WMI in PowerShell 3.0

Finding Namespaces and Classes in WMI

New CIM Cmdlets shipping in PowerShell 3.0 have made it easier to discover WMI namespaces and classes.

```
Using Tab completion for CIM Cmdlet Parameters ( Tab+Space in ISE shows a drop down)
Get-CimInstance –Namespace <Tab>  #Finding top level namespaces
#Tab completion for class names
#If namespace is not specified, shows classes from default root/cimv2 namespace
Get-CimInstance –ClassName *Bios<Tab>
Get-CimInstance –Namespace root/Microsoft/Windows/smb –ClassName <tab>
Note: Tab completion only works for local machine.

Using Get-CimClass for advanced class search
Get-CimClass #All classes in root/cimv2
Get-CimClass -MethodName Stop* #Find classes that have a method like Stop*
Get-CimClass -PropertyName Handle #Find classes that have a property names handle
Get-CimClass -ClassName *Partition -QualifierName Association #Find Association classes
Get-CimClass -Namespace root/Microsoft/Windows/smb -class *Smb* -QualifierName Indication

Note: Get-CimClass only works for computers that support Schema retrieval operations (GetClass and
EnumerateClasses). WMI supports these operations for a rich client experience.
```

Getting data from WMI

```
Get-CimInstance -ClassName Win32_Service  #Find instances of Win32_Service class
#Output of Get-CimInstance is Microsoft.Management.Infrastructure.CimInstance#<ClassName>
#Getting data through a WQL Query
Get-CimInstance -Query "Select * from Win32_Service Where Name like 'app%'"
#Get only a subset of Properties – typically used to reduce network/memory footprint
Get-CimInstance -ClassName Win32_Service -KeyOnly
Get-CimInstance -ClassName Win32_Service -Property Name, Status
#A CimInstance is a snapshot of the object state from server on client.
$a = Get-CimInstance -ClassName Win32_Process
Get-CimInstance -InputObject $a[0] #Note object passed as input object is not changed

#If you have scripts using WMI cmdlets, you will find it easy to migrate them to new CIM Cmdlets
```

Peeping into CimInstance

CimInstance class has the following properties

- .CimInstanceProperties - List of properties of this class.
- .CimClass - Schema provided by CIM for this class*.
- .CimClass.CimClassMethods - Methods supported by this class.
- .CimSystemProperties - System properties like namespace.

```
Note: *For Cim Schema to be accurate, CIM Server must support class schema retrieval operations.
```

CimInstance is portable - supports full serialization and deserialization

```
Get-CimInstance Win32_Service -Filter 'Name Like "app%"'|export-clixml t1.xml
$x = import-clixml .\t1.xml
$x[0].pstypenames
diff ($x) (Get-CimInstance win32_service -Filter 'Name Like "app%"')
```

Working with Associations

```
# Get instance of Win32_LogicalDisk class with DriveType==3 (hard drives)
$disk1, $diskn = Get-CimInstance -class Win32_LogicalDisk -Filter 'DriveType = 3'
# Get the associated instancec disk1
Get-CimAssociatedInstance -CimInstance $disk1
# Given an instance of Win32_LogicalDisk give the associated instances of specific type
Get-CimAssociatedInstance -CimInstance $disk1 -ResultClassName Win32_DiskPartition


$service = Get-CimInstance Win32_Service -Filter 'Name Like "winrm%"'
#Find Services that WinRM service depends on
Get-CimAssociatedInstance -InputObject $service -Association Win32_DependentService
```

What is CIM/WMI?

CIM: Common Information Model (CIM) is the DMTF standard [DSP0004] for describing the structure and behavior of the managed resources such as storage, network, or software components.

WMI: Windows Management Instrumentation (WMI) is a CIM server that implements the CIM standard on Windows.

What is WS-Man/WinRM?

WS-Man: WS-Management (WS-Man) protocol is a SOAP-based, firewall-friendly protocol for the management clients to communicate with CIM severs.

WinRM: Windows Remote Management (WinRM) is the Microsoft implementation of the WS-Man protocol on Windows.

What is WQL?

The WMI Query Language (WQL) is used by management clients to query for data from WMI.

WQL is very similar, but not identical, to the CIM Query Language (CQL) defined by the DMTF.

What are new CIM Cmdlets?

PowerShell 2.0 shipped with WMI and WsMan cmdlets. Why to have another set of cmdlets in 3.0?

WMI cmdlets (like Get-WmiObject) work over DCOM and work with WMI/Windows only.

WsMan cmdlets (like Get-WsManInstance) work over WS-Man protocol, but they are not IT Pro friendly.

New Cim cmdlets provide best of both worlds

- Rich PowerShell experience, no more XML
- Work over both WsMan (remote default) and DCOM (local default)
- Work with non-Windows devices that implement WS-Man protocol
- Simplify discovery of namespace of classes in WMI.

Old WMI and WsMan Cmdlets are still supported in Win8. It is easy to change scripts to new standard-based CIM cmdlets.

```
#Get a list of CIM cmdlets
Get-Command –Module CimCmdlets
```

What is an Association

An association represents a relationship between two or more instances of managed resources like disk and volumes or directories and files. Given an instance of a class, a CIM server returns all instances related to the given instance. You can also filter the results by specifying a target class or the name of the association relationship.

Invoking a CIM Method

```
#Finding method of a Class
$c = Get-CimClass Win32_Process
$c.CimClassMethods #You can also use .CimClass property of a CimInstance
#Invoking a method on an instance
$a = Get-CimInstance Win32_Process -Filter "Name Like 'PowerShell%'"
$a | Invoke-CimMethod -MethodName GetOwner #$a binds to InputObject parameter
#Invoke a class static method - icim is the alias for Invoke-CimMethod
icim -ClassName Win32_Process -MethodName Create -Arguments @{CommandLine="calc.exe"}
```

Performing CIM Operations

```
#Creating an instance. CIM Provider should support CreateInstance intrinsic method
New-CimInstance -Class Win32_Environment -Property @{Name="testvar"; VariableValue="testvalue";
UserName="fareast\osajid"}

#Modifying an instance. CIM Provider should support ModifyInstance intrinsic method
$a = Get-CimInstance -Class Win32_Environment -Filter "Name='testvar'" #; VariableValue="testvalue";
UserName="fareast\osajid"}
Set-CimInstance -InputObject $a -Property @{VariableValue="ChangedValue"} -PassThru

#Same result can be achieved through setting the VariableValue property of $a
$a.VariableValue="ChangedValue" #To update the object on the server, call Set-CimInstance next
Set-CimInstance -InputObject $a -PassThru

#Removing an instance. CIM Provider should support RemoveInstance intrinsic method
Remove-CimInstance -InputObject $a
```

Events – CIM Indications

```
$filter = "SELECT * FROM CIM_InstModification WHERE TargetInstance ISA 'Win32_LocalTime'"
# Subscribe for events using the filter
Register-CimIndicationEvent -Query $filter  -SourceIdentifier "Timer"
# Get the events using the PowerShell eventing mechanism
Get-Event -SourceIdentifier Timer
Unregister-Event -SourceIdentifier "Timer"

#Subscribe for the event
$Action = {$process = $Event.SourceEventArgs.NewEvent;write-host New process Name = $process.ProcessName
Id = $process.ProcessId }
Register-CimIndicationEvent -ClassName Win32_ProcessStartTrace   -Action $Action -SourceIdentifier
"ProcessWatch"
Unregister-Event -SourceIdentifier "ProcessWatch"
```

Working with remote servers

```
#CIM Cmdlets have -ComputerName and -CimSession parameters for managing remote servers
Get-CimInstance Win32_Service -ComputerName Server1
#By default WsMan protocol is used when a ComputerName is passed (including localhost or 127.0.0.1)
#If multiple operations are performed against the same server, creating a CimSession is recommended.
$s = New-CimSession -CN server1
gcim Win32_Service -CimSession $s

#Managing down level Windows servers
#There are two ways to manage down level Windows Servers
#  Install Windows Management Framework 3.0 (recommended)
#  OR use DCOM protocol
$so = New-CimSessionOption -Protocol DCOM
$s = New-CimSession -CN server1 -SessionOption $so
Get-CimInstance Win32_Service -CimSession $s


#PSComputerName property of CimInstance shows the source computer name
gcim Win32_Process -CN server1,server2 | Select Name, PsComputerName

#If a CN or CimSession was passed to fetch a CimInstance, it does not have to be specified again for
subsequent operations.
gcim Win32_Process -CN server1,server2 | icim -MethodName GetOwner
```

What are various CIM Operations?

CIM classes should implement methods explicitly defined in their specifications (called extrinsic) and a set of standard predefined methods. The predefined methods are called intrinsic, and they are

- Enumerate instances of a class
- Enumerate associated instances
- Get instances by executing a query on server.
- Get a specific instance of a class
- Create a new instance of class
- Modify an instance of a class
- Delete instance of a class
- Invoke extrinsic method on a class or instance
- Enumerate Classes in a namespace
- Get class schema
- Subscribe for indications
- Unsubscribe from indications

You will notice that CIM cmdlets are modeled on CIM Operations.

What is a CIM Indication?

CIM Indication is a representation of an event in the managed system. A CIM client can subscribe for receiving indications by providing the indication type and the filtering expression, which selects events that will be delivered to the client.

What is a CimSession

A CimSession represents connection to a CIM Server. There is no physical permanent connection established with the server, so a CimSession is a very lightweight client side connection object. CimSession can be used to manage any server that supports WsMan protocol.

Creating CIM-based cmdlets

Developers and advanced IT Pros can use CDXML to wrap existing CIM classes to provide a more PS friendly task abstraction.
See **http://go.microsoft.com/fwlink/?LinkId=252460** for details

Developers can create cmdlets in native code by implementing a CIM class and writing CDXML to go with it.

More Information

WMI Blog : http://blogs.msdn.com/b/wmi/
PowerShell Blog : http://blogs.msdn.com/b/powershell/
Script Center : http://technet.microsoft.com/en-us/scriptcenter/bb410849
Scripting Guys : http://blogs.technet.com/b/heyscriptingguy/