

Monitoring Performance, Events and Alerts with Windows PowerShell

Jeffery Hicks
SAPIEN Technologies

Pre-requisites for this presentation:

1) Some PowerShell experience

Level: Intermediate

- PowerShell and WMI
- Using the .NET Management Classes
- Querying Performance Classes in WMI
- Monitoring Events
- All demos will be available at blog.sapien.com



WMI and PowerShell

- PowerShell v1.0 has limited remoting capabilities
- Windows Management Instrumentation is the tool of choice
- Get-WMIObject is the cmdlet (although we may access .NET management classes directly)

Get-WMIObject

```
Get-WMIObject [-namespace  
root\cimv2]  
[-class] Win32_LogicalDisk  
[-computername SRV03]  
[-credential PSCredential]
```

```
Get-WMIObject -query "Select  
Size,Freespace from  
win32_logicaldisk where  
drivetype=3"  
[-computername SRV02]  
[-credential PSCredential]
```

Browsing WMI

- Get-WMIObject –list
- Use a WMI Browser



Performance Monitoring

- WMI PerfRawData
- WMI PerfFormattedData
- System.Diagnostics

- Raw and ready
- You'll likely have to prepare data depending on the property
- Time consuming to develop

- Already cooked
- May need a little re-formatting to make it more meaning (e.g. converting total number of seconds to hours or days)

WMI PerfMon Classes

- Formatted Data vs Raw Data

```
PS C:\> get-wmiobject -list | where  
{$_.name -match "RawData"}
```

```
PS C:\> get-wmiobject -list | where  
{$_.name -match "FormattedData"}
```

- Access the same WMI classes but through .NET
- Get a PerfMon Category
- Find your counters
- Get instances
- Return values

```
[System.Diagnostics.PerformanceCounter  
Category]::GetCategories() | Sort  
Categoryname |  
format-table  
Categoryname,CategoryHelp -wrap
```

```
#get a specific category
```

```
$category=New-Object
```

```
    system.diagnostics.performancecountercategory
```

```
    "Memory"
```

```
#list counters for this category
```

```
#we need an instance to look at
```

```
$instance=$category.getinstancenames()[0]
```

```
$category.getcounters($instance)
```

#show all instances of a given
category

```
$category.getinstancenames()
```


- Create Performance Counter Object
- Category,Counter,Instance

New-Object

```
System.Diagnostics.PerformanceCounter  
"Process", "% Privileged  
Time", $process
```

- **NextValue()** Obtains a counter sample and returns the calculated value for it.
- **NextSample()** Obtains a counter sample, and returns the raw value for it.
- You might need to format results in either case



- Watching for new event logs
- Watching for file additions
- Watching for process changes
- Watching for service changes

.NET Management Classes

- System.Management namespace
- System.Management.WQLEventQuery
- System.Management.ManagementScope
- System.Management.ManagementEventWatcher
- System.Management.EventWatcherOptions

- Use Trace Classes
- Watch for `__Instance` events

```
get-wmiobject -list | where {$_.name -match  
    "trace"}
```

Win32_SystemTrace

Win32_ProcessTrace

Win32_ProcessStartTrace

Win32_ProcessStopTrace

Win32_ModuleTrace

Win32_ModuleLoadTrace

Win32_ThreadTrace

Win32_ThreadStartTrace

Win32_ThreadStopTrace

- Get-NewProcessEvent.ps1
- Get-ProcessEvent.ps1

- __InstanceCreationEvent
- __InstanceModificationEvent
- __InstanceDeletionEvent

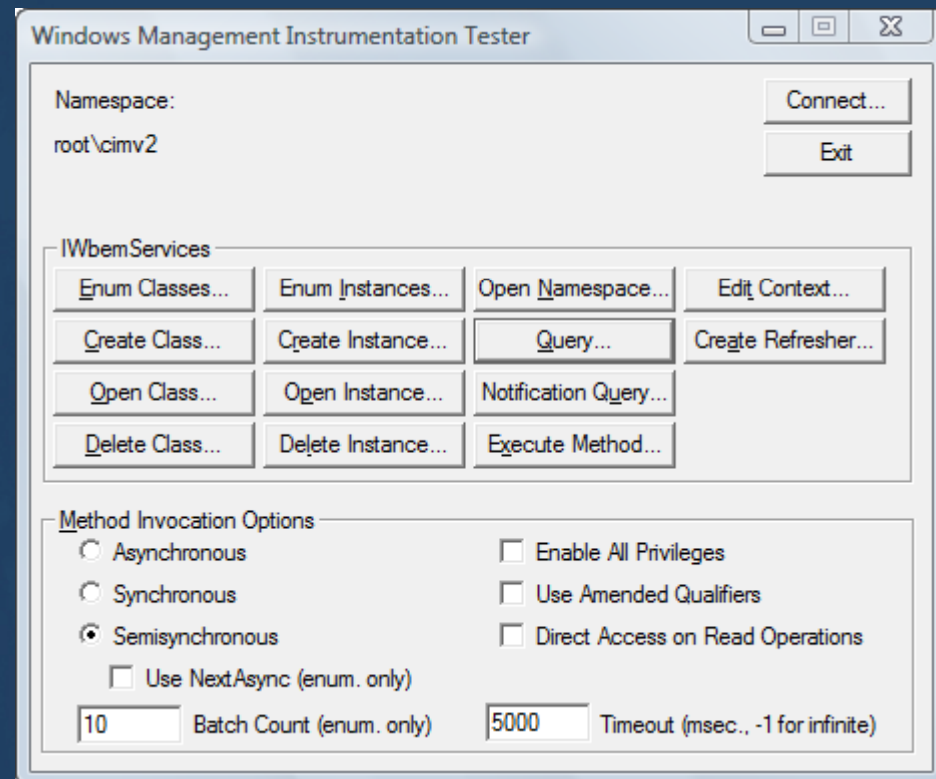
TargetInstance

- TargetInstance ISA *WMIClass*
- __InstanceModificationEvent creates a PreviousInstance object
- Some WMI classes will need additional Where clauses

- Demo-palooza



- Test everything I've shown you in WBEMTest



- Blog.SAPIEN.com
- www.ScriptingAnswers.com
- www.PowerShellCommunity.org
- www.ThePowerShellGuy.com
- www.leeholmes.com/blog

- Windows PowerShell v1.0: TFM 2nd edition (Jones & Hicks)
- Windows PowerShell Cookbook (Holmes)
- Windows PowerShell in Action (Payette)



It's QUESTION TIME !!

Thank you

- jhicks@sapien.com
- Follow me: twitter.com/jeffhicks

