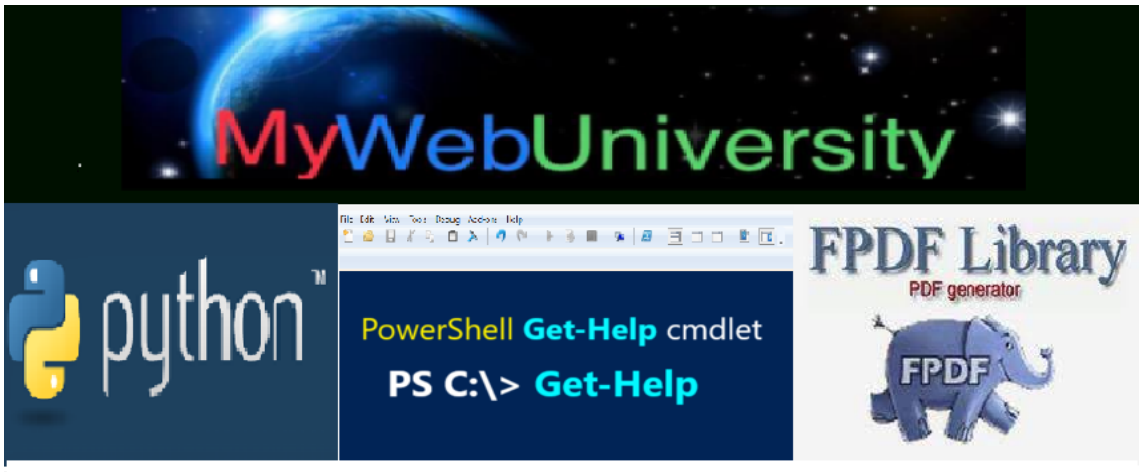

Full credit is given to all the above companies including the Operating System that this PDF file was generated!

Windows PowerShell Get-Help on Cmdlet 'Get-Member'

PS:\>Get-HELP Get-Member -Full

NAME

Get-Member

SYNOPSIS

Gets the properties and methods of objects.

SYNTAX

Get-Member [[-Name] <System.String[]>] [-Force] [-InputObject <System.Management.Automation.PSObject>] [-MemberType {AliasProperty | CodeProperty | Property |

NoteProperty | ScriptProperty | Properties | PropertySet | Method | CodeMethod | ScriptMethod | Methods | ParameterizedProperty | MemberSet | Event | Dynamic | All}]

[-Static] [-View {Extended | Adapted | Base | All}] [<CommonParameters>]

DESCRIPTION

The `Get-Member` cmdlet gets the members, the properties and methods, of objects.

To specify the object, use the InputObject parameter or pipe an object to `Get-Member`. To get information about static members, the members of the class, not of the

instance, use the Static parameter. To get only certain types of members, such as NoteProperties , use the MemberType parameter.

`Get-Member` returns a list of members that's sorted alphabetically. Methods are listed first, followed by the properties.

PARAMETERS

-Force <System.Management.Automation.SwitchParameter>

Adds the intrinsic members and the compiler-generated get_ and set_ methods to the display. The following list describes the properties that are added when you

use the Force parameter:

- `PSBase`: The original properties of the .NET object without extension or adaptation. These are the properties defined for the object class. - `PSAdapted`:

The properties and methods defined in the PowerShell extended type system.

- `PSExtended`: The properties and methods that were added in the `Types.ps1xml` files or using the

`Add-Member` cmdlet. - `PSObject`: The adapter that converts the base object to a PowerShell PSObject object. - `PSTypeNames`: A list of object types that

describe the object, in order of specificity. When formatting the object, PowerShell searches for the types in the `Format.ps1xml` files in the PowerShell

installation directory (`$PSHOME`). It uses the formatting definition for the first type that it finds.

By default, `Get-Member` gets these properties in all views except Base and Adapted , but doesn't display them.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-InputObject <System.Management.Automation.PSObject>

Specifies the object whose members are retrieved.

Using the InputObject parameter isn't the same as piping an object to `Get-Member`. The differences are as follows:

- When you pipe a collection of objects to `Get-Member`, `Get-Member` gets the members of the individual objects in the collection, such as the properties of

each string in an array of strings. - When you use InputObject to submit a collection of objects, `Get-Member` gets the members of the collection, such as the

properties of the array in an array of strings.

Required? false

Position? named

Default value None

Accept pipeline input? True (ByValue)

Accept wildcard characters? false

-MemberType <System.Management.Automation.PSMemberTypes>

Specifies the member type that this cmdlet gets. The default is `All`.

The acceptable values for this parameter are:

- `AliasProperty`

- `CodeProperty`

- `Property`

- `NoteProperty`

- `ScriptProperty`

- `Properties`

- `PropertySet`

- `Method`

- `CodeMethod`

- `ScriptMethod`

- `Methods`

- `ParameterizedProperty`

- `MemberSet`

- `Event`

- `Dynamic`

- `All`

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 MemberType parameter as an array of values or as a comma-separated string of those values. The cmdlet will combine 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.

For information about these values, see PSMemberTypes Enumeration (/dotnet/api/system.management.automation.psmembertypes).

Not all objects have every type of member. If you specify a member type that the object doesn't have, PowerShell returns a null value. To get related types of

members, such as all extended members, use the View parameter. If you use the MemberType parameter with the Static or View parameters, `Get-Member` gets the

members that belong to both sets.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Name <System.String[]>

Specifies the names of one or more properties or methods of the object. `Get-Member` gets only the specified properties and methods.

If you use the Name parameter with the MemberType , View , or Static parameter, `Get-Member` gets only the members that satisfy the criteria of all parameters.

To get a static member by name, use the Static parameter with the Name parameter.

Required? false

Position? 0

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Static <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet gets only the static properties and methods of the object. Static properties and methods are defined on the class of objects, not on

any particular instance of the class.

If you use the Static parameter with the View or Force parameters, the cmdlet ignores those parameters. If you use the Static parameter with the MemberType

parameter, `Get-Member` gets only the members that belong to both sets.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-View <System.Management.Automation.PSMemberViewTypes>

Specifies that this cmdlet gets only particular types properties and methods. Specify one or more of the values. The default is Adapted , Extended .

The acceptable values for this parameter are:

- Base. Gets only the original properties and methods of the .NET object (without extension or adaptation). - Adapted. Gets only the properties and methods

defined in the PowerShell extended type system.

- Extended. Gets only the properties and methods that were added in a `Types.ps1xml` files or by

using the `Add-Member` cmdlet. - All. Gets the members in the Base, Adapted, and Extended views.

The View parameter determines the members retrieved, not just the display of those members.

To get particular member types, such as script properties, use the MemberType parameter. If you use the MemberType and View parameters in the same command,

`Get-Member` gets the members that belong to both sets. If you use the Static and View parameters in the same command, the View parameter is ignored.

Required? false

Position? named

Default value Adapted, Extended

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.PSObject

You can pipe any object to this cmdlet.

OUTPUTS

Microsoft.PowerShell.Commands.MemberDefinition

This cmdlet returns a MemberDefinition for each property or method that its gets.

NOTES

Windows PowerShell includes the following aliases for `Get-Member`:

- `gm`

You can get information about a collection object either using the InputObject parameter or by piping the object, preceded by a comma, to `Get-Member`.

You can use the `$This` automatic variable in script blocks that define the values of new properties and methods. The `$This` variable refers to the instance of

the object to which the properties and methods are being added. For more information about the `$This` variable, see about_Automatic_Variables

(../Microsoft.PowerShell.Core/About/about_Automatic_Variables.md).

If you pass an object representing a type , like a type literal such as `[int]`, `Get-Member` return information about the

`[System.RuntimeType]` type. However,

when you use the Static parameter, `Get-Member` returns the static members of the specific type represented by the `System.RuntimeType` instance.

-------- Example 1: Get the members of process objects --------

```
Get-Service | Get-Member

TypeName: System.ServiceProcess.ServiceController

Name                      MemberType     Definition
----                      ----------     ----------
Name                      AliasProperty  Name = ServiceName
RequiredServices          AliasProperty  RequiredServices = ServicesDependedOn
Disposed                  Event          System.EventHandler Disposed(System.Object, System.EventArgs)
Close                     Method         void Close()
Continue                  Method         void Continue()
CreateObjRef              Method         System.Runtime.Remoting.ObjRef CreateObjRef(type requestedType)
Dispose                   Method         void Dispose(), void IDisposable.Dispose()
Equals                    Method         bool Equals(System.Object obj)
ExecuteCommand            Method         void ExecuteCommand(int command)
GetHashCode               Method         int GetHashCode()
GetLifetimeService        Method         System.Object GetLifetimeService()
GetType                   Method         type GetType()
InitializeLifetimeService Method         System.Object InitializeLifetimeService()
Pause                     Method         void Pause()
Refresh                   Method         void Refresh()
Start                     Method         void Start(), void Start(string[] args)
Stop                      Method         void Stop()
WaitForStatus             Method         void WaitForStatus(System.ServiceProcess.ServiceControllerSt...
CanPauseAndContinue       Property       bool CanPauseAndContinue {get;}
CanShutdown               Property       bool CanShutdown {get;}
CanStop                   Property       bool CanStop {get;}
```

```
Container                Property      System.ComponentModel.IContainer Container {get;}
DependentServices        Property      System.ServiceProcess.ServiceController[] DependentServices {get;}
DisplayName              Property      string DisplayName {get;set;}
MachineName              Property      string MachineName {get;set;}
ServiceHandle            Property      System.Runtime.InteropServices.SafeHandle ServiceHandle {get;}
ServiceName              Property      string ServiceName {get;set;}
ServicesDependedOn       Property      System.ServiceProcess.ServiceController[] ServicesDependedOn {get;}
ServiceType              Property      System.ServiceProcess.ServiceType ServiceType {get;}
Site                     Property      System.ComponentModel.ISite Site {get;set;}
StartType                Property      System.ServiceProcess.ServiceStartMode StartType {get;}
Status                   Property      System.ServiceProcess.ServiceControllerStatus Status {get;}
ToString                 ScriptMethod  System.Object ToString();
```

---------- Example 2: Get members of service objects ----------

```
Get-Service | Get-Member -Force
(Get-Service Schedule).PSBase
```

The `Get-Member` command uses the Force parameter to add the intrinsic members and compiler-generated members of the objects to the display. You can use these properties and methods in the same way that you would use an adapted method of the object. The second command shows how to display the value of the PSBase property of the Schedule service. For more information on intrinsic members, see about_Intrinsic_Members (../Microsoft.PowerShell.Core/About/about_Intrinsic_Members.md)

------ Example 3: Get extended members of service objects ------

```
Get-Service | Get-Member -View Extended

TypeName: System.ServiceProcess.ServiceController

Name            MemberType    Definition
----            ----------    ----------
```

Name AliasProperty Name = ServiceName

RequiredServices AliasProperty RequiredServices = ServicesDependedOn

ToString ScriptMethod System.Object ToString();

The `Get-Member` command uses the View parameter to get only the extended members of the service objects. In this case, the extended member is the Name property,

which is an alias property of the ServiceName property.

---- Example 4: Get script properties of event log objects ----

Get-WinEvent -LogName System -MaxEvents 1 | Get-Member -MemberType NoteProperty

TypeName: System.Diagnostics.Eventing.Reader.EventLogRecord

Name MemberType Definition

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

Message NoteProperty string Message=The machine-default permission settings do not grant Local ...

The MemberType parameter gets only objects with a value of `NoteProperty` for their MemberType property.

The command returns the Message property of the EventLogRecord object.

------- Example 5: Get objects with a specified property -------

$list = "Get-Process", "Get-Service", "Get-Culture", "Get-PSDrive", "Get-ExecutionPolicy"

foreach ($cmdlet in $list) {& $cmdlet | Get-Member -Name MachineName}

TypeName: System.Diagnostics.Process

Name MemberType Definition

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

MachineName Property string MachineName {get;}

TypeName: System.ServiceProcess.ServiceController

```
Name        MemberType Definition
----        ---------- ----------
MachineName Property   string MachineName {get;set;}
```

The results show that only process objects and service objects have a MachineName property.

---------- Example 6: Get members for an array ----------

```
$array = @(1,'hello')
$array | Get-Member

   TypeName: System.Int32

Name        MemberType Definition
----        ---------- ----------
CompareTo   Method     int CompareTo(System.Object value), int CompareTo(int value), int ICompar...
Equals      Method     bool Equals(System.Object obj), bool Equals(int obj), bool IEquatable[int...
GetHashCode Method     int GetHashCode()
GetType     Method     type GetType()
GetTypeCode Method     System.TypeCode GetTypeCode(), System.TypeCode IConvertible.GetTypeCode()
ToBoolean   Method     bool IConvertible.ToBoolean(System.IFormatProvider provider)
ToByte      Method     byte IConvertible.ToByte(System.IFormatProvider provider)
...

   TypeName: System.String

Name               MemberType           Definition
----               ----------           ----------
Clone              Method               System.Object Clone(), System.Object ICloneable.Clone()
CompareTo          Method               int CompareTo(System.Object value), int CompareTo(str...
Contains           Method               bool Contains(string value), bool Contains(string val...
CopyTo             Method               void CopyTo(int sourceIndex, char[] destination, int ...
EndsWith           Method               bool EndsWith(string value), bool EndsWith(string val...
EnumerateRunes     Method               System.Text.StringRuneEnumerator EnumerateRunes()
```

```
Equals              Method              bool Equals(System.Object obj), bool Equals(string va...
GetEnumerator       Method              System.CharEnumerator GetEnumerator(), System.Collect...
GetHashCode         Method              int GetHashCode(), int GetHashCode(System.StringCompa...
...

Get-Member -InputObject $array

TypeName: System.Object[]

Name          MemberType          Definition
----          ----------          ----------
Add           Method              int IList.Add(System.Object value)
Address       Method              System.Object&, System.Private.CoreLib, Version=4.0.0.0, Cu...
Clear         Method              void IList.Clear()
Clone         Method              System.Object Clone(), System.Object ICloneable.Clone()
CompareTo     Method              int IStructuralComparable.CompareTo(System.Object other, Sy...
...
```

The `$array` variable contains an Int32 object and a string object, as seen when the array is piped to `Get-Member`. When `$array` is passed using the InputObject parameter `Get-Member` returns the members of the Object[] type.

--- Example 7: Determine which object properties you can set ---

```
$File = Get-Item c:\test\textFile.txt
$File.PSObject.Properties | Where-Object isSettable | Select-Object -Property Name

Name
----
PSPath
PSParentPath
PSChildName
PSDrive
PSProvider
```

PSIsContainer

IsReadOnly

CreationTime

CreationTimeUtc

LastAccessTime

LastAccessTimeUtc

LastWriteTime

LastWriteTimeUtc

Attributes

Example 8: List the properties of an object in the order they were created

```
$Asset = New-Object -TypeName PSObject
$d = [ordered]@{Name="Server30";System="Server Core";PSVersion="4.0"}
$Asset | Add-Member -NotePropertyMembers $d -TypeName Asset
$Asset.PSObject.Properties | Select-Object Name, Value
```

```
Name      Value
----      -----
Name      Server30
System    Server Core
PSVersion 4.0
```

RELATED LINKS

Online Version: https://learn.microsoft.com/powershell/module/microsoft.powershell.utility/get-member?view=powershell-5.1&WT.mc_id=ps-gethelp

Add-Member