Microsoft Azure

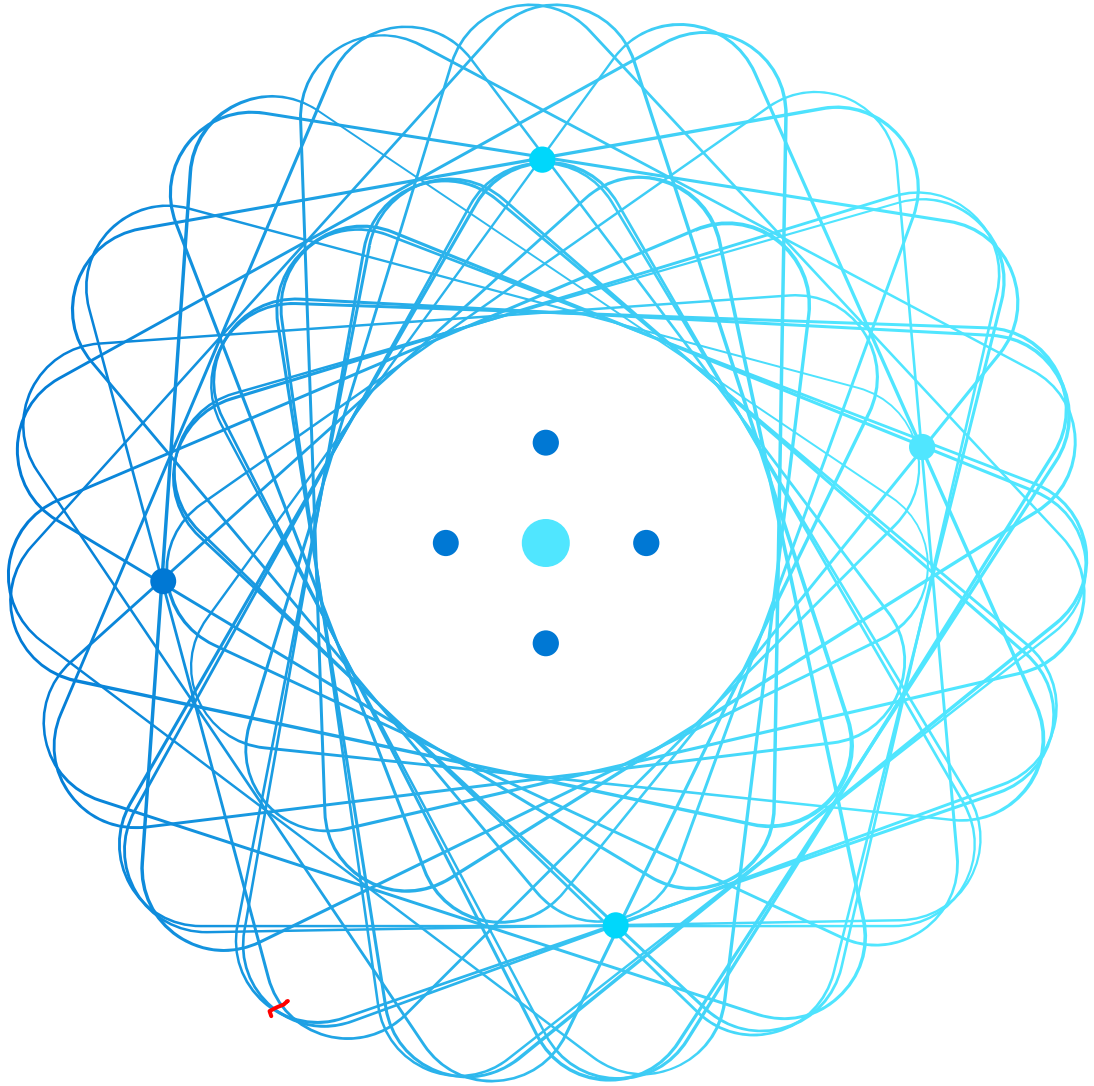
>_

10961

AZ-040 Automating Administration with PowerShell

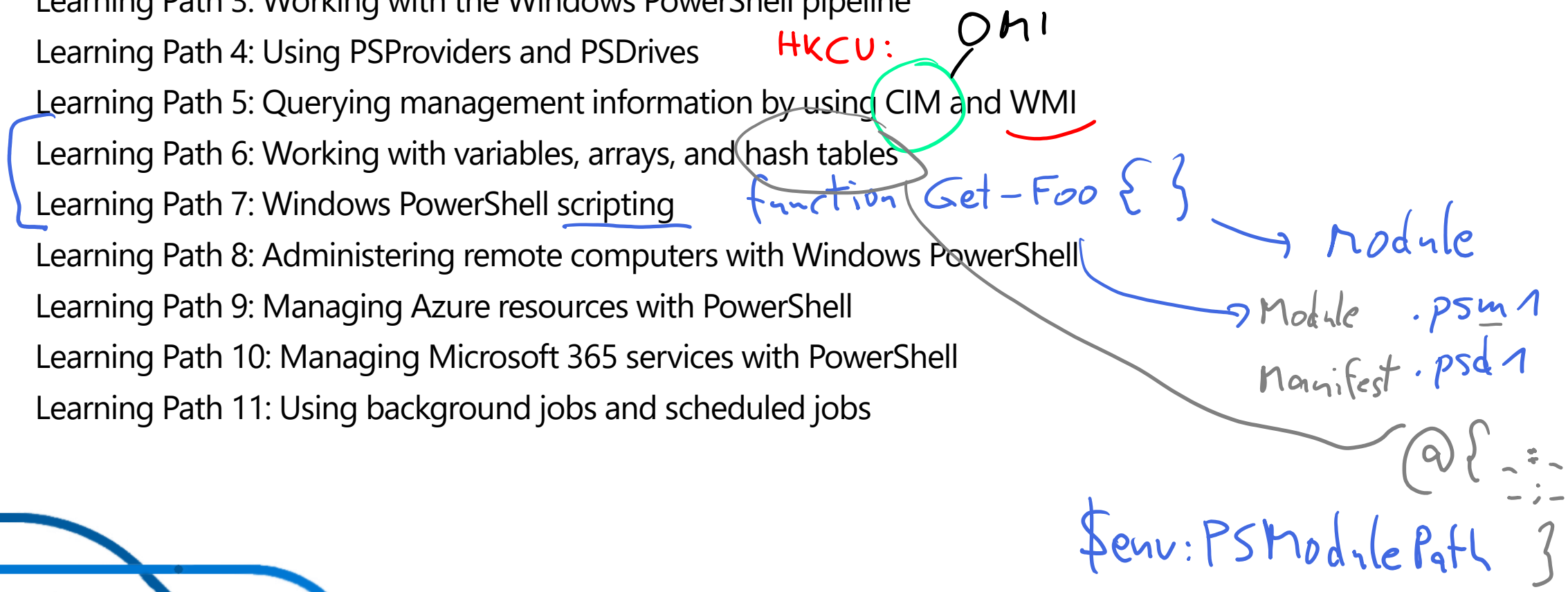
Tag 3

Guten Morgen!



Course outline

Learning Path 1: Getting started with Windows PowerShell
Learning Path 2: Windows PowerShell for local systems administration
Learning Path 3: Working with the Windows PowerShell pipeline
Learning Path 4: Using PSProviders and PSDrives
Learning Path 5: Querying management information by using CIM and WMI
Learning Path 6: Working with variables, arrays, and hash tables
Learning Path 7: Windows PowerShell scripting
Learning Path 8: Administering remote computers with Windows PowerShell
Learning Path 9: Managing Azure resources with PowerShell
Learning Path 10: Managing Microsoft 365 services with PowerShell
Learning Path 11: Using background jobs and scheduled jobs



Learning Path 5: Query management information by using Common Information Model and Windows Management Instrumentation

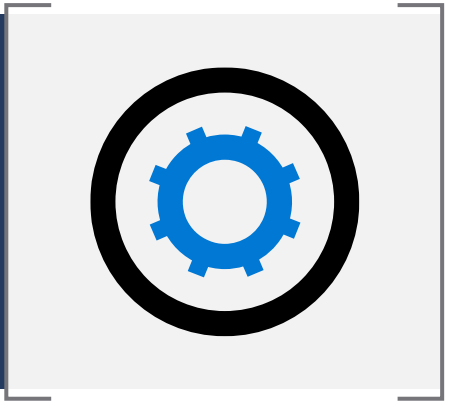
Overview

Windows Management Instrumentation (WMI) and Common Information Model (CIM) provide local and remote access to a repository of management information, including robust information available from the operating system, from computer hardware, and from installed software. In this Learning Path, you'll learn about these two technologies.

Modules:

- Review CIM and WMI
- Query configuration information by using CIM and WMI
- Query and manipulate repository objects by using CIM and WMI methods

Section break 1



Review CIM and WMI

Module overview

In this Module, you'll learn about the architecture of CIM and WMI. Both technologies connect to a common information repository that contains management information that you can query and manipulate. The repository contains information about a computer system or device, including hardware, software, hardware drivers, components, roles, services, user settings, and just about every configurable item and the current state of that item. Familiarity with the framework and syntax of CIM and WMI will help you know and control many aspects of an operating system environment.

Units:

- Architecture and technologies
- Understanding the repository
- Finding documentation
- Demonstration: Finding documentation for classes

Architecture and technologies

- CIM cmdlets
  - Local connections use COM.
  - Ad hoc remote connections use WS-MAN.
  - CIM sessions for remote connections can use WS-MAN or DCOM.
- WMI cmdlets
  - Local connections use COM.
  - Ad hoc remote connections use DCOM.
- CIM or WMI?
  - CIM is preferred because it's current.
  - CIM cmdlets provide easier network connectivity.

Understanding the repository

- The repository used by CIM and WMI is organized into namespaces.
- Namespaces organize related classes:
  - Use tab completion in the *-Namespace* parameter of the **Get-CimInstance** cmdlet to browse the namespace structure.
- Classes represent manageable components.
- An instance is an actual occurrence of a class.
- An instance has:
  - Properties that describe the instance's attributes.
  - Methods that cause the instance to perform an action.

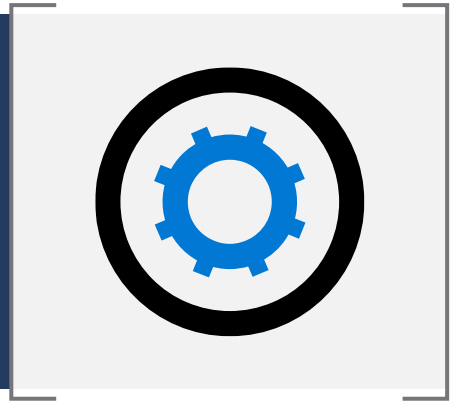
Finding documentation

- The fastest way to find documentation is to enter a repository class name into an internet search engine.
- Classes in the **root\CIMv2** namespace are typically well-documented.
- Classes from other namespaces are typically not well-documented.
- You can get some information by using **Get-Member**.

Demonstration: Finding documentation for classes

In this demonstration, you will learn how to use a search engine to locate the online documentation for the **Win32_BIOS** class.

Section break 2



Query data by using CIM and WMI

Module overview

One of the most common uses for WMI and CIM is querying configuration information from computers. In this Module, you'll learn more about the structure of the namespaces that contain classes and how to query instances of a class. You'll also learn how to query remote computers by using ad hoc connections and CIM sessions.

Units:

- Listing namespaces
- Demonstration: Listing local repository namespaces by using WMI
- Listing classes
- Demonstration: Listing and sorting the classes from a CIM namespace
- Querying instances
- Demonstration: Querying class instances
- Connecting to remote computers
- Using CIM sessions
- Demonstration: Using CIMSession objects

Listing namespaces

- By listing namespaces, you can discover the contents of the repository on your computer.
- To list the namespaces, run the following command:
  - **Get-WmiObject -Namespace root -List -Recurse | Select -Unique __NAMESPACE**
- CIM commands offer tab completion for the *-Namespace* parameter.

Demonstration: Listing local repository namespaces by using WMI

In this demonstration, you will learn how to use the **Get-WmiObject** cmdlet to list namespaces.

Listing classes

- List classes for a namespace by using WIM or CIM:
  - **Get-WmiObject -Namespace root\CIMv2 -List**
  - **Get-CimClass -Namespace root\CIMv2**
- Search for classes with wildcards:
  - **Get-CimClass *network***

| ~~where~~

Filter left!

Demonstration: Listing and sorting the classes from a CIM namespace

In this demonstration, you will learn how to:

- List the classes in **root\SecurityCenter2**.
- List the classes in **root\CIMv2** and sort by **CimClassName**.
- List the classes in **root\CIMv2** that have **network** in the class name.

Querying instances

- Query instances by using WMI or CIM:
  - **Get-WmiObject -Class Win32_LogicalDisk**
  - **Get-CimInstance -ClassName Win32_LogicalDisk**
- Use filtering with CIM or WMI commands to return only matching instances:
  - **Get-CimInstance -ClassName Win32_LogicalDisk -Filter "DriveType=3"**
- Use a WQL query with CIM or WMI commands to return only matching instances:
  - **Get-CimInstance -Query "SELECT * FROM Win32_LogicalDisk WHERE DriveType = 3"**

Demonstration: Querying class instances

In this demonstration, you will learn how to:

1. Use **Get-WmiObject** to query all instances of **Win32_Service**.
2. Use **Get-CimInstance** to query all instances of **Win32_Process**.
3. Use **Get-CimInstance** to query instances of **Win32_LogicalDisk** with a **DriveType** property of **3**.
4. Use **Get-Ciminstance** with a WQL query to display all instances of **Win32_NetworkAdapter**.

Connecting to remote computers

- CIM and WMI cmdlets provide ad hoc reconnections to remote computers by using **-ComputerName**
  - **Get-CimInstance -ClassName Win32_BIOS -ComputerName LON-DC1**
  - You can specify multiple computer names.
- Only WMI cmdlets support using *–Credential*
  - **Get-WmiObject -ComputerName LON-DC1 -Credential ADATUM\Administrator -Class Win32_BIOS**
  - Credentials can be stored in a variable to avoid password prompts.
- Ad hoc connection protocol
  - WMI: DCOM
  - CIM: WS-MAN

RPC :135 → dyn. Port :-C
WinRM :5985
:5986 SSL :-)

Using CIM sessions

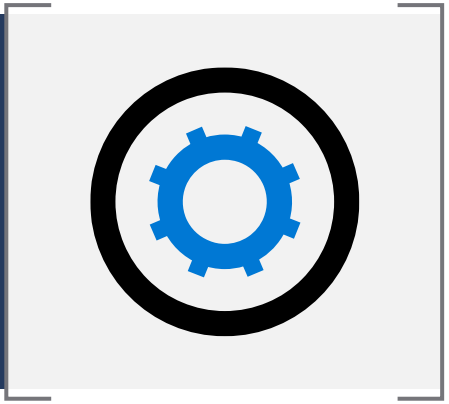
- A CIM session:
  - Is a persistent configuration object for remote connections.
  - Can be reused for multiple commands.
  - Has settings not available for ad-hoc connections.
- Store a CIM session in a variable:
  - **$sessions = New-CimSession -ComputerName LON-CL1,LON-CL2**
- Use the variable when running CIM commands:
  - **Get-CimInstance -CimSession $sessions -ClassName Win32_OperatingSy**stem
- Configure session options
  - **$opt = New-CimSessionOption -Protocol Dcom**
  - **$DcomSession = New-CimSession -ComputerName LON-DC1 -SessionOption $opt**
- Remove sessions
  - **$DcomSession | Remove-CimSession**

Demonstration: Using CIMSession objects

In this demonstration, you will learn how to:

1. Create a CIM session to **LON-DC1**.
2. Query the **Win32_OperatingSystem** class by using the CIM session.
3. Remove the CIM session.

Section break 3



Making changes by using CIM and WMI

Module overview

In this Module, you'll learn to use CIM and WMI to make changes by using methods. The methods available to you vary depending on the type of object. Discovering and understanding these methods is an important step in querying and manipulating the repository information.

Units:

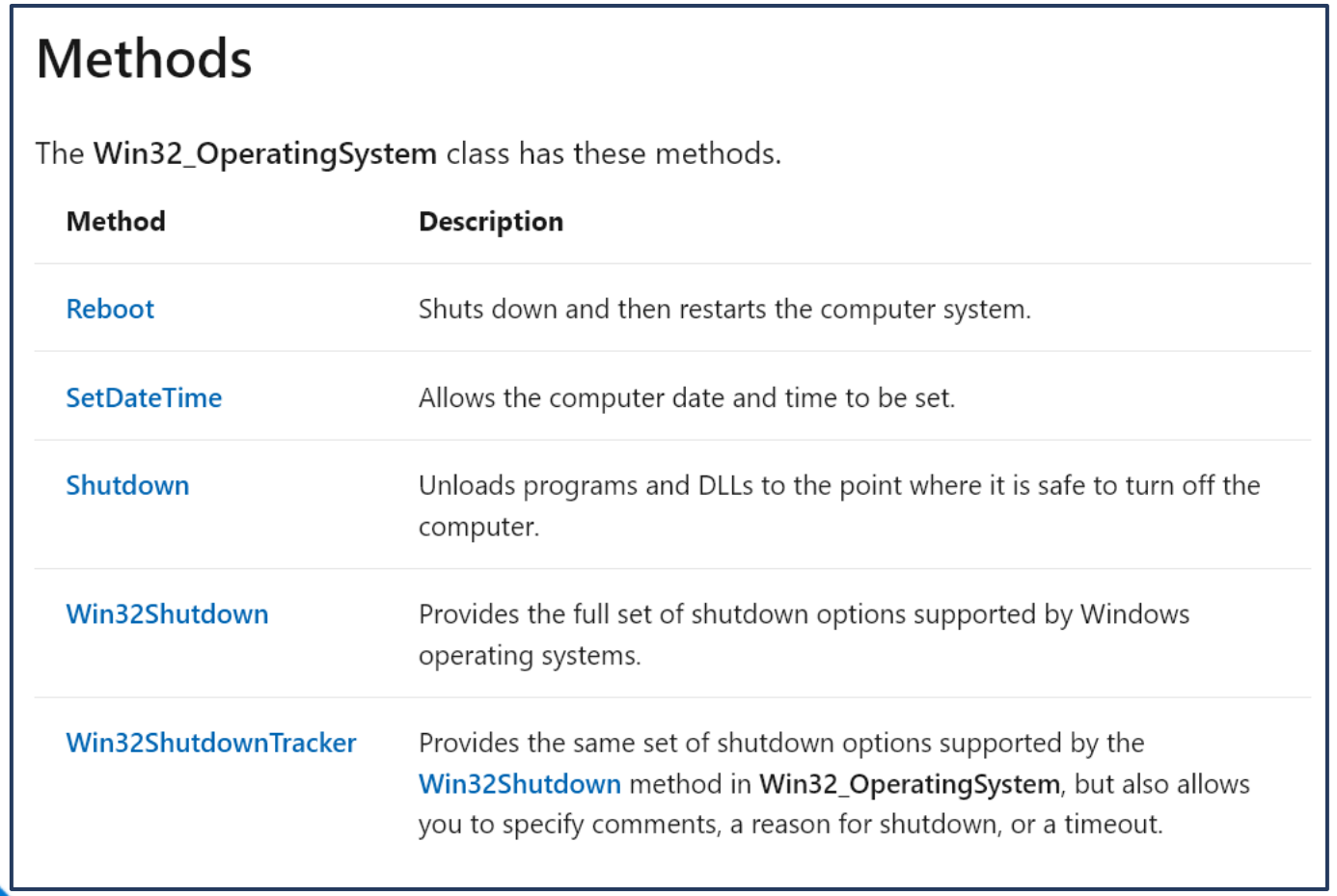
- Discovering methods
- Finding documentation for methods
- Demonstration: Finding methods and documentation
- Invoking methods
- Demonstration: Invoking methods of repository objects

Discovering methods

- A method is an action you can perform on an object.
- The methods available vary depending on the class.
- Review methods for an object instance:
  - **Get-WmiObject -Class Win32_Service | Get-Member -MemberType Method**
  - **Get-CimInstance -Class Win32_Service | Get-Member -MemberType Method**
- Review methods for a specific class:
  - **Get-CimClass -Class Win32_Service | Select-Object -ExpandProperty CimClassMethods**

Finding documentation for methods

- Documentation for methods is part of the class documentation.
- Use an internet search to find the documentation for a class.



Methods

The **Win32_OperatingSystem** class has these methods.

Method	Description
Reboot	Shuts down and then restarts the computer system.
SetDateTime	Allows the computer date and time to be set.
Shutdown	Unloads programs and DLLs to the point where it is safe to turn off the computer.
Win32Shutdown	Provides the full set of shutdown options supported by Windows operating systems.
Win32ShutdownTracker	Provides the same set of shutdown options supported by the Win32Shutdown method in **Win32_OperatingSystem**, but also allows you to specify comments, a reason for shutdown, or a timeout.

Demonstration: Finding methods and documentation

In this demonstration, you will learn how to:

1. Use **Get-WmiObject** to display members for the **Win32_Service** class.
2. Use **Get-CimClass** to display the **CimClassMethods** property for the **Win32_Service** class.
3. Use **Get-CimClass** to display members for the **Win32_Service** class.
4. Find documentation for the **Change** method of the **Win32_Service** class.

Invoking methods

Set-Service

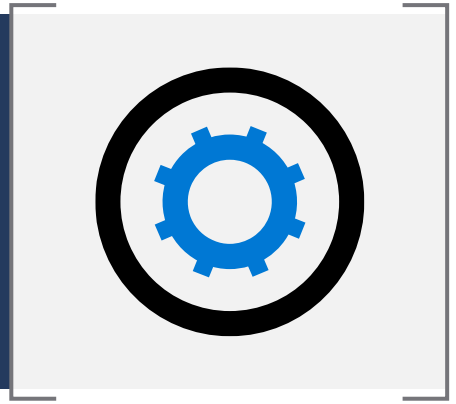
- Direct invocation works only for WMI objects.
  - **$WmiSpoolerService = Get-WmiObject -Class Win32_Service -Filter "Name='Spooler'"**
  - **$WmiSpoolerService.ChangeStartMode("Manual")**
- For WMI objects, method parameters are ordinal.
  - **$WmiSpoolerService | Invoke-WmiMethod -Name ChangeStartMode -Argument "Manual"**
- For CIM objects, method parameters are named.
  - **Invoke-CimMethod -ClassName Win32_Process -MethodName Create -Arguments @{CommandLine='Mspaint.exe'}**

Demonstration: Invoking methods of repository objects

In this demonstration, you will learn how to:

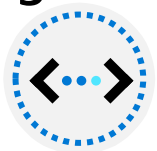
1. Use **Invoke-CimMethod** and the **Reboot** method of the **Win32_OperatingSystem** class remotely on **LON-DC1**.
2. Use the **Create** method of the **Win32_Process** class to start **mspaint.exe**.
3. Use the **Terminate** method of the **Win32_Process** class to close Microsoft Paint.

Section break 4

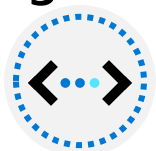


Lab: Querying information by using WMI and CIM

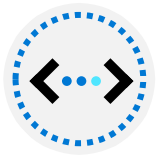
Exercise 1: Querying information by using WMI



Exercise 2: Querying information by using CIM



Exercise 3: Invoking methods



Sign-in information for the exercises:

Virtual machines: **AZ-040T00A-LON-DC1** and **AZ-040T00A-LON-CL1**

Username: **Adatum\Administrator**

Password: **Pa55w.rd**

Lab scenario

You have to query management information from several computers. You start by querying the information from your local computer and from one test computer in your environment.

Lab-review questions

Which class do you query to review IP addresses?

When would you prefer to use WMI instead of CIM?

When might you want to remotely reboot a computer by using the **Win32_OperatingSystem** class?

Lab-review answers

Which class do you query to review IP addresses?

The **Win32_NetworkAdapterConfiguration** class contains IP address information.

When would you prefer to use WMI instead of CIM?

You'd use WMI instead of CIM if there were any features not available in CIM that you required. This is rare.

When might you want to remotely reboot a computer by using the **Win32_OperatingSystem** class?

If you've made configuration changes that require a reboot to take effect, you can trigger a reboot at the end of your configuration process.

References

[Win32 Provider]

[CIM Classes (WMI)]

[Event log message indicates that the Windows Installer reconfigured all installed applications]

[WQL (SQL for WMI)]