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 'Remove-Job'

PS:\>Get-HELP Remove-Job -Full

NAME

Remove-Job

SYNOPSIS

Deletes a PowerShell background job.

SYNTAX

Remove-Job [-Command <System.String[]>] [-Confirm] [-WhatIf] [<CommonParameters>]

Remove-Job [-Filter] <System.Collections.Hashtable> [-Force] [-Confirm] [-WhatIf] [<CommonParameters>]

Remove-Job [-Id] <System.Int32[]> [-Force] [-Confirm] [-WhatIf] [<CommonParameters>]

Remove-Job [-Job] <System.Management.Automation.Job[]> [-Force] [-Confirm] [-WhatIf] [<CommonParameters>]

Remove-Job [-Name] <System.String[]> [-Force] [-Confirm] [-WhatIf] [<CommonParameters>]

Remove-Job [-InstanceId] <System.Guid[]> [-Force] [-Confirm] [-WhatIf] [<CommonParameters>]

Remove-Job [-State] {AtBreakpoint | Blocked | Completed | Disconnected | Failed | NotStarted | Running | Stopped | Stopping | Suspended | Suspending} [-Confirm]
[-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Remove-Job` cmdlet deletes PowerShell background jobs that were started by the `Start-Job` cmdlet or by cmdlets such as `Invoke-Command` that support the AsJob
parameter.

You can use `Remove-Job` to delete all jobs or delete selected jobs. The jobs are identified by their Name , ID , Instance ID , Command , or State . Or, a job object
can be sent down the pipeline to `Remove-Job`. Without parameters or parameter values, `Remove-Job` has no effect.

Since PowerShell 3.0, `Remove-Job` can delete custom job types, such as scheduled jobs and workflow jobs. For example, `Remove-Job` deletes the scheduled job, all
instances of the scheduled job on disk, and the results of all triggered job instances.

If you try to delete a running job, `Remove-Job` fails. Use the `Stop-Job` cmdlet to stop a running job. Or, use `Remove-Job` with the Force parameter to delete a
running job.

Jobs remain in the global job cache until you delete the background job or close the PowerShell session.

PARAMETERS

-Command <System.String[]>

Deletes jobs that include the specified words in the command. You can enter a comma-separated array.

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

Accept wildcard characters? false

-Filter <System.Collections.Hashtable>

Deletes jobs that satisfy all the conditions established in the associated hash table. Enter a hash table where the keys are job properties and the values are job

property values.

This parameter works only on custom job types, such as workflow jobs and scheduled jobs. It doesn't work on standard background jobs, such as those created by

using the `Start-Job`.

This parameter is introduced in PowerShell 3.0.

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

-Force <System.Management.Automation.SwitchParameter>

Deletes a job even if the job's state is Running . If the Force parameter isn't specified, `Remove-Job` doesn't delete running jobs.

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

-Id <System.Int32[]>

Deletes background jobs with the specified Id . You can enter a comma-separated array. The job's Id is a unique integer that identifies a job within the current

session.

To find a job's Id , use `Get-Job` without parameters.

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

-InstanceId <System.Guid[]>

Deletes jobs with the specified InstanceId . You can enter a comma-separated array. An InstanceId is a unique GUID that identifies a job.

To find a job's InstanceId , use `Get-Job`.

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

-Job <System.Management.Automation.Job[]>

Specifies the jobs to be deleted. Enter a variable that contains the jobs or a command that gets the jobs. You can enter a comma-separated array.

You can send job objects down the pipeline to `Remove-Job`.

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

-Name <System.String[]>

Only deletes jobs with the specified friendly name. Wildcards are permitted. You can enter a comma-separated array.

Friendly names for jobs aren't guaranteed to be unique, even within a PowerShell session. Use the WhatIf and Confirm parameters when you delete files by name.

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

-State <System.Management.Automation.JobState>

Only deletes jobs with the specified state. To delete jobs with a state of Running , use the Force parameter.

Accepted values:

- AtBreakpoint

- Blocked

- Completed

- Disconnected

- Failed

- NotStarted

- Running

- Stopped

```
    - Stopping

    - Suspended

    - Suspending

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

-Confirm <System.Management.Automation.SwitchParameter>
    Prompts you for confirmation before `Remove-Job` is run.

    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 `Remove-Job` runs. The cmdlet isn't 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.Job

You can pipe a Job object to this cmdlet.

OUTPUTS

None

This cmdlet returns no output.

NOTES

Windows PowerShell includes the following aliases for `Remove-Job`:

- `rjb`

A PowerShell job creates a new process. When the job completes, the process exits. When `Remove-Job` is run, the job's state is removed.

If a job stops before completion and its process hasn't exited, the process is forcibly terminated.

---------- Example 1: Delete a job by using its name ----------

```
$batch = Get-Job -Name BatchJob
$batch | Remove-Job
```

`Get-Job` uses the Name parameter to specify the job, BatchJob . The job object is stored in the `$batch` variable. The object in `$batch` is sent down the pipeline

to `Remove-Job`.

An alternative is to use the Job parameter, such as `Remove-Job -Job $batch`.

----------- Example 2: Delete all jobs in a session -----------

```
Get-job | Remove-Job
```

`Get-Job` gets all the jobs in the current PowerShell session. The job objects are sent down the pipeline to `Remove-Job`.

-------------- Example 3: Delete NotStarted jobs --------------

```
Remove-Job -State NotStarted
```

`Remove-Job` uses the State parameter to specify the job status.

------- Example 4: Delete jobs by using a friendly name -------

```
Remove-Job -Name *batch -Force
```

`Remove-Job` uses the Name parameter to specify a job name pattern. The pattern includes the asterisk (` `) wildcard to find all job names that end with batch . The

Force * parameter deletes jobs that running.

-- Example 5: Delete a job that was created by Invoke-Command --

```
$job = Invoke-Command -ComputerName Server01 -ScriptBlock {Get-Process} -AsJob
$job | Remove-Job
```

`Invoke-Command` runs a job on the Server01 computer. The AsJob parameter runs the ScriptBlock as a background job. The job object is stored in the `$job` variable.

The `$job` variable object is sent down the pipeline to `Remove-Job`.

Example 6: Delete a job that was created by Invoke-Command and Start-Job

```
$S = New-PSSession -ComputerName Server01
Invoke-Command -Session $S -ScriptBlock {Start-Job -ScriptBlock {Get-Process} -Name MyJob}
Invoke-Command -Session $S -ScriptBlock {Remove-Job -Name MyJob}
```

`New-PSSession` creates a PSSession , a persistent connection, to the Server01 computer. The connection is saved in the `$S` variable.

`Invoke-Command` connects to the session saved in `$S`. The ScriptBlock uses `Start-Job` to start a remote job. The job runs a `Get-Process` command and uses the Name
parameter to specify a friendly job name, MyJob .

`Invoke-Command` uses the `$S` session and runs `Remove-Job`. The Name parameter specifies that the job named MyJob is deleted.

------- Example 7: Delete a job by using its InstanceId -------

```
$job = Start-Job -ScriptBlock {Get-Process PowerShell}
$job | Format-List -Property *
Remove-Job -InstanceId ad02b942-8007-4407-87f3-d23e71955872

State         : Completed
HasMoreData   : True
StatusMessage :
Location      : localhost
Command       : Get-Process PowerShell
JobStateInfo  : Completed
Finished      : System.Threading.ManualResetEvent
InstanceId    : ad02b942-8007-4407-87f3-d23e71955872
Id            : 3
Name          : Job3
ChildJobs     : {Job4}
PSBeginTime   : 7/26/2019 11:36:56
PSEndTime     : 7/26/2019 11:36:57
PSJobTypeName : BackgroundJob
Output        : {}
Error         : {}
Progress      : {}
```

```
    Verbose       : {}
    Debug         : {}
    Warning       : {}
    Information   : {}
```

`Start-Job` starts a background job and the job object is saved in the `$job` variable.

The object in `$job` is sent down the pipeline to `Format-List`. The Property parameter uses an asterisk (`*`) to specify that all the object's properties are
displayed in a list.

`Remove-Job` uses the InstanceId parameter to specify the job to delete.

RELATED LINKS

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

about_Jobs

about_Job_Details

about_Remote_Jobs

Get-Job

Invoke-Command

Receive-Job

Resume-Job

Start-Job

Stop-Job

Suspend-Job

Wait-Job