VMUG usercon

Zero to Hero with PowerShell & PowerCLI

Mike

mikenelsonio

nelmedia

mikenelson-io

- Almost 40 years in tech
- Principal Technical Evangelist @ Pure Storage
- Experience from Helpdesk to Architect
- Scripter, not a coder
- Passion for community, teaching, learning
- Beer, BBQ, & Gadgets

/MyPresentations

VMUG usercon

Why do you, or why would you, use PowerCLI?

Fact: PowerCLI does not exist without PowerShell

PowerShell

aka “PoSH”

Started as a scripting framework for automation & evolved into a command line interface (CLI) and a scripting language.

Native executables, cmdlets, scripts, functions, aliases, modules, help, profiles, parameters, and so much more.

Versions

.Net Framework

<=5.1

Windows

.NET Core

6.x

7.x

Windows
Linux
MacOS

Core Stuff

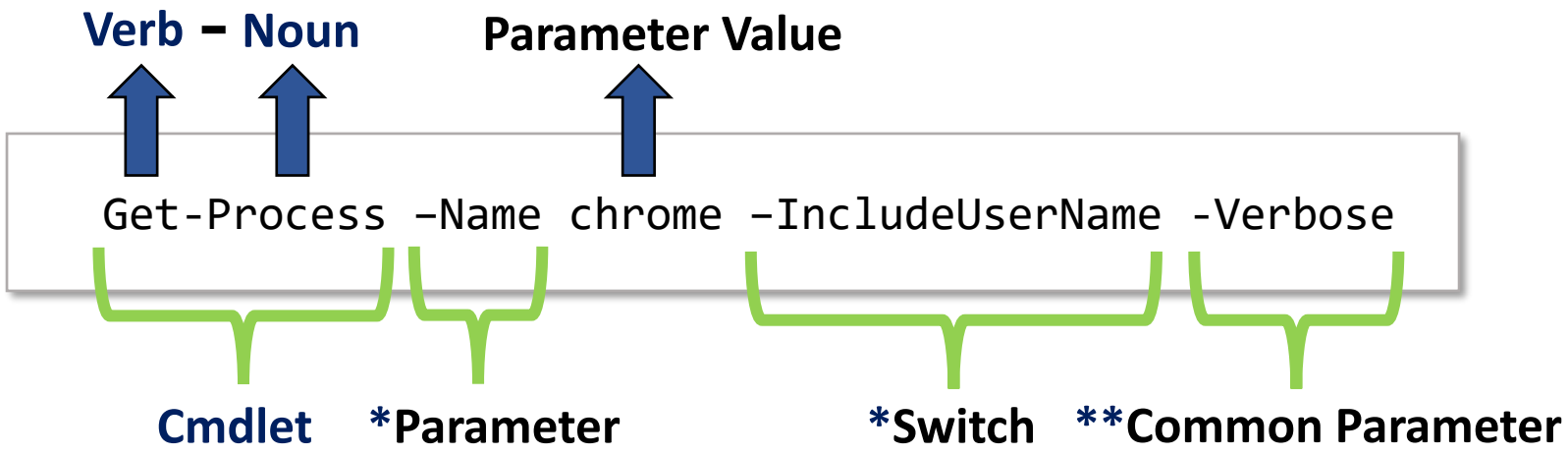
- Profiles

Basically “logon scripts” for PowerShell. Their main function is to pre-set the environment, but also does other things.

- Cmdlets (“command-lets”)

The soul of PowerShell. It is a type of command in PowerShell. Used at the command line or in a script (.ps1 file).

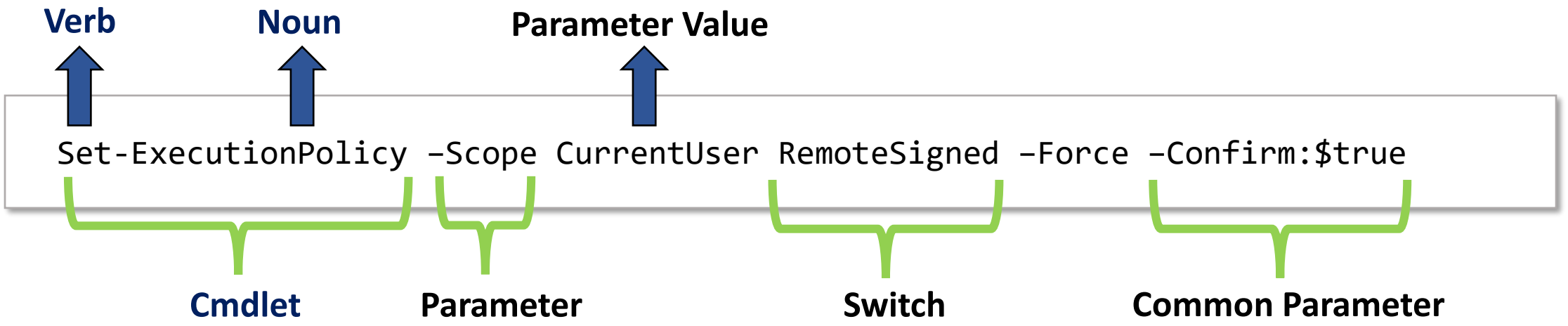
Example Syntax



* = Optional or required

** = the cmdlet may or may not support

Example Syntax v2



Core Stuff

- Profiles
- cmdlets
- Variables

A unit of *memory* to which a value is stored.

- Parameters

Allows the user to provide input and/or options to cmdlets.

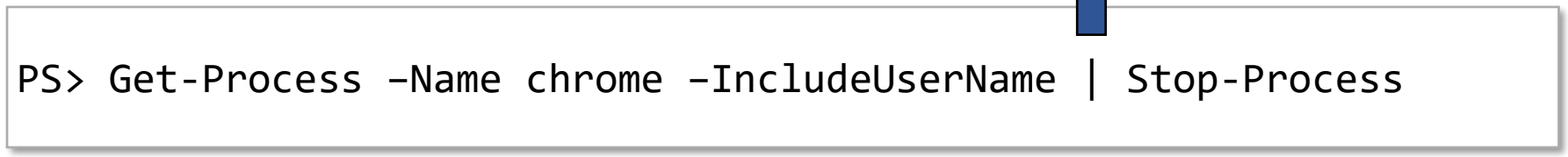
- Pipelines

Allows for the “piping” of output objects to and from other cmdlets.

- Functions
- Modules
- Scripts

Pipelines

Pipeline operator



```
PS> Get-Process –Name chrome –IncludeUserName | Stop-Process
```

Object(s) returned by first cmdlet are sent (‘piped’) to the second cmdlet

“One-liner”

Pipelines

“One-liner”

Get all Windows VMs that need updated tools, then update all the tools at once

```
PS> Get-VM -Location 'MyDatacenter' | Where-Object { $_.ExtensionData.Guest.ToolsVersionStatus
 -eq 'guestToolsNeedUpgrade' -and $_.PowerState -like 'PoweredOn' } |
Get-VMGuest | Where-Object { $_.GuestFamily -like 'WindowsGuest'} |
Update-Tools -NoReboot -RunAsync
```

cmd.exe max character limit? 8,191

PowerShell max character limit? 32,764

PowerShell command separator? “||” Ex. Get-Process || Get-Disk

Core Stuff

- Profiles
- cmdlets
- Variables

A unit of *memory* to which a value is stored.

- Parameters

Allows the user to provide input and/or options to cmdlets.

- Pipelines

Allows for the “piping” of output objects to other cmdlets, functions, or commands.

- Functions
- Modules
- Scripts

Functions

A list of PowerShell statements that run like you had entered them on the command line.

```
function Get-ChromeProcess { Get-Process chrome }

function Get-ChromeProcess {
  $a = Get-Process chrome
    if ($a –eq $null) {
      Write-Host “No Chrome process present”}
    return $a
}

Get-ChromeProcess
```

To run a function, simply “call” it.

Scripts & Modules

- Review: Scripts are .ps1 files
- Modules are .psm1 files, which can contain commands, providers, variables, functions, help context, aliases, workflows, etc., all bundled into a single file.
- A .psd1 file is a module manifest file, which is basically a definition file for a module.
- Modules can be autoloaded by PowerShell.

Example Manifest & Modules

VMware-vCD-TenantReport.psd1

Get-NicDetails.psm1

Get-NewAndRemovedVMs.psm1

Core Commands to Know

Get-Command

Show-Command

Get-Help

-ShowWindow

Get-Member

Update-Help

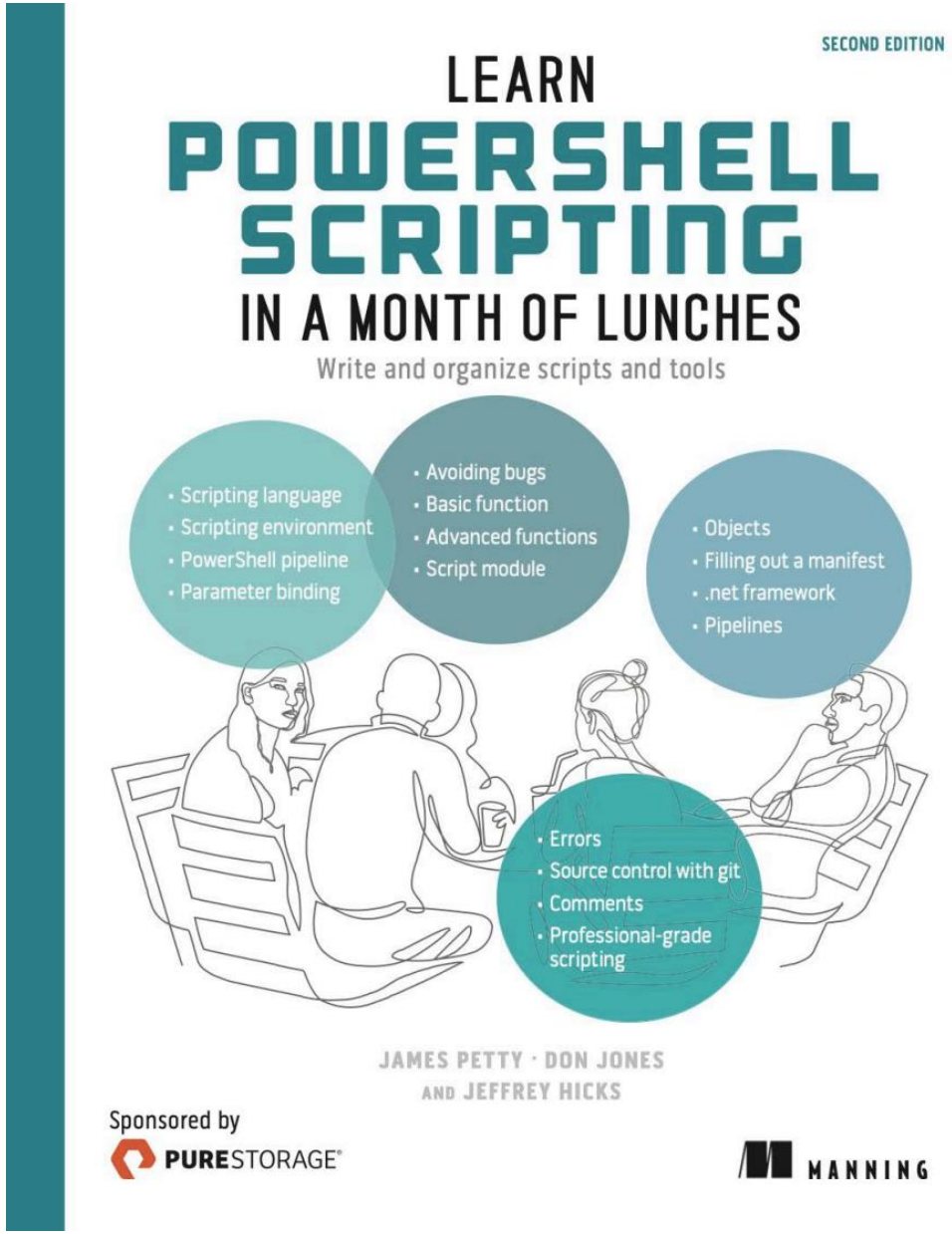
Update-Module

PowerCLI

- **~1000** cmdlets in 28+ modules
- Install-Module VMware.PowerCLI
- Cmdlet collisions such as “Get-VM”
  - Import-Module VMware.PowerCLI –Prefix “Vmx”
    - Get-VM becomes Get-VmxVM
  - Use –NoClobber parameter

28 modules!

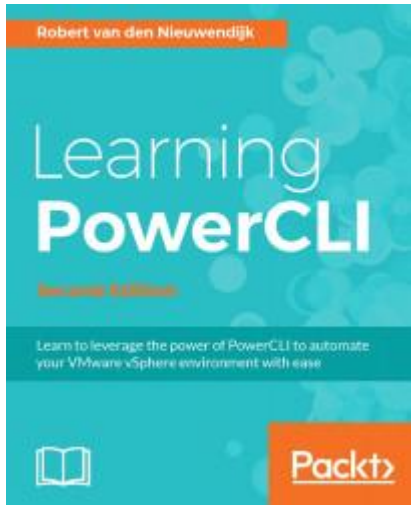
```
RequiredModules = @(
@{"ModuleName"="VMware.VimAutomation.Sdk";"ModuleVersion"="13.1.0.21605170"}
@{"ModuleName"="VMware.VimAutomation.Common";"ModuleVersion"="13.1.0.21605386"}
@{"ModuleName"="VMware.Vim";"ModuleVersion"="8.1.0.21605554"}
@{"ModuleName"="VMware.VimAutomation.Core";"ModuleVersion"="13.1.0.21606170"}
@{"ModuleName"="VMware.VimAutomation.Srm";"ModuleVersion"="12.7.0.20091290"}
@{"ModuleName"="VMware.VimAutomation.License";"ModuleVersion"="12.0.0.15939670"}
@{"ModuleName"="VMware.VimAutomation.Vds";"ModuleVersion"="13.1.0.21610933"}
@{"ModuleName"="VMware.CloudServices";"ModuleVersion"="12.6.0.19606210"}
@{"ModuleName"="VMware.VimAutomation.Vmc";"ModuleVersion"="13.0.0.20797723"}
@{"ModuleName"="VMware.VimAutomation.Nsxt";"ModuleVersion"="13.1.0.21606089"}
@{"ModuleName"="VMware.VimAutomation.vROps";"ModuleVersion"="13.1.0.21611158"}
@{"ModuleName"="VMware.VimAutomation.Cis.Core";"ModuleVersion"="13.1.0.21605976"}
@{"ModuleName"="VMware.VimAutomation.HorizonView";"ModuleVersion"="13.1.0.21610272"}
@{"ModuleName"="VMware.VimAutomation.Cloud";"ModuleVersion"="13.1.0.21611174"}
@{"ModuleName"="VMware.DeployAutomation";"ModuleVersion"="8.0.0.21610665"}
@{"ModuleName"="VMware.ImageBuilder";"ModuleVersion"="8.0.0.21610262"}
@{"ModuleName"="VMware.VimAutomation.Storage";"ModuleVersion"="13.1.0.21606282"}
@{"ModuleName"="VMware.VimAutomation.StorageUtility";"ModuleVersion"="1.6.0.0"}
@{"ModuleName"="VMware.VumAutomation";"ModuleVersion"="12.7.0.20091294"}
@{"ModuleName"="VMware.VimAutomation.Security";"ModuleVersion"="13.1.0.21606510"}
@{"ModuleName"="VMware.VimAutomation.Hcx";"ModuleVersion"="13.0.0.20803747"}
@{"ModuleName"="VMware.VimAutomation.WorkloadManagement";"ModuleVersion"="12.4.0.18627055"}
@{"ModuleName"="VMware.Sdk.Runtime";"ModuleVersion"="1.0.1111.21624264"}
@{"ModuleName"="VMware.Sdk.vSphere";"ModuleVersion"="8.0.1111.21624264"}
@{"ModuleName"="VMware.PowerCLI.VCenter";"ModuleVersion"="12.6.0.19600125"}
@{"ModuleName"="VMware.Sdk.Nsx.Policy";"ModuleVersion"="4.1.0.21605558"}
@{"ModuleName"="VMware.Sdk.Srm";"ModuleVersion"="8.7.0.21605564"}
@{"ModuleName"="VMware.Sdk.Vr";"ModuleVersion"="8.7.0.21605566"}
# @{"ModuleName"="VMware.Sdk.Vcf.CloudBuilder";"ModuleVersion"="0.0.0.0"}
# @{"ModuleName"="VMware.Sdk.Vcf.SddcManager";"ModuleVersion"="0.0.0.0"}
```

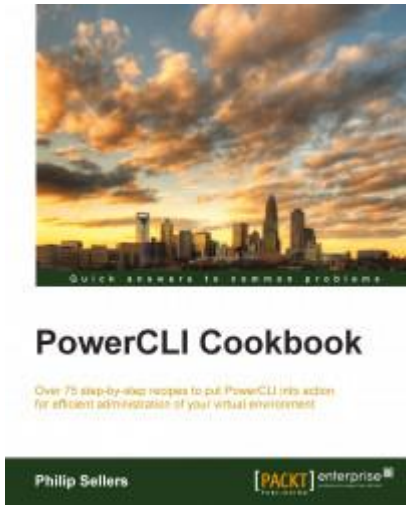


VMUG usercon

https://www.purestorage.com/resources/type-a/powershell-in-a-month-of-lunches.html

VMUG usercon





DEMOS

Thank You!
@mikenelsonio

VMUG usercon

TAKE THE SESSION SURVEY

VMUG usercon