSession

Id Name          ComputerName    State        ConfigurationName     Availability
-- ----          ------------    -----        -----------------     ------------
 1 Backups       Localhost       Opened       Microsoft.PowerShell  Available
```

```
Id Name          ComputerName    State        ConfigurationName     Availability
-- ----          ------------    -----        -----------------     ------------
 1 Backups       Localhost       Disconnected  Microsoft.PowerShell         None
```

```
Get-PSSession | Connect-PSSession
```

```
Id Name          ComputerName    State        ConfigurationName     Availability
-- ----          ------------    -----        -----------------     ------------
 1 Backups       Localhost       Opened       Microsoft.PowerShell     Available
```

This example shows the effect of disconnecting and then reconnecting to a session.

The first command uses the `Get-PSSession` cmdlet. Without the ComputerName parameter, the command gets only sessions that were created in the current session.

The output shows that the command gets the `Backups` session on the local computer. The State of the session is `Opened` and the Availability is `Available`.

The second command uses the `Get-PSSession` cmdlet to get the PSSession objects that were created in the current session and the `Disconnect-PSSession` cmdlet to
disconnect the sessions. The output shows that the `Backups` session was disconnected. The State of the session is `Disconnected` and the Availability is `None`.

The third command uses the `Get-PSSession` cmdlet to get the PSSession objects that were created in the current session and the `Connect-PSSession` cmdlet to
reconnect the sessions. The output shows that the `Backups` session was reconnected. The State of the session is `Opened` and the Availability is `Available`.

If you use the `Connect-PSSession` cmdlet on a session that is not disconnected, the command does not affect the session and it does not generate any errors.

--- Example 3: Series of commands in an enterprise scenario ---

```
$s = New-PSSession -ComputerName Server01 -Name ITTask -ConfigurationName ITTasks

Invoke-Command -Session $s -ScriptBlock {Start-Job -FilePath \\Server30\Scripts\Backup-SQLDatabase.ps1}


Id     Name           State        HasMoreData     Location           Command

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

2      Job2           Running      True            Server01           \\Server30\Scripts\Backup...


Disconnect-PSSession -Session $s -OutputBufferingMode Drop -IdleTimeoutSec 60*60*15


Id Name          ComputerName    State        ConfigurationName     Availability

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

 1 ITTask        Server01        Disconnected  ITTasks              None


Get-PSSession -ComputerName Server01 -Name ITTask


Id Name          ComputerName    State        ConfigurationName     Availability

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

 1 ITTask        Server01        Disconnected  ITTasks              None


$s = Connect-PSSession -ComputerName Server01 -Name ITTask


Id Name          ComputerName    State        ConfigurationName     Availability

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

 1 ITTask        Server01        Opened       ITTasks              Available


Invoke-Command -Session $s -ScriptBlock {Get-Job}


Id     Name           State        HasMoreData     Location           Command

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

2      Job2           Completed    True            Server01           \\Server30\Scripts\Backup...


Invoke-Command -Session $s -ScriptBlock {$BackupSpecs = Receive-Job -JobName Job2}

        Invoke-Command -Session $s -ScriptBlock {\\Server30\Scripts\New-SQLDatabase.ps1 -InitData
$BackupSpecs.Initialization}
    Disconnect-PSSession -Session $s -OutputBufferingMode Drop -IdleTimeoutSec 60*60*15


    Id Name            ComputerName    State         ConfigurationName     Availability
    -- ----            ------------    -----         -----------------     ------------
     1 ITTask          Server01        Disconnected  ITTasks               None
```

The ninth command disconnects from the session in the `$s` variable.The administrator closes PowerShell and closes the computer. She can reconnect to the session on

the next day and check the script status from her work computer.

RELATED LINKS

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

Disconnect-PSSession

Enter-PSSession

Exit-PSSession

Get-PSSession

Get-PSSessionConfiguration

New-PSSession

New-PSSessionOption

New-PSTransportOption

Receive-PSSession

Register-PSSessionConfiguration

Remove-PSSession