



We use optional cookies to improve your experience on our websites, such as through social media connections, and to display personalized advertising based on your online activity. If you reject optional cookies, only cookies necessary to provide you the services will be used. You may change your selection by clicking "Manage Cookies" at the bottom of the page.  
[Privacy Statement](#) [Third-Party Cookies](#)

Accept

Reject

Manage cookies

# Microsoft Ignite

Nov 19–22, 2024

Register now >



Learn

Discover ▾

Product documentation ▾

Development languages ▾

Topics ▾



Sign in



## New-PSDrive

Reference

Feedback

Module: [Microsoft.PowerShell.Management](#)

### In this article

[Syntax](#)

[Description](#)

[Examples](#)

[Parameters](#)

[Show 4 more](#)

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

## Syntax

```
New-PSDrive
    [-Name] <String>
    [-PSProvider] <String>
    [-Root] <String>
    [-Description <String>]
    [-Scope <String>]
    [-Persist]
    [-Credential <PSCredential>]
    [-WhatIf]
    [-Confirm]
    [<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](#).
- 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](#).

#### 📌 Note

Unless you use the **Scope** parameter, PSDrives are created in the scope in which the `New-PSDrive` command is run.

## Examples

### Example 1: Create a temporary drive mapped to a network share

This example creates a temporary PowerShell drive that's 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

This example creates a temporary PowerShell drive that provides access to a directory on the local computer.

```
$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

This example creates a temporary PowerShell drive that provides access to a registry key. It creates a drive named `MyCompany` that is mapped to the `HKLM:\Software\MyCompany` registry key.

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

----	-----	----
MyCompany	Registry	HKLM:\Software\MyCompany

`New-PSDrive` uses the **Name** parameter to specify PowerShell drive named `MyCompany` and the **PSPProvider** 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

This example maps a network drive that's authenticated with a domain service account's credentials. For more information about the **PSCredential** object that stores credentials and how passwords are stored as a **SecureString**, see the **Credential** parameter's description.

```
$cred = Get-Credential -Credential Contoso\ServiceAccount
New-PSDrive -Name "S" -Root "\\Server01\Scripts" -Persist -F
Net Use
```

Status	Local	Remote	Network
-----	-----	-----	-----
OK	S:	\\Server01\Scripts	Microsoft v

 **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

This example shows the difference between a persistent mapped network drive and a temporary PowerShell drive that is mapped to the same network share.

If you close the PowerShell session and then open a new session, the temporary `PSDrive:` isn't available, but the persistent `X:` drive is available. When deciding which method to use to map network drives, consider how you'll use the drive. For example, whether it has to be persistent, and whether the drive has to be visible to other Windows features.

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

# Use the Persist parameter of New-PSDrive to create the X:
```

```
# 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 cmdlet to examine the drives.
# The drives appear to be the same, although the network share is mapped to a drive letter.
# 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)
Equals	Method	System.Boolean Equals(Object)
GetHashCode	Method	System.Int32 GetHashCode()
...		

```
# Net Use and Get-CimInstance for the Win32_LogicalDisk class and Win32_NetworkConnection class find only the persistent drives.
# PowerShell temporary drives are known only to PowerShell.
Net Use
Get-CimInstance Win32_LogicalDisk | Format-Table -Property Name, Provider, Root
Get-CimInstance Win32_NetworkConnection
```

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

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

LocalName	RemoteName	ConnectionState
-----	-----	-----
X:	\\products\public	Disconnected



## Example 6: Create persistent drive in a script

PSDrives are created in the scope in which the `New-PSDrive` command is run. When the command is run within a script, the drive mapping is local to the script. When the script exits, the drive is no longer available.

```
New-PSDrive -Persist -Name "X" -PSProvider "FileSystem" -Root "C:\Program Files\Microsoft Windows Defender\Windows Defender Resources\WinSxS\xml\
```

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.

## Parameters

### -Confirm

Prompts you for confirmation before running the cmdlet.

 Expand table

Type:	SwitchParameter
Aliases:	cf
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

### -Credential

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** object and the password is stored as a **SecureString**.

ⓘ **Note**

For more information about **SecureString** data protection, see [How secure is SecureString?](#).

⌵ Expand table

Type:	<b>PSCredential</b>
Position:	Named
Default value:	Current user
Required:	False
Accept pipeline input:	True
Accept wildcard characters:	False

**-Description**

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`.

 Expand table

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True
Accept wildcard characters:	False

**-Name**

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.

 Expand table

Type:	String
Position:	0
Default value:	None
Required:	True
Accept pipeline input:	True
Accept wildcard characters:	False

**-Persist**

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 **PSPProvider** 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.

 Expand table

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True
Accept wildcard characters:	False


**-PSProvider**

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.

 Expand table

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True
Accept wildcard characters:	False

**-Root**

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.

 Expand table

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True
Accept wildcard characters:	False

**-Scope**

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](#).

 Expand table

Type:	String
Position:	Named
Default value:	Local
Required:	False
Accept pipeline input:	True
Accept wildcard characters:	False

**-WhatIf**

Shows what would happen if the cmdlet runs. The cmdlet isn't run.

 Expand table

Type:	SwitchParameter
Aliases:	wi
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

## Inputs

None

You can't pipe objects to this cmdlet

## Outputs

PSDriveInfo

This cmdlet returns a **PSDriveInfo** object representing the created drive.

## Notes

PowerShell includes the following aliases for `Get-PSDrive`:

- All platforms:
  - `ndr`
- Windows:
  - `mount`

`New-PSDrive` is designed to work with the data exposed by any provider. To list the providers available in your session, use `Get-PSProvider`. For more information about providers, see [about\\_Providers](#).

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.

## Related Links

- [about\\_Providers](#)
- [Get-Credential](#)
- [Get-PSDrive](#)
- [Remove-PSDrive](#)


### Collaborate with us on GitHub


The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see [our contributor guide](#).



### PowerShell feedback

PowerShell is an open source project. Select a link to provide feedback:

 [Open a documentation issue](#)

 [Provide product feedback](#)



