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

PowerShell Get-Help on command 'Invoke-WmiMethod'

PS C:\Users\wahid> Get-Help Invoke-WmiMethod

NAME

Invoke-WmiMethod

SYNOPSIS

Calls WMI methods.

SYNTAX

Invoke-WmiMethod [-Class] <System.String> [-Name] <System.String>
[-ArgumentList <System.Object[]>] [-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}]
[-Locale <System.String>] [-Namespace <System.String>] [-ThrottleLimit
<System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-WmiMethod [-Name] <System.String> [-ArgumentList <System.Object[]>]
[-AsJob] -InputObject <System.Management.ManagementObject> [-ThrottleLimit
<System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-WmiMethod [-Name] <System.String> [-ArgumentList <System.Object[]>] [-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}] [-Locale <System.String>] [-Namespace <System.String>] -Path <System.String> [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-WmiMethod [-Name] <System.String> [-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}] [-Locale <System.String>] [-Namespace <System.String>] [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-WmiMethod [-Name] <System.String> [-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}] [-Locale <System.String>] [-Namespace <System.String>] [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-WmiMethod [-Name] <System.String> [-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}] [-Locale <System.String>] [-Namespace <System.String>] [-ThrottleLimit <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Invoke-WmiMethod` cmdlet calls the methods of Windows Management Instrumentation (WMI) objects.

New Common Information Model (CIM) cmdlets, introduced in Windows PowerShell 3.0, perform the same tasks as the WMI cmdlets. The CIM cmdlets comply with WS-Management (WSMan) standards and with the CIM standard, which enables the cmdlets to use the same techniques to manage Windows computers and those running other operating systems. Instead of using `Invoke-WmiMethod`, consider using Invoke-CimMethod (../cimcmdlets/invoke-cimmethod.md).

PARAMETERS

-ArgumentList <System.Object[]>

Specifies the parameters to pass to the called method. The value of this parameter must be an array of objects, and they must appear in the order required by the called method. The `Invoke-CimCommand` cmdlet does not have these limitations.

To determine the order in which to list those objects, run the `GetMethodParameters()` method on the WMI class, as illustrated in Example 1, near the end of this topic.

> [!IMPORTANT] > If the first value is an array that contains more than one element, a second value of `$null` is > required. Otherwise, the command generates an error, such as > `Unable to cast object of type 'System.Byte' to type 'System.Array'.`. See example 4 above.

-AsJob <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet 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 `Invoke-WmiMethod` is used against a remote computer, 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 (the Job cmdlets). To get the job results, use the `Receive-Job` cmdlet.

To use this parameter with remote computers, the local and remote computers must be configured for remoting. Additionally, you must start Windows PowerShell by using the Run as administrator option in Windows Vista and later versions of Windows. For more information, see about_Remote_Requirements (../Microsoft.PowerShell.Core/About/about_Remote_Requirements.md).

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. The acceptable values for this parameter 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 Windows NT LAN Manager (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 WMI class that contains a static method to call.

-ComputerName <System.String[]>
Specifies, as a string array, the computers that this cmdlet runs the
command on. The default is the local computer.

Type the NetBIOS name, an IP address, or a fully qualified domain name of
one or more computers. To specify the local computer, type the computer
name, a dot (`.`), or `localhost`.

This parameter does not rely on Windows PowerShell remoting. You can use
the ComputerName parameter even if your computer is not configured to run
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 will be prompted for a password.

-EnableAllPrivileges <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet enables all the privileges of the current user before the command makes the WMI call.

-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, which is usually set to `3`: 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.)

-InputObject <System.Management.ManagementObject>

Specifies a ManagementObject object to use as input. When this parameter is used, all other parameters except the Flag and Argument parameters are ignored.

-Locale <System.String>

Specifies the preferred locale for WMI objects. Specify the value of the Locale parameter as an array in the `MS_<LCID>` format in the preferred order.

-Name <System.String>

Specifies the name of the method to be invoked. This parameter is mandatory and cannot be null or empty.

-Namespace <System.String>

When used with the Class parameter, this parameter specifies the WMI repository namespace where the referenced WMI class or object is located.

-Path <System.String>

Specifies the WMI object path of a WMI class, or specifies the WMI object path of an instance of a WMI class. The class or the instance that you specify must contain the method that is specified in the Name parameter.

-ThrottleLimit <System.Int32>

Specifies a throttle value for the number of WMI operations that can be executed simultaneously. This parameter is used together with the AsJob parameter. The throttle limit applies only to the current command, not to the session or to the computer.

-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 is not 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: List the required order of WMI method parameters -

```
Get-WmiObject Win32_Volume |
    Get-Member -MemberType Method -Name Format |
    Select-Object -ExpandProperty Definition

System.Management.ManagementBaseObject Format(System.String FileSystem,
System.Boolean QuickFormat,
 System.UInt32 ClusterSize, System.String Label, System.Boolean
EnableCompression,
```

System.UInt32 Version)

To invoke WMI in PowerShell 3.0 differs from alternate methods, and requires
that object values are entered in a specific order.
-------- Example 2: Start an instance of an application --------

([Wmiclass]'Win32_Process').Create.OverloadDefinitions

System.Management.ManagementBaseObject Create(System.String CommandLine,
System.String CurrentDirectory,
System.Management.ManagementObject#Win32_ProcessStartup
ProcessStartupInformation)

Invoke-WmiMethod -Path Win32_Process -Name Create -ArgumentList
C:\Windows\system32\notepad.exe

__GENUS : 2
__CLASS : __PARAMETERS
__SUPERCLASS :
__DYNASTY : __PARAMETERS
__RELPATH :
__PROPERTY_COUNT : 2
__DERIVATION : {}
__SERVER :
__NAMESPACE :
__PATH :
ProcessId : 11312
ReturnValue : 0
PSComputerName :

This command starts an instance of Notepad by calling the `Create` method of
the Win32_Process class.

The ReturnValue property is populated with a `0`, and the ProcessId property is populated with an integer (the next process ID number) if the command is completed.

---------- Example 3: Rename a file ----------

```
$invokeWmiMethodSplat = @{
    Path = "CIM_DataFile.Name='C:\scripts\test.txt'"
    Name = 'Rename'
    ArgumentList = 'C:\scripts\test_bu.txt'
}
Invoke-WmiMethod @invokeWmiMethodSplat
```

```
__GENUS          : 2
__CLASS          : __PARAMETERS
__SUPERCLASS     :
__DYNASTY        : __PARAMETERS
__RELPATH        :
__PROPERTY_COUNT : 1
__DERIVATION     : {}
__SERVER         :
__NAMESPACE      :
__PATH           :
ReturnValue      : 0
```

This command renames a file. It uses the Path parameter to reference an instance of the CIM_DataFile class. Then, it applies the Rename method to that particular instance.

The ReturnValue property is populated with a `0` if the command is completed.

- Example 4: Passing an array of values using `-ArgumentList` -

```
$acl = Get-Acl test.txt
$binSD = $acl.GetSecurityDescriptorBinaryForm()
```

```
    $invokeWmiMethodSplat = @{
        Class = 'Win32_SecurityDescriptorHelper'
        Name = 'BinarySDToSDDL'
        ArgumentList = $binSD, $null
    }
    Invoke-WmiMethod @invokeWmiMethodSplat
```

REMARKS

To see the examples, type: "get-help Invoke-WmiMethod -examples".

For more information, type: "get-help Invoke-WmiMethod -detailed".

For technical information, type: "get-help Invoke-WmiMethod -full".

For online help, type: "get-help Invoke-WmiMethod -online"