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

PowerShell Get-Help on command 'New-PSDrive'

PS C:\Users\wahid> Get-Help New-PSDrive

NAME

New-PSDrive

SYNOPSIS

Creates temporary and persistent drives that are associated with a location in an item data store.

SYNTAX

New-PSDrive [-Name] <System.String> [-PSProvider] <System.String> [-Root] <System.String> [-Credential <System.Management.Automation.PSCredential>] [-Description <System.String>] [-Persist] [-Scope <System.String>] [-UseTransaction] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `New-PSDrive` cmdlet creates temporary and persistent drives that are mapped to or associated with a location in a data store, such as a network drive, a directory on the local computer, or a registry key, and persistent Windows mapped network drives that are associated with a file system location on a remote computer.

Temporary drives exist only in the current PowerShell session and in sessions that you create in the current session. They can have any name that is valid in PowerShell and can be mapped to any local or remote resource. You can use temporary PowerShell drives to access data in the associated data store, just as you would do with any mapped network drive. You can change locations into the drive, by using `Set-Location`, and access the contents of the drive by using `Get-Item` or `Get-ChildItem`.

Because temporary drives are known only to PowerShell, you can't access them by using File Explorer, Windows Management Instrumentation (WMI), Component Object Model (COM), Microsoft .NET Framework, or with tools such as `net use`.

The following features were added to `New-PSDrive` in PowerShell 3.0:

- Mapped network drives. You can use the Persist parameter of `New-PSDrive` to create Windows mapped network drives. Unlike temporary PowerShell drives, Windows mapped network drives aren't session-specific. They're saved in Windows and they can be managed by using standard Windows tools, such as File Explorer and net use . Mapped network drives must have a drive-letter name and be connected to a remote file system location. When your command is scoped locally, no dot-sourcing, the Persist parameter doesn't persist the creation of a PSDrive beyond the scope in which the command is running. If you're running `New-PSDrive` inside a script, and you want the drive to persist indefinitely, you must dot-source the script. For best results, to force a new drive to persist indefinitely, add the Scope parameter to your command, and set its value to Global . For more information about dot-sourcing, see about_Scripts (../Microsoft.PowerShell.Core/About/about_Scripts.md#script-scope-and-dot-sourcing). - External drives. When an external drive is connected to the computer, PowerShell automatically adds a PSDrive to the file system that represents the new drive. You don't have to restart PowerShell. Similarly, when an external drive is disconnected from the computer, PowerShell automatically deletes the PSDrive that represents the

removed drive. - Credentials for Universal Naming Convention (UNC) paths.

When the value of the Root parameter is a UNC path, such as `\Server\Share`, the credential specified in the value of the Credential parameter is used to create the PSDrive . Otherwise, Credential isn't effective when you're creating new file system drives.

Some code samples use splatting to reduce the line length and improve readability. For more information, see about_Splatting (../Microsoft.PowerShell.Core/About/about_Splatting.md).

> [!NOTE] > Unless you use the Scope parameter, PSDrives are created in the scope in which the > `New-PSDrive` command is run.

PARAMETERS

-Credential <System.Management.Automation.PSCredential>

Specifies a user account that has permission to do this action. The default is the current user.

Since PowerShell 3.0, when the value of the Root parameter is a UNC path, you can use credentials to create file system drives.

Type a user name, 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).

-Description <System.String>
Specifies a brief text description of the drive. Type any string.

To see the descriptions of all the session's drives, `Get-PSDrive |
Format-Table Name, Description`.

To see the description of a particular drive, type `(Get-PSDrive
<DriveName>).Description`.

-Name <System.String>
Specifies a name for the new drive. For persistent mapped network drives,
use a drive letter. For temporary PowerShell drives, you aren't limited to
drive letters, use any valid string.

-Persist <System.Management.Automation.SwitchParameter>
Indicates that this cmdlet creates a Windows mapped network drive. The
Persist parameter is only available on Windows.

Mapped network drives are saved in Windows on the local computer. They're
persistent, not session-specific, and can be viewed and managed in File
Explorer and other tools.

When you scope the command locally, without dot-sourcing, the Persist
parameter doesn't persist the creation of a PSDrive beyond the scope in
which you run the command. If you run `New-PSDrive` inside a script, and
you want the new drive to persist indefinitely, you must dot-source the
script. For best results, to force a new drive to persist, specify Global
as the value of the Scope parameter and include Persist in your command.

The name of the drive must be a letter, such as `D` or `E`. The value of

Root parameter must be a UNC path of a different computer. The PSProvider parameter's value must be `FileSystem`.

To disconnect a Windows mapped network drive, use the `Remove-PSDrive` cmdlet. When you disconnect a Windows mapped network drive, the mapping is permanently deleted from the computer, not just deleted from the current session.

Mapped network drives are specific to a user account. Mapped drives created in elevated sessions or sessions using the credential of another user aren't visible in sessions started using different credentials.

-PSProvider <System.String>

Specifies the PowerShell provider that supports drives of this kind.

For example, if the drive is associated with a network share or file system directory, the PowerShell provider is `FileSystem`. If the drive is associated with a registry key, the provider is `Registry`.

Temporary PowerShell drives can be associated with any PowerShell provider. Mapped network drives can be associated only with the `FileSystem` provider.

To see a list of the providers in your PowerShell session, use the `Get-PSProvider` cmdlet.

-Root <System.String>

Specifies the data store location to which a PowerShell drive is mapped.

For example, specify a network share, such as `\Server01\Public`, a local directory, such as `C:\Program Files`, or a registry key, such as `HKLM:\Software\Microsoft`.

Temporary PowerShell drives can be associated with a local or remote location on any supported provider drive. Mapped network drives can be associated only with a file system location on a remote computer.

-Scope <System.String>

Specifies a scope for the drive. The acceptable values for this parameter are: Global , Local , and Script , or a number relative to the current scope. Scopes number 0 through the number of scopes. The current scope number is 0 and its parent is 1. For more information, see about_Scopes (../Microsoft.PowerShell.Core/About/about_Scopes.md).

-UseTransaction <System.Management.Automation.SwitchParameter>

Includes the command in the active transaction. This parameter is valid only when a transaction is in progress. For more information, see about_Transactions (../Microsoft.PowerShell.Core/About/about_Transactions.md).

-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 a temporary drive mapped to a network share

```
New-PSDrive -Name "Public" -PSProvider "FileSystem" -Root "\\Server01\Public"
```

```
Name       Provider      Root
----       --------      ----
Public     FileSystem    \\Server01\Public
```

`New-PSDrive` uses the Name parameter to specify PowerShell drive named `Public` and the PSProvider parameter to specify the PowerShell `FileSystem` provider. The Root parameter specifies the network share's UNC path.

To view the contents from a PowerShell session: `Get-ChildItem -Path Public:`

Example 2: Create a temporary drive mapped to a local directory

```
$parameters = @{
   Name = "MyDocs"
   PSProvider = "FileSystem"
   Root = "C:\Users\User01\Documents"
   Description = "Maps to my My Documents folder."
}
New-PSDrive @parameters
```

```
Name       Provider      Root
----       --------      ----
MyDocs     FileSystem    C:\Users\User01\Documents
```

Splatting creates the parameter keys and values. The Name parameter specifies the drive name, MyDocs . The PSProvider parameter specifies the PowerShell `FileSystem` provider. Root specifies the local computer's directory. The Description parameter describes the drive's purpose. `New-PSDrive` uses the splatted parameters to create the `MyDocs` drive.

To view the contents from a PowerShell session: `Get-ChildItem -Path MyDocs:`

---- Example 3: Create a temporary drive for a registry key ----

```
New-PSDrive -Name "MyCompany" -PSProvider "Registry" -Root
```

"HKLM:\Software\MyCompany"

Name Provider Root

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

MyCompany Registry HKLM:\Software\MyCompany

`New-PSDrive` uses the Name parameter to specify PowerShell drive named `MyCompany` and the PSProvider parameter to specify the PowerShell `Registry` provider. The Root parameter specifies the registry location.

To view the contents from a PowerShell session: `Get-ChildItem -Path MyCompany:`

Example 4: Create a persistent mapped network drive using credentials

$cred = Get-Credential -Credential Contoso\ServiceAccount

New-PSDrive -Name "S" -Root "\\Server01\Scripts" -Persist -PSProvider "FileSystem" -Credential $cred

Net Use

Status Local Remote Network

OK S: \\Server01\Scripts Microsoft Windows Network

> [!NOTE] > Remember, if you use the above snippet in a script, set the Scope parameter value to > "Global" to ensure the drive persists outside the current scope.

The `$cred` variable stores a PSCredential object that contains the service account's credentials. `Get-Credential` prompts you to enter the password that's stored in a SecureString .

`New-PSDrive` creates the mapped network drive by using several parameters. Name specifies the `S` drive letter that Windows accepts. and Root defines

`\Server01\Scripts` as the location on a remote computer. Persist creates a Windows mapped network drive that's saved on the local computer. PSProvider specifies the `FileSystem` provider. Credential uses the `$cred` variable to get the service account credentials for authentication.

The mapped drive can be viewed on the local computer in PowerShell sessions, File Explorer, and with tools such as net use . To view the contents from a PowerShell session: `Get-ChildItem -Path S:`

------ Example 5: Create persistent and temporary drives ------

```
# Create a temporary PowerShell drive called PSDrive:
# that's mapped to the \\Server01\Public network share.
New-PSDrive -Name "PSDrive" -PSProvider "FileSystem" -Root "\\Server01\Public"

# Use the Persist parameter of New-PSDrive to create the X: mapped network
drive,
# which is also mapped to the \\Server01\Public network share.
New-PSDrive -Persist -Name "X" -PSProvider "FileSystem" -Root
"\\Server01\Public"

# Now, you can use the Get-PSDrive drive cmdlet to examine the two drives.
# The drives appear to be the same, although the network share name appears
only
# in the root of the PSDrive: drive.
Get-PSDrive -Name "PSDrive", "X"

Name       Provider      Root
----       --------      ----

PsDrive    FileSystem    \\Server01\public
X          FileSystem    X:\

# Get-Member cmdlet shows that the drives have the same object type,
```

```
# System.Management.Automation.PSDriveInfo.
Get-PSDrive "PSDrive", "x" | Get-Member

TypeName: System.Management.Automation.PSDriveInfo

Name              MemberType   Definition
----              ----------   ----------
CompareTo         Method       System.Int32 CompareTo(PSDriveInfo drive),
Equals            Method       System.Boolean Equals(Object obj),
GetHashCode       Method       System.Int32 GetHashCode()
...

# Net Use and Get-CimInstance for the Win32_LogicalDisk class,
# and Win32_NetworkConnection class find only the persistent X: drive.
# PowerShell temporary drives are known only to PowerShell.
Net Use
Get-CimInstance Win32_LogicalDisk | Format-Table -Property DeviceID
Get-CimInstance Win32_NetworkConnection

Status      Local     Remote                  Network
-------------------------------------------------------
OK          X:        \\contoso-pc\data       Microsoft Windows Network

deviceid
--------
C:
D:
X:

LocalName   RemoteName            ConnectionState         Status
---------   ----------            ---------------         ------
X:          \\products\public     Disconnected            Unavailable
```

-------- Example 6: Create persistent drive in a script --------

New-PSDrive -Persist -Name "X" -PSProvider "FileSystem" -Root

"\\Server01\Public" -Scope Global

To ensure that the drive is available outside of the script you must use the

Scope parameter to create the drive in the Global scope.

REMARKS

To see the examples, type: "get-help New-PSDrive -examples".

For more information, type: "get-help New-PSDrive -detailed".

For technical information, type: "get-help New-PSDrive -full".

For online help, type: "get-help New-PSDrive -online"