

The Wayback Machine - <https://web.archive.org/web/20040717120558/http://msdn.microsoft...>

[MSDN Home](#) > [MSDN Library](#) > [Setup and System Administration](#) > [Windows Management Instrumentation \(WMI\)](#) > [Windows Management Instrumentation](#) > [WMI Reference](#) > [WMI Classes](#) > [Win32 Classes](#) > [Operating System Classes](#)

Platform SDK: Windows Management Instrumentation

Win32_Process

The **Win32_Process** [WMI class](#) represents a sequence of events on a Windows® operating system. A descendent or member of this class is a sequence that consists of an interaction of one or more processors or interpreters, some executable code, and a set of inputs, for example, a client application running on a Windows system.

The following syntax is simplified from Managed Object Format (MOF) code and includes all inherited properties.

```
class Win32_Process : CIM_Process
{
    string Caption;
    string CommandLine;
    string CreationClassName;
    datetime CreationDate;
    string CSCreationClassName;
    string CSName;
    string Description;
    string ExecutablePath;
    uint16 ExecutionState;
    string Handle;
    uint32 HandleCount;
    datetime InstallDate;
    uint64 KernelModeTime;
    uint32 MaximumWorkingSetSize;
    uint32 MinimumWorkingSetSize;
    string Name;
    string OSCreationClassName;
    string OSName;
    uint64 OtherOperationCount;
    uint64 OtherTransferCount;
    uint32 PageFaults;
    uint32 PageFileUsage;
    uint32 ParentProcessId;
    uint32 PeakPageFileUsage;
    uint64 PeakVirtualSize;
    uint32 PeakWorkingSetSize;
    uint32 Priority;
    uint64 PrivatePageCount;
    uint32 ProcessId;
    uint32 QuotaNonPagedPoolUsage;
    uint32 QuotaPagedPoolUsage;
    uint32 QuotaPeakNonPagedPoolUsage;
    uint32 QuotaPeakPagedPoolUsage;
    uint64 ReadOperationCount;
    uint64 ReadTransferCount;
    uint32 SessionId;
    string Status;
    datetime TerminationDate;
    uint32 ThreadCount;
    uint64 UserModeTime;
    uint64 VirtualSize;
    string WindowsVersion;
```

```

uint64 WorkingSetSize;
uint64 WriteOperationCount;
uint64 WriteTransferCount;
};

```

Methods

The **Win32_Process** class defines the following methods.

Method	Description
AttachDebugger	Launches the currently registered debugger for a process.
Create	Creates a new process.
GetOwner	Retrieves the user name and domain name under which the process is running.
GetOwnerSid	Retrieves the security identifier (SID) for the owner of a process.
SetPriority	Changes the execution priority of a process.
Terminate	Terminates a process and all of its threads.

Properties

The **Win32_Process** class has the following properties.

Caption

Data type: **string**

Access type: Read-only

Short description of an object—a one-line string.

CommandLine

Data type: **string**

Access type: Read-only

Command line used to start a specific process, if applicable. This property is new for Windows XP.

CreationClassName

Data type: **string**

Access type: Read-only

Qualifiers: [Key](#), **MaxLen(256)**

Name of the first concrete class in the inheritance chain that is used to create an instance. You can use this property with other key properties of the class to identify uniquely all instances of the class and its subclasses. This property is inherited from

[CIM_System](#)

CreationDate

Data type: **datetime**

Access type: Read-only

Date the process begins executing.

CSCreationClassName

Data type: **string**

Access type: Read-only

Creation class name of the scoping computer system.

CSName

Data type: **string**

Access type: Read-only

Name of the scoping computer system.

Description

Data type: **string**

Access type: Read-only

Description of an object.

ExecutablePath

Data type: **string**

Access type: Read-only

Qualifiers: [Privileges](#)(**SeDebugPrivilege**)

Path to the executable file of the process.

Example: C:\WINDOWS\EXPLORER.EXE

ExecutionState

Data type: **uint16**

Access type: Read-only

This property is not implemented and does not get populated for any instance of this class. This property is always NULL.

Handle

Data type: **string**

Access type: Read-only

Process identifier.

HandleCount

Data type: **uint32**

Access type: Read-only

Total number of open handles owned by the process. HandleCount is the sum of the handles currently open by each thread in this process. A handle is used to examine or modify the system resources. Each handle has an entry in a table that is maintained internally. Entries contain the addresses of the resources and data to identify the resource type.

InstallDate

Data type: **datetime**

Access type: Read-only

Date an object is installed. The object may be installed without a value being written to this property.

KernelModeTime

Data type: **uint64**

Access type: Read-only

Time in kernel mode, in 100 nanosecond units. If this information is not available, use a value of 0 (zero).

MaximumWorkingSetSize

Data type: **uint32**

Access type: Read-only

Qualifiers: **Privileges(SeDebugPrivilege), Units(Kilobytes)**

Maximum working set size of the process. The working set of a process is the set of memory pages visible to the process in physical RAM. These pages are resident, and available for an application to use without triggering a page fault.

Example: 1413120

MinimumWorkingSetSize

Data type: **uint32**

Access type: Read-only

Qualifiers: **Privileges(SeDebugPrivilege), Units(Kilobytes)**

Minimum working set size of the process. The working set of a process is the set of memory pages visible to the process in physical RAM. These pages are resident and available for an application to use without triggering a page fault.

Example: 20480

Name

Data type: **string**

Access type: Read-only

Label for an object. When inherited by a subclass, the property can be overridden to be a key property.

OSCreationClassName

Data type: **string**

Access type: Read-only

Creation class name of the scoping operating system.

OSName

Data type: **string**

Access type: Read-only

Name of the scoping operating system.

OtherOperationCount

Data type: **uint64**

Access type: Read-only

Number of I/O operations performed that are not read or write operations.

OtherTransferCount

Data type: **uint64**

Access type: Read-only

Qualifiers: **Units(Bytes)**

Amount of data transferred during operations that are not read or write operations.

PageFaults

Data type: **uint32**

Access type: Read-only

Number of page faults that a process generates.

Example: 10

PageFileUsage

Data type: **uint32**

Access type: Read-only

Qualifiers: **Units(Kilobytes)**

Amount of page file space that a process is using currently.

Example: 102435

ParentProcessId

Data type: **uint32**

Access type: Read-only

Unique identifier of the process that creates a process. Process identifier numbers are reused, so they only identify a process for the lifetime of that process. It is possible that the process identified by **ParentProcessId** is terminated, so **ParentProcessId** may not refer to a running process. It is also possible that **ParentProcessId** incorrectly refers to a process that reuses a process identifier. You can use the **CreationDate** property to determine whether the specified parent was created after the process represented by this Win32_Process instance was created.

PeakPageFileUsage

Data type: **uint32**

Access type: Read-only

Qualifiers: **Units(Kilobytes)**

Maximum amount of page file space used during the life of a process.

Example: 102367

PeakVirtualSize

Data type: **uint64**

Access type: Read-only

Qualifiers: **Units(Bytes)**

Maximum virtual address space a process uses at any one time. Using virtual address space does not necessarily imply corresponding use of either disk or main memory pages. However, virtual space is finite, and by using too much the process might not be able to load libraries.

PeakWorkingSetSize

Data type: **uint32**

Access type: Read-only

Qualifiers: **Units(Kilobytes)**

Peak working set size of a process.

Example: 1413120

Priority

Data type: **uint32**

Access type: Read-only

Scheduling priority of a process within an operating system. The higher the value, the higher priority a process receives. Priority values can range from 0 (zero), which is the lowest priority to 31, which is highest priority.

Example: 7

PrivatePageCount

Data type: **uint64**

Access type: Read-only

Current number of pages allocated that are accessible only to the process represented by this Win32_Process instance.

ProcessId

Data type: **uint32**

Access type: Read-only

Global process identifier that you can use to identify a process. The value is valid from the time a process is created until it is terminated.

QuotaNonPagedPoolUsage

Data type: **uint32**

Access type: Read-only

Quota amount of non-paged pool usage for a process.

Example: 15

QuotaPagedPoolUsage

Data type: **uint32**

Access type: Read-only

Quota amount of paged pool usage for a process.

Example: 22

QuotaPeakNonPagedPoolUsage

Data type: **uint32**

Access type: Read-only

Peak quota amount of non-paged pool usage for a process.

Example: 31

QuotaPeakPagedPoolUsage

Data type: **uint32**

Access type: Read-only

Peak quota amount of paged pool usage for a process.

Example: 31

ReadOperationCount

Data type: **uint64**

Access type: Read-only

Number of read operations performed.

ReadTransferCount

Data type: **uint64**

Access type: Read-only

Qualifiers: **Units(Bytes)**

Amount of data read.

SessionId

Data type: **uint32**

Access type: Read-only

Unique identifier that an operating system generates when a session is created. A session spans a period of time from log on until log off from a specific system.

Status

Data type: **string**

Access type: Read-only

This property is not implemented and does not get populated for any instance of this class. It is always NULL.

TerminationDate

Data type: **datetime**

Access type: Read-only

Process was stopped or terminated. To get the termination time, a handle to the process must be held open. Otherwise, this property returns NULL.

ThreadCount

Data type: **uint32**

Access type: Read-only

Number of active threads in a process. An instruction is the basic unit of execution in a processor, and a thread is the object that executes an instruction. Each running process has at least one thread. This property is for computers running Windows NT only.

UserModeTime

Data type: **uint64**

Access type: Read-only

Time in user mode, in 100 nanosecond units. If this information is not available, use a value of 0 (zero).

VirtualSize

Data type: **uint64**

Access type: Read-only

Qualifiers: **Units(Bytes)**

Current size of the virtual address space that a process is using. Using virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and by using too much, the process might not be able to load libraries.

WindowsVersionData type: **string**

Access type: Read-only

Version of Windows in which the process is running.

Example: 4.0

WorkingSetSizeData type: **uint64**

Access type: Read-only

Amount of memory in bytes that a process needs to execute efficiently—for an operating system that uses page-based memory management. If the system does not have enough memory (less than the working set size), thrashing occurs. If the size of the working set is not known, use NULL or 0 (zero). If working set data is provided, you can monitor the information to understand the changing memory requirements of a process.

WriteOperationCountData type: **uint64**

Access type: Read-only

Number of write operations performed.

WriteTransferCountData type: **uint64**

Access type: Read-only

Qualifiers: **Units(Bytes)**

Amount of data written.

Examples

The following VBScript example shows how to obtain the owner of each process on a local computer. You can use this script to obtain data from a remote computer, for example, to determine which users have processes running on a Windows 2000 terminal server, substitute the name of the remote computer for "."—in the first line. You must also be an administrator on the remote machine.

```
strComputer = "."
Set objWMIService = GetObject("winmgmts:" & "{impersonationLevel=impersonate}!\\" & strComputer & "\root\cimv2")

Set colProcesses = objWMIService.ExecQuery( "select * from win32_process" )
For Each objProcess in colProcesses

    If objProcess.GetOwner ( User, Domain ) = 0 Then
        Wscript.Echo "Process " & objProcess.Caption & " belongs to " & Domain & "\" & User
    Else
        Wscript.Echo "Problem " & Rtn & " getting the owner for process " & objProcess.Caption
    End If
Next
```

The following VBScript example shows how to obtain the log on session associated with a running process. A process must be running Notepad.exe before the script starts. The example locates the instances of Win32_LogonSession associated with the Win32_Process that represents Notepad.exe. The Win32_SessionProcess class is specified as the association class. For more information, see [ASSOCIATORS OF Statement](#).

On Error Resume Next

```
Set objWMIService = GetObject("winmgmts:" & "{impersonationLevel=impersonate}!\\" & "." & "\root\cimv2")
```

```
Set colProcesses = objWMIService.ExecQuery("Select * from Win32_Process Where Name = 'Notepad.exe'")
```

```
For Each objProcess in colProcesses
```

```
    ProcessId = objProcess.ProcessId
```

```
    Set colLogonSessions = objWMIService.ExecQuery("Associators of {Win32_Process='" & ProcessId & "'} " & _  
        "Where Resultclass = Win32_LogonSession Assocclass = Win32_SessionProcess", "WQL", 48)
```

```
        If Err <> 0 Then
```

```
            WScript.Echo "Error on associators query " & Err.number & Err.Description
```

```
            WScript.Quit
```

```
        End If
```

```
        For Each LogonSession in colLogonSessions
```

```
            Wscript.Echo " Logon id is " & LogonSession.LogonId
```

```
        Next
```

```
Next
```

Remarks

The **Win32_Process** class is derived from [CIM_Process](#).

Requirements

Client: Requires Windows XP, Windows 2000 Professional, or Windows NT Workstation 4.0 SP4 and later.

Server: Requires Windows Server 2003, Windows 2000 Server, or Windows NT Server 4.0 SP4 and later.

Header: Declared in Cimwin32.mof.

Namespace: Included in \root\cimv2.

See Also

[Operating System Classes](#)

Last updated: May 2004 | [What did you think of this topic?](#) | [Order a Platform SDK CD](#)
[© 2004 Microsoft Corporation. All rights reserved. Terms of use.](#)

[Manage Your Profile](#) | [Legal](#) | [Contact Us](#) | [MSDN Flash Newsletter](#)

©2004 Microsoft Corporation. All rights reserved. [Terms of Use](#) | [Trademarks](#) | [Privacy Statement](#)

