WMIC - Take Command-line Control over WMI

Table Of Contents

Chapter 1

WMIC - Take Command-line Control over WMI

Chapter 1

WMIC - Take Command-line Control over WMI

By Ethan Wilansky

This article is from the March 2002 issue of Windows & .NET Magazine.

Microsoft is creating a lot of good reasons to make the command prompt in Windows XP and the Windows Server 2003 family your home for systems management. Windows Management Instrumentation Command-line (WMIC), which uses the power of Windows Management Instrumentation (WMI) to enable systems management from the command line, is one of those reasons.

WMI has been an important part of Microsoft's systems management initiative since Microsoft Systems Management Server (SMS) 2.0 and has grown in popularity since the introduction of Windows 2000. However, until the introduction of WMIC, you couldn't easily access the WMI repository or the WMI namespace from a command prompt.

On This Page

What Is WMIC?
How to Run WMIC
WMIC Command-Line Components
Putting WMIC to Work

What Is WMIC?

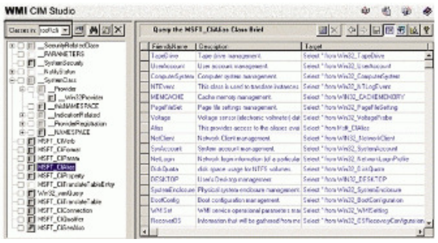
WMIC extends WMI for operation from several command-line interfaces and through batch scripts. Before WMIC, you used WMI-based applications (such as SMS), the WMI Scripting API, or tools such as CIM Studio to manage WMI-enabled computers. Without a firm grasp on a programming language such as C++ or a scripting language such as VBScript and a basic understanding of the WMI namespace, do-it-yourself systems management with WMI was difficult. WMIC changes this situation by giving you a powerful, user-friendly interface to the WMI namespace.

WMIC is more intuitive than WMI, in large part because of *aliases*. Aliases take simple commands that you enter at the command line, then act upon the WMI namespace in a predefined way, such as constructing a complex WMI Query Language (WQL) command from a simple WMIC alias Get command. Thus, aliases act as friendly syntax intermediaries between you and the namespace. For example, when you run a simple WMIC command such as

```
useraccount list brief
```

from the WMIC command prompt to get user account information, the Useraccount alias performs a WQL query of the Win32_Useraccount class and displays specific data from this class in text format. WMIC also displays the Win32_Useraccount class's properties at the console in text format. WMIC can return the results of a command in other formats, such as XML, HTML, and Comma Separated Value (CSV).

WMIC stores aliases as instances of a class in the WMI schema. The default alias classMSFT_CliAlias—and other classes that support WMIC are stored in the schema's default namespace, or role—root\cli. Think of a role as simply another WMI namespace that's specifically designed to support WMIC. The default role, root\cli, connects to the root\cimv2 namespace to operate on the classes within root\cimv2. You won't typically use CIM Studio when you run WMIC, but you can use CIM Studio to find the root\cli namespace in a list of WMI namespaces. Figure 1 shows the root\cli node and some of the properties in the MSFT_CliAlias class.



FriendlyName	Description	Target
TapeDrive	Tape drive management.	Select * from Win32_TapeDrive
UserAccount	User account management.	Select * from Win32_UserAccount
ComputerSystem	Computer system management.	Select * from Win32_ComputerSystem
NTEvent	This class is used to translate instances	Select * from Win32_NTLogEvent
MEMCACHE	Cache memory management.	Select * from WIN32_CACHEMEMORY
PageFileSet	Page file settings management.	Select * from Win32_PageFileSetting
Voltage	Voltage sensor (electronic voltmeter) dat	Select * from Win32_VoltageProbe
Alias	This provides access to the aliases avai	Select * from Msft_CliAlias
NetClient	Network Client management.	Select * from WIN32_NetworkClient
SysAccount	System account management.	Select * from Win32_SystemAccount
NetLogin	Network login information (of a particular	Select * from Win32_NetworkLoginProfile
DiskQuota	disk space usage for NTFS volumes.	Select * from Win32_DiskQuota
DESKTOP	User's Desktop management	Select * from WIN32_DESKTOP
SystemEnclosure	Physical system enclosure management.	Select * from Win32_SystemEnclosure
BootConfig	Boot configuration management.	Select * from Win32_BootConfiguration
WMISet	WMI service operational parameters mai	Select * from Win32_WMISetting
RecoverOS	Information that will be gathered from me	Select * from Win32_OSRecoveryConfiguration

Figure 1: The root\cli namespace and MSFT_CliAlias

You can add new aliases to the root\cli namespace and to other namespaces. You can also access the WMI namespace directly with the Class and Path commands. Later in this article, I show you how to use these commands.

Top of page

How to Run WMIC

To invoke the WMIC command prompt, type

wmic

in the standard XP or .NET Server command interpreter (cmd.exe), a Telnet session, or the Run dialog box. These startup methods are available locally or through a .NET Server Terminal Services session.

The first time you start WMIC, Windows installs it on the computer and displays the WMIC command prompt, wmic:root\cli>. At this prompt, you enter commands interactively. For example, the command

```
process where name='outlook.exe' call terminate
```

closes Microsoft Outlook if it's running. After the command runs, the WMIC command prompt reappears. Note that you must type each command in this article on one line, not split across multiple lines, for the commands to run properly. (Some commands in the tables and listings in this article are split across multiple lines for publication purposes.)

WMIC also runs in noninteractive mode. This mode is especially useful when you need to complete a single task or you need to run WMIC commands in a batch file. To run a command in noninteractive mode, you start WMIC and run the command in the same line of code. For example, you can open a command prompt and type

```
wmic /node:SERVER1 printer list status
```

to check the status of printers attached to SERVER1. In this example, WMIC starts, uses the /node switch to establish a remote connection to SERVER1, and runs a command to display printer status information. After the command runs and displays results, the command prompt reappears.

You can remotely access any WMI-enabled computer from any computer running WMIC. The computer you want to access doesn't require WMIC. For example, you can start WMIC from a computer running XP Professional Edition to manage a mix of computers running XP, Win2K, Windows NT 4.0, Windows Me, and Windows 9x.

Top of page

WMIC Command-Line Components

WMIC uses global switches, aliases, verbs, commands, and command-line help to empower the interface. Global switches are settings that apply to and for the entire WMIC session. For example, the /trace:on switch enables error tracing. While this switch is on, WMIC returns error information for every command you execute. The /node switch lets you access a remote computer. The /interactive:on switch ensures that WMIC prompts you for confirmation before performing delete operations. Other global switches include /role, /user, /implevel, and /namespace.

As I explained earlier, aliases are the friendly syntax intermediaries between you and the WMI namespace. Verbs are the actions you want to take when specifying an alias. I've already shown you the List and Call verbs. Table 1 describes the other WMIC verbs and provides a sample command for each one.

Table 1 WMIC Verbs

Verb	**Sample Command**	**Description**
Assoc	group where name= 'administrators' assoc	Shows all the associations that the Administrators group has with the system. For example, Administrators group members and the drives that they own appear in the list of properties displayed.
	os assoc	Displays information about the OS and installed patches and hotfixes.
Create	environment create name="progloc", username="wkst01 \ ethanw",variablevalue= "%programfiles%\prog01"	Adds a variable named Progloc and sets its value to a folder below the Program Files folder. For example, the sample command is adding this variable to the Ethanw user account on the WKST01 workgroup computer.
Delete	environment where(name= "progloc") delete	Deletes the Progloc environment variable. To avoid unintended deletions when testing a WMIC command that uses the Delete verb, use the /interactive:on global switch. You'll then be prompted to confirm each deletion.
Get	partition get bootpartition, description, deviceid, bootable	Returns the boot-partition Boolean (true or false), description string, and device ID properties of the partition alias.
Set	path WIN32_USERACCOUNT where(name="user01") set disabled="true"	Disables the User01 user account on a member server or workstation.

You use commands to control access to WMIC and the WMI namespace. Notice that the last sample command in the table uses Path and the WIN32_USERACCOUNT class rather than the Useraccount alias. Path is a WMIC command that lets you directly access one or more instances in the WMI namespace rather than reaching them through an alias. The Path command is especially useful when no alias exists for the systems management task you want to complete. You can extend WMIC with new aliases and new roles, but using the Path command is easier if you have a firm grasp on the WMI namespace.

WMIC also supports the Class, Context, Quit, and Exit commands. The Class command lets you directly access a class in the WMI schema or create an instance of an existing class. The difference between the Path and Class commands is that the Path command acts on an instance and its properties (e.g., to retrieve management data), while the Class command acts on the class definition. For example, to retrieve all the properties of the WIN32_SOFTWAREELEMENT class, you can type

```
class WIN32_SOFTWAREELEMENT get
```

The output to the console is in HTML format. Later, I show you how to use the /output global switch to redirect the output to an HTML file and view the attributes from a Web browser. The Class command with the Assoc verb shows the namespace path to a class and the other classes associated with the class. You can use the Class command to delete a class and to create an instance of a class but not to create a class.

The Context command shows the current settings of the global switches. The Quit and Exit commands simply leave the WMIC command prompt and return you to the previous shell (e.g., the Telnet prompt, the XP command prompt).

Command-line Help is the way to become familiar with WMIC. Table 2 shows the characters to type at a WMIC command prompt to find the specified information.

Table 2 Command-Line Help

Command	**Example**	**Description**
/? or -?		Shows the syntax of all global switches and aliases
/<switch_name> /?	/user /?	Shows information about one global switch
<command_name> /?	class /?	Shows information about one command
<alias_name> /?	memcache /?	Shows information about one alias
<alias_name>	temperature	Shows information about one alias and verb combination
<alias_name> <verb_name> /?	temperature get /?	Shows information about one alias and verb combination
/?:Full	irq get /?:Full	Shows verbose help information

Top of page

Putting WMIC to Work

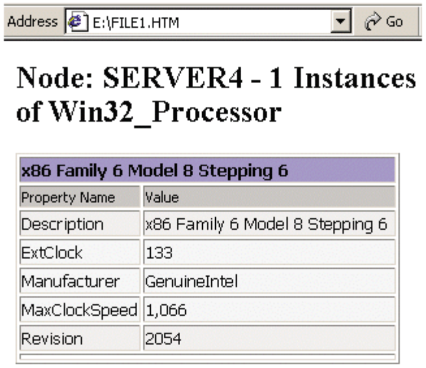
Now that you understand the components of the WMIC command line, let's look at how to run WMIC from a batch file and send output to the console and to an HTML or XML file. When you run WMIC from a batch file, the XP command prompt appears after the commands run. Running WMIC from a batch file lets you repeat common tasks without having to retype a complicated sequence of commands. For example, Listing 1 shows a command you can place in a batch file to display at the console selected processor information about two computers, SERVER1 and SERVER4. The /format switch is verb-specific rather than global because it works with only the Get and List verbs.

```
LISTING 1: Code to Display
Results at the
Console from a WMIC Batch File
wmic /node:SERVER1, SERVER4
     cpu get name, caption,
     maxclockspeed, systemname
     /format:textvaluelist.xsl
```

WMIC batch files can use variables. Instead of specifying computer names in the batch file, you can specify variables in the format %1, %2, and so on, as Listing 2 shows. You can place this code in a batch file, then when you run it, type one or two computer names after the batch file name. Alternatively, you can create a separate text file that contains a CSV list or a carriage-return-delimited list of computer names. You call the text file with the /node global switch and the text file name with the @ symbol as a prefix. The @ symbol tells the /node switch that the following parameter is a filename, not a computer name.

```
LISTING 2: Code to Use Variables in a WMIC Batch File
@echo off
if "%1"=="" goto msg
if "%2"=="" goto single
wmic /node:%1, %2 cpu get name,
     caption, maxclockspeed,
     systemname
     /format:textvaluelist.xsl
 goto end
:single
 wmic /node:%1 cpu get name,
   caption, maxclockspeed,
   systemname
   /format:textvaluelist.xsl
goto end
:msg
 echo you must specify at least
    one computer name.
:end
```

The console isn't the only place to send results. You can instruct WMIC to send output to a file in XML, HTML, or Managed Object Format (MOF) format. MOF is the native WMI file format for classes and class instances in the WMI repository on a WMI-enabled computer. Listing 3 shows code that directs the output of SERVER4 processor information to an HTML file. The /output global switch instructs WMIC to send the output to file1.htm. The /format verb-specific switch instructs WMIC to transform the native XML output into an HTML form. You can create Extensible Stylesheet Language (XSL) files to format output or use any of the XSL files stored in the \%systemroot%\system32\wbem folder of any computer with WMIC installed. For example, you can use the csv.xsl file to format the output into a CSV list of results, or the htable.xsl file to create an HTML table of results. Figure 2 shows file1.htm open in a Web browser.



Node: SERVER4 - 1 Instances of Win32_Processor

x86 Family 6 Model 8 Stepping 6	
Property Name	Value
Description	x86 Family 6 Model 8 Stepping 6
ExtClock	133
Manufacturer	GenuineIntel
MaxClockSpeed	1,066
Revision	2054

Figure 2: Selected processor information in HTML format

```
LISTING 3: Code to Direct WMIC Output to an HTML File
wmic /node:SERVER4
     /output:e:\file1.htm cpu get
      description, maxclockspeed,
      extclock, manufacturer,
      revision /format:hform.xsl
```

```
LISTING 4: Code to Direct Class
 Command Output to an HTML File
wmic /output:e:\se_class.htm
      class WIN32_SOFTWAREELEMENT
      get
```

```
LISTING 5: Code to Generate XML Output from a WMIC Command
  wmic cpu get maxclockspeed
       /translate:basicxml
      /format:rawxml.xsl
```

Earlier, I mentioned that the default output of the Class command with the Get verb is in HTML format. Therefore, to send output of this type of command to an HTML file, you specify the /output switch without a /format switch, as Listing 4 shows. The /record and /append global switches also let you capture information from the WMIC command line. Use the WMIC Command-line Help facility to find more information about these switches.

To output information in XML format, use the /translate switch and the Basicxml keyword to convert the greater than (>) and less than (<) symbols into meaningful characters in XML. For example, Listing 5 shows how to create raw XML output. You can then import the XML data into a database or some other repository that understands the XML tags in the output. The output created from the code in Listing 5 contains the WMIC command, the command-line request, the target nodes, the context of the global switches, and the command results.

For good reason, WMIC is a key piece of Microsoft's command-line initiative for XP and .NET Server. WMIC enables robust command-line systems-management access to the WMI namespace, wherever WMI is running on the network. The command-line components take a little time to master, but after you do master them, a whole world of systems management lies at your fingertips.

We at Microsoft Corporation hope that the information in this work is valuable to you. Your use of the information contained in this work, however, is at your sole risk. All information in this work is provided "as -is", without any warranty, whether express or implied, of its accuracy, completeness, fitness for a particular purpose, title or non-infringement, and none of the third-party products or information mentioned in the work are authored, recommended, supported or guaranteed by Microsoft Corporation. Microsoft Corporation shall not be liable for any damages you may sustain by using this information, whether direct, indirect, special, incidental or consequential, even if it has been advised of the possibility of such damages. All prices for products mentioned in this document are subject to change without notice.

Top of page

Top of page