PowerShell Quick Reference

v2.9 by Dimitri Koens **www.dimensionit.tv**

Please contact me with questions, remarks, etc regarding this document at dimitrikoens@gmail.com. Join me on Linked in and Facebook to receive valuable information regarding PowerShell, SCOM, SQL Server and Virtualization.

Quick Start

Get-Process # displays a list of running processes
Get-Process | Select-Object Name, Company # selects several columns
Get-Process | Select-Object Name, Company | Format-Table -AutoSize # uses minimal column width
Get-Process | Select-Object Name, Company | Format-List # displays a list instead of a table
Get-Process | Sort-Object ID –descending # sorts on process id instead of name
Get-Process | Where { $_.vm –gt 150MB } # selects processes where virtual memory is greater than 150MB

Built-in Help functionality

Get-Help
Get-Help Get-Process –full
Get-Help Get-Process –examples # view the example commands
Get-Help about* # lists all the about-articles, use the full article name to view its contents, e.g. about_scripts

Get-Command # Display all commands
Get-Command *process* # display all commands containing the word "process"

Get-Process | Get-Member # display the properties and methods of the output

The following two examples help you to protect you from … you! ☺
whatif parameter displays the command without actually executing it
Get-Process PowerShell | Stop-Process –whatif

Get-Process PowerShell | Stop-Process –confirm # asks for confirmation

Common aliases

Command	Alias	Command	Alias	Command	Alias
Clear-Host	cls, clear	**Get-Member**	Gm	**Remove-Item**	ri, del, erase, rmdir, rd, rm
Copy-Item	copy, cpi, cp	**Get-Process**	gps, ps	**Rename-Item**	rni, ren
ForEach-Object	foreach, %	**Get-Service**	Gsv	**Select-Object**	Select
Format-List	FL	**Get-WmiObject**	Gwmi	**Set-Location**	sl, cd, chdir
Format-Table	FT	**Group-Object**	Group	**Sort-Object**	Sort
Get-ChildItem	gci, dir, ls	**Move-Item**	mi, move, mv	**Stop-Process**	spps, kill
Get-Content	gc, type, cat	**Out-Host**	Oh	**Where-Object**	where, ?
Get-Help	help, man	**ise (alleen v2)**	Powershell_ise.exe	**Write-Output**	echo, write

Get-Alias | Group-Object Definition # display all possible aliases per command
Select-String: like Findstr or Grep. No alias defined! Use: Set-Alias Grep Select-String

Operators

Operator	Meaning	Operator	Meaning
-eq, -ne	Equal, not equal: 5 –eq 5	**-match, -notmatch, -cmatch**	regular expression match: "Rick" -match "[DMNR]ick"
-gt, ge	greater than, greater than or equals: 6 –gt 5	**-contains, - notcontains**	Array contains specific value: "red", "blue" –contains "blue"
-lt, -le	less than, less than or equals: 5 –lt 6	**-is, -isnot**	type comparison: 15 –is [int]
-like, -notlike, -clike	pattern comparison using wildcards: "Samantha" -like "sam*"	**-f**	Formatting: $a=2987654; "free space: {0:N0} bytes" -f $a

Punctuation Marks

(expression) { code block } [item in array]
` backtick is the escape character, mostly found on the key combined with tilde-sign ~

"string with automatich variable expansion"
'string without automatic variable expansion'

Keyboard shortcuts

Tab: command completion	F7: display history popup	Ctrl ←, Ctrl →: jump one word left or right
Esc: clear the command line	F8: lookup last command that starts with current input. Try this: Get-Process; <enter>; Get<F8>	| More: <Ctrl-C> quit, <q> quit, <space> scroll page, <enter> scroll one line
Use arrow up and down to browse previous commands	Home, End: jump to start or end of current command line	Within ISE: F5 = Run, F8 = Run Selection

Security

The .ps1 extension	**Execution Policy** (Set- and Get-ExecutionPolicy)	**To prevent command hijacking**
Associated with Notepad. When a user receives a PowerShell script through e-mail and doubleclicks it then the script just opens in notepad instead of executing (like the i-love-you virus did).	Restricted (default), AllSigned, RemoteSigned, Unrestricted (not recommended) All scripts that are not on local fixed disks, like CD's/DVD's and drive mappings to network shares, but also attachements in e-mail and chat-programs are considered remote	You can only run commands from the current location by specifying the path to the script. Example: .\script.ps1 instead of script.ps1.

Working with files

Get-Process | Out-File p.txt –append
Get-Process | Export-CSV p.csv
Get-Process | Export-CliXML p.xml
Import-CSV p.csv # displays using Format-List because there are more than four columns and object type is not recognized
Import-CliXML p.xml # displays using Format-Table because object type is recognized (try to add: | Get-Member)

Variables

```
$i = 1   # storing value 1 in variable $i
$i++   # incrementing $i with 1, resulting in 2
```

Looping

```
for ($i = 1; $i -le 10; $i++) { $i }   # displays numbers 1 through 10. See the Active Directory section for a practical example.
```

While loop only executes when condition is true	Do … While loop, always executes, at least once	Do … Until loop, always executes, at least once
$i = 1	$a = 1	$a = 1
While ($i -le 10) { $i; $i++ }	Do {$a; $a++} While ($a -lt 10)	Do {$a; $a++} Until ($a –gt 10)

Typical example of a Do … Until loop

```
$RequiredLength = 12
Do {        $password = read-host -prompt "Password, please"
            if ($password.length -lt $RequiredLength) { "password is too short!" }
} Until ($password.length -ge $RequiredLength)
```

WMI

```
Get-WmiObject  –list
# inspecting shares through WMI
Get-WmiObject Win32_Share
$s = Get-WmiObject Win32_Share | Where { $_.Name –eq “C$” }
$s  ;  $s | Get-Member  check name en caption
# we’ll need the wmiclass type to create objects through WMI
$sc=[WMICLASS]"Win32_Share"
$sc.create("C:\", "mynewshare", 0)   # creating a new share

# automating defragmentation (please check with your SAN administrator!)
$Cvolume = Get-WmiObject Win32_Volume | Where { $_.name –eq "C:\" }
$df = $Cvolume.DefragAnalysis()   # can take several minutes
$df   # inspecting the result
If ( $df.DefragRecommended ) { $Cvolume.defrag($true) }

Get-WmiObject Win32_OperatingSystem –computername (Get-Content servers.txt) | Format-Table __SERVER,Version,ServicePackMajorVersion,ServicePackMinorVersion
```

Active Directory

Requirements: PowerShell v2, Active Directory Module for Windows PowerShell (on a Domain Controller, also part of RSAT). Windows Server 2008 R2 Domain Controller or install ADMGS on a W2003/2008 Domain Controller. Open port TCP/9389.

```
Import-Module ActiveDirectory   # imports the Active Directory module for PowerShell
Get-Command –module ActiveDirectory   # displays all 76 commands in PowerShell v2
New-ADOrganizationalUnit “Employees” -Path "DC=Contoso,DC=com"   # creates a new OU
Get-ADOrganizationalUnit -Filter “*” | FT Name, DistinguishedName –AutoSize
New-ADUser TestUser1   # creates a disabled user in the Users container

# The next script takes a plain text password as input and creates an enabled user account in the Employees OU
$userpwd = ConvertTo-SecureString -AsPlainText "Pa$$w0rd" –Force   # converts plaintext to secure string
New-ADUser TestUser2 -AccountPassword $userpwd -Enabled $true -Path 'OU=Employees,DC=Contoso,DC=com'

For ($i=1; $i –le 10; $i++) { New-ADUser –name Testuser$i }   # creates ten new testusers
```

Functions

```
Function Newest-Eventlog { Param ($log=”system”, $newest=5) Get-Eventlog $log –newest $newest }
Newest-Eventlog   # try with parameters like –log application –newest 10
```

Background Jobs

```
Start-Job -ScriptBlock { Get-Process PowerShell }
Get-Job
Get-Job -Id 1| Receive-Job   # use the -Keep parameter to keep the data in memory
Start-Job { Sleep 60 }   # starts a new job which just waits for 60 seconds
Wait-Job -Id 3   # wait for a job to complete, or use: Stop-Job -Id 3   # stops job
Remove-Job -Id 3   # remove a completed job
```

Remoting

Requirements: PowerShell v2 on local and remote systems. Run Enable-PSRemoting on remote system. Open port TCP/5985.

```
Enter-PSSession –ComputerName Server2
<use your PowerShell skills now on the remote machine>
Exit-PSSession
```

Error handling and Debugging

```
$ErrorActionPreference   # displays the default action when an error occurs
Dir c:, x:, c:   # should result in a file listing of the c-drive, followed by an error, followed by a listing of the c-drive
$ErrorActionPreference = ‘Continue’   # or ‘SilentlyContinue’ or ‘Inquire’ or ‘Stop’
# Use the –ErrorAction parameter to use a other error action for the current command only
$Error[0] | Get-Member
```

More information on Internet (and the previous page…)

http://blogs.msdn.com/b/powershell/
http://technet.microsoft.com/en-us/scriptcenter/dd742419.aspx
http://poshoholic.com/
http://thepowershellguy.com/
http://www.powershellmagazine.com/
http://www.computerperformance.co.uk/powershell/
http://powerwf.com/products.aspx
http://social.technet.microsoft.com/wiki/contents/articles/windows-powershell-survival-guide.aspx