


John Savill's
PowerShell Master Class




© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

1

Who am I?




@NTFAQGuy

- MCSE NT 4 through 2016, Azure, CISSP, ITIL v3
- Author of the Windows FAQ
- Latest book "Mastering Server 2016 Hyper-V"
- Speaker at Tech Ed, Windows Connections and Ignite
- Large number of Pluralsight courses
- <https://savilltech.com/>
- <http://www.youtube.com/NTFAQGuy>

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

2

Agenda




- Because this is online I will be updating it as required
- Key module themes:
 - PowerShell Fundamentals
 - Remote Management with PowerShell
 - Connecting commands together and mastering objects
 - Creating a PowerShell script
 - Advanced scripting techniques
 - Parsing data and working with objects
 - Desired State Configuration
 - PowerShell Workflows, Azure Automation and Azure Functions
 - The best of the rest

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

3

PowerShell Fundamentals




Before PowerShell
PowerShell Interfaces
Structure of PowerShell Cmdlets and Modules

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

4

The World Without PowerShell




- Many administrators are used to the command prompt
- The command prompt has some built-in commands such as dir and can call separate executables such as ipconfig
- Batch files can be created to perform a fixed set of tasks
- Some pipelining (|) is possible but the output from commands is a text string

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

5

WMI, VBScript, JScript




- There are many scripting languages available which provide a richer language than available in batch files
- Each has its own format and different levels of capability
- Different syntax needed between scripting languages and command line so requires learning multiple systems and environments
- Not highly extensible

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

6

Enter PowerShell




- Formally codename Monad a loooong time ago
- Provides an environment for both executing live commands and creating powerful scripts
- Built on the .NET framework (and now .NET Core) and is focused around objects enabling rich sequences to be created which can perform complex tasks
- Uses a common syntax and common verb-noun pair command naming for cmdlets

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

7

Simpler Language and Standards Based




- Thousands of cmdlets so simplicity is important
- PowerShell is focused on "least cognitive distance" between what you want to do and the actual command
- <verb>-<noun>
- New core cmdlets related to new functionality and also changes to existing cmdlets to match simpler syntax and more features (e.g. Get-Childitem)
- PowerShell can manage non-Windows such as Linux as WSMAN and CIM (WMI2) are the default protocols for management
- PowerShell Core utilizes SSH in addition to WSMAN
- Remote Management enabled by default

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

8

PowerShell Availability




- PowerShell v2 is available for
 - Windows Server 2008 SP1 +
 - Windows Vista SP 2+
 - Windows XP SP 2 +
- Pre-installed in Windows Server 2008 R2 and Windows 7
- .NET Framework 2.0 SP1 required
- .NET Framework 3.5 SP1 required for ISE

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

9

PowerShell v3



- Standard in Windows Server 2012 and Windows 8
- Also available as part of Windows Management Framework (WMF) 3.0
 - <http://www.microsoft.com/en-us/download/details.aspx?id=34595>
- Available for Windows Server 2008 SP2, 2008 R2 SP1 and Windows 7 SP1
- Can be installed side-by-side with PowerShell v2
- Consider this the absolute minimum version

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

10

PowerShell v4




- Standard in Windows Server 2012 R2 and Windows 8.1
- Also available as part of Windows Management Framework (WMF) 4.0
 - <http://www.microsoft.com/en-us/download/details.aspx?id=40855>
- Available for Windows Server 2008 R2 SP1, Windows Server 2012 and Windows 7 SP1
- Still have version 2 available

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

11

PowerShell v5



- New cmdlets such as Get-ItemPropertyValue (to look at property values directly).
- Windows PowerShell Integrated Scripting Environment (ISE) updates
- Windows PowerShell Desired State Configuration (DSC) performance and optimizations (and bug fixes)
- Network Switch cmdlets (in the NetworkSwitch module), which allow management of switches that are Windows Server 2012 R2 certified to have management related to global configuration, VLAN configuration, and Layer 2 port configurations performed.
- OneGet, which allows the discovery and installation of packages from around the web, including services such as Chocolatey
- Part of WMF 5.0

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

12

Easy Access to PowerShell

- Right click on Start and by default shows Command Prompt and Command Prompt (Admin)
- This can be changed to PowerShell through Taskbar properties
- When using PowerShell sometimes you want to run as Admin
- Should say "Administrator" at top
- Can customize layout, font, opacity (Win 10)
- Can also pin to taskbar



© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

19

Using ISE

- The standard PowerShell interface has basic autocomplete
- Integrated Scripting Interface features intellisense scripting and debugging and help windows
- While learning (and even using) the ISE may be your way to go
- ISE does require WPF so not available on Server Core instances and takes a little longer to start
- Snippets are your friend!
- Only supports Windows PowerShell

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

20

Visual Studio Code

- A free, lightweight, cross-platform editor for many languages
- This is the development environment that is the future and has a great deal of investment
- Add-in available to add full syntax and debug support for PowerShell from <https://marketplace.visualstudio.com/items?itemName=ms-vscode.PowerShell>
- Many extensions available including support multiple for terminals
- Native integration with GIT change control
- Supports Windows PowerShell and PowerShell Core (and a lot more)

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

21

Start with Interactive Shell

- Best way to get started is using PowerShell as an interactive shell
- The shell environment can be customized just like a command prompt window
- Start with navigating round the file system
- Dir works!
- But the switches don't, e.g. `dir /s`, why?
- Intrinsic vs external cmd.exe commands

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

22

Using non-PowerShell in PowerShell

- Difference between commands built into the shell (cmd.exe) and executables
- Search for dir.exe or dir.com, you won't find it. This is part of cmd.exe
- It works in PowerShell but only because PowerShell has an alias that resolves to Get-ChildItem
- Therefore the switches used must be those for Get-ChildItem and NOT dir
- If you call an executable, e.g. IPCONFIG.exe then the existing switches work

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

23

Aliases are your friend

- At least initially
- For most cmd.exe intrinsic commands PowerShell has an alias to the PowerShell cmdlet
- However start to learn the proper PowerShell methods rather than relying on aliases long term
- In scripts use the full commands rather than aliases

```
Get-Alias
```

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

24

John Savill's
PowerShell Master Class

Navigating with PowerShell

- Regular commands will work but file system is not the only hierarchical storage system
 - Registry
 - Certificate Store
 - Active Directory
 - RDS
 - IIS
- These can also be navigated and configured like a file system

```
Get-PSDrive
```

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

25

John Savill's
PowerShell Master Class

Tab Support

- Can use tab character for:
 - Cmdlet completion
 - Cycling through parameters
 - Cycling through valid values for parameters (where the parameter has a set of valid values)
 - Cycling through files and folders

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

26

John Savill's
PowerShell Master Class

PowerShell Modules

- Modules are loaded which contain PowerShell functionalities such as cmdlets
- `Get-module` to see those loaded
- `Get-module -listavailable` to see all available
- `Import-module <module>` to add into PowerShell instance
- `Get-Command -Module <module>` to list commands in a module

```
Get-Command -Module Hyper-V | Select  
-Unique Noun | Sort Noun
```

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

27

John Savill's
PowerShell Master Class

Module Versions and Updating

- Modules go through updates
- Check version with
 - `(Get-Module <module name>).Version`
- Closer integration with repositories like GitHub
 - This enables initial installation and update
 - `Install-Module Az`
 - `Update-Module Az`

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

28

John Savill's
PowerShell Master Class

Getting Help

- PowerShell has great help
- `Get-Help <cmdlet>`
- Also available
 - `-full`
 - `-detailed`
 - `-examples`
 - `-online` (to display in web)
 - `-showwindow`
- `Get-Command -Module <module>`
- `Get-Command -Noun <noun>`

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

29

John Savill's
PowerShell Master Class


Updatable Help

- Help for cmdlets is pulled from the Internet and installed into correct location for module (language specific)
 - `Update-Help` (run elevated)
 - `Save-Help` (save to a folder and use this with above Update-Help cmdlet)
- `Get-Help` will automatically pull down latest help
- `Get-Help <cmdlet> -Online` for web based help information
 - `Get-Help New-VHD -Online`
 - `(Get-Command New-VHD).HelpUri`

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

30

Coming up Next



Connecting Commands Together and Mastering Objects

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

31

Connecting Commands Together and Mastering Objects


2

PowerShell Pipeline
Passing and Using Objects
Tricks With Objects

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

32

Semicolon




- Do not have to terminate commands
- You can if you want
- If you want to run multiple commands on one line that are not using objects down the pipe use ;
- <command 1>;<command 2>

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

33

Keeping the objects



- PowerShell does not convert output to text but rather maintains the objects
- This enables easy chaining of commands


```
Get-VMNetworkAdapter <VM Name> |
Set-VMNetworkAdapter -AllowTeaming On
```
- Normal commands can be used with PowerShell

```
ipconfig | select-string -pattern 255
```

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

34

The Pipeline




- PowerShell commands within a session keeps the output as objects
- These objects can be passed along the pipeline to be manipulated by later commands
- Can check the type of an object

```
<var>.GetType()
<var>.GetType().fullname
```

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

35

Default Pipeline Output



- Typically at the end of the pipeline the data is sent to screen
- There are various different Out verb cmdlets
- The default is **Out-Default**
- This directs to **Out-Host**
- Other targets such as
 - **Out-Printer**
 - **Out-GridView** (this is cool and combine with **-passthru** to select then perform an action)
 - **Out-Null** to suppress

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

36

Command > file.txt

- In a command session the > character can be used to send the output to a file
 - `Tasklist > procs.txt`
- This also works in PowerShell
 - `Get-Process > procs.txt`
- Reality is > is a shortcut taking advantage of the pipeline
 - `Get-Process | Out-File procs.txt`

John Savill's PowerShell Resources

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

37

Export Data

- Simple use of pipeline is exporting of the data
- Initially data may be output to screen with a default formatting
 - `Get-Process | Format-Table/Format-List`
- Exporting out to other formats such as a CSV file and XML file
 - `get-process | Export-csv c:\stuff\proc.csv`
 - `get-process | Export-clsxml c:\stuff\proc.xml`
- When importing back keeps the objects(kind of!)
 - `Import-Csv C:\stuff\proc.csv | where {$_.name -eq "notepad"} | Stop-Process`

John Savill's PowerShell Resources

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

38

Limiting Objects Returned

- Can use `Sort-Object`
- Then combine with `-descending,-ascending`
- Send to `Select-Object` with `-first n,-last n`
- For example to get the newest 5 processes use:
 - `Get-Process | Sort-Object -Descending -Property StartTime | Select-Object -First 5`
- Can use `Measure-Object` to count output
 - `Get-Process | Measure-Object`
 - `Get-Process | Measure-Object WS -Sum`

John Savill's PowerShell Resources

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

39

Comparing Objects

- Often you may wish to compare data
- `Compare-Object` can be used to compare two sets of data
- You can select how the compare of objects works
 - `-Property <name>` - finds differences in instances of objects in the results
 - Without this any change would be displayed

John Savill's PowerShell Resources

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

40

Advanced Outputs

- There are various `ConvertTo` cmdlets for advanced output
 - HTML
 - JSON
 - CSV
 - XML
 - Secure String
- Would combine with `Out-File` to output the formatted output to a file

John Savill's PowerShell Resources

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

41

Doing Things with Objects

- Outputting to file is great but the real power is using cmdlets together and keeping the objects
- Cmdlets need to be able to accept as input objects output from the previous cmdlet in the pipeline
- For example (don't run this!)
 - `Get-Process | Stop-Process`
- The above would be very bad
 - `Get-Process -Name notepad | Stop-Process`
- Can use `-passthru` with some commands to allow objects to continue down the pipeline
 - `get-aduser bruce | Disable-ADAccount -PassThru`

John Savill's PowerShell Resources

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

42

Copyright 2019 John Savill
Do not redistribute

7

-confirm and -whatif

- These are parameters that can be added to other commands
- -confirm will prompt before performing any actions
- -whatif will display what would happen without actually performing any real actions
 - Very useful for testing purposes

```
get-aduser -filter * | Remove-ADUser -whatif
```

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

43

More -Confirm

- Every cmdlet has an impact level
- If the impact level is equal or higher than the confirmation preference
 - \$confirmPreference
- To avoid confirmation when not desired add -Confirm:\$false

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

44

Find Members of an Object

- **Get-Member!**
- This will list all the properties, events and methods for the object passed
- This is an easy way to find out what you can do with objects

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

45

What is \$_ ???

- Objects are passed from one command to another but sometimes an attribute of a passed object needs to be used
- **\$** represents the current pipeline object and lets you access a property of a piped in object instead of the entire object

```
Get-Process | Where-Object {$_.name -eq "notepad"} `
| Stop-Process
```

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

46

Simple Syntax Example

```
Get-Process | where {$_.HandleCount -gt 900}
Get-Process | where {$psitem.HandleCount -gt 900}
Get-Process | where HandleCount -gt 900
```

- Can use gps instead of Get-Process (just an alias)
- I prefer to still use \$_ but I'm just set in my ways

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

47

Advanced \$_ use


```
$UnattendFile = "C:\stuff\unattend.xml"
$xml = [xml](gc $UnattendFile)
$child = $xml.CreateElement("TimeZone", $xml.unattend.NamespaceURI)
$child.InnerXml = "Central Standard Time"
$null = $xml.unattend.settings.Where({$.Pass -eq 'oobeSystem'}).component.appendchild($child)
$xml.Save($UnattendFile)
```

- Or
\$resources = Get-AzResourceProvider -ProviderNamespace Microsoft.Compute
\$resources.ResourceTypes.Where({\$_ .ResourceTypeName -eq 'virtualMachines'}).Locations

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author

49

Coming up Next



Remote Management with PowerShell

© Copyright 2019 John Savill. All rights reserved.
No part of this presentation may be used without express permission from the author