A PowerShell Module for Hyper-V

Author: James O'Neill (http://www.blogs.technet.com/jamesone)

Getting Started

The commands described below are implemented in the form of a PowerShell version 2 Module which is distributed as a single ZIP file, from http://PShyperV.codeplex.com with this document being made available separately.
A previous version was implemented for version 1 of PowerShell but this version and remains available on codeplex. The old version will work on PowerShell V2, but the new version will not work on PowerShell V1. Both versions work with the original release of Hyper-V and the R2 release.

The installation process consists of placing the PowerShell files into a suitable folder.
The module is then imported into a PowerShell session using the command `Import-Module <path\>Hyperv.psd1`
If no path is specified, PowerShell uses the system environment variable PSModulePath and examines each folder it references to find a folder using the name specified for the module, containing a manifest file of the same name. So, for example if you place the files into the c:\users*yourID*\documents\windowsPowershell\modules\HyperV you can just issue the command

```
install-module HyperV
```

(Install module can be shortened to IPMO).
The zip file includes a simple `Install.cmd` file which will make the following changes

- Install PowerShell on Hyper-V Server or Windows Server 2008-R2 Core installations. This produces a harmless error on full installations of Server 2008 , which can be ignored.
- Create a folder under `program files` and add this folder to the PSModulePath environment variable
- Place the files in a subfolder of this a folder
- Set registry settings to give an ideal PowerShell console window and to allow PowerShell to run unsigned scripts.

You may do any or all of these things manually, and good practice says you should review a registry file before allowing it to change your system. The Scripts are not signed and if you wish to implement a policy of signing scripts you should verify the script for yourself and then sign it.
This build includes a menu – similar to the PowerShell configurator found on codeplex at http://psconfig.codeplex.com/ you can launch the menu with the command Show-Hypervmenu
Alternatively when the module has been imported, you can discover its commands by entering the commands

```
get-command -module HyperV
```

or

```
get-command -module Hyperv| get-help | format-table  name,synopsis -auto
```

Finally `get-help` *`Command-Name`* `-full` will give you the online help for a command

PowerShell Verbs and nouns

PowerShell has a standard Verb-Noun format for command names. The introduction of V2 saw a push to standardize verbs to the names published at http://msdn.microsoft.com/en-us/library/ms714428(VS.85).aspx

Standard PowerShell verbs used

Add	Adds a resource to a container, or attaches an item to another item. This verb is paired with the Remove verb. The resource may be a new or existing one.
Compress	Compacts the data of a resource.
Convert	Changes the data from one representation to another.
Copy	Copies a resource to another name or to another container.
Dismount	Detaches a named entity from a location. This verb is paired with the *Mount* verb.
Expand	Officially Powershell defines this as "*Restores the data of a resource that has been compressed to its original state.*" The library uses it for "Increase the size of"
Export	Encapsulates the primary input into a persistent data store, such as a file, or into an interchange format. This verb is paired with the *Import* verb.
Get	Retrieves a resource.
Import	Creates a resource from data that is stored in a persistent data store (such as a file) or in an interchange format.
Invoke	Performs an action, such as running a command or a method.
Merge	Creates a single resource from multiple resources.
Mount	Attaches a named entity to a location. This verb is paired with the *Dismount* verb.
New	Officially this *Creates an empty resource that is not associated with any content.,* the library uses it for creating an free standing object – for example New-VHD creates a new disk image file, without mounting it in a drive – rather than an object which is attached to something (for which *Add* is used)
Out	Sends data out of the environment.
Ping	Determines if a resource is active and if it is responding to requests.
Remove	Deletes a resource from a container. This verb is paired with the *Add* verb.
Rename	Changes the name of a resource.
Restore	Sets a resource to a predefined state.
Save	Preserves data to avoid loss. The library uses it for machine states, and it paired with *Start*, and an alternative to *stop*
Select	Locates a resource in a container. Traditionally, e.g. in SQL, this is used to apply criteria to the container. In the library this is used to when there is a need to prompt for user input to indentify the resource
Set	Creates a resource that contains some data, or replaces data on an existing resource. This verb is paired with the *Get* verb.
Start	Initiates an operation. This verb is paired with the *Stop* verb.
Stop	Discontinues an activity. This verb is paired with the *Start* verb.
Test	Verifies the operation or consistency of a resource.
Update	Sets a resource to a new state.
Wait	Pauses an operation until a specified event occurs.

Non Standard PowerShell Verbs from version 1 of the library

In version 1 of the library some non-standard verbs were used these were

Apply	***Apply***-Snapshot is now replaced with ***Restore-***Snapshot
Choose	Was an attempt to use a different verb for "Ask for user input to identify items". ***Select*** traditionally meant only "Return a subset of data based on the provided criteria", but that definition has been expanded to encompass both kinds of selection So ***choose*** functions have become ***Select*** ones.
Compact	Compact-VHD has become ***Compress***-VHD.
List	***List***-VMState has become ***Get***-VMSummary
Shutdown	Shutdown-VM used the integration components to start a shutdown, the command is now ***Invoke-VMShutdown***
UnMount	Unmount-VHD is replaced with the preferred verb ***Dismount***

To avoid breaking scripts based on the old library, Aliases have been provided to map the new names to the old ones, and these are found in the file **V1 compatibility.ps1** If you do not need this compatibility you may safely remove the reference to it from the Module manifest **HyperV.psd1**.

PowerShell Parameters

PowerShell allows parameters to be passed by name and by position.

```
Set-VMMemory "Tenby" 768MB
```

Is the same as

```
Set-VMMemory -VM "Tenby" -memory 768MB
```

(PowerShell understands suffixes KB, MB, GB, TB when writing numbers)

Generally commands return an *object*, for example `Get-VM "Tenby"` returns a WMI object representing a virtual machine with the display name of "Tenby".

These objects can be stored in PowerShell *variables* or *piped* into other commands; when piping the parameters must be named

The following commands all do the same thing

```
Get-VM "tenby" | Set-VMMemory -memory 768MB
```

and

```
$MyVm = Get-VM "tenby"
Set-VMMemory $myVM 768MB
```

One general design principle is commands will either accept an object (the WMI object for the VM), or accept the information which allows the function to get the object (the name of the VM).

A Second principle in PowerShell is that where it is practical to processed multiple, they can be passed as a comma separated list; for example

```
Set-VMMemory "Tenby","Oxford"  1GB
```

will set the memory on two VMs.

Note that a common mistake in Powershell (especially among those new to it) would be enter parameters which should be distinct, separated with commas, for example

```
Set-VMmemory "Tenby",1GB
```

- this will assume that you want to set the memory on two VMs named "Tenby" and "1GB" but haven't said what you want the memory to be.

In many of the functions, parameters support alternative names, so `Get-VM` takes a `-name` parameter which has an alias of "`VMName`". The names can be shorted provided they are *not ambiguous* (`-V` might mean `-Verbose` or `-VMName`), but `-VM` can only mean `VMName`.

Many built-in PowerShell commands support standard parameters `-whatif` (which runs the command without applying any changes) `-confirm` (which runs the command and prompts before applying changes) `-verbose` (which generates additional output) and `-force` (which suppresses any prompt) – these are widely supported in the module.

Summary of functions provided

Although the library uses 31 different verbs and 69 different nouns (2139 possible combinations), in practice many nouns are only used with one verb, and some verbs are used with only one noun: in practice there are just over 100 commands, and 17 Main nouns and 6 main verbs account for most of the library

	Total Verbs	Add	Get	New	Remove	Select	Set
Tree	2					X	
VHD	9		X	X			
VM	12		X	X	X	X	X
VMCPUCount	2		X				X
VMDisk	3	X	X				X
VMDrive	2	X			X		
VMFloppyDisk	3	X	X		X		
VMHost	2		X				X
VMIntegrationComponent	2		X				X
VMKVP	3	X	X		X		
VMLiveMigrationNetwork	2		X			X	
VMMemory	2		X				X
VMNIC	4	X	X		X	X	
VMNICSwitch	2		X				X
VMNICVLAN	2		X				X
VMRASD	4	X		X	X		X
VMSCSIController	2	X			X		
VMSerialPort	2		X				X
VMSnapshot	7		X	X	X	X	
VMState	2						X
VMSwitch	3		X		X	X	
Grand Total	**122**	**10**	**33**	**11**	**10**	**12**	**12**

In addition there 16 aliases for backwards compatibility

Commands in the library, grouped by function

Virtual Machine Operations

Start-VM	Puts the specified Hyper-V Virtual Machine into the running state.
Stop-VM	Puts the specified Hyper-V Virtual Machine into the Stopped state.
Save-VM	Puts the specified Hyper-V Virtual Machine into the Suspended (saved) state.
Set-VMState	Sets the state of a given VM to running, stopped, suspended etc.
Invoke-VMShutdown	Instructs the OS in specified Virtual Machine to begin an orderly shutdown.
Get-Vm	Returns WMI objects representing Hyper-V Virtual Machines
Select-VM	Allows the user to select a virtual machine from a list.
Get-VMSummary	Returns summary information about running VMs
Get-VMThumbnail	Creates a JPG image of a running VM
Get-VMBuildScript	Outputs a script which will build the VM
New-VM	Creates a new Hyper-V Virtual Machine.
Remove-VM	Removes the specified VM from the server.
Get-VMSettingData	Gets the Setting data object for one or more Virtual Machines.
Set-VM	Sets Name, Notes, Boot order, start-up, shutdown and Recovery options
Set-VMRASD	Modifies VM hardware described by Resource Allocation Setting Data
Remove-VMRASD	Removes VM hardware described by Resource Allocation Setting Data
Add-VMRASD	Adds an item of VM hardware described by Resource Allocation Setting Data
New-VMRASD	Creates a resource Allocation Setting Data object.
Ping-VM	Pings a virtual machine with ICMP Ping
Test-VmHeartbeat	Tests the heartbeat integration component, to see if the VM is alive
Import-VM	Imports a virtual Machine which was previously exported.
Export-Vm	Exports a VM
New-VmConnectSession	Opens a VMConnect session to the specified VM on the specified server.

VM Memory

Set-VMMemory	Sets the memory resources allocated to one or more Virtual Machines.
Get-VMMemory	Gets the memory resources allocated to one or more Virtual Machines.

VM Processors

Get-VMCPUCount	Gets the CPU Resources allocated to one or more Virtual Machines.
Set-VMCPUCount	Sets the CPU Resources allocated to one or more Virtual Machines.
Get-VMProcessor	Gets Virtual CPU devices in use on one or more running Virtual Machines.

VM Serial Ports

Get-VMSerialPort	Gets the virtual Serial ports on one or more Virtual Machines.
Set-VMSerialPort	Connects a serial port on a Virtual Machine to a named pipe.

VM Floppy disk, Hard disk and DVD drives and controllers

Get-VMFloppyDisk	Returns the floppy disk(S) attached to one or more VMs
Add-VMFloppyDisk	Adds a floppy disk to a VM.
Remove-VMFloppyDisk	Removes floppy(s) disk attached to one or more VM(s)
Add-VMSCSIController	Adds a Synthetic SCSI controller to one or more VM(s)
Remove-VMSCSIcontroller	Removes a SCSI controller from one or more virtual Machine(s)
Get-VMDiskController	Returns Disk controllers attached to a VM
Get-VMDriveByController	Returns the drives attached to a given disk controller.
Add-VMDrive	Adds a drive to a controller on the specified VM.
Remove-VMdrive	Removes a drive and/or the disk image in it from a VM
Get-VMDiskByDrive	returns the disk mounted in a given drive
Get-VMDisk	Returns all disks attached one or more VMs,
Add-VMDisk	Adds a disk image to a VM, mounting it in a drive
Set-VMDisk	Changes the Disk mounted in a Drive on a VM
Add-VMPassThrough	Connects a Passthrough disk to a VM
Select-VMPhysicalDisk	Allows the user to choose a disk to become a pass-through disk
Add-VMNewHardDisk	Creates and attaches a new virtual hard disk in one command.

Virtual Hard Disk and Floppy disk images

New-VFD	Creates one or more virtual Floppy disk file(s)
New-VHD	Creates one or more virtual hard disk file(s)
Get-VHDDefaultPath	Gets the default path for Virtual Hard Disk (VHD) files.
Get-VHD	Gets VHD files from a specified folder on a hyper-v server.
Get-VHDInfo	Gets detailed information about one or more VHD files
Get-VHDMountPoint	Returns the mount point for a VHD file, if it is mounted
Mount-VHD	Mounts a VHD file to make it appear to be a disk in the parent partition
Dismount-VHD	Dismounts a previously mounted VHD
Expand-VHD	Increases the size of a VHD
Connect-VHDParent	Reattaches a differencing VHD to its parent
Merge-VHD	Merges VHDs [from snapshots]
Test-VHD	Tests the working state of a disk - for example if its parent can be found
Compress-VHD	Compacts one or more dynamic VHD files
Convert-VHD	Creates a new VHD of a different type based on an existing VHD

VM to Host integration

Get-VMIntegrationComponent	Gets the integration Component data for one or more Virtual Machines.
Set-VMIntegrationComponent	Enables or disables integration Components on one or more Virtual Machines.
Get-VMKVP	Gets Key/Value pairs for one or more Virtual Machines.
Add-VMKVP	Adds Key/Value pairs to be sent to one or more Virtual Machines.
Remove-VMKVP	Removes a Key/Value pair from the set sent to one or more Virtual Machines.

VM Ethernet Cards

Get-VmNic	Returns information about Network Interface Cards
Select-VmNic	Allows the user to select a Network Interface Cards connected toa VM
Add-VmNic	Creates a new legacy or VM-bus Network Interface Card on a Virtual Machine.
Remove-VMNic	Removes a Network Interface Card from a VM
Set-VMNICAddress	Changes the MAC address of a Network Interface Card
Get-VMNicPort	Returns the switch port object for a a Network Interface Card
Get-VMNicSwitch	Returns the switch connected to a a Network Interface Card
Set-VMNICSwitch	Connects a Network Interface Card to a Virtual switch
Get-VMNicVlan	Gets the VLAN ID associated with a Network Interface Card
Set-VMNicVlan	Sets the VLAN ID associated with a Network Interface Card
Get-VMByMACaddress	Discovers which VM owns a particular MAC address

Virtual Network Switches

Select-VMExternalEthernet	Selects an avaialable host network Interface
New-VMPrivateSwitch	Creates a virtual network switch, without a NIC in the parent partition
New-VMInternalSwitch	Creates a virtual network switch, bound to a virtual NIC in the parent partition
New-VMExternalSwitch	Creates a virtual network switch, bound to a physical network card
Remove-VMSwitchNIC	Removes the parent partition NIC associated with a Virtual switch
Remove-VMSwitch	Deletes a virtual network switch
Get-VMSwitch	Returns one or more Virtual Switch objects
Select-VMSwitch	Returns a Virtual Switch objects selected by the user
New-VMSwitchPort	Creates a new port on a virtual network switch

Hyper-V Servers.

Set-VMHost	Configures the settings for the Hyper-V service on the host server
Get-VMHost	Lists Hyper-V servers registered with Active Directory.

Snapshots

Get-VMSnapshot	Returns SnapShots for one or more VM(s)
Get-VMSnapshotTree	Displays all the Snapshots of a VM in a tree view
Select-VMSnapshot	Allows the user to select a snapshot for a VM
New-VMSnapshot	Creates a new snapshot of one more VMs.
Rename-VMsnapshot	Changes the display name of a snapshot
Update-VMSnapshot	Replaces an existing snapshot with a new one
Restore-VMsnapshot	Rolls a VM back (or forward) to a snapshot
Remove-VMSnapshot	Removes one or more snapshots from a VM.

Cluster Operations

Move-VM	Moves one or more VMs between cluster nodes using live migration
Select-ClusterSharedVolume	Allows the user to select a clustered shared volume from a list
Sync-VMClusterConfig	Synchronizes the configuration of a VM across all members of a cluster
Get-VMClustergroup	Returns the cluster group of which the VM is a member
Select-VMLiveMigrationNetwork	Allows the user to select cluster networks to be used live migrating
Get-VMLiveMigrationNetwork	Returns a list of Cluster networks in use for live migrating VMs

User Interface

Show-HypervMenu	Displays a menu to manage hyperv
Show-VMMenu	Displays a menu to manage and individual VM
Show-VMDiskMenu	Displays a menu to manage an individual VM's disks
Show-VHDMenu	Displays a menu to manage hyperv Virtual hard disks
Out-Tree	Outputs objects in a tree format
Select-Tree	Allows the user to select from objects in a tree format
Select-List	Returns an object selected by the user from a table of numbered rows.
Select-Item	Returns a zero-based integer indicating the user's selection

Miscellaneous Admin functions

Test-WMIJob	Checks the status of background WMI jobs
Test-WMIResult	Checks the result returned by calling a WMI method
Convert-DiskIDtoDrive	Converts a logical disk index to a drive letter.
Get-FirstAvailableDriveLetter	Returns a CHAR indicating the first available drive letter.
Wait-ForDisk	Waits for disk to come on line
Test-Admin	Checks to see if the current session has administrator privileges
ConvertTo-Enum	Converts Hash Table data to an enumeration type definition.
Select-Enumtype	Returns a value selected by the user from an enumeration type
New-Zip	Creates a new , empty Zip file
Get-ZIPContent	Returns information about the contents of a Zip file
Add-ZIPContent	Adds content to a Zip file
Copy-ZipContent	Copies content out of a zip file

Detailed Explanation of the functions

Start-VM

Puts the specified Hyper-V Virtual Machine into the Running state.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to start. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
Wait		Specifies that the command should not return until the WMI job is complete. This does not guarantee that the Operating system in the VM is responsive. To check that the OS has loaded use the HeartBeatTimeout Parameter.
HeartBeatTimeOut		If specified the VM is checked for a response from the Heartbeat integration component every 5 seconds until one is found or the timeout expires.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description

Starting a VM restores the contents of memory if they were saved. A background WMI job is created to change the state.

Examples

```
Start-vm "London-DC" -HeartBeatTimeOut 300
```

Starts the VM named "London-DC" and waits up to 5 minutes for a heartbeat before running the next command

Notes

Start-VM Calls set-VMstate passing it the VM and server it received.

See Also:

Set-VmState
Stop-VM

Stop-VM

Puts the specified Hyper-V Virtual Machine into the Stopped state.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to stop. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
Wait		Specifies that the command should not return until the WMI job is complete.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Stopping a VM powers it off without saving its state. If it is in a saved state, the saved state information is deleted. A background WMI job is created to change the state.

Examples
```
Stop-VM (Select-VM -Server "JAMES-2008" -Multiple)
```
Prompts the user to select one or more of the VMs on the server named "JAMES-2008", and stops the selected VMs.
```
Get-VM -running | Stop-VM
```
Stop all VMs that are in the Running state.

Notes
Stop-VM calls Set-VMstate passing it the VM and Server parameters it received.

See Also:
Set-VmState
Start-Vm

Save-VM
Puts the specified Hyper-V Virtual Machine into the Suspended (saved) state.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to save. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
Wait		Specifies that the command should not return until the WMI job is complete.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Saving a VM stores the contents of memory before powering it off. A background WMI job is created to change the state.

Examples
```
Save-VM (Select-VM -Server "JAMES-2008" -Multiple)
```
Prompts the user to select one or more of the VMs on the server named "JAMES-2008", and saves the selected VM(s).
```
Get-VM -running | Save-VM
```
Saves all VMs that are in the Running state.

Notes
Save-VM has an alias of suspend VM It calls Set-VMstate passing it the VM and Server parameters it received.

See Also:
Set-VmState

Set-VMState
Sets the state of a given VM to Running, Stopped, Suspended etc.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
State *Required*		The desired new state If a valid integer or state name is passed it is converted to the VMState Enum

Server	Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out

Examples

```
Get-VM -Name "Core-%" | Set-VMState -State [VMState]::Running
```

Starts all VMs with names that start with "CORE-".

```
Set-VMState -VM "Core","Tenby" -Server "JAMES-2008" - State "Running"
```

Starts the VMs named "Core" and "Tenby" on the server "JAMES-2008" to the running state using the name of the state.

Notes

VMState is intended for use by Start-, Stop- and Save- VM, but may be called directly

See Also:

RequestStateChange Method on MSDN

Invoke-VMShutdown

Instructs the OS in specified Virtual Machine to begin an orderly shutdown.

Parameters

VM *Required* (Input from pipeline) Wildcards	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Reason	Sent to the guest OS for information about why the remote shutdown was started.
Server	Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
Wait	Specifies that the command should not return until the WMI job is complete.
PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description

Stopping a VM does not close down the operating system inside it. The Hyper-V integration components include a Shutdown component which tells guest OS to start a clean shut down. This command invokes that component.

Examples

```
Invoke-VMShutdown (Select-VM -Server "JAMES-2008" -Multiple)
```

Prompts the user to select one or more of the VMs on the server "JAMES-2008", and shuts down the selected VMs.

```
Get-VM -Running | Invoke-VMShutdown
```

Shuts down all VMs that are in the Running state on the local server

See Also:

ShutdownComponent - InitiateShutdown Method on MSDN

Ping-VM

Pings a virtual machine with ICMP Ping

Parameters

VM *Required* (Input from pipeline) Wildcards	The Virtual Machine(s) to check. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.

Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.

Description
First attempts to use integration components to discover the Fully Qualified Domain Name of the VM, then attempts to ping that FQDN using ICMP from the machine where the command is running.

Examples
```
Ping-VM "Tenby" -server james2008
```
Attempts to ping from the local machine to the VM named "Tenby" on the server James-2008. This relies on the integration components being present and the FQDN they return being resolvable on the local machine.
```
get-vm -r | foreach-object {if ((Ping-VM $_).statusCode -ne 0) {"$($_.elementname) is inaccessible"} }
```
Gets the running VMs on the local server, pings each of them and outputs a message for any which are running but can't be pinged.

See Also:
Get-VMKvp
WIn32_PingStatus WMI object on MSDN

Test-VmHeartbeat
Tests the HeartBeat integration component, to see if the VM is alive

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to check. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
HeartBeatTimeOut		If specified the VM is checked for a response from the heartbeat integration component every 5 seconds until one is found or the timeout expires.

Description
This is not a 100% reliable test because the OS in the VM may not have integration components available, they may not be installed or may be disabled. But where they are installed the provided the best indication of VM state.

Examples
```
start-vm "London DC" ; Test-vmheartBeat "London DC" -Timeout 300; start-vm "London SQL"
```
Starts the VM named "London DC" and waits up to 5 minutes for its heartbeat(this is the same as calling Start-vm "London-DC" -HeartBeatTimeOut 300). Then starts VM "London SQL"

See Also:
MSVM_HeartbeatComponentClass on MSDN

Get-Vm
Returns WMI objects representing Hyper-V Virtual Machines

Parameters
Name	(Input from pipeline) Wildcards	Specifies the name (or name pattern) to look for. WMI wild cards use % , but * will be converted to % to allow familiar style to be used. Can be used with an Alias of VMName (which may be shorten to VM).
Server		Specifies the name of the Hyper-V Server to query for Virtual Machines. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
Suspended		Only Return Virtual Machines that are in the SUSPENDED state.
Running		Only return Virtual Machines that are in the RUNNING state.
Stopped		Only return Virtual Machines that are in the STOPPED state.

Description
Gets Virtual machine objects on one or more Hyper-V servers, the selection can be based on name or state of the VM.

Examples
```
Get-VM
```
Returns WMI MSVM_ComputerSystem objects for all Virtual Machines on the local server (the parent partition is filtered out).
```
Get-VM -Name "Windows 2008 Ent Full TS"
```
Returns a single WMI MSVM_ComputerSystem object for the VM named "Server 2008 ENT Full TS"
```
Get-VM -Name "%2008% -Server "JAMES-2008"
```
Returns WMI MSVM_ComputerSystem objects for the VMs with "2008" in their names from the server named JAMES-2008.

See Also:
New-VM
Remove-VM
Set-VM
MsVM_ComputerSystem Class on MSDN

Select-VM
Allows the user to select a virtual machine from a list.

Parameters
Server Specifies the Hyper-V server on which Virtual Machines reside. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.

Multiple If specified allows multiple VMs to be selected from the list. If omitted only a single VM may be selected.

Description
If only one VM is available then it is returned otherwise a list of VMs is displayed and the user is prompted to make a selection.

Examples
```
Select-vm -multiple
```
Lets the user select one or more VMs from a list of those on the local machine
```
Select-vm -Server James-2008,Jackie-2008
```
Lets the user select a single VM from a list of those on the cluster nodes named "James-2008" and "Jackie-2008"

Get-VMSummary
Returns summary information about running VMs

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to provide information about. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.

Description
For each VM a custom object is returned with the following properties Host The Hyper-V server hosting the VM VMElementName The Display Name for the VM Name The GUID which uniquely identifies the VM CreationTime The Time the VM was created EnabledState The current state of the VM as a Enum Notes The Notes field for the VM CPUCount The number of CPUs assigned to the VM CPULoad The instantaneous CPU load on the VM CPULoadHistory An Array of CPU loads for the last 100 second MemoryUsage Memory used by the VM Heartbeat The State of the VM heartbeat as a number HeartbeatText The State of the VM heartbeat as converted to text Uptime The VM uptime in Milliseconds UptimeFormatted The VM uptime formatted as Hours, minutes and seconds GuestOS The Guest OS Identity string returned by KVP exchange Snapshots The number of Snapshots which exist of the virtual Machine Jobs The asynchronous jobs associated with the VM FQDN The fully qualified domain name returned by KVP exchange IpAddress The IP address found when attempting to ping the FQDN

Examples

```
Get-VMSummary -server james-2008,jackie-2008 | ft -a
```

Outputs formatted status for all VMs on the servers named "James-2008" and "Jackie-2008" (subject to the constraints of screen width)

```
Get-VMSmmary "Windows 2008 Ent Full TS"
```

Outputs status for the VM named "Server 2008 ENT Full TS" on the local server

---

Get-VMThumbnail

Creates a JPG image of a running VM

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to capture a picture of. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
width		The width of the Image in Pixels (default 800)
Height		The Height of the Image in Pixels (default 600)
Path		The path to save the image to, if no name is specified the VM name is used. If no directory is included the current one is used
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used.
Passthru		Specifies that the Bitmap image should be returned as an object and not saved as a file

Description

Creates a JPG image of a running VM, at a given width and height which can either be saved to a file or passed on to another command

Examples

```
Get-VMJPEG core
```

Gets a 800x600 jpeg for the machine named core, and writes it as core.jpg in the current folder.

```
While ($true) { Get-VMJPEG -vm "core" -w 640 -h 480 ` -path ((get-
date).toLongTimeString().replace(":","-") + ".JPG") Sleep -Seconds 10 }
```

Creates a loop which continues until interrupted; in the loop creates an image of the VM “Core” with a file name based on the current time, then waits 10 seconds and repeats.

```
Get-vm -Running -server "James-2008" | Get-VMJPEG -w 320 -h 240 -path images
```

Discovers running VMs on Server named "James-2008", for each writes a 320x240 size image to the folder named images.

See Also:

GetVirtualSystemThumbnailImage method on MSDN :

Get-VMBuildScript

Outputs a script which will build the VM

Parameters

VM	(Input from pipeline) Wildcards	The Virtual Machine(s) to get build scripts for. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe. If no VM is specified the script is out put for all VMs
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.

Description

Outputs New-VM, add-VMxxxx, Set-VMxxx commands to recreate the same VM

Examples

```
Get-VMBuildScript "Tenby"
```

Outputs a script to build the VM named "Tenby"

New-VM

Creates a new Hyper-V Virtual Machine.

Parameters

Name *Required*	(Input from pipeline)	Specifies the display name of the new Virtual Machine. Hyper-V does not require these names to be unique.
Path		The location for VM files, including snapshots. Hyper-V has a default location, which will be used if no path is specified
Server		Specifies the Hyper-V server on which the Virtual Machine will reside. By default "." , the local computer, is used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description

Creates a new Hyper-V Virtual Machine, and returns a WMI object representing it. After the machine is created you should set memory, CPU count, network adapters and disks

Examples

```
New-VM -Name "Tenby" -Server "James-2008"
```

Creates a new VM named "Tenby" on the server named "James-2008".

See Also:

Get-VM
Remove VM
DefineVirtualSystem Method on MSDN

Remove-VM

Removes the specified VM from the server.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to remove. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description

Delete a VM and its snapshots, but does not delete its Virtual Hard Disk files

Examples

```
Remove-VM -VM "Tenby"
```

Removes the VM named "Tenby" from the local server

See Also:

New-VM
DestroyVirtualSystem Method on MSDN

Get-VMSettingData

Gets the Setting data object for one or more Virtual Machines.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. If no VM is specified the information will be returned for all VMs on the server. It may be passed via the pipe.

Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description
Returns an MSVM_VirtualSystemSettingData object describes the boot devices for a VM, its display name and notes.

See Also:
MSVM_VirtualSystemSettingData object on MSDN

Set-VM
Sets Name, Notes, Boot order, start-up, shutdown and Recovery options

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to configure. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Name		The new display name for the VM
BootOrder		An array of enums representing boot devices. If valid integers or strings are passed they will be converted to enums
Notes		The new contents of the VM Notes field
AutoRecovery		An enum representing a recovery action. If valid integers or strings are passed they will be converted to enums
AutoShutdown		An enum representing a Shutdown action. If valid integers or strings are passed they will be converted to enums
AutoStartUP		An enum representing a start-up action. If valid integers or strings are passed they will be converted to enums
AutoDelay		The number of seconds to delay an automated startup (to allow other VMs to boot first)
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Sets Name, Notes, Boot order, start-up, shutdown and Recovery options

Examples
```
Set-vm $vm -bootorder CD,IDE,net,Floppy
```
Sets the boot order for the machine whose config is $vm to CD, IDE, Network, Floppy.
```
set-vm -vm "core" -autoStart AlwaysStartup
```
Sets the VM named core on the local host to start whenever the host OS boots
```
Set-vm "CORE-%" -bootorder CD,IDE,net,Floppy -autoStart
```
Take all the machines whose names begin CORE- and set their boot order and start-up action

See Also:
Get-VMSettingData
Msvm_VirtualSystemGlobalSettingData WMI object on MSDN
MSVM_VirtualSystemSettingData WMI object on MSDN

Set-VMRASD
Modifies virtual hardware described by Resource Allocation Setting Data to a VM

Parameters

VM *Required*	(Input from pipeline)	The Virtual Machine to modify, as a WMI object

RASD *Required* — The Resource Allocation Settings Data WMI Object representing the updated hardware.

PSC — THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command

Force — Ensures that the user is not prompted before the action is carried out

Description
Exists to support other commands. Not intended to be called from the command line

See Also:
ModifyVirtualSystemResources method of MSVM_VirtualSystemManagementService on MSDN

New-VMRASD
Creates a resource Allocation Setting Data object.

Parameters
Restype *Required* — Resource type

ResSubType — Resource Sub type

Server *Required* — Specifies the Hyper-V server on which to create the object. By default "." , the local computer, is used.

Description
Exists to support other commands. Not intended to be called from the command line

See Also:
MSVM_AllocationCapabilities on MSDN
MSVM_SettingsDefineCapabilities on MSDN

Add-VMRASD
Adds virtual hardware described by Resource Allocation Setting Data to a VM

Parameters
VM *Required* — (Input from pipeline) The Virtual Machine to modify, as a WMI object

RASD *Required* — The Resource Allocation Settings Data WMI Object representing the new hardware.

PSC — THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command

Force — Ensures that the user is not prompted before the action is carried out

Description
Exists to support other commands. Not intended to be called from the command line

See Also:
AddVirtualSystemResources method of MSVM_VirtualSystemManagementService on MSDN

Remove-VMRASD
Removes virtual hardware described by Resource Allocation Setting Data to a VM

Parameters
VM *Required* — (Input from pipeline) The Virtual Machine to modify, as a WMI object

RASD *Required* — The Resource Allocation Settings Data WMI Object representing the hardware to be removed

PSC — THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to

		the command
Force		Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Exists to support other commands. Not intended to be called from the command line

See Also:
RemoveVirtualSystemResources method of MSVM_VirtualSystemManagementService on MSDN

Import-VM
Imports a virtual Machine which was previously exported.

Parameters

Paths *Required*	(Input from pipeline)	string which contains the name of a VHD file or an array which contains these. If no Path is passed as a parameter, the Import-VM looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path.
Server		Specifies the Hyper-V server on which the import is to be performed. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
ReImportVM		If specified the ZIP files for an import are copied back to their location and the VM specified is deleted. Note that no check is performed to confirm the deleted VM is the same one which will be impoted
ReUseIDs		Indicates the re-imported VM should use the same IDs as when it was exported. This can cause a conflict if the old VM still exists
Wait		Specifies that the command should not return until the WMI job is complete.
Preserve		Specifies that a zip file should be made containing the files which are consumed at input.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Imports a virtual Machine which was previously exported.

See Also:
Export-VM
ImportVirtualSystem method on MSDN

Export-Vm
Exports a VM

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to export. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Path *Required*		The location where the exported VM files will be placed.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
CopyState		If present indicates that the state, such as virtual hard disks, saved state files, and memory content files, should be exported.
Wait		Specifies that the command should not return until the WMI job is complete.
Preserve		Specifies that a zip file should be made containing the files which are consumed at input

PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out

Description
The command supports the option to the copy the state of the VM (as in the GUI) And can also make a backup copy of the files which will be consumed during import

See Also:
Import-VM
ExportVirtualSystem method on MSDN

New-VmConnectSession
Opens a VMConnect session to the specified VM on the specified server.

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to connect to. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.

Description
VMConnect uses RDP to connect to a "Virtual KVM switch" on the virtualized graphics card, keyboard and mouse of the Virtual Machine, rather than any Remote Desktop Service running in its Operating System. The connection is made by invoking %programFiles%\hyper-v\VMConnect.exe. the same program which is called by the GUI Hyper-V manager

Examples
```
New-VMConnectSession -VM "tenby"
```
Launches a VMConnect session to a Virtual Machine named "Tenby" on the local server.
```
New-VMConnectSession -VM $tenby -server James-2008
```
Launches a VMConnect session to the VM described by the $tenby variable, located on the server James-2008
```
$vmConnectPID = New-VMConnectSession -vm "tenby" ; Start-Sleep -Seconds 60 ; Stop-Process -Id $vmConnectPID
```
Launches a VMConnect session to the VM named "Tenby" on the local server, waits one minute, and then closes VMConnect.

Notes

Get-VMMemory
Gets the memory resources allocated to one or more Virtual Machines.

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. If no VM is specified the information will be returned for all VMs on the server. It may be passed via the pipe.
Server *Required*		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer,, is used. An array of server names may be passed but wildcards can not be used.

Examples
```
Get-VMMemory -VM "Tenby" -Server "JAMES-2008"
```
Gets the memory allocated to the VM named "Tenby" on the host "JAMES-2008".

See Also:
Set-VMMemory
MSVM_MemorySettingData object on MSDN

Set-VMMemory

Sets the memory resources allocated to one or more Virtual Machines.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe..
Memory *Required*		Specifies the amount of memory that should be allocated to the Virtual Machine. If greater than 2MB it is assumed expressed in bytes, otherwise it is treaded as MB
Server *Required*		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Examples

```
Get-VM -Name "Core-%" | Set-VMMemory -Memory 1073741824
```

Allocates 1GB of memory to all of the VMs on the local server with a name that starts with "CORE-".

```
Set-VMMemory -VM "Tenby" -Memory 1.5GB -Server James-2008
```

Allocates 1.5GB of memory to the VM named Tenby on the server JAMES-2008

See Also:

Get-VMMemory

MSVM_MemorySettingData object on MSDN

Get-VMCPUCount

Gets the CPU Resources allocated to one or more Virtual Machines.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects.If no VM is specified the information will be returned for all VMs on the server. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Examples

```
Get-VMCPUCount core
```

Returns the CPU settings for the VM named core on the local server.

See Also:

Set-VMCPUCount

MsVM_ProcessorSettingData object on MSDN

Set-VMCPUCount

Sets the CPU Resources allocated to one or more Virtual Machines.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe..
CPUCount		The Number of Virtual Processors to assign to the VM. It must be in the range 1-4. It can not be greater than the number of physical cores in the host. Some OSes support fewer processors.
Limit		The maximum time this processor is allowed to use. A VM defaults to a limit of 100% unless configured otherwise

Reservation	Amount of CPU time reserved for a VM. A VM defaults to a reservation of 0% unless configured otherwise
Weight	Relative Weighting used to assign CPU time to this VM when the host processor is fully utilized
Server	Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out

Description
Sets the CPU Resources allocated to one or more Virtual Machines.

Examples
```
Set-VMCPUCount "tenby" 2 -Server "James-2008"
```
Assigns 2 CPUs to the VM named Tenby on Server James-2008
```
Get-vm Core-% | Set-VMCPUCount -CPUCount 2
```
Gives 2 CPUs to all VMs on the local machine whose names begin CORE-

See Also:
Get-VMCpuCount
MsVM_ProcessorSettingData object on MSDN

Get-VMProcessor
Gets the virtual CPU devices connected to one or more Virtual Machines.

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. If no VM is specified the information will be returned for all VMs on the server. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description
Get-VMCPUCount returns the number of virtual processors assigned to a VM: Get-VMProcessor returns the objects representing those processors when the VM is running.

Examples
```
Get-VMProcessor "core"
```
Returns the Virtual CPU objects for the VM named core on the local server.

See Also:
Get-VMCPUCount
MsVM_Processor object on MSDN

Get-VMSerialPort
Gets the virtual Serial port devices connected to one or more Virtual Machines.

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
PortNumber		The Serial Port number, either 1 or 2. If not specified both serial ports will be returned
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description
Gets the virtual Serial port devices connected to one or more Virtual Machines.

Examples
```
Get-VMSerialPort "core"
```
Gets the settings for both seral ports of the VM named "Core" on the local server

See Also:
Set-VMSerialPort
Msvm_SerialPort Class on MSDN

Set-VMSerialPort
Connects a serial port on the specified Hyper-V Virtual Machine to a named pipe.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe..
PortNumber		The port to connect to (1 or 2)
Connection		The named-pipe path to the connect the port to
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Connects a serial port on the specified Hyper-V Virtual Machine to a named pipe.

Examples
```
Set-VMSerialPort "CORE" 2 "\\.\PIPE\WIBBLE"
```
Connects serial port 2 on the VM "Tenby" on the local server

See Also:
Get-VMSerialPort
Msvm_SerialPort Class on MSDN

Get-VMFloppyDisk
Returns the floppy disk(S) attached to one or more VMs

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description
Returns MSVM_ResourceAllocationSettingData objects for the Floppy disk(s). The accompanying format.ps1xml file formats MSVM_ResourceAllocationSettingData objects to show the VM's element name, the resource's Element name, its subtype (Microsoft virtual Floppy disk) and its connection (the path to the VFD file)

Examples
```
Get-VMFloppyDisk (get-vm -server james-2008) | foreach {$_.connection}
```
Produces a list of all the VFD files in the floppy drives of the VMs on the server James-2008

See Also:
Add-VMFloppyDisk
Remove-VMFloppyDisk

Add-VMFloppyDisk

Adds a floppy disk to a VM.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Path		
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description

HyperV VMs are created a single floppy drive. Add-VMFloppyDisk mounts a virtual Floppy disk (VFD) image file in the drive.

Examples

```
add-VMFloppyDisk $core "C:\Users\Public\Documents\Microsoft Hyper-V\Blank Floppy Disk\blank.VFD"
```

Adds a floppy disk to the machine Pointed to by $Core , the VFD being Blank.vfd.

Remove-VMFloppyDisk

Removes floppy(s) disk attached to one or more VM(s)

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to modify . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description

Removes any floppy disk attached to the VM(s) provided. If no Floppy is present no error occurs

See Also:

Get-VMFloppyDisk
Add-VMFloppyDisk

Add-VMSCSIController

Adds a Synthetic SCSI controller to one or more VM(s)

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Name		The display name for the controller. The Default name is held in the preference variable lstr_VMBusSCSILabel and can be changed.

Server	Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out

Description
Adds a Synthetic SCSI controller to one or more VM(s)

Examples
```
Add-VMSCSIController $tenby
```
Adds a VMBus SCSI Controller to VM whose info is in $tenby
```
Get-vm Core-% -server james-2008 | Add-VMSCSIController
```
Adds a SCSI Controller to all VMs whose names begin CORE- on the server named "James-2008"

Remove-VMSCSIcontroller
Removes a SCSI controller from one or more virtual Machine(s)

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
ControllerID		The Zero-based ID of the intended disk controller. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Removes a SCSI controller from one or more virtual Machine(s)

Examples
```
Remove-VMSCSIController $tenby 0
```
Remove the first VMBus SCSI Controller to VM whose info is in $tenby

See Also:
Get-VMDiskcontroller
Add-VMSCSIController

Get-VMDiskController
Returns Disk controllers attached to a VM

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
ControllerID		The Zero-based ID of the intended disk controller
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

SCSI	If specified indicates that only SCSI devices is intended. If neither IDE nor SCSI is specified (or both) then both types of controller are returned
IDE	If specified indicates that only IDE devices is intended. If neither IDE nor SCSI is specified (or both) then both types of controller are returned

Description
Returns MSVM_ResourceAllocationSettingData objects for the disk controller(s). The accompanying format.ps1xml file formats MSVM_ResourceAllocationSettingData objects to show the VM's element name, the resource's Element name, its subtype (Microsoft Emulated IDE Controller or Microsoft Synthetic SCSI controller) and its connection (empty for disk controllers)

Examples
```
Get-VM -server James-2008| Get-VMDiskController -IDE -SCSI
```
Returns all the DiskControllers for all the VMs on Server James-2008
```
Get-VMDiskController $Tenby -SCSI -controllerID 0
```
Returns SCSI controller 0 in the VM pointed to by $Tenby

Notes
To return the nth SCSI and nth IDE controller BOTH switches must be specified as well as the controller ID.

See Also:
Get-VMDriveByController
Add-VMScsicontroller

Get-VMDriveByController
Returns the drives attached to a given disk controller.

Parameters
Controller *Required*	(Input from pipeline)	The Controller parameter is a single MSVM_ResourceAllocationSettingData WMI object which represents an IDE or SCSI controller or an array which contains these.
Lun	Wildcards	The Zero-based Logical Unit number which identifies a drive on a disk controller. If no LUN is passed, all drives are returned

Description
Returns the drives attached to a given disk controller.

Examples
```
Get-VMDiskController "Tenby" -server "James-2008" -IDE -controllerID 0 | Get-
VMDriveByController
```
Gets IDE disk controller 0 for the VM named "Tenby" on the server named "james-2008" and then gets the Drives attached to it
```
$drive=Get-VMDriveByController $controller –lun 0
```
Gets the first disk attached to the Controller specified by $controller and stores the result in $drive

See Also:
Get-VMDiskController
Get-VMDiskByDrive

Add-VMDrive
Adds a drive to a controller on the specified VM.

Description
Adds a drive (either a DVD or Hard disk drive) to the specified LUN slot on the specified IDE controller of a VM

Examples
```
Add-VMDRIVE "tenby" 1 1 -server james-2008
```
Adds a virtual DVD to IDE controller 1, disk slot 1 on the VM named Tenby on Server James-2008
```
Add-VMDRIVE $tenby 0 3 -SCSI
```
Adds a Virtual Hard Disk drive to SCSI controller 0, LUN 3 on the VM whose info is in $tenby
```
Get-vm Core-% | Add-VMDRIVE -controllerID 0 -lun 1 -DVD
```
Adds a DVD drive to IDE controller 0, disk slot 1 on all the VMs on the local server whose name begins with CORE-

Remove-VMdrive

Removes a drive and/or the disk image in it from a VM

Description

Removes a drive and/or the disk image in it from a VM

Examples

```
Remove-VMdrive "Tenby" 0 1 -SCSI -DiskOnly -Server "James-2008"
```

Removes the disk mounted in Scsi Controller 0, Lun 1 on the VM named tenby on the server named james-2008

```
Remove-VMdrive $Core 1 1 -IDE
```

Removes the disk drive and any associated disk from IDE controller 1, Lun 1 on the VM point to by $core.

```
Remove-VMdrive "Tenby" 0 1 -SCSI -DiskOnly -Server "James-2008"
```

Remove the Disk from the drive at device 1 of SCSI controller 0 of the VM named "Tenby" on the Server "James-2008"

```
Remove-VMdrive $Core 1 1 -IDE
```

Remove the Disk and drive at device 1 of IDE controller 1 in the VM pointed to by $core

```
get-vmdisk | where {$_.diskpath -eq "C:\Windows\system32\vmguest.iso"} | foreach
{remove-vmDrive -diskonly -vm $_.vmelementName -c $_.controllerId -L $_.driveLun}
```

Find all instances of the guest extensions ISO and remove them.

See Also:

Add-VMDrive
Get-VMDisk
Get-VMDiskController
Get-VMDriveByController

Get-VMDiskByDrive

Returns the disk mounted in a given drive

Parameters

Drive *Required*	(Input from pipeline)	A single MSVM_ResourceAllocationSettingData WMI object which represents a hard-disk drive or DVD drive or an array which contains these. If an array is passed the function calls itself recursively for each item.

Description

Returns the disk mounted in a given drive

Examples

```
Get-VMDiskController "Tenby" -server "James-2008" -IDE -controllerID 0 | Get-
VMDriveByController | get-vmdiskByDrive
```

Gets the disks in the drives attached to IDE controller 0 in the VM named Tenby on Server James-2008

```
get-vmdiskByDrive $drive
```

Gets the disk in the drive pointed to by $drive

See Also:

Get-VMDriveByController

Add-VMDisk

Adds a disk image to a VM, mounting it in a drive

Description

Mounts a disk (usually a VHD file, but also an ISO file or the pass-through pat to an optical disk) into an existing logical drive on a VM

Examples

```
Add-VMDisk $tenby 0 1 "C:\update.iso" -DVD
```

Adds a DVD image C:\update.iso, to disk 1, controller 0 on the VM whose info is in $tenby

```
Add-VMDisk $tenby 0 0 ((get-VHDdefaultPath) +"\tenby.VHD")
```

Adds a virtal hard disk named tenby.VHD in the Default folder , to disk 0, controller 0 on the VM whose info is in $tenby

---

Get-VMDisk

Returns all disks attached one or more VMs,

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe. If not specified al VMs are queried.
snapshot		If the snapshot switch is included disks attached to snapshots are included

Description

The function returns objects with 10 properties VMElementname The display name of the VM VMGUID Its GUID (ComputerSystem's "name" property) ControllerName The Element name property of the disk controller object ControllerInstanceID The WMI Instance ID of the disk controller object ControllerID The logical controller number DriveName The element Name of the drive object DriveInstanceID The WMI InstanceID for the drive DriveLun The Slot occupied by the drive (disk's address property) of the disk object) DiskPath The path to the disk (disk's connection property) DiskName The Element name property of the disk object DiskInstnaceID The WMI Instance ID for the disk

Examples

```
Get-VMDisk (Select-vm -server "James-2008" -multi) | format-table -autosize -property
VMname, DriveName, @{Label="Conected to"; expression={"{0,5} {1}:{2}" -f
$_.Controllername.split(" ")[0], $_.ControllerID,$_.DriveLun }} , DiskPath
```

Gets the disks (without snapshot disks) for the chosen VMs on Server James-2008, and returns the result as a table with the VMName, Drive name, the connection path, and the DiskPath.

```
Get-VMDisk | foreach {$_.diskpath}
```

Returns a list of disks in use

```
Get-VMDisk | where {$_.ControllerName -match "^IDE"}
```

Returns a list of disks attached to IDE controllers

See Also:

Get-VMDiskController
Get-VMDriveByController
Get-VMDiskByDrive
Get-VHDInfo

Set-VMDisk

Changes the Disk mounted in a Drive on a VM

Description

Changes the Disk mounted in a Drive on a VM

Examples

```
Set-VMDisk Tenby 0 1 (Get-WmiObject -Query "Select * From win32_cdromdrive Where
ID='D:' " ).deviceID
```

Sets the DVD on controller 0, device 1 for the VM named "Tenby" on the local Server to point to physical drive D: on the host.

```
Set-VMDisk $Core 0 0 "\\?\Volume{d1f72a03-d43a-11dc-8bf1-806e6f6e6963}\Virtual Hard
Disks\Core.VHD"
```

Sets the Disk on controller 0, device 0 of the VM pointed to by $core to Core.VHD using GUID (not drive letter) notation

---

Select-VMPhysicalDisk

Allows the user to make a choice between disks to become a pass-through disk

Description

Allows the user to make a choice between disks to become a pass-through disk. If more than one disk is available a menu will be displayed.

Examples

```
Add-VMPassThrough $vm 0 1 (Select-VMPhysicaldisk)
```

If there are no physical disks available to Hyper-v (i.e. connected and marked offline), this will do nothing. if there is exactly one disk that is used. If there is more than one the user is prompted to select one The physical disk is bound to IDE controller 0 , Lun 1 on the VM held in $VM

See Also:
Add-VMPassthroughDisk
Msvm_ResourcePool Class on MSDN

Add-VMPassThrough
Connects a Passthrough disk to a VM

Description
Hyper-V allows disks which are seen as Off-line by the parent partition to be connected to a LUN in VM, bypassing the use of VHD files Add-VMPassthrough makes the connection

Examples
```
Add-VMPassThrough $vm 0 1 (Select-VMPhysicaldisk)
```
If there are no physical disks available to Hyper-v (i.e. connected and marked offline), this will do nothing. if there is exactly one disk that is used. If there is more than one the user is prompted to select one The physical disk is bound to IDE controller 0 , Lun 1 on the VM held in $VM

Add-VMNewHardDisk
Creates and attaches a new Virtual Hard Disk in one command.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
ControllerID		The Zero-based ID of the intended disk controller
LUN		The Zero-based Logical Unit number which identifies a drive on a disk controller
VHDPath		The path to the VHD file. If not specified the name of the VM will be used. If the .VHD file extension is omitted it will be added and if only a file name is passed the server's default folder (not the current working directory) will be assumed.
Size		The Size of the disk in bytes
ParentVHD		To create a differencing disk the parent-disk path must be specified and the size and fixed parameters are then ignored
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Fixed		Indicates a fixed disk is required. If neither ParentVHD nor fixed is specified a dynamic VHD will be created
SCSI		If specified indicates that a SCSI device is intended, if not specified IDE is used
Force		Ensures that the user is not prompted before the action is carried out

Description
Add-VMNewHardDisk combines New-VHD, Add-VMDrive and Add-VMDisk. It calls NEW-VHD with ParentDisk, Size, Fixed, VHDPath and Server parameters. If no VHDPath is passed the VHD is created in the default folder using a name which matches the VM. It then calls Add-VMDrive with the Server, VM, SCSI, Controller and LUN parameters to create the drive Finally it calls Add-VMDisk with VM, controllerID, Lun, VHDPath, Server and SCSI parameters. Note that the function assumes that the controller exists.

Examples
```
Add-VMNewHardDisk -vm $vm -controllerID 0 -lun 3 -VHDpath "$(get-
VHDdefaultPath)\foo31.VHD" -size 20gb -scsi
```
Adds a 20GB dynamic disk, named foo31.VHD in the default folder, to the VM defined in $VM, on SCSI controller 0, Lun 3.

New-Vfd
Creates one or more virtual Floppy disk file(s)

Parameters

VFDPaths *Required*	(Input from pipeline)	The path to the VFD file is required. If the .VFD file extension is omitted it will be added and if only a file name is passed the server's default VHD folder will be assumed (Hyper-V does not have the concept of a default VFD folder).
Server		Specifies the Hyper-V server on which a the VFD will be created By default "." , the local computer, is used.
Wait		Indicates that the command should wait until the disk creation job completes.

Description
Creates one or more virtual Floppy disk file(s)

Examples
```
New-VFD "Floppy.VFD"
```
Creates a new floppy disk named FLOPPY.VFD in the default folder.

See Also:
Add-VMFlopyDisk
CreateVirtualFloppyDisk Method of the Msvm_ImageManagementService Class on MSDN

New-VHD
Creates one or more Virtual Hard Disk file(s)

Parameters

VHDPaths *Required*	(Input from pipeline)	The path to the VHD file is required. If the .VHD file extension is omitted it will be added and if only a file name is passed the server's default folder (not the current working directory) will be assumed.
Size		The Size of the disk in bytes
parentVHD		To create a differencing disk the ParentDisk path must be specified and the size and fixed parameters are then ignored
Server		Specifies the Hyper-V server on which a the VHD will be created By default "." , the local computer, is used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Fixed		Indicates a fixed disk is required. If neither ParentVHD nor fixed is specified a dynamic VHD will be created
Wait		Indicates that the command should wait until the disk creation job completes. This can take several minutes in the case of large fixed disks.
Force		Ensures that the user is not prompted before the action is carried out.

Description
Creates one or more Virtual Hard Disk file(s)

See Also:
Add-VMDisk
Get-VMDisk
Add-VMNewHardDisk
Msvm_ImageManagementService Class on MSDN

Get-VHDDefaultPath
gets the default path for Virtual Hard Disk (VHD) files.

Parameters

Server	Specifies the Hyper-V server to query By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description
The default path for VHD files is "C:\users\public\documents\hyper-v\Virtual Hard Disks" Get-VHDDefaultPath obtains the path for a given server.

See Also:
New-VHD
Get-VHD
Msvm_VirtualSystemManagementServiceSettingData Class on MSDN

Get-VHD
Gets VHD files from a specified folder on a hyper-v server.

Parameters
Server Specifies the Hyper-V server to query By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Path The folder to search for VHD files on the specified server(s). If not specified the default VHD folder will be queried

Description
Gets VHD files from a specified folder on a hyper-v server (by default the local server, but a remote server may be specified). The default path for VHD files is “C:\users\public\documents\hyper-v\Virtual Hard Disks” Get-VHDDefaultPath obtains the path for a given server.

Get-VHDInfo
Gets detailed information about one or more VHD files

Parameters
VHDPaths *Required* (Input from pipeline) A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server’s default folder (not the current working directory) will be assumed.

Server Specifies the Hyper-V server on which the VHD resides. By default "." , the local computer, is used.

Description
For each VHD requested, Get-VHDInfo returns an object with the following properties VHDPath The path to the file, Path An alternate name for VHDPath, FileSize The size of the file on disk InSavedState Indicates whether the disk is associated with a VM in a saved state InUse Indicates whether the disk is mounted MaxInternalSize The size as seen by the virtual machine ParentPath The parent for a differencing disk (empty for other disks) Type 2=”Fixed”,3=”Dynamic”,4=”Differencing” TypeName The type Number converted to text.

Examples
```
cd (Get-VHDDefaultPath) ; dir *.VHD | get-VHDinfo
```
Moves to the default folder for VHDs gets all the VHD files and passes them into Get-VHDInfo.
```
(Get-VHDInfo 'C:\Users\Public\Documents\Microsoft Hyper-V\Virtual Hard
Disks\Core.VHD').parentPath
```
Returns the parent path of a single differencing disk
```
Get-VMDisk “core%” | forEach {Get-VHDInfo $_.Diskpath} | measure-object -Sum filesize
```
Gets all the disks on virtual machines with names beginning with CORE, gets the disk info for each one, and calculates the sum their file sizes.

See Also:
Get-VHD
Get-VMDisk
GetVirtualHardDiskInfo Method of the Msvm_ImageManagementService Class on MSDN

Test-VHD
Tests the working state of a disk - for example if its parent can be found

Parameters
VHDPaths *Required* (Input from pipeline) A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is

	piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working directory) will be assumed.
Server	Specifies the Hyper-V server on which the VHD file resides. By default "." , the local computer, is used.

Description
Tests the working state of a disk - for example if its parent can be found

Examples
```
dir "$(Get-VHDDefaultPath)\*.VHD" | Test-VHD
```
Gets all the VHD files in the default folder and checks them
```
Get-VMDisk | %{$_.DiskPath} | where {$_.endswith(".VHD")} | Test-VHD
```
Gets all disk on the VMs and validates them

Compress-VHD
Compacts one or more dynamic VHD files

Parameters

VHDPaths *Required*	(Input from pipeline)	A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working directory) will be assumed.
Server		Specifies the Hyper-V server on which the VHD resides. By default "." , the local computer, is used.
Wait		If the wait switch is specified the function will display a progress indicator until the job completes.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Unused space in dynamic VHD files can be reclaimed. This does not affect the maximum size or the partitions and data inside the VHD.

Examples
```
Compress-VHD (get-VHDdefaultPath) +"\tenby.VHD"
```
Compacts the VHD as a background job. You can check status with Get-WmiObject -NameSpace root\virtualization msVM_storagejob | ft jobStatus, description, percentcomplete -auto

See Also:
Expand-VHD
Get-VHD
Merge-VHD
Convert-VHD
CompactVirtualHardDisk Method of the Msvm_ImageManagementService Class on MSDN

Connect-VHDParent
Reattaches a differencing VHD to its parent

Parameters

VHDPaths *Required*	(Input from pipeline)	A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working

	directory) will be assumed.
ParentPath *Required*	The location of the moved or recovered parent VHD file. This can be a scriptblock or a string
Server	Specifies the Hyper-V server on which the VHD resides. By default "." , the local computer, is used.
Wait	If the wait switch is specified the function will display a progress indicator until the job completes.
PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out

Description
If the parent of differencing disk is moved or renamed, the VHD must be reattached to it before it can be used. The parent disk must be an identical copy or the same file as the original.

See Also:
get-VHD
get-VHDInfo
Test-VHD
ReconnectParentVirtualHardDisk Method of the Msvm_ImageManagementService Class on MSDN

Convert-VHD
Creates a new VHD of a different type based on an existing VHD

Parameters
VHDPaths *Required*	(Input from pipeline)	A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working directory) will be assumed.
DestPath *Required*		The location where the resulting file should be saved. This can be a scriptblock or a string
Type *Required*		An enum indicating the type of VHD, if a valid integer (2,3) or valid string "Fixed","Dynamic" is passed it will be converted to an enum
Server		Specifies the Hyper-V server on which the VHD resides. By default "." , the local computer, is used.
Wait		If the wait switch is specified the function will display a progress indicator until the job completes.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Creates a new VHD of a different type based on an existing VHD; for example takes a differencing disk and creates a new fixed or dynamic disk.

Examples
```
convert-VHD core temp -type [VHDType]::dynamic
```
Will merge a differencing disk "CORE.VHD" in the default folder (and its parent(s)) into a new disk called temp.VHD also in the default folder
```
Convert-VHD "$( Get-VHDDefaultPath )\Temp.VHD" F:\backups\MyDisk.VHD -type
[VHDType]::fixed
```
Will convert a disk to a fixed one on a different drive.
```
dir *.VHD | get-VHDinfo | where-object {$_.type -eq 3} | convert-VHD -type "Fixed" -
path {$_.replace(".VHD","-FIXED.VHD")}
```

Finds VHD files in the current folder and isolates the Dynamic ones, and converts them to fixed files with -FIXED appended to the file name.

```
pushd (Get-VHDDefaultPath) ; dir *.VHD | get-VHDinfo | where-object {$_.type -eq 3} | foreach {convert-VHD $($_.path) '.\temp.VHD' -type 2 -wait ; del $($_.path) ; "ren temp.VHD $($_.path)"} ; popd
```

Move to the default VHD folder, get the VHD files, isolate the dynamic ones, convert them to a fixed size one named temp , delete the original , rename temp to the orginal name

Notes

Note that moving a disk type from differing to anything else is a convert not a merge

See Also:

New-VHD
Merge-VHD
ConvertVirtualHardDisk Method of the Msvm_ImageManagementService Class on MSDN

Merge-VHD

Merges VHDs (from snapshots)

Parameters

VHDPaths *Required*	(Input from pipeline)	A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working directory) will be assumed.
DestPath *Required*	(Input from pipeline)	The location where the resulting file should be saved. This can be a scriptblock or a string
Server		Specifies the Hyper-V server on which the VHD resides. By default "." , the local computer, is used.
Wait		If the wait switch is specified the function will display a progress indicator until the job completes.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description

If a valid file is provided the function attempts to merge the differencing disk specified by VHDpath into its ancestor specified by DestPath

See Also:

New-VHD
Convert-VHD
MergeVirtualHardDisk Method of the Msvm_ImageManagementService Class on MSDN

Expand-VHD

Increases the size of a VHD

Parameters

VHDPaths *Required*	(Input from pipeline)	A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working directory) will be assumed.
Size *Required*		The size for the disk in bytes. It must be bigger than the current size. The Hyper-V gui only allows whole GB to be selected, but a fraction may be used.
Server		Specifies the Hyper-V server on which the VHD resides. By default "." , the local computer, is

		used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Wait		If the wait switch is specified the function will display a progress indicator until the job completes.
Force		Ensures that the user is not prompted before the action is carried out

Description
If a valid file is provided the function attempts to extend the VHD at that location using the Image Management Service to the value specified in the size parameter

Examples
```
Expand-VHD 'C:\users\Public\Documents\Microsoft Hyper-V\Virtual Hard Disks\Tenby.VHD' 22gb
```
Expands the named disk to 22GB in size

Notes
This will not expand the partition(s) on the disk, that needs to be done separately.

See Also:
New-VHD
Get-VHDInfo
ExpandVirtualHardDisk Method of the Msvm_ImageManagementService Class on MSDN

Mount-VHD
Mounts a VHD file to make it appear to be a disk in the parent partition

Parameters
VHDPaths *Required*	(Input from pipeline)	A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working directory) will be assumed.
Partition		If a partition number is provided it will be assigned a drive letter, according to a the DriveLetter and No DriveLetter Parameters
DriveLetter		If specified , and partition has been passed, the specified partition will be assigned the requested drive letter
NoDriveLetter		If specified the requested partition will be assigned no drive letter If a partition is specified with neither DriveLetter nor NoDriveLetter , it will be assigned a free drive letter by the system.
Offline		Specifies the disk is to be mounted offline.

Description
Mounts a VHD file to make it appear to be a disk in the parent partition

Examples
```
dir "$(Get-VHDDefaultPath)\*.VHD" | Mount-VHD –offline
```
Gets all the VHD files in the default folder, and mounts them in an Offline state
```
Mount-VHD tenby
```
Mounts tenby.VHD from the default folder, and brings the disk on line
```
Mount-VHD –path "C:\users\public\documents\hyper-v\Virtual Hard Disks\tenby.VHD" –Partition 2 –letter H
```
Mounts tenby.VHD, brings the disk on line and assigns drive H: to the Second partition

See Also:
New-VHD
Dismount-VHD
Mount Method of the Msvm_ImageManagementService Class on MSDN

Dismount-VHD

Dismounts a previously mounted VHD

Parameters

VHDPaths (Input *Required* from pipeline)
A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working directory) will be assumed.

PSC
THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command

Force
Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description

If a valid file is provided then the VHD file at that location is dismounted using the Image Management Service.

Examples

```
Dismount-VHD (get-VHDdefaultPath) +"\tenby.VHD"
```

Dismounts the VHD

```
dir "$(Get-VHDDefaultPath)\*.VHD" | Dismount-VHD
```

Attempts to Dismount all the disks in the folder - will fail gracefully if they are not mounted

See Also:

Mount VHD

Mount Method of the Msvm_ImageManagementService Class on MSDN

Get-VHDMountPoint

Returns the mount point for a VHD file, if it is mounted

Parameters

VHDPaths (Input from *Required* pipeline)
A string which contains the name of a VHD file or an array which contains these. If no VHDPath is passed as a parameter, the function looks for input to be piped to it. If an object is piped, the command will look for Fullname, Path or Diskpath properties, in that order, to use as the path. If the .VHD file extension is omitted it will be added and if the path only contains a file name the VHD, then the server's default folder (not the current working directory) will be assumed.

Description

Returns the mount point (drive letter) for a VHD file, if it is mounted.

See Also:

Mount-VHD

Msvm_MountedStorageImage Class on MSDN

Get-VMIntegrationComponent

Gets the integration Component data for one or more Virtual Machines.

Parameters

VM (Input from *Required* pipeline) Wildcards
The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. If no VM is specified the information will be returned for all VMs on the server. It may be passed via the pipe.

Server
Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description
Returns Resource Allocation Setting Data objects describing the Integration components for a VM

See Also:
Set-VMIntegrationComponent
Integration Components Classes on MSDN

Set-VMIntegrationComponent
Enables or disables integration Components on one or more Virtual Machines.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe..
ComponentName		The Name(s) of the Integration component(s) to be modified
State		Either Running or Stopped
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Enables or disables integration Components on one or more Virtual Machines.

See Also:
Get-VMIntegrationComponent
Integration Components Classes on MSDN

Get-VMKVP
Gets Key/Value pairs for one or more Virtual Machines.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. If no VM is specified the information will be returned for all VMs on the server. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description
Hyper-V provides an integration component to move a small amount of data between the registry in a guest VM and the host. This data is in the form of Keys (names) and associated Values.

Examples
```
(Get-VMKVP "Windows 2008 Ent Full TS").OSName
```
Migt returns "Windows Server (R) 2008 Enterprise" - the OS that VM is running.
```
Get-vmkvp % -server james-2008
```
Returns the Key/Value pairs sent back by all the VMs on the Server James-2008
```
Get-Vm -running | Get-VMKVP
```
Returns the Key/Value pairs for running VMs on the local Server

Notes
The values sent Automatically to the child VM can be found in HKLM:\SOFTWARE\Microsoft\Virtual Machine\guest\Parameters

The values sent Programmatically to the child VM can be found in HKLM:\SOFTWARE\Microsoft\Virtual Machine\External
Those sent by the Child VM are in HKLM:\SOFTWARE\Microsoft\Virtual Machine\auto
If the VM isn't running its Key/Value Pair Exchange Service does NOT persist the values. So stopped VMs won't return anything !

See Also:
Add-VMKVP
Msvm_KvpExchangeComponent on MSDN

Add-VMKVP
Adds Key/Value pairs to be sent to one or more Virtual Machines.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe..
Key *Required*		The Name of the key
Value *Required*		The Value Associated with the key
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Notes
The values sent Automatically to the child VM can be found in HKLM:\SOFTWARE\Microsoft\Virtual Machine\guest\Parameters
The values sent Programmatically to the child VM can be found in HKLM:\SOFTWARE\Microsoft\Virtual Machine\External
Those sent by the Child VM are in HKLM:\SOFTWARE\Microsoft\Virtual Machine\auto
If the VM isn't running its Key/Value Pair Exchange Service does NOT persist the values. So stopped VMs won't return anything !

See Also:
Get-VMKVP
Remove-VMKVP
Msvm_KvpExchangeComponent on MSDN

Remove-VMKVP
Removes a Key/Value pair from the set sent to one or more Virtual Machines.

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe..
Key *Required*		The Name of the key
Server *Required*		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to

	the command
Force	Ensures that the user is not prompted before the action is carried out

See Also:
Get-VMKVP
Add-VMKVP
Msvm_KvpExchangeComponent on MSDN

Get-VmNic
Returns information about Network Interface Cards

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe. If no VM is specified the information will be returned for all VMs on the server. If no VM is specified, NICs from all VMs Will be returned
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer is used. An array of server names by be passed but wildcards cannot be used.
Legacy		Select legacy (emulated) NICs. If both -legacy and -vmbus OR neither is specified all types of NIC Will be returned
VMBUs		Select VM-Bus(synthetic) NICs. If both -legacy and -vmbus OR neither is specified all types of NIC Will be returned

Description
Returns information about legacy and/or VMbus Network Interface Cards on one or more Virtual Machine(s).

Examples
```
Get-VMNic $core -legacy -vmbus
```
Returns both Legacy and VMbus NICs found on the VM pointed to by $core

See Also:
Select-VMNIC
Add-VMNic
Remove-VMNic
Msvm_EmulatedEthernetPortSettingData Class on MSDN
Msvm_SyntheticEthernetPortSettingData Class on MSDN

Get-VMNicPort
Returns the switch port object for a a Network Interface Card

Parameters

NIC *Required*	(Input from pipeline) An object representing the a Network Interface Card whose port is sought

Description
Returns the switch port object for a a Network Interface Card: This is the logical port on a switch, not the switch itself. It can be used to find the switch.

Examples
```
Get-VMNic $core -legacy -vmbus | get-vmNicPort
```
Returns the switch-ports on the NICs of the VM pointed to by $core

See Also:
Select-VMNic
Get-VMNic
Get-VMNICSwitch

Get-VMNicSwitch
Returns the switch connected to a a Network Interface Card

Parameters

NIC (Input from pipeline) An object representing the a Network Interface Card whose switch is sought
Required

Description

Returns the WMI object representing the Virtual Network Switch connected to a a Network Interface Card

Examples

```
(Get-VMNic $vm -legacy -vmbus | get-vmNicSwitch) | foreach-object {$_.elementName}
```

Returns names of Switches used by the VM pointed to by $core

See Also:

Get-VMNIC
Get-VMSwitch

Get-VMNicVLAN

Gets the VLAN ID associated with a Network Interface Card

Parameters

NIC (Input from pipeline) An object representing the NIC to query
Required

Description

Gets the VLAN ID associated with a Network Interface Card

Examples

```
get-vmnic | ForEach-Object{Add-Member -input $_ -MemberType noteproperty -Value $(Get-VMNICVLAN $_) -Name "VLAN" -PassThru}
```

Gets all available NICs and adds the VLAN as a property of each

See Also:

Set-VMNicVLAN
Msvm_BindsTo Class on MSDN
Msvm_NetworkElementSettingData Class on MSDN

Select-VmNic

Allows the user to select a Network Interface Cards connected to a VM

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe. If no VM is specified, NICs from all VMs Will be returned
Server *Required*		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer is used. An array of server names by be passed but wildcards cannot be used.

Description

If a VM has only a single Network Interface Card, that NIC will be returned, if it has more than one, a list will be displayed to allow the user to make a selection.

Examples

```
Select-vmnic $Core
```

Allows the user to choose from the NICs on the server pointed to by $core

See Also:

Get-VMNIC
Add-VMNic
Remove-VMNic

Add-VmNic

Creates a new legacy or VM-bus Network Interface Card on a Virtual Machine.

Parameters

VM	(Input from	The Virtual Machine(s) to modify. This may be a WMI object representing a VM, a string

Required	pipeline) Wildcards	object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Virtualswitch	Wildcards	The Virtual switch to connect. This may be a WMI object representing a switch or a string object containing the name of a switch (if a wildcard is used it must resolve to a single switch). If not specified, the new NIC is created in a disconnected state.
MAC		MAC address. If not specified the Network card will be assigned a MAC address automatically, the first time it is used.
GUID		VM-Bus NICS are identified by a GUID. If a new VM is created, using an existing VHD file the OS in the VM will see a new NIC unless the original GUID is used.
Server *Required*		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer is used. An array of server names by be passed but wildcards cannot be used.
Legacy		Create a legacy (emulated) NIC. If not specified all types a VM-bus (Synthetic) NIC Will be created.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description

Creates a new Network Interface Card on a virtual machine. It can create Legacy (emulated) or VM-bus (Synthetic) nics, assign a fixed MAC address, and specify the GUID which identifies a VM-bus NIC.

Examples

```
Add-VMNic $core (select-VMSwitch)
```

Adds a VM-bus nic to the VM pointed to by $core , choosing the connection from a list of switches.

```
Add-VMNIC "tenby" (Select-VMswitch) -legacy
```

Adds a legacy nic to the VM named Tenby on the local server, choosing the connection from a list of switches.

```
get-vm core-% -Server James-2008 | add-vmnic -virtualSwitch "Internal Virtual Network"
-legacy
```

Adds a legacy nic to those VMs on Server James-2008 which have names beginning Core- and binds them to the "Internal virtual network" switch.

See Also:

Get-VMNIC
Remove-VMNic
Msvm_EmulatedEthernetPortSettingData Class on MSDN
Msvm_SyntheticEthernetPortSettingData Class on MSDN

Set-VMNICSwitch

Connects a Network Interface Card to a Virtual switch

Parameters

NIC *Required*		The NIC to Modify
Switch *Required*		The Virtual switch to connect. This may be a WMI object representing a switch or a string object containing the name of a switch (if a wildcard is used it must resolve to a single switch).
VM *Required*	Wildcards	The VM does not need to be passed to the function (it is determined from the NIC) but is retained for compatibility reasons.
VirtualSwitch		The Server does not need to be passed to the function (it is determined from the NIC) but is retained for compatibility reasons.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Connects a Virtual Network Interface Card to a different a virtual switch. If no switch is specified, the NIC is disconnected

Examples
```
Set-VMNICSwitch $core (Select-vmNic $core) (Select-VMswitch $core.__server)
```
Re-connects a NIC on the VM pointed to by $core, If there are multiple NICs the user will prompted to select one, and they will be prompted to select a switch if there is more than one.

See Also:
Get-VMNicSwitch
SelectVMSwitch

Set-VMNICAddress
Changes the MAC address of a Network Interface Card

Parameters
NIC	(Input from pipeline)	An object representing the NIC to be modified
Required		
MAC		MAC address. This must be 12 Hex digits
Required		
VM		The VM does not need to be passed to the function (it is determined from the NIC) but is retained for compatibility reasons.
Server		The Server does not need to be passed to the function (it is determined from the NIC) but is retained for compatibility reasons.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Normally the MAC address of a Network Interface Card is set at the first boot after the card is added to the virtual machine to one of a range of MAC addresses defined for the server. However the address can be set or changed maually.

Examples
```
Set-VMNICAddress $core (Select-vmNic $core) "00155D010101"
```
Sets the MAC address of a NIC on the pointed to by $core, if there are multiple NICs the user will prompted to select one

See Also:
Get-VMNic
New-Nic
Select-VMNic
Msvm_EmulatedEthernetPortSettingData Class on MSDN
Msvm_SyntheticEthernetPortSettingData Class on MSDN

Set-VMNicVLAN
Sets the VLAN ID associated with a Network Interface Card

Parameters
NIC	(Input from pipeline)	An object representing the NIC to update
Required		
VLANID		The ID to Assign to the VLAN
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Sets the VLAN ID associated with a Network Interface Card

See Also:
Get-VMNicVLAN
Msvm_BindsTo Class on MSDN
Msvm_NetworkElementSettingData Class on MSDN
MSVM_hostedAccessPoint class on MSDN

Remove-VMNic
Removes a Network Interface Card from a VM

Parameters

NIC	(Input from pipeline) *Required*	The NIC to remove from the VM
VM		The VM does not need to be passed to the function (it is determined from the NIC) but is retained for compatibility reasons
Server		The Server does not need to be passed to the function (it is determined from the NIC) but is retained for compatibility reasons
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Removes a Network Interface Card from a VM.

Examples
```
Remove-VMNIC $core (Select-vmNic $core)
```
Removes a NIC on server pointed to by $core, if there are multiple NICs the user will prompted to select one
```
Get-vmNic -legacy -VM "Lab*" | remove-VMNic -force
```
Gets all the legacy NICS on virtual machines with names beginning “Lab” and removes them without prompting

See Also:
Add-VMNIC
Get-VMNIC
Select-VMNIC
Msvm_EmulatedEthernetPortSettingData Class on MSDN
Msvm_SyntheticEthernetPortSettingData Class on MSDN

Get-VMByMACaddress
Discovers which VM owns a particular mac address

Parameters

Mac	(Input from pipeline) *Required*	The MAC address to seek. The command uses regular expression matching so a partial address or regex can be used in the MAC field
Server		Specifies the Hyper-V server to query for network interfaces By default "." , the local computer is used.

Examples
```
Get-VMbymacAddress "00155D000101"
```
Returns detils of the VM with the NIC given the address 00155D000101
```
get-vm (get-vmbyMacAddress "00155DD0BEEF").vm
```
Returns the WMI object representing that VM.

Get-VMSwitch
Returns one or more Virtual Switch objects

Parameters

VirtualSwitchName	(Input from	The name of a Switch being sought (which may include wildcards.) WMI queries

	pipeline) Wildcards	use % as a wildcard, but the command will convert * to %. Name can be used as an Alias for VirtualSwitchName (as well as contractions of it)
Server *Required*		Specifies the Hyper-V server on which the switches reside. By default "." , the local computer is used. An array of server names by be passed but wildcards cannot be used.

Description
If no name is specified returns all the Virtual Network Switches on a host, Otherwise returns the switch(es) which match the given name

See Also:
New-VmExternalSwitch
New-VMInternalSwitch
New-VMPrivateSwitch
Remove-VMSwitch
Select-VMSwitch
Msvm_VirtualSwitch Class on MSDN

Select-VMSwitch
Returns a Virtual Switch objects

Parameters
Server *Required* Specifies the Hyper-V server on which the switches reside. By default "." , the local computer is used. An array of server names by be passed but wildcards cannot be used.

Description
If only one switch is found on a server, it is returned, if more than one exists a list is provided to allow the user to make a selection.

New-VMSwitchPort
Returns a newly created Switch port

Parameters
VirtualSwitch *Required*	(Input from pipeline) Wildcards	The Virtual switch to connect. This may be a WMI object representing a switch or a string object containing the name of a switch (if a wildcard is used it must resolve to a single switch).
Server *Required*		Specifies the Hyper-V server on which the a named switch is found. By default "." , the local computer is used. An array of server names by be passed but wildcards cannot be used.

Description
This command is intended to be called by other commands, and not directly from the command line

Notes
Not intended to be called directly, used by Add-VMNIC and SetVMNICConnection

See Also:
CreateSwitchPort Method of the Msvm_VirtualSwitchManagementService Class on MSDN

Select-VMExternalEthernet
Selects an avaialable host network Interface

Parameters
Server Specifies the Hyper-V server to query for network interfaces By default "." , the local computer is used.

Description
When a External virtual network switch is created it needs to be told which network interface on the host computer it should be bound to. If there is more than one available a list is presented to allow the user to make a selection.

See Also:
NewVMExternalSwitch
Msvm_ExternalEthernetPort on MSDN

New-VMPrivateSwitch

Creates a virtual network switch, which isn't bound to a NIC in the parent partition

Parameters

VirtualSwitchName The name for the new switch
Required

Ports The number of ports to be created on the new switch

Server Specifies the Hyper-V server to query for network interfaces By default "." , the local computer is used.

Force Ensures that the user is not prompted before the action is carried out

Description

Creates a virtual network switch, which isn't bound to a NIC in the parent partition

Examples

```
New-VMPrivateSwitch "VM network" -server "HVCore"
```

Creates a Switch on the server named "HVCore". The network will not be accessible in the host OS, and will be named "VM Network" in the Hyper-V administration tools

See Also:

New-VMInternalSwitch
New-VMExternalSwitch
Remove-VMSwitch
CreateSwitch method of MsVM_VirtualSwitchManagementService on MSDN

New-VMInternalSwitch

Creates a virtual network switch, bound to a virtual NIC in the parent partition

Parameters

VirtualSwitchName The name for the new switch
Required

Ports The number of ports to be created on the new switch

Server Specifies the Hyper-V server to query for network interfaces By default "." , the local computer is used.

Force Ensures that the user is not prompted before the action is carried out

Description

Creates a virtual network switch, and a virtual NIC in the parent partition, and binds them together.

Examples

```
New-VMInternalSwitch "Host and VM network"
```

Creates a Switch and virtual NIC in the host. The device name for the NIC in the host and the Network name in Hyper-V will be "Host and VM Network"

See Also:

New-VMPrivateSwitch
New-VMExternalSwtich
Remove-VMSwitch
Remove-VMSwitchNIC
CreateSwitch method of MsVM_VirtualSwitchManagementService on MSDN
CreateInternalEthernetPortDynamicMac method of MsVM_VirtualSwitchManagementService on MSDN

New-VMExternalSwitch

Creates a virtual network switch, bound to a physical network card

Parameters

VirtualSwitchName The name for the new switch
Required

ExternalEthernet Either the name of a Network interface, passed as a string, or a WMI object representing an interface

Ports The number of ports to be created on the new switch

Server Specifies the Hyper-V server to query for network interfaces By default "." , the local computer is used.

PSC THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command

Force Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Creates a virtual network switch, and binds it to a network card which is available in the parent partition

Examples
```
Select-VMExternalEthernet | New-VMExternalSwitch -VirtualSwitchName "Wired virtual Network"
```
Allows the user to choose if there are multiple available NICs and binds the selected one to a new switch. The device name for the NIC in created in the host and the Network name in Hyper-V will be "Wired virtual Network"
```
New-VMExternalSwitch -VirtualSwitchName "Wired virtual Network" -ext "Broadcom" -Server Core
```
Finds a Nic with a name begining "Broadcom" on the server named "core", and binds it to a new switch. The device name for the NIC in created in the host and the Network name in Hyper-V will be "Wired virtual Network"

See Also:
New-VMPrivateSwitch
New-VMExternalSwtich
Remove-VMSwitch
SetupSwitch method of MsVM_VirtualSwitchManagementService on MSDN

Remove-VMSwitchNIC
Removes the parent partition NIC associated with a Virtual switch

Parameters
Name The name of the NIC to be removed
Required

Server Specifies the Hyper-V server to query for the NIC . By default "." , the local computer is used.

PSC THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command

Force Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Removes the parent partition Network Interface Card associated with a Virtual switch. A virtual NIC is always created in the parent partition for internal and external networks. In Hyper-V R2 there is an option to not allow a management partition to access the an external NIC used by VMs. The same result can be achieved by using remove-vmswitchNic

Notes
This is intended to be called from other commands, and not from the command line.

See Also:
DeleteInternalEthernetPort Method of the Msvm_VirtualSwitchManagementService Class

---

Remove-VMSwitch
Deletes a virtual network switch

Parameters
virtualSwitch A string containing the name for the switch to be deleted, or a WMI object representing the switch
Required

Server	Specifies the Hyper-V server to query for network interfaces By default "." , the local computer is used.
PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Deletes a virtual network switch

See Also:
New-VMPrivateSwitch
New-VMExternalSwtich
New-VMInternalSwitch
DeleteSwitch method of MsVM_VirtualSwitchManagementService on MSDN

Get-VMHost
Lists Hyper-V servers registered with active directory

Parameters
Domain (Input from pipeline) Specifies the Active directory container to search from: defaults to the root domain.

Description
Queries active directory for Hyper-v connection points - in other words, servers registered with the domain running Hyper-V

Examples
```
get-vmhost "DC=ite,DC=contoso,DC=com" | foreach {$_; Get-vmsummary -server $_}
```
Queries the domain ITE.Contoso.com for Hyper-V servers, prints the name of each and dumps the state of VMs on each

---

Set-VMHost
Configures the settings for the Hyper-V service itself

Parameters
ExtDataPath	If passed changes the external data path used for Snapshots etc
VHDPath	If passed changes the default location for Virtual Hard disk files
MinMAC	Must be exactly 12 Hexadecimal digits. If set, changes the minimum MAC address for dynamically assigned addresses
MaxMAC	Must be exactly 12 Hexadecimal digits. If set, changes the maximum MAC address for dynamically assigned addresses
OwnerContact	Contact information for a person responsible for the server
OwnerName	Name of a person responsible for the server
Server	Specifies a single Hyper-V server to be updated. By default "." , the local computer is used.
PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and Recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out

Description
Configures the settings for the Hyper-V service, including contact details, MAC Address range, default VHD location, and default data files location

See Also:
Get-VMHost
ModifyServiceSettings Method of the Msvm_VirtualSystemManagementService Class on MSDN

Get-VMSnapshot

Returns SnapShots for one or more VM(s)

Parameters

VM	(Input from pipeline) Wildcards	The Virtual Machine(s) to query . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe. If no VM is passed, all VMs on the server are queried
Name		The name of a snapshot to return
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
Newest		If Specified only the newest Snapshot is returned
Root		if specified only the root snapshot is returned
Current		If specified only the current snapshot is returned

Description

Returns one or more MSVM_VirtualSystemSettingData WMI object(s) representing snapshots. An individual snapshot can be selected by specifying its name, and the newest snapshot can be selected by specifying the -Newest switch.

Examples

```
Get-Vmsnapshot $Core -newest
```

Returns the newest snapshot on the VM pointed to by $core

See Also:

Get-vmsnapshotTree
New-Vmsnapshot
Remove-VMSnapshot
Rename-VMSnapShot
Restore-VMSnapShot
Select-VMSnapshot

Get-VMSnapshotTree

Displays all the Snapshots of a VM in a tree view

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description

The function displays the snapshots of a Virtual machine arranged in tree format to make it easy to see which snapshots are descended from which others.

See Also:

Get-vmSnapshot
Select-VMSnapshot

Select-VMSnapshot

Allows the user to select a snapshot for a VM

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to query This may be a WMI object representing a VM, a string object containing the name of a VM. It may be passed via the pipe.

Server	Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.

Description
If there is only a single snapshot it is returned, if there is more than one a tree view of the snapshots for the VM is displayed for the user to make a selection.

Examples
```
Select-Vmsnapshot $Core
```
Gets the Snapshots of the machine pointed to by $core and if there are multiple snap shots prompts the user to select one from a tree

See Also:
Get-VMSnapshot
Get-VMSnapshotTree

New-VMSnapshot
Creates a new snapshot of one more VMs.

Parameters
VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Note		Sets an explanatory note on the snapshot
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
Wait		If the wait switch is specified the function will display a progress indicator until the job completes.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Creates a new snapshot of one more VMs.

Examples
```
new-vmsnapshot $Core
```
Takes a snapshot of the VM pointed to by $core
```
get-vm "core%" -server "James-2008" | new-VmSnapshot -wait
```
Gets the VMs with names beginning "Core" on the server "James-2008" and snapshots them one by one

See Also:
Get-vmsnapshot
Get-vmsnapshotTree
Remove-VMSnapshot
Rename-VMSnapShot
Restore-VMSnapShot
Select-VMSnapshot
CreateVirtualSystemSnapshot Method of the Msvm_VirtualSystemManagementService Class on MSDN

Rename-VMsnapshot
Changes the display name of a snapshot

Description
Changes the display name of a snapshot

Examples

```
Rename-vmSnapshot –v $core –s (Select-vmsnapshot $core).elementName –n "default"
```

Prompts the user to select one of the snap shots on the VM pointed to by $core and renames it to “default”

See Also:

Get-VMSnapshot
New-VMSnapshot
Update-VMSnapshot

Update-VMSnapshot

Replaces an existing snapshot with a new one

Parameters

VM *Required*	(Input from pipeline) Wildcards	The Virtual Machine(s) to update . This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
SnapName		The name of the snapshot to update. If no name is provided the newest snapshot is used (and its name re-used)
Note		A note to add to the newly created snapshot
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out. Update-VMSnapShot calls other commands which have a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description

This function is a wrapper for New-Snapshot, remove-Snapshot, and rename-snapshot. If no snapshot name is specified the function gets the most recent snapshot. It renames that snapshot to “Delete-me”. It creates a new snapshot, using the note if one is specified, and renames the new snapshot to the given name. Finally it removes the old version of the snapshot.

See Also:

Get-VMSnapshot
New-VmSnapshot
Rename-VMSnapshot
Remove-VMSnapshot

Restore-VMSnapshot

Rolls a VM back (or forward) to a snapshot

Parameters

SnapShot *Required*	(Input from pipeline)	An Object representing the snapshot to be removed - may be passed via the pipeline.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".
Restart		Restarts the VM when the Snapshot has been restored
Wait		If the wait switch is specified the function will display a progress indicator until the job completes.

Description
Rolls a VM back (or forward) to a snapshot

Examples
```
Select-vmsnapshot $Core | Restore-vmsnapshot
```
Lets the user select a snapshot on the VM pointed to by $core and applies it.

See Also:
Get-VMSnapshot
New-VMSnapshot
Remove-VMSnapshot
Select-VMSnapshot
ApplyVirtualSystemSnapshot Method of the Msvm_VirtualSystemManagementService Class on MSDN

Remove-VMSnapshot
Removes one or more snapshots from a VM.

Parameters

Snapshot (Input from pipeline) *Required*	An Object representing the snapshot to be removed - may be passed via the pipeline.
Tree	If specified remove-vmSnapshot will remove snapshots which are children of the selected one.
Wait	If the wait switch is specified the function will display a progress indicator until the job completes.
PSC	THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command
Force	Ensures that the user is not prompted before the action is carried out. This command has a "ConfirmImpact" level of High: so the default is to request confirmation (unless the $confirmPreference variable is set to "none".

Description
Removes one or more snapshots from a VM.

Examples
```
Select-vmsnapshot $Core | remove-vmsnapshot -tree
```
Prompts the user to select a snapshot on the VM pointed to by $core, and removes it any child snapshots

See Also:
Get-vmsnapshot
Select-VMSnapshot
RemoveVirtualSystemSnapshot Method of the Msvm_VirtualSystemManagementService Class on MSDN

Move-VM
Moves one or more VMs between cluster nodes using live migration

Parameters

VM	(Input from pipeline) Wildcards	The Virtual Machine(s) to migrate. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe. If no VM is specified the script is out put for all VMs
Destination		The name of the cluster node to which Virtual Machines should be migrated.
Server		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
Moves VMs between cluster nodes. If multiple VMs are specified they are moved sequentially. If the cluster commands are not loaded or a VM is not running, or not configured as a cluster application then a warning is returned

Examples
```
Move-VM
```
Moves all virtual machines on the local server to the first other node the cluster - this is only recommended for 2 node clusters
```
Get-Vm -server NodeA -running | move-vm -Destination Node
```
Gets VMs running on the Server named "Node A" and moves them to the server "nodeB"

See Also:
Get-VMClusterGroup
Get-VMLiveMigrationNetwork
Set-VMLiveMigrationNetwork

Select-ClusterSharedVolume
Allows the user to select a clustered shared volume from a list

Parameters
Server	Specifies the Hyper-V server on which the cluster shared volume resides

Description
Allows the user to select a clustered shared volume from a list. If cluster functions are not loaded, returns an error

Examples
```
$path = (Select-ClusterSharedVolume).volname ; new-vm -name "Ha" -path $path
```
Gets the path a of a user determined Cluster shared volume and uses it as the data directory for a new Virtual Machine

See Also:
Get-ClusterSharedVolume

Sync-VMClusterConfig
Synchronizes the configuration of a VM across all members of a cluster

Parameters
VM	(Input from pipeline) Wildcards	The Virtual Machine(s) to synchronise. This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe.
Server *Required*		Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards can not be used.
PSC		THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc., with nested and recursive calls to the command
Force		Ensures that the user is not prompted before the action is carried out

Description
If the configuration of a clustered VM is changed on the node where it is resident, it is necessary to tell the cluster to update its information so that all nodes are consistent. If this is not done VMs may not be able to migrate. The failoverClusters commands have not been imported this command generates a warning.

See Also:
Get-VMClusterGroup
Get-ClusterResource
Update-ClusterVirtualMachineConfiguration

Get-VMLiveMigrationNetwork
Returns a list of Cluster networks in use for live migrating VMs

Parameters

Server Specifies the Hyper-V server / Cluster node to query. By default "." , the local computer is used.

Description

Returns a list of Cluster networks in use for live migrating VMs

See Also:

Select-VMLiveMigrationNetwork
Get-ClusterResourceType
Get-ClusterParameter

Select-VMLiveMigrationNetwork

Allows the user to select cluster networks to be used live migrating

Parameters

Server Specifies the Hyper-V server / Cluster node to query. By default "." , the local computer is used.

PSC THIS PARAMETER SHOULD NEVER BE PASSED FROM THE COMMAND LINE. It is used to ensure correct handling of Whatif, confirm, verbose etc, with nested and recursive calls to the command

Force Ensures that the user is not prompted before the action is carried out

Description

Shows a list of clusternetworks from which the user can choose the network(s) to be used for live migration.

See Also:

Get-VMLiveMigrationNetwork
Get-ClusterNetwork
Get-ClusterResourceType
set-ClusterParameter

Get-VMClusterGroup

Returns the Cluster Group of which the VM is a member

Parameters

VM (Input from pipeline) Wildcards — The Virtual Machine(s) to test This may be a WMI object representing a VM, a string object containing the name of a VM (or a wildcard which matches multiple VMs), or an array containing a mixture of such objects. It may be passed via the pipe. If no VM is specified the script is out put for all VMs

Server — Specifies the Hyper-V server on which a named Virtual Machine resides. By default "." , the local computer, is used. An array of server names may be passed but wildcards cannot be used.

Description

Returns the cluster group of which the VM is a member or null if the cluster commands are not loaded or the VM is not a member of the cluster. This can be used as a test for Highly available VMs

Examples

```
Get-VMClusterGroup "Tenby"
```

Returns the cluster group for the Virtual Machine named "or null if it is not clustered.

```
Get-VMClusterGroup | where-object {$_.state -eq "Offline"} | Move-ClusterGroup
```

Moves offline VMs from to the other node of a cluster. To move running VMs use Move-VM.

Select-Enumtype

Returns a value selected by the user from an Enumeration type

Parameters

eType The Enumeration type to use to build the selection list
Required

Default The Value returned if nothing is selected from the list (if not specified the function will return null)

Description

When an enum type has been created to allow easy mapping of names to values, this function prompts the user to select one of the names, and returns the corresponding value.

Examples

```
$inboundAction = (Select-EnumType fwaction)
```

The user is offered choices "Allow" or "Block" = 0 the two possible options in the fwAction (firewall action) enum, and the variable is set to 1 or 0, the associated values based on that choice.

Select-List

Returns an object selected by the user from a table of numbered rows.

Parameters

InputObject *Required*	(Input from pipeline)	An array of objects which provide the data for the selection list. The data returned comes from this parameter as well. If a single item is provided, it is returned without showing the list.
Property *Required*		One or more property names used to format the selection list.
Multiple		Specifies that multiple items can be selected.

Description

Takes a collection of objects and the list of properties to be displayed. This information is used to show a table with numbered rows. The user is prompted to make either a single or multiple choice, by row numbers and the selected objects are returned.

Select-Item

Returns a zero-based integer indicating the user's selection from a list

Description

A list of choices is displayed one after the other, and the users selection is returned as a zero-based integer. Can display a caption followed by a message before the choices

Examples

```
select-item -Caption "Configuring RemoteDesktop" -Message "Do you want to: " -choice "
&Disable Remote Desktop","&Enable Remote Desktop","&Cancel" -default 1
```

will display the following

Configuring RemoteDesktop Do you want to: [D] Disable Remote Desktop [E] Enable Remote Desktop [C] Cancel [?] Help (default is "E"):

The function accepts D, E ,or C as input and returns 0 for Disable, 1 for Enable and 2 for Cancel

```
Get-choice @{"&Tea"="A drink made from leaves";"&Coffee"="A drink made from beans"}
```

will display the following

Please make a selection Choices are presented below [T] Tea [C] Coffee [?] Help (default is "T"):

Pressing ? will produce help as follows

T - A drink made from leaves C - A drink made from beans

See Also:

[PromptForChoice Method of PSHostUserInterface on MSDN]

Select-Tree

Allows the user to select from objects in a tree format

Parameters

Items *Required*	The collection of items to output in the tree
StartAt *Required*	The first item in the branch of the tree to output
Path	The name of the property name which identifies a location in the tree. It does not need to be a fully qualified path, but must be unique
Parent	The name of the property which holds the path of a node's parent.
Label	The name of the property which holds the text to display
Indent	The function is called recursively and this parameter specifies how far each branch of the tree is indented.
Multiple	Determines whether a single or multiple selection is to made from the tree

Description
The function takes a collection of objects which must have a property to display (label), a property identify the position in the tree (path) and a property which identifies a node's parent (parent).

See Also:
Out-Tree

Out-Tree
Outputs objects in a tree format

Parameters
Items *Required* The collection of items to output in the tree

StartAt *Required* The first item in the branch of the tree to output

Path The name of the property name which identifies a location in the tree. It does not need to be a fully qualified path, but must be unique

Parent The name of the property which holds the path of a node's parent.

Label The name of the property which holds the text to display

Indent The function is called recursively and this parameter specifies how far each branch of the tree is indented.

Description
The function takes a collection of objects which must have a property to display (label), a property identify the position in the tree (path) and a property which identifies a node's parent (parent).

Examples
```
out-tree -items (dir -rec | where {$_.mode -like "D*"}) -start (get-item $pwd) -path pspath -parent psparentPath -label name
```
Outputs a tree of subfolders of the current folder, the starting item is the current folder, and the path for each item is the PsPath property, its parent is in the PsParentPath propery. The name property is output

See Also:
Select-Tree

Test-Admin
Checks to see if the current session has administrator privileges

Description
Some tasks require administrator privileges, and on Windows Vista/Server 2008 and later this may require the user to choose "Run as administrator" when starting PowerShell. This tests to see if the session is elevated or not. Some tasks require administrator privileges, and on Windows Vista/Server 2008 and later this may require the user to choose "Run as administrator" when starting PowerShell. Test-admin returns a boolean indicated the elevated status - true = elevated, false = not elevated

Examples
```
Test-Admin
```
Returns true if the session was started as administrator, false otherwise

---

Test-WMIJob
Checks the status of background WMI jobs

Parameters
Job *Required* (Input from pipeline) A WMI object representing a job. If a string is passed containing a WMI path it is converted to the WMI object automatically.

StatusOnly If not specified the Job WMI object is returned. If it is specified only the JobState Property is returned, saving the caller the step of getting the property

Wait If not specified the job will only be examined once. If specified it will be examined every 250 milliseconds as long as its state is "Running"

Description The Description to be displayed while waiting for the job to complete

Description
Some wmi functions create a background job, it may be necessary to wait for a job to complete, or simply to check its current state.

Test-WMIResult
Checks the result returned by calling a WMI method

Parameters
Result *Required*	The WMI Object returned by calling the method
JobWaitText	The text to be displayed on a progress indicator while waiting for the job to complete (if -wait is specified and a background job exists)
SuccessText	The text written to Verbose: if the method was successful
FailText	The Text to be included with the error and written to error if the method fails
wait	If specified will wait for a background job to complete. If not specified a message will be written to Warning indicating the job is still running if it has not completed

Description
When calling a WMI method the result will often by an object containing a return value, which indicates immediate success, the creation of a task to perform the job, or an error. Test-WMI result looks at this code and if necessary uses Test-WMIJob to see what the outcome of the job, and hence the method, was

Notes
This command is expected to be used by other commands, not from the command line

See Also:
Test-WmiJob

Convert-DiskIDtoDrive
Converts a logical disk index to a drive letter.

Description
Converts a logical disk index to a drive letter.

Examples
```
PS > Convert-DiskIDToDrive 0
```
Returns the drive letters of the partitions on the specified disk. For Example C:, D:

Get-FirstAvailableDriveLetter
Returns a CHAR indicating the first available drive letter.

Description
Returns a CHAR indicating the first available drive letter.

Wait-ForDisk
Waits for disk to come on line

Parameters
MountPoint *Required*	The drive letter being sought. If the trailing colon is omitted, the command will add it.
Attempts	The number of times to look for the logical disk object before returning false
MSPause	The number of milliseconds to pause between attempts

Description
Waits for a predefined time to see if a Win32_logicalDisk WMI object can be found with a given drive letter.

WMI Objects used by the functions

WMI Class name	Used in Functions
Cim_Datafile	Get-VHD, Show-VHDMenu, Show-VHDMenu, Show-VHDMenu, Show-VHDMenu, Show-VHDMenu
Msvm_ActiveConnection	Show-HypervMenu
Msvm_AllocationCapabilities	New-VMRASD
Msvm_BindsTo	Get-VMNICVLAN, Set-VMNICVLAN, Set-VMNICVLAN
Msvm_ComputerSystem	Add-VMNewHardDisk, Add-VMDisk, Add-VMDisk, Add-VMDrive, Add-VMFloppyDisk, Add-VMPassThrough, Add-VMSCSIController, Get-VMDisk, Get-VMFloppyDisk, Remove-VMDrive, Remove-VMFloppyDisk, Set-VMDisk, Show-VMMenu, Add-VMNIC, Get-VMNIC, Set-VMNICAddress, Set-VMNICSwitch, Get-VMSnapshot, Get-VMSnapshotTree, New-VMSnapshot, Rename-VMSnapshot, Restore-VMSnapshot, Select-VMSnapshot, Update-VMSnapshot, Export-VM, Get-VM, Get-VM, Get-VM, Get-VMBuildScript, Get-VMClusterGroup, Get-VMSummary, Get-VMThumbnail, Invoke-VMShutdown, Move-VM, New-VMConnectSession, Ping-VM, Remove-VM, Set-VM, Set-VM, Set-VMState, Test-VMHeartBeat, Get-VMIntegrationComponent, Get-VMKVP, Get-VMProcessor, Get-VMSettingData, Remove-VMKVP, Set-VMCPUCount, Set-VMMemory, Set-VMSerialPort, Sync-VMClusterConfig
Msvm_DiskDrive	Select-VMPhysicalDisk, Show-VMDiskMenu, Show-VMDiskMenu
Msvm_EmulatedEthernetPortSettingData	Get-VMByMACAddress, Get-VMNIC
Msvm_ExternalEthernetPort	Show-HypervMenu, Show-HypervMenu, New-VMExternalSwitch, New-VMExternalSwitch, Select-VMExternalEthernet, Set-VMNICVLAN
Msvm_HeartbeatComponent	Test-VMHeartBeat
Msvm_HostedAccessPoint	Set-VMNICVLAN
Msvm_InternalEthernetPort	Show-HypervMenu, Remove-VMSwitchNIC
Msvm_KvpExchangeComponent	Get-VMKVP
Msvm_KvpExchangeDataItem	Add-VMKVP, Remove-VMKVP
Msvm_MemorySettingData	Get-VMMemory
Msvm_MountedStorageImage	Get-VHDMountPoint, Mount-VHD
Msvm_NetWorkElementSettingData	Get-VMNICVLAN, Set-VMNICVLAN, Set-VMNICVLAN
Msvm_PreviousSettingData	Get-VMSnapshot
Msvm_Processor	Get-VMProcessor
Msvm_ProcessorSettingData	Get-VMCPUCount
Msvm_ResourceAllocationSettingData	Add-VMFloppyDisk, Get-VMDiskByDrive, Get-VMDiskByDrive, Get-VMDiskController, Get-VMDriveByController, Get-VMDriveByController, Get-VMFloppyDisk, Remove-VMDrive, Set-VMDisk, Show-VMDiskMenu, Show-VMDiskMenu, Show-VMDiskMenu, Get-VMSerialPort
Msvm_ResourcePool	Select-VMPhysicalDisk, Show-VMDiskMenu, Show-VMDiskMenu
Msvm_SettingsDefineCapabilities	New-VMRASD
Msvm_ShutdownComponent	Invoke-VMShutdown
Msvm_SwitchLANEndpoint	New-VMInternalSwitch, Set-VMNICVLAN
Msvm_SwitchPort	Show-HypervMenu, Set-VMNICVLAN
Msvm_SyntheticEthernetPortSettingData	Get-VMByMACAddress, Get-VMNIC
Msvm_VirtualSwitch	Get-VMNICSwitch, Get-VMSwitch, New-VMSwitchPort, Remove-VMSwitch
Msvm_VirtualSwitchManagementService	New-VMExternalSwitch, New-VMInternalSwitch, New-VMPrivateSwitch, New-VMSwitchPort, Remove-VMSwitch, Remove-VMSwitchNIC, Set-VMNICSwitch
Msvm_VirtualSystemGlobalSettingData	Show-VMMenu, Get-VMBuildScript, New-VM, Set-VM

Msvm_VirtualSystemManagementService	New-VMSnapshot, Remove-VMSnapshot, Restore-VMSnapshot, Export-VM, Get-VMSummary, Import-VM, New-VM, Remove-VM, Set-VM, Set-VMHost, Add-VMKVP, Add-VMRASD, Remove-VMKVP, Remove-VMRASD, Set-VMRASD
Msvm_VirtualSystemManagementServiceSettingData	Get-VhdDefaultPath, Show-HypervMenu, Show-HypervMenu, Set-VMHost
Msvm_VirtualSystemSettingData	Get-VMByMACAddress, Set-VMNICAddress, Set-VMNICSwitch, Get-VMSnapshot, New-VMSnapshot, Remove-VMSnapshot, Rename-VMSnapshot, Restore-VMSnapshot, Get-VM, Get-VM, Get-VMSettingData, Get-VMSettingData
Msvm_VlanEndpoint	Remove-VMSwitch, Remove-VMSwitch
Win32_CdromDrive	Add-VMDisk, Add-VMDisk, Set-VMDisk, Set-VMDisk, Show-VMDiskMenu
Win32_Directory	Get-VHD, Show-VHDMenu, Show-VHDMenu
Win32_DiskDrive	Get-VHDMountPoint, Mount-VHD
Win32_DiskPartition	Get-VHDMountPoint, Convert-DiskIDtoDrive, Convert-DiskIDtoDrive
Win32_LogicalDisk	Wait-ForDisk, Convert-DiskIDtoDrive, Get-FirstAvailableDriveLetter
Win32_OperatingSystem	Show-VMDiskMenu
Win32_PingStatus	Ping-VM