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

PowerShell Get-Help on command 'Invoke-CimMethod'

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

NAME

Invoke-CimMethod

SYNOPSIS

Invokes a method of a CIM class.

SYNTAX

Invoke-CimMethod [-CimClass] <Microsoft.Management.Infrastructure.CimClass>
[[-Arguments] <System.Collections.IDictionary>] [-MethodName] <System.String>
[-ComputerName <System.String[]>] [-OperationTimeoutSec <System.UInt32>]
[-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-CimMethod [-CimClass] <Microsoft.Management.Infrastructure.CimClass>
[[-Arguments] <System.Collections.IDictionary>] [-MethodName] <System.String>
-CimSession <Microsoft.Management.Infrastructure.CimSession[]>
[-OperationTimeoutSec <System.UInt32>] [-Confirm] [-WhatIf]
[<CommonParameters>]

Invoke-CimMethod [-ClassName] <System.String> [[-Arguments]
<System.Collections.IDictionary>] [-MethodName] <System.String> -CimSession

<Microsoft.Management.Infrastructure.CimSession[]> [-Namespace
<System.String>] [-OperationTimeoutSec <System.UInt32>] [-Confirm] [-WhatIf]
[<CommonParameters>]

Invoke-CimMethod [-InputObject]
<Microsoft.Management.Infrastructure.CimInstance> [[-Arguments]
<System.Collections.IDictionary>] [-MethodName] <System.String> -CimSession
<Microsoft.Management.Infrastructure.CimSession[]> [-OperationTimeoutSec
<System.UInt32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-CimMethod [[-Arguments] <System.Collections.IDictionary>] [-MethodName]
<System.String> -CimSession <Microsoft.Management.Infrastructure.CimSession[]>
[-Namespace <System.String>] [-OperationTimeoutSec <System.UInt32>] -Query
<System.String> [-QueryDialect <System.String>] [-Confirm] [-WhatIf]
[<CommonParameters>]

Invoke-CimMethod [[-Arguments] <System.Collections.IDictionary>] [-MethodName]
<System.String> -CimSession <Microsoft.Management.Infrastructure.CimSession[]>
[-Namespace <System.String>] [-OperationTimeoutSec <System.UInt32>]
[-ResourceUri <System.Uri>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-CimMethod [-ClassName] <System.String> [[-Arguments]
<System.Collections.IDictionary>] [-MethodName] <System.String> [-ComputerName
<System.String[]>] [-Namespace <System.String>] [-OperationTimeoutSec
<System.UInt32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-CimMethod [[-Arguments] <System.Collections.IDictionary>] [-MethodName]
<System.String> [-ComputerName <System.String[]>] [-Namespace <System.String>]
[-OperationTimeoutSec <System.UInt32>] [-ResourceUri <System.Uri>] [-Confirm]
[-WhatIf] [<CommonParameters>]

Invoke-CimMethod [-InputObject]
<Microsoft.Management.Infrastructure.CimInstance> [[-Arguments]

<System.Collections.IDictionary>] [-MethodName] <System.String> [-ComputerName <System.String[]>] [-OperationTimeoutSec <System.UInt32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Invoke-CimMethod [[-Arguments] <System.Collections.IDictionary>] [-MethodName] <System.String> [-ComputerName <System.String[]>] [-Namespace <System.String>] [-OperationTimeoutSec <System.UInt32>] -Query <System.String> [-QueryDialect <System.String>] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Invoke-CimMethod` cmdlet invokes a method of a CIM class or CIM instance using the name-value pairs specified by the Arguments parameter.

If the InputObject parameter is not specified, the cmdlet works in one of the following ways:

- If neither the ComputerName parameter nor the CimSession parameter is specified, then this cmdlet works on local Windows Management Instrumentation (WMI) using a Component Object Model (COM) session. - If either the ComputerName parameter or the CimSession parameter is specified, then this cmdlet works against the CIM server specified by either the ComputerName parameter or the CimSession parameter.

If the InputObject parameter is specified, the cmdlet works in one of the following ways:

- If neither the ComputerName parameter nor the CimSession parameter is specified, then this cmdlet uses the CIM session or computer name from the input object. - If the either the ComputerName parameter or the CimSession parameter is specified, then this cmdlet uses the either the CimSession parameter value or ComputerName parameter value. This is not a common scenario.

PARAMETERS

-Arguments <System.Collections.IDictionary>

Specifies the parameters to pass to the called method. Specify the values for this parameter as name-value pairs, stored in a hash table. The order of the values entered isn't important.

-CimClass <Microsoft.Management.Infrastructure.CimClass>

Specifies a CIM class object that represents a CIM class definition on the server. Use this parameter when invoking a static method of a class.

You can use the `Get-CimClass` cmdlet to retrieve a class definition from the server.

Using this parameter results in better client side schema validations.

-CimSession <Microsoft.Management.Infrastructure.CimSession[]>

Runs the command using the specified CIM session. Enter a variable that contains the CIM session, or a command that creates or gets the CIM session, such as the `New-CimSession` or `Get-CimSession` cmdlets. For more information, see about_CimSession (../Microsoft.PowerShell.Core/About/about_CimSession.md).

-ClassName <System.String>

Specifies the name of the CIM class for which to perform the operation. This parameter is only used for static methods. You can use tab completion to browse the list of classes, because PowerShell gets a list of classes from the local WMI server to provide a list of class names.

-ComputerName <System.String[]>

Specifies the name of the computer on which you want to run the CIM operation. You can specify a fully qualified domain name (FQDN), a NetBIOS

name, or an IP address.

When using this parameter, the cmdlet creates a temporary session to the specified computer using the WsMan protocol. Otherwise, the cmdlet performs the operation on the local computer using Component Object Model (COM).

Connect using a CIM session for better performance when multiple operations are being performed on the same computer.

-InputObject <Microsoft.Management.Infrastructure.CimInstance>

Specifies a CIM instance object to use as input to invoke a method. This parameter can only be used to invoke instance methods. To invoke class static methods, use the Class parameter or the CimClass parameter.

-MethodName <System.String>

Specifies the name of the CIM method to invoke. This parameter is mandatory and cannot be null or empty. To invoke static method of a CIM class use the ClassName or the CimClass parameter.

-Namespace <System.String>

Specifies the namespace for the CIM operation. The default namespace is root/cimv2 . You can use tab completion to browse the list of namespaces, because PowerShell gets a list of namespaces from the local WMI server to provide the list of namespaces.

-OperationTimeoutSec <System.UInt32>

Specifies the amount of time that the cmdlet waits for a response from the computer. By default, the value is 0, which means that the cmdlet uses the default timeout value for the server.

If the OperationTimeoutSec parameter is set to a value less than the default connection retry timeout of 3 minutes, network failures that last

more than the value of the OperationTimeoutSec parameter are not recoverable.

-Query <System.String>

Specifies a query to run on the CIM server. A method is invoked on the instances received as a result of the query. You can specify the query dialect using the QueryDialect parameter.

If the value specified contains double quotes (`"`), single quotes (`'`), or a backslash (``), you must escape those characters by prefixing them with the backslash (``) character. If the value specified uses the WQL LIKE operator, then you must escape the following characters by enclosing them in square brackets (`[]`): percent (`%`), underscore (`_`), or opening square bracket (`[`).

-QueryDialect <System.String>

Specifies the query language used for the Query parameter. The acceptable values for this parameter are: WQL or CQL .

The default value is WQL .

-ResourceUri <System.Uri>

Specifies the resource uniform resource identifier (URI) of the resource class or instance. The URI is used to identify a specific type of resource, such as disks or processes, on a computer.

A URI consists of a prefix and a path to a resource. For example:

`http://schemas.microsoft.com/wbem/wsman/1/wmi/root/cimv2/Win32_LogicalDisk`

`http://intel.com/wbem/wscim/1/amt-schema/1/AMT_GeneralSettings`

By default, if you do not specify this parameter, the DMTF standard resource URI `http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/` is used and the class name is appended to it. ResourceURI can only be used with CIM sessions created using the WSMan protocol, or when specifying the ComputerName parameter, which creates a CIM session using WSMan.

When you specify this parameter without specifying the ComputerName parameter, or when you specify a CIM session created using DCOM protocol, you get an error. The DCOM protocol does not support the ResourceURI parameter.

If both the ResourceUri parameter and the Filter parameter are specified, the Filter parameter is ignored.

-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: Invoke a method ----------

Invoke-CimMethod -Query 'select * from Win32_Process where name like "notepad%"' -MethodName "Terminate"

----- Example 2: Invoke a method using CIM instance object -----

```
$x = Get-CimInstance -Query 'Select * from Win32_Process where name like
"notepad%"'
Invoke-CimMethod -InputObject $x -MethodName GetOwner



------ Example 3: Invoke a static method using arguments ------

Invoke-CimMethod -ClassName Win32_Process -MethodName "Create" -Arguments @{
  CommandLine = 'notepad.exe'; CurrentDirectory = "C:\windows\system32"
}



---------- Example 4: Client-side validation ----------

$c = Get-CimClass -ClassName Win32_Process
Invoke-CimMethod -CimClass $c -MethodName "xyz" -Arguments @{ CommandLine =
'notepad.exe' }
```

REMARKS

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

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

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

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