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

PowerShell Get-Help on command 'Set-Alias'

PS C:\Users\wahid> Get-Help Set-Alias

NAME

Set-Alias

SYNOPSIS

Creates or changes an alias for a cmdlet or other command in the current PowerShell session.

SYNTAX

Set-Alias [-Name] <System.String> [-Value] <System.String> [-Description <System.String>] [-Force] [-Option {AllScope | Constant | None | Private | ReadOnly | Unspecified}] [-PassThru] [-Scope {Global | Local | Private | Numbered scopes | Script}] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Set-Alias` cmdlet creates or changes an alias for a cmdlet or a command, such as a function, script, file, or other executable. An alias is an alternate name that refers to a cmdlet or command. For example, `sal` is the alias for the `Set-Alias` cmdlet. For more information, see about_Aliases (../Microsoft.PowerShell.Core/About/about_Aliases.md).

A cmdlet can have multiple aliases, but an alias can only be associated with one cmdlet. You can use `Set-Alias` to reassign an existing alias to another cmdlet, or change an alias's properties, such as the description.

An alias that's created or changed by `Set-Alias` isn't permanent and is only available during the current PowerShell session. When the PowerShell session is closed, the alias is removed.

PARAMETERS

-Description <System.String>

Specifies a description of the alias. You can type any string. If the description includes spaces, enclose it in single quotation marks.

-Force <System.Management.Automation.SwitchParameter>

Use the Force parameter to change or delete an alias that has the Option parameter set to ReadOnly .

The Force parameter can't change or delete an alias with the Option parameter set to Constant .

-Name <System.String>

Specifies the name of a new alias. An alias name can contain alphanumeric characters and hyphens. Alias names can't be numeric, such as 123.

-Option <System.Management.Automation.ScopedItemOptions>

Sets the Option property value of the alias. Values such as `ReadOnly` and `Constant` protect an alias from unintended changes. To see the Option property of all aliases in the session, type `Get-Alias | Format-Table -Property Name, Options -Autosize`.

The acceptable values for this parameter are as follows:

- `AllScope` - The alias is copied to any new scopes that are created.

- `Constant` - Can't be changed or deleted.

- `None` - Sets no options and is the default.

- `Private` - The alias is available only in the current scope.

- `ReadOnly` - Can't be changed or deleted unless the Force parameter is used. - `Unspecified`

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 combines 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.

-PassThru <System.Management.Automation.SwitchParameter>

Returns an object that represents the alias. Use a format cmdlet such as `Format-List` to display the object. By default, `Set-Alias` doesn't generate any output.

-Scope <System.String>

Specifies the scope this alias is valid in. The default value is Local . For more information, see about_Scopes (../Microsoft.PowerShell.Core/About/about_Scopes.md).

The acceptable values are as follows:

- `Global`

- `Local`

- `Private`

- `Numbered scopes`

- `Script`

-Value <System.String>
Specifies the name of the cmdlet or command that the alias runs. The Value parameter is the alias's Definition property.

-Confirm <System.Management.Automation.SwitchParameter>
Prompts you for confirmation before running the cmdlet.

-WhatIf <System.Management.Automation.SwitchParameter>
Shows what would happen if the cmdlet runs. The cmdlet isn't run.

<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: Create an alias for a cmdlet ----------

PS> Set-Alias -Name list -Value Get-ChildItem

PS> Get-Alias -Name list

CommandType Name
---------- ----
Alias list -> Get-ChildItem

The `Set-Alias` cmdlet creates an alias in the current PowerShell session. The Name parameter specifies the alias's name, `list`. The Value parameter specifies the cmdlet that the alias runs.

To run the alias, type `list` on the PowerShell command line.
- Example 2: Reassign an existing alias to a different cmdlet -

```
PS> Get-Alias -Name list

CommandType     Name
-----------     ----
Alias           list -> Get-ChildItem

PS> Set-Alias -Name list -Value Get-Location

PS> Get-Alias -Name list

CommandType     Name
-----------     ----
Alias           list -> Get-Location
```

The `Get-Alias` cmdlet uses the Name parameter to display the `list` alias. The `list` alias is associated with the `Get-ChildItem` cmdlet. When the `list` alias is run, the items in the current directory are displayed.

The `Set-Alias` cmdlet uses the Name parameter to specify the `list` alias. The Value parameter associates the alias to the `Get-Location` cmdlet.

The `Get-Alias` cmdlet uses the Name parameter to display the `list` alias. The `list` alias is associated with the `Get-Location` cmdlet. When the `list` alias is run, the current directory's location is displayed.

-------- Example 3: Create and change a read-only alias --------

```
Set-Alias -Name loc -Value Get-Location -Option ReadOnly -PassThru |
    Format-List -Property *

DisplayName         : loc -> Get-Location
Definition          : Get-Location
Options             : ReadOnly
Description         :
Name                : loc
CommandType         : Alias

$Parameters = @{
    Name        =  'loc'
    Value       =  (Get-Location)
    Option      =  'ReadOnly'
    Description =  'Displays the current directory'
    Force       = $true
    PassThru    = $true
}
Set-Alias @Parameters | Format-List -Property *

DisplayName         : loc -> Get-Location
Definition          : Get-Location
Options             : ReadOnly
Description         : Displays the current directory
Name                : loc
CommandType         : Alias
```

The `Set-Alias` cmdlet creates an alias in the current PowerShell session. The Name parameter specifies the alias's name, `loc`. The Value parameter specifies the `Get-Location` cmdlet that the alias runs. The Option parameter specifies the ReadOnly value. The PassThru parameter represents the alias object and sends the object down the pipeline to the `Format-List` cmdlet.

`Format-List` uses the Property parameter with an asterisk (`*`) so that every property is displayed. The example output shows a partial list of those properties.

The `loc` alias is changed with the addition of two parameters. Description adds text to explain the alias's purpose. The Force parameter is needed because the `loc` alias is read-only. If the Force parameter isn't used, the change fails.

------- Example 4: Create an alias to an executable file -------

```
PS> Set-Alias -Name np -Value C:\Windows\notepad.exe

PS> Get-Alias -Name np

CommandType     Name
-----------     ----
Alias           np -> notepad.exe
```

The `Set-Alias` cmdlet creates an alias in the current PowerShell session. The Name parameter specifies the alias's name, `np`. The Value parameter specifies the path and application name `C:\Windows\notepad.exe`. The `Get-Alias` cmdlet uses the Name parameter to show that the `np` alias is associated with `notepad.exe`.

To run the alias, type `np` on the PowerShell command line to open `notepad.exe`.

--- Example 5: Create an alias for a command with parameters ---

```
Function CD32 {Set-Location -Path C:\Windows\System32}

Set-Alias -Name Go -Value CD32
```

A function named `CD32` is created. The function uses the `Set-Location`

cmdlet with the Path parameter to specify the directory, `C:\Windows\System32`.

The `Set-Alias` cmdlet creates an alias to the function in the current PowerShell session. The Name parameter specifies the alias's name, `Go`. The Value parameter specifies the function's name, `CD32`.

To run the alias, type `Go` on the PowerShell command line. The `CD32` function runs and changes to the directory `C:\Windows\System32`.

------- Example 6: Update options for an existing alias -------

Set-Alias -Name Go -Option ReadOnly, Private

The alias `Go` should already exist. After running the command, the alias can't be changed without using the Force parameter and is only available in the current scope.

REMARKS

To see the examples, type: "get-help Set-Alias -examples".

For more information, type: "get-help Set-Alias -detailed".

For technical information, type: "get-help Set-Alias -full".

For online help, type: "get-help Set-Alias -online"