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

## PowerShell Get-Help on command 'Get-WmiObject'

```
PS C:\Users\wahid> Get-Help Get-WmiObject


NAME
    Get-WmiObject


SYNOPSIS
    Gets instances of Windows Management Instrumentation (WMI) classes or
    information about the available classes.



SYNTAX
    Get-WmiObject [[-Class] <System.String>] [[-Property] <System.String[]>]
    [-Amended] [-AsJob] [-Authentication {Default | None | Connect | Call | Packet
    | PacketIntegrity | PacketPrivacy | Unchanged}] [-Authority <System.String>]
    [-ComputerName <System.String[]>] [-Credential
    <System.Management.Automation.PSCredential>] [-DirectRead]
    [-EnableAllPrivileges] [-Filter <System.String>] [-Impersonation {Default |
    Anonymous | Identify | Impersonate | Delegate}] [-Locale <System.String>]
    [-Namespace <System.String>] [-ThrottleLimit <System.Int32>]
    [<CommonParameters>]

    Get-WmiObject [[-Class] <System.String>] [-Amended] [-AsJob] [-Authentication
    {Default | None | Connect | Call | Packet | PacketIntegrity | PacketPrivacy |
    Unchanged}] [-Authority <System.String>] [-ComputerName <System.String[]>]
    [-Credential <System.Management.Automation.PSCredential>]
    [-EnableAllPrivileges] [-Impersonation {Default | Anonymous | Identify |
    Impersonate | Delegate}] [-List] [-Locale <System.String>] [-Namespace
    <System.String>] [-Recurse] [-ThrottleLimit <System.Int32>]
    [<CommonParameters>]

    Get-WmiObject [-Amended] [-AsJob] [-Authentication {Default | None | Connect |
    Call | Packet | PacketIntegrity | PacketPrivacy | Unchanged}] [-Authority
    <System.String>] [-ComputerName <System.String[]>] [-Credential
    <System.Management.Automation.PSCredential>] [-DirectRead]
    [-EnableAllPrivileges] [-Impersonation {Default | Anonymous | Identify |
    Impersonate | Delegate}] [-Locale <System.String>] [-Namespace
    <System.String>] -Query <System.String> [-ThrottleLimit <System.Int32>]
    [<CommonParameters>]
```

## DESCRIPTION

Starting in PowerShell 3.0, this cmdlet has been superseded by `Get-CimInstance`.

The `Get-WmiObject` cmdlet gets instances of WMI classes or information about the available WMI classes. To specify a remote computer, use the ComputerName parameter. If the List parameter is specified, the cmdlet gets information about the WMI classes that are available in a specified namespace. If the Query parameter is specified, the cmdlet runs a WMI query language (WQL) statement.

The `Get-WmiObject` cmdlet does not use Windows PowerShell remoting to perform remote operations. You can use the ComputerName parameter of the `Get-WmiObject` cmdlet even if your computer does not meet the requirements for Windows PowerShell remoting or is not configured for remoting in Windows PowerShell.

Beginning in Windows PowerShell 3.0, the __Server property of the object that `Get-WmiObject` returns has a PSComputerName alias. This makes it easier to include the source computer name in output and reports.

## PARAMETERS

-Amended <System.Management.Automation.SwitchParameter>

Gets or sets a value that indicates whether the objects that are returned from WMI should contain amended information. Typically, amended information is localizable information, such as object and property descriptions, that is attached to the WMI object.

-AsJob <System.Management.Automation.SwitchParameter>

Runs the command as a background job. Use this parameter to run commands that take a long time to finish.

When you use the AsJob parameter, the command returns an object that represents the background job and then displays the command prompt. You can continue to work in the session while the job finishes. If `Get-WmiObject` is used with the ComputerName parameter, the job is created on the local computer, and the results from remote computers are automatically returned to the local computer. To manage the job, use the cmdlets that contain the `Job` noun. To get the job results, use the `Receive-Job` cmdlet.

For more information about Windows PowerShell background jobs, see about_Jobs (../Microsoft.PowerShell.Core/about/about_Jobs.md)and about_Remote_Jobs (../Microsoft.PowerShell.Core/about/about_Remote_Jobs.md).

-Authentication <System.Management.AuthenticationLevel>

Specifies the authentication level to be used with the WMI connection.

Valid values are:

- `-1`: Unchanged - `0`: Default - `1`: None (No authentication in
performed.) - `2`: Connect (Authentication is performed only when the
client establishes a relationship with   the application.) - `3`: Call
(Authentication is performed only at the beginning of each call when the
application   receives the request.) - `4`: Packet (Authentication is
performed on all the data that is received from the client.) - `5`:
PacketIntegrity (All the data that is transferred between the client and
the application   is authenticated and verified.) - `6`: PacketPrivacy
(The properties of the other authentication levels are used, and all the
data is encrypted.)

-Authority <System.String>

Specifies the authority to use to authenticate the WMI connection. You can specify standard NTLM or Kerberos authentication. To use NTLM, set the authority setting to `ntlmdomain:<DomainName>`, where `<DomainName>` identifies a valid NTLM domain name. To use Kerberos, specify `kerberos:<DomainName><ServerName>`. You cannot include the authority setting when you connect to the local computer.

-Class <System.String>

Specifies the name of a WMI class. When this parameter is used, the cmdlet retrieves instances of the WMI class.

-ComputerName <System.String[]>

Specifies the target computer for the management operation. Enter a fully qualified domain name (FQDN), a NetBIOS name, or an IP address. When the remote computer is in a different domain than the local computer, the fully qualified domain name is required.

The default is the local computer. To specify the local computer, such as in a list of computer names, use `localhost`, the local computer name, or a dot (`.`).

When specifying a remote computer, your current account or the one you specify with the Credential parameter must have appropriate permissions to access the information.

This parameter does not rely on Windows PowerShell remoting, which uses WS-Management. You can use the ComputerName parameter of `Get-WmiObject` even if your computer is not configured to run WS-Management remote commands.

-Credential <System.Management.Automation.PSCredential>

Specifies a user account that has permission to perform this action. The default is the current user. Type a user name, such as `User01`, `Domain01\User01`, or `User@Contoso.com`. Or, enter a PSCredential object, such as an object that is returned by the `Get-Credential` cmdlet. When you type a user name, you are prompted for a password. Credentials cannot be used when targeting the local computer.

-DirectRead <System.Management.Automation.SwitchParameter>

Specifies whether direct access to the WMI provider is requested for the specified class without any regard to its base class or to its derived classes.

-EnableAllPrivileges <System.Management.Automation.SwitchParameter>

Enables all the privileges of the current user before the command makes the WMI call.

-Filter <System.String>

Specifies a Where clause to use as a filter. Uses the syntax of the WMI Query Language (WQL).

> [!IMPORTANT] > Do not include the Where keyword in the value of the

parameter. For example, the following > commands return only the logical disks that have a DeviceID of `c:` and services that have the > name 'WinRM' without using the Where keyword.

`Get-WmiObject Win32_LogicalDisk -filter "DeviceID = 'c:' "`

`Get-WmiObject win32_service -filter "name='WinRM'"`

-Impersonation <System.Management.ImpersonationLevel>

Specifies the impersonation level to use.

The acceptable values for this parameter are:

- `0`: Default . Reads the local registry for the default impersonation level. The default is usually set to Impersonate . - `1`: Anonymous . Hides the credentials of the caller. - `2`: Identify . Allows objects to query the credentials of the caller. - `3`: Impersonate . Allows objects to use the credentials of the caller. - `4`: Delegate . Allows objects to permit other objects to use the credentials of the caller.

-List <System.Management.Automation.SwitchParameter>

Gets the names of the WMI classes in the WMI repository namespace that is specified by the Namespace parameter.

If you specify the List parameter, but not the Namespace parameter, `Get-WmiObject` uses the Root\Cimv2 namespace by default. This cmdlet does not use the Default Namespace registry entry in the `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WBEM\Scripting` registry key to determine the default namespace.

-Locale <System.String>

Specifies the preferred locale for WMI objects. Enter a value in `MS_<LCID>` format.

-Namespace <System.String>

When used with the Class parameter, the Namespace parameter specifies the WMI repository namespace where the specified WMI class is located. When used with the List parameter, it specifies the namespace from which to gather WMI class information.

-Property <System.String[]>

Specifies the WMI class properties that this cmdlet gets information from. Enter the property names.

-Query <System.String>

Runs the specified WMI Query Language (WQL) statement. This parameter does not support event queries.

-Recurse <System.Management.Automation.SwitchParameter>

Searches the current namespace and all other namespaces for the class name that is specified by the Class parameter.

-ThrottleLimit <System.Int32>

Specifies the maximum number of WMI operations that can be executed simultaneously. This parameter is valid only when the AsJob parameter is used in the command.

<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: Get processes on the local computer --------

```
Get-WmiObject -Class Win32_Process
```

-------- Example 2: Gets services on a remote computer --------

```
Get-WmiObject -Class Win32_Service -ComputerName 10.1.4.62
```

Example 3: Get WMI classes in the root or default namespace of the local computer

```
Get-WmiObject -Namespace "root/default" -List
```

----- Example 4: Get a named service on multiple computers -----

```
Get-WmiObject -Query "select * from win32_service where name='WinRM'"
-ComputerName Server01, Server02 |
  Format-List -Property PSComputerName, Name, ExitCode, Name, ProcessID,
StartMode, State, Status

PSComputerName : SERVER01
Name           : WinRM
ExitCode       : 0
Name           : WinRM
ProcessID      : 844
StartMode      : Auto
State          : Running
Status         : OK

PSComputerName : SERVER02
Name           : WinRM
ExitCode       : 0
Name           : WinRM
```

```
ProcessID    : 932
StartMode    : Auto
State        : Running
Status       : OK
```

A pipeline operator (`|`) sends the output to the `Format-List` cmdlet, which adds the PSComputerName property to the default output. PSComputerName is an alias of the __Server property of the objects that `Get-WmiObject` returns. This alias was introduced in PowerShell 3.0.

-------- Example 5: Stop a service on a remote computer --------

```
(Get-WmiObject -Class Win32_Service -Filter "name='WinRM'" -ComputerName
Server01).StopService()
```

This is equivalent to using the `Stop-Service` cmdlet.

-------- Example 6: Get the BIOS on the local computer --------

```
Get-WmiObject -Class Win32_Bios | Format-List -Property *
```

```
Status                : OK
Name                  : Phoenix ROM BIOS PLUS Version 1.10 A05
Caption               : Phoenix ROM BIOS PLUS Version 1.10 A05
SMBIOSPresent         : True
__GENUS               : 2
__CLASS               : Win32_BIOS
__SUPERCLASS          : CIM_BIOSElement
__DYNASTY             : CIM_ManagedSystemElement
__RELPATH             : Win32_BIOS.Name="Phoenix ROM BIOS PLUS Version 1.10
__PROPERTY_COUNT      : 27
__DERIVATION          : {CIM_BIOSElement, CIM_SoftwareElement,
CIM_LogicalElement,
__SERVER              : Server01
__NAMESPACE           : root\cimv2
```

```
__PATH              : \\SERVER01\root\cimv2:Win32_BIOS.Name="Phoenix ROM BIOS
BiosCharacteristics  : {7, 9, 10, 11...}
BIOSVersion          : {DELL   - 15, Phoenix ROM BIOS PLUS Version 1.10 A05}
BuildNumber          :
CodeSet              :
CurrentLanguage      : en|US|iso8859-1
Description          : Phoenix ROM BIOS PLUS Version 1.10 A05
IdentificationCode   :
InstallableLanguages  : 1
InstallDate          :
LanguageEdition      :
ListOfLanguages      : {en|US|iso8859-1}
Manufacturer         : Dell Inc.
OtherTargetOS        :
PrimaryBIOS          : True
ReleaseDate          : 20101103000000.000000+000
SerialNumber         : 8VDM9P1
SMBIOSBIOSVersion    : A05
SMBIOSMajorVersion   : 2
SMBIOSMinorVersion   : 6
SoftwareElementID    : Phoenix ROM BIOS PLUS Version 1.10 A05
SoftwareElementState  : 3
TargetOperatingSystem : 0
Version              : DELL   - 15
Scope                : System.Management.ManagementScope
Path                 : \\SERVER01\root\cimv2:Win32_BIOS.Name="Phoenix ROM BIOS
Options              : System.Management.ObjectGetOptions
ClassPath            : \\JUNE-PC\root\cimv2:Win32_BIOS
Properties           : {BiosCharacteristics, BIOSVersion, BuildNumber,
Caption...}
SystemProperties     : {__GENUS, __CLASS, __SUPERCLASS, __DYNASTY...}
Qualifiers           : {dynamic, Locale, provider, UUID}
Site                 :
```

```
Container            :
```

------- Example 7: Get the services on a remote computer -------

```
Get-WmiObject Win32_Service -Credential FABRIKAM\administrator -ComputerName
Fabrikam
```

> [!NOTE] > Credentials cannot be used when targeting the local computer.

REMARKS

To see the examples, type: "get-help Get-WmiObject -examples".

For more information, type: "get-help Get-WmiObject -detailed".

For technical information, type: "get-help Get-WmiObject -full".

For online help, type: "get-help Get-WmiObject -online"