
POWERSHELL JUMPSTART FOR SQL SERVER DBAS

Thanks to our Sponsors!

Yearly Partners



Gold Sponsors





Microsoft®
SQL Server®



www.mikefal.net

IDEA
ACE



Mike Fal

Microsoft
CERTIFIED
Solutions Expert

Data Platform

Don't Panic!

```
Strns = Get-ChildItem $dir -recurse | Where-Object {$_.name -like "*.trn"} | sort-object LastWriteTime
}
else{
    $full = Get-ChildItem $dir | Where-Object {$_.name -like "*.bak"} | Sort-Object LastWriteTime -desc | Select-Object -first 1
    $diff = Get-ChildItem $dir | Where-Object {$_.name -like "*.dff"} | sort-object LastWriteTime -desc | select-object -first 1
    $trns = Get-ChildItem $dir | Where-Object {$_.name -like "*.trn"} | sort-object LastWriteTime
}

#initialize and process full backup
$outputfile = Join-Path -Path $outputdir -ChildPath "restore_$database.sql"
$restore = Get-RestoreObject $database $full
$shfull = Get-Header $restore $smosrv
if($database.Length -eq 0)
{
    $database = $shfull.DatabaseName
    $restore.Database=$database
}

$LSNCheck = $shfull.CheckpointLSN
$files = $restore.ReadFileList($smosrv)
foreach($file in $files){
    $pfile = $file.PhysicalName
    if($newdata.Length -gt 0 -and $file.Type -eq "D"){
        $pfile=$newdata + $pfile.Substring($pfile.LastIndexOf("\"))
    }

    if($newdata.Length -gt 0 -and $file.Type -eq "L"){
        $pfile=$newlog + $pfile.Substring($pfile.LastIndexOf("\"))
    }

    $newfile = New-Object("Microsoft.SqlServer.Management.Smo.RelocateFile") ($file.LogicalName,$pfile)
    $restore.RelocateFiles.Add($newfile) | out-null
}

$ssqlout += "/*****"
$ssqlout += "Restore Database Script Generated $(Get-Date)"
$ssqlout += "Database: "+$database
$ssqlout += "*****/"
$ssqlout += "--FULL RESTORE"
If($owner){$ssqlout += "EXECUTE AS LOGIN = '$owner';"}
$ssqlout += $restore.Script($smosrv)

#process differential backups
if($diff -ne $null){
    $restore = Get-RestoreObject $database $diff
    $hdiff = Get-Header $restore $smosrv

    if($hdiff.DatabaseBackupLSN -eq $LSNCheck){
        $ssqlout += "--DIFF RESTORE"
        $ssqlout += $restore.Script($smosrv)
        $LSNCheck = $hdiff.LastLSN
    }
    else{
```

**Don't focus on the code,
focus on the concepts.**

Ask questions!

Get-Agenda

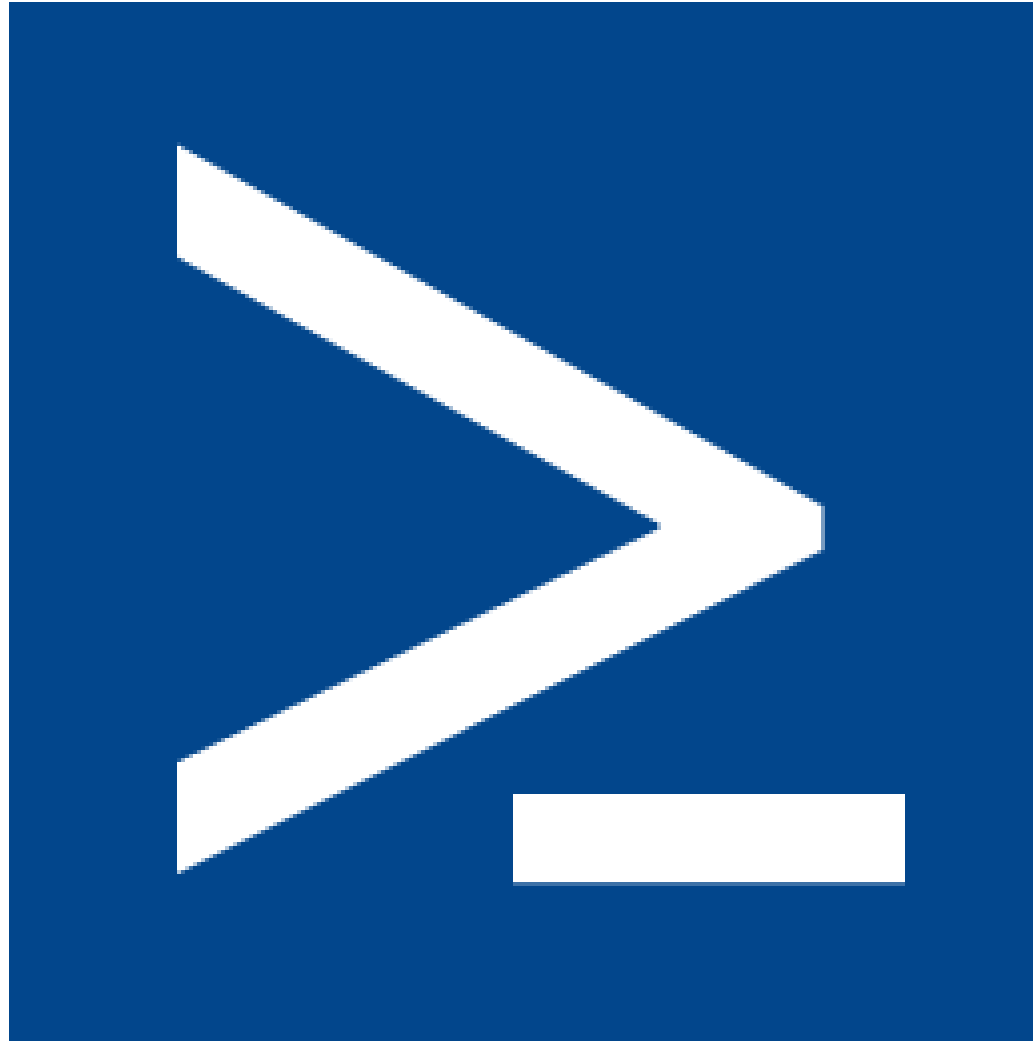
The What and Why of Powershell

Language Basics

Working with SQL Server

And Then What?

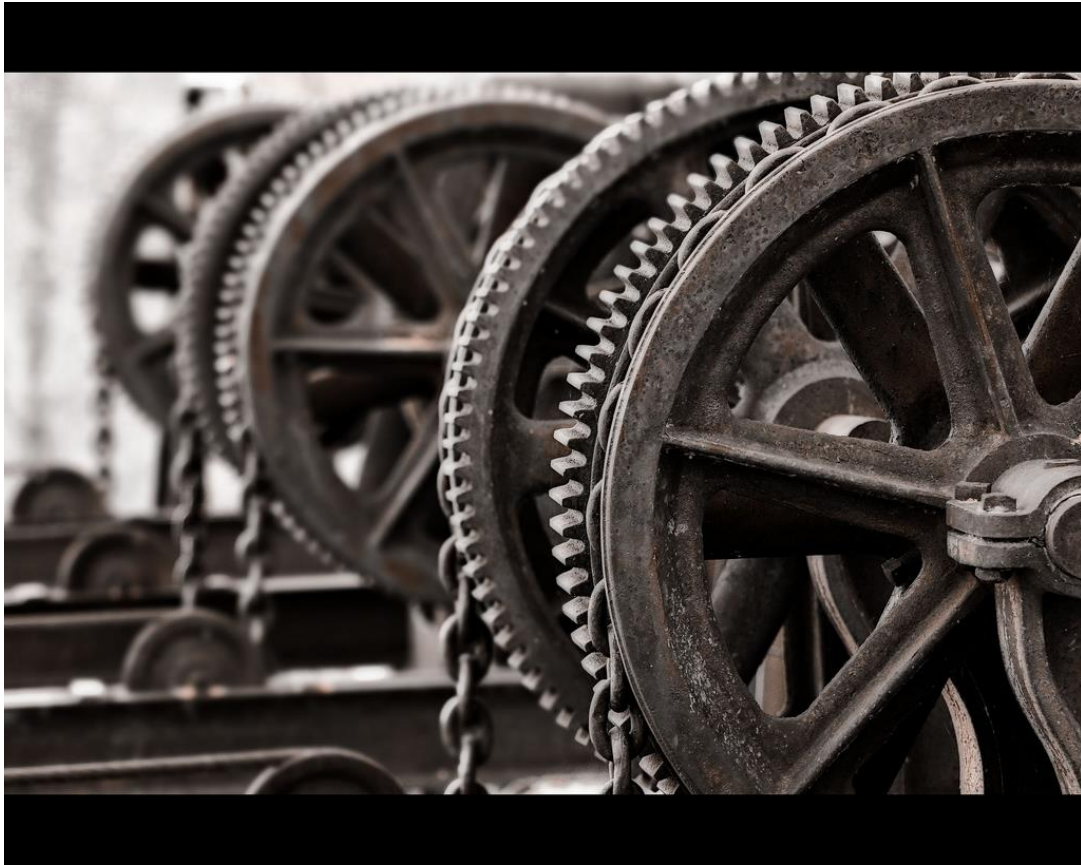
What is Powershell?



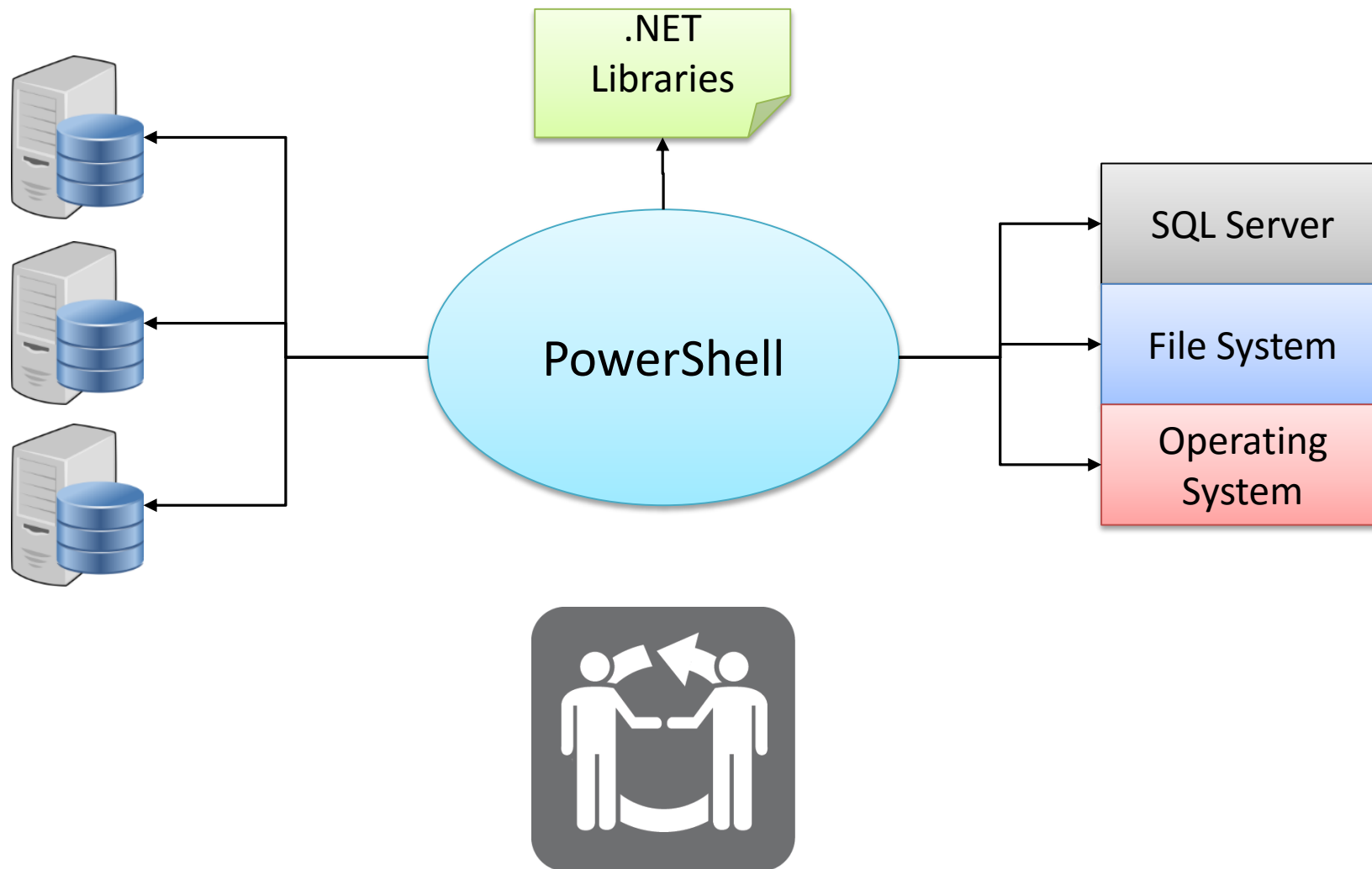
The Facts

- Envisioned by Jeffery Snover – 2002
 - The Monad Manifesto
- Released as Powershell RC 1 – April 2006
 - Originally called Project “Monad”
- Current available version: 4.0
 - 5.0 is in preview!

But for what?

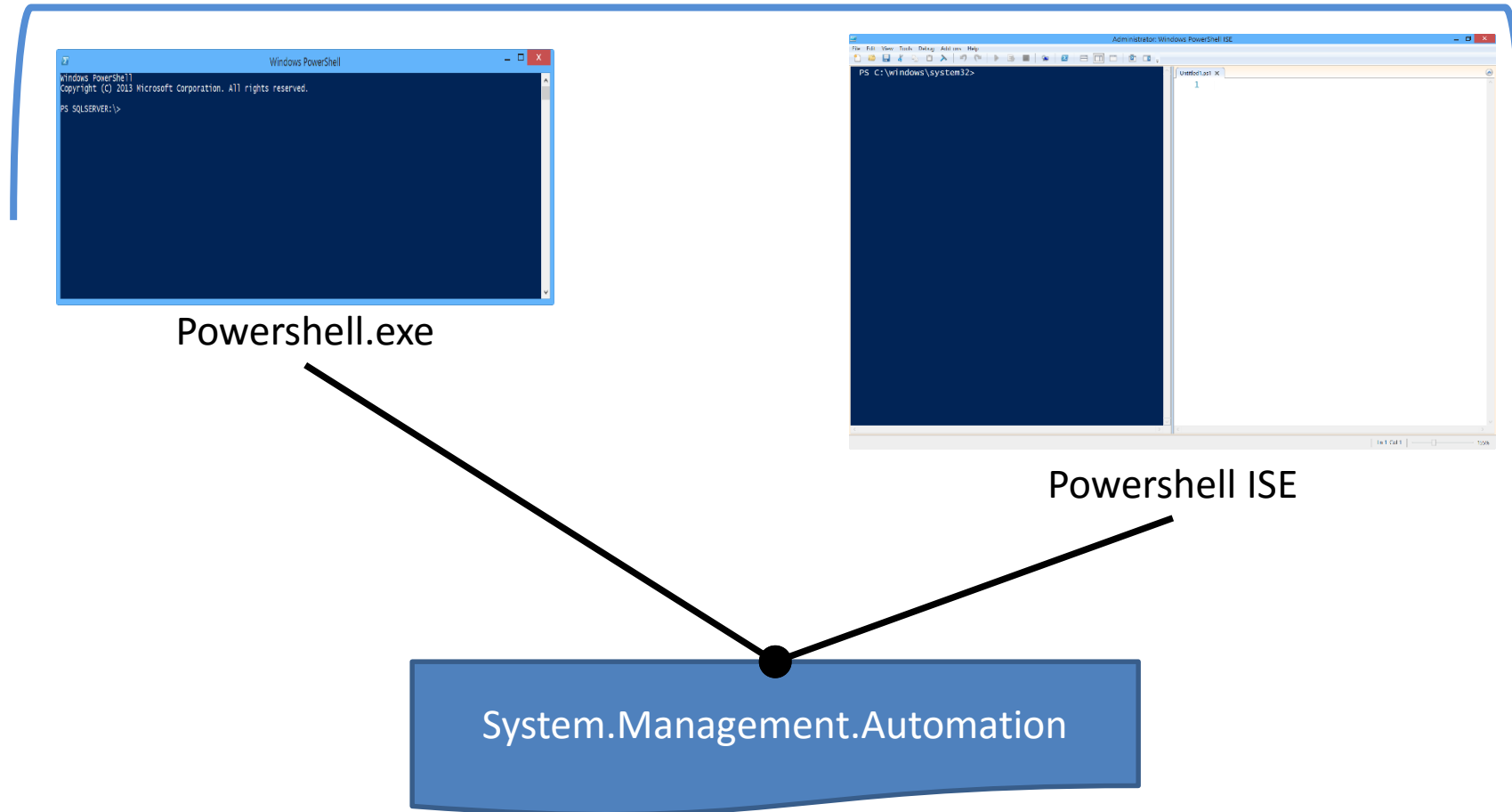


Why PowerShell?

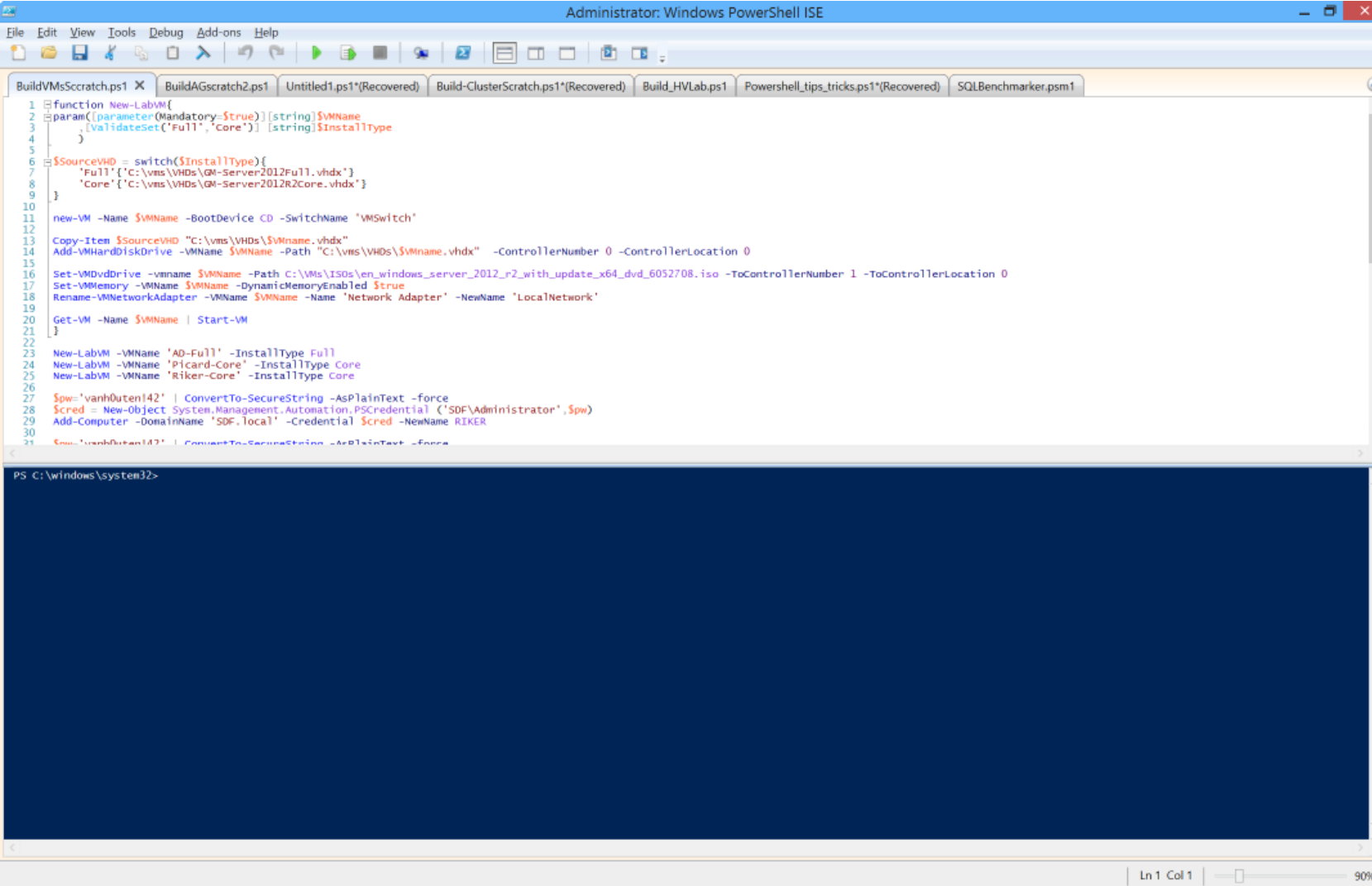


Pieces and Parts

Hosts



Demo – The ISE



The screenshot shows the Windows PowerShell ISE interface. The title bar reads "Administrator: Windows PowerShell ISE". The menu bar includes File, Edit, View, Tools, Debug, Add-ons, and Help. The toolbar contains various icons for file operations and execution. The script editor displays a PowerShell script with line numbers 1 through 31. The script defines a function named New-LabVM, which takes two parameters: \$VMName (mandatory, string) and \$InstallType (validates to 'Full' or 'Core', string). The function logic includes setting the source VHD based on the install type, creating a new VM, adding a hard disk, setting the DVD drive to a Windows Server 2012 R2 ISO, enabling dynamic memory, renaming the network adapter, and starting the VM. At the bottom of the script, three VMs are created: 'AD-Full' (Full install), 'Picard-Core' (Core install), and 'Riker-Core' (Core install). The script also sets a password for the administrator and adds a local user named 'RIKER'. The console window at the bottom shows the prompt "PS C:\windows\system32>" and is currently empty. The status bar at the bottom right indicates "Ln 1 Col 1" and "90%" zoom.

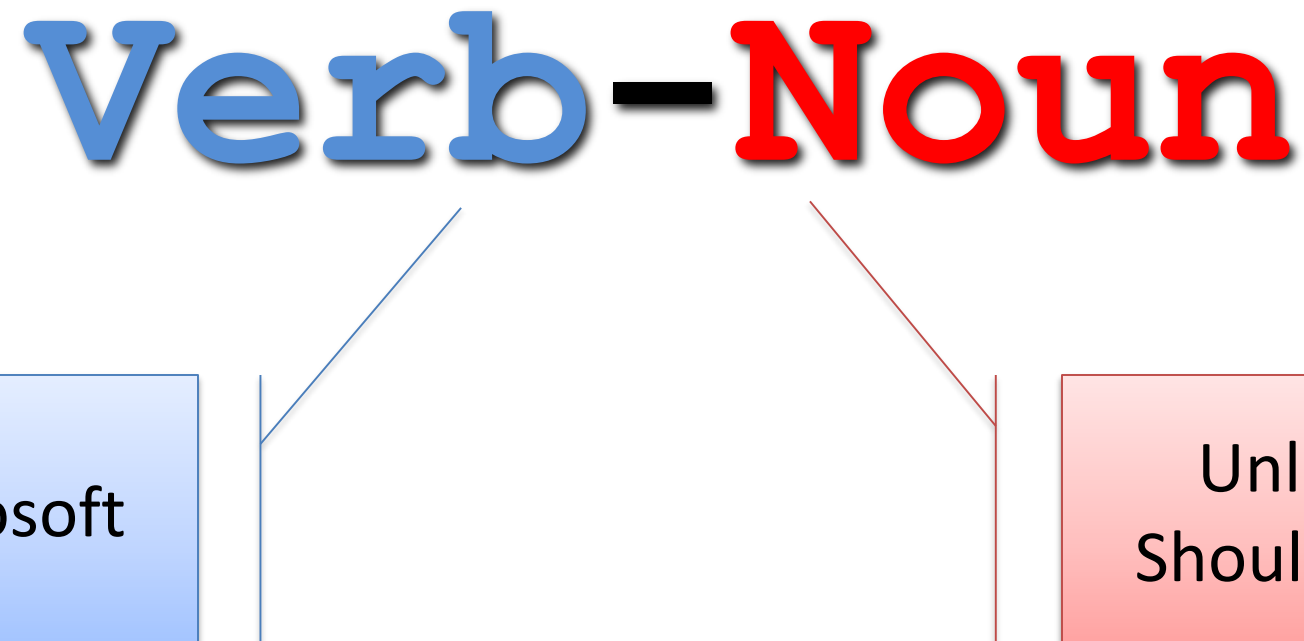
```
1 function New-LabVM{
2 param([parameter(Mandatory=$true)][string]$VMName
3       ,[ValidateSet('Full','Core')][string]$InstallType
4       )
5
6 $SourceVHD = switch($InstallType){
7     'Full' {'C:\vms\VHDs\GM-Server2012Full.vhdx'}
8     'Core' {'C:\vms\VHDs\GM-Server2012R2Core.vhdx'}
9 }
10
11 new-vm -Name $VMName -BootDevice CD -SwitchName 'VMSwitch'
12
13 Copy-Item $SourceVHD "C:\vms\VHDs\$VMName.vhdx"
14 Add-VMHardDiskDrive -VMName $VMName -Path "C:\vms\VHDs\$VMName.vhdx" -ControllerNumber 0 -ControllerLocation 0
15
16 Set-VMdvdDrive -vmname $VMName -Path C:\VMS\ISOs\en_windows_server_2012_r2_with_update_x64_dvd_6052708.iso -ToControllerNumber 1 -ToControllerLocation 0
17 Set-VMemory -VMName $VMName -DynamicMemoryEnabled $true
18 Rename-VMNetworkAdapter -VMName $VMName -Name 'Network Adapter' -NewName 'LocalNetwork'
19
20 Get-VM -Name $VMName | Start-VM
21 }
22
23 New-LabVM -VMName 'AD-Full' -InstallType Full
24 New-LabVM -VMName 'Picard-Core' -InstallType Core
25 New-LabVM -VMName 'Riker-Core' -InstallType Core
26
27 $pw='vanh0uten142' | ConvertTo-SecureString -AsPlainText -force
28 $cred = New-Object System.Management.Automation.PSCredential ('SDF\Administrator',$pw)
29 Add-Computer -DomainName 'SDF.local' -Credential $cred -NewName RIKER
30
31 $pw='vanh0uten142' | ConvertTo-SecureString -AsPlainText -force
```

PS C:\windows\system32>

cmdlets

Fundamental unit of “getting stuff done”

Verb—**Noun**



Limited by Microsoft

Unlimited values,
Should be descriptive

cmdlet examples

Get-Help

New-Item

Remove-Module

Learn Within Powershell

Get-Command

List Commands:
New
-Module SQLPS

Get-Help

Show help info:
man, help
-ShowWindow

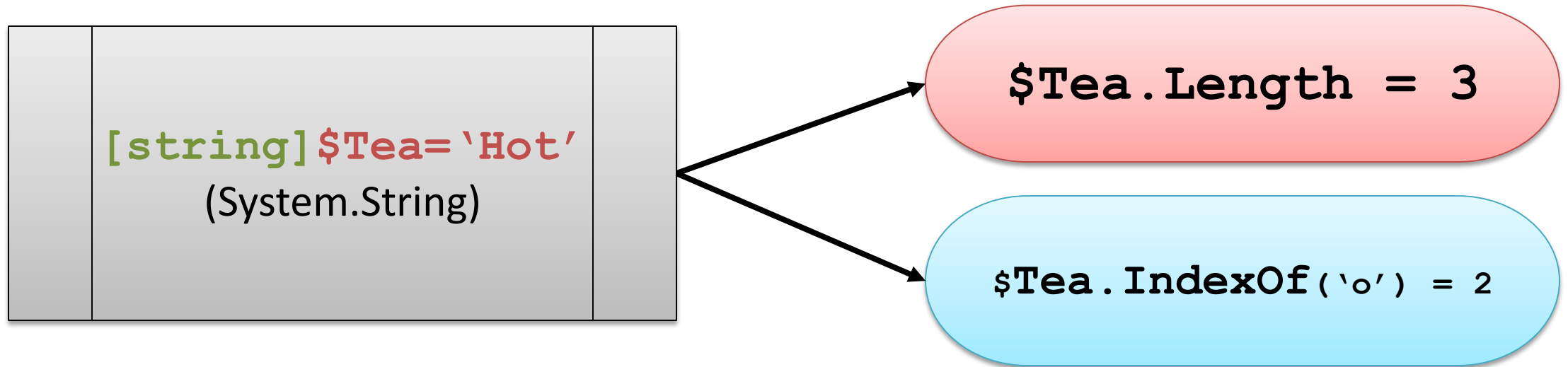
Get-Member

Methods and properties

Object Oriented Thinking

Everything is a .Net object!

- Properties (attributes)
- Methods (functions, do things)



Variables

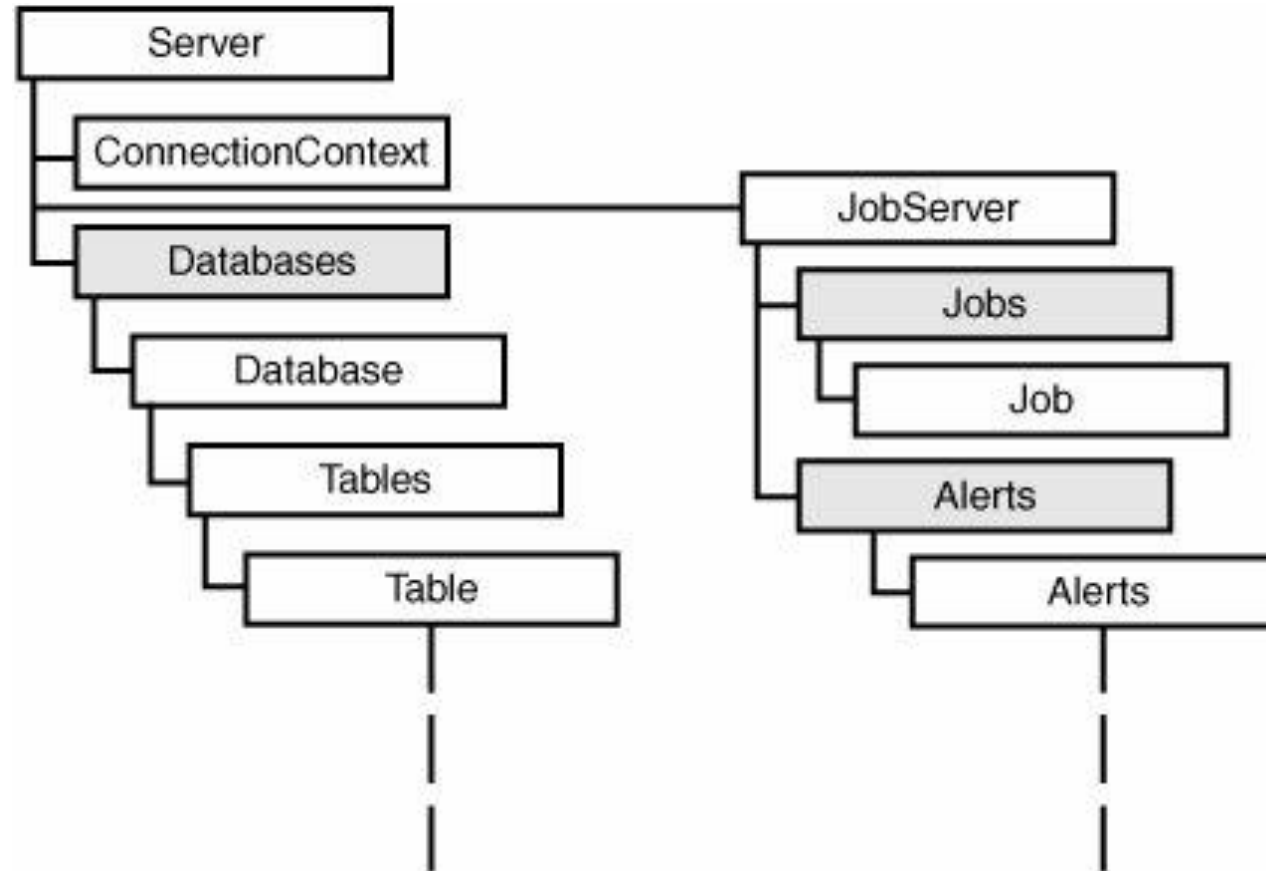
‘\$’ indicates a variable

```
[string]$Tea = 'Hot'
```

```
$Tea = 'Hot'
```

```
$TeaTime = Get-Date
```

Demo – CMDLETS and Objects



The SQL Server Module



Import-Module SQLPS (SQL 2012+ client)

SQL Server cmdlets SQL Server Provider

Demo – Working with SQL Server

```
PS SQLSERVER:\> Get-Command -Module SQLPS
```

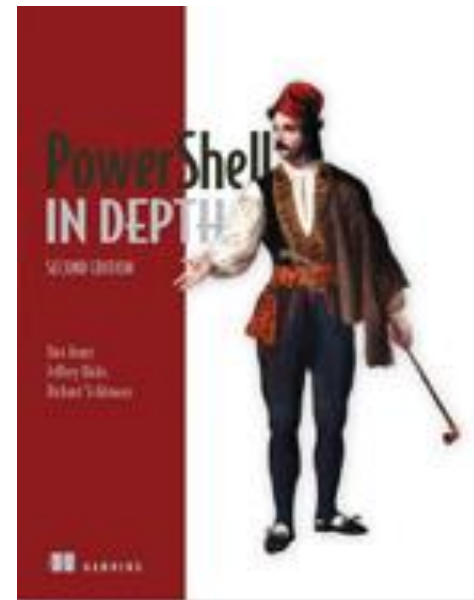
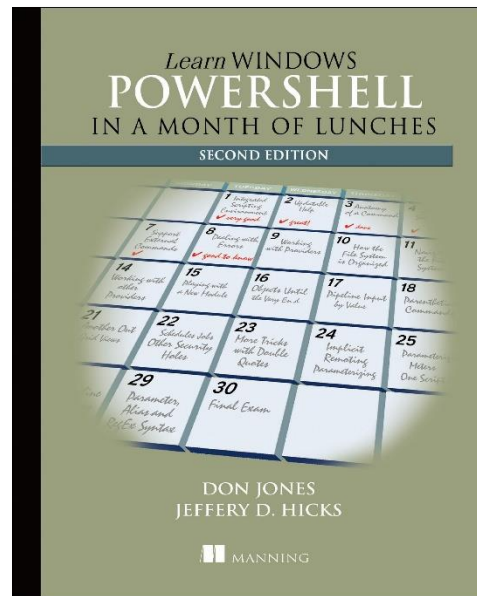
CommandType	Name	ModuleName
-----	----	-----
Function	SQLSERVER:	SQLPS
Cmdlet	Add-SqlAvailabilityDatabase	SQLPS
Cmdlet	Add-SqlAvailabilityGroupListenerStaticIp	SQLPS
Cmdlet	Add-SqlFirewallRule	SQLPS
Cmdlet	Backup-SqlDatabase	SQLPS
Cmdlet	Convert-UrnToPath	SQLPS
Cmdlet	Decode-SqlName	SQLPS
Cmdlet	Disable-SqlAlwaysOn	SQLPS
Cmdlet	Enable-SqlAlwaysOn	SQLPS
Cmdlet	Encode-SqlName	SQLPS
Cmdlet	Get-SqlCredential	SQLPS
Cmdlet	Get-SqlDatabase	SQLPS
Cmdlet	Get-SqlInstance	SQLPS
Cmdlet	Get-SqlSmartAdmin	SQLPS
Cmdlet	Invoke-PolicyEvaluation	SQLPS
Cmdlet	Invoke-Sqlcmd	SQLPS
Cmdlet	Join-SqlAvailabilityGroup	SQLPS
Cmdlet	New-SqlAvailabilityGroup	SQLPS
Cmdlet	New-SqlAvailabilityGroupListener	SQLPS
Cmdlet	New-SqlAvailabilityReplica	SQLPS
Cmdlet	New-SqlBackupEncryptionOption	SQLPS
Cmdlet	New-SqlCredential	SQLPS
Cmdlet	New-SqlHADREndpoint	SQLPS

So now what?



Books

- [Powershell In A Month of Lunches](#)
- [Powershell in Depth, 2nd Edition](#)



General Powershell

- The Scripting Guys
(<http://blogs.technet.com/b/heyscriptingguy/>)
- Jeff Hicks
(<http://jdhitsolutions.com/blog/>)
- Powershell.org
(<http://powershell.org/>)

Bloggers

- Ben Miller
(<http://www.dbaduck.com/>)
- Allen White
([http://sqlblog.com/blogs/allen white/](http://sqlblog.com/blogs/allen_white/))
- Kendal Van Dyke
(<http://www.kendalvandyke.com/>)
- Laerte Junior
(<https://www.simple-talk.com/author/laerte-junior/>)

Online

Microsoft Virtual Academy

PluralSight (\$\$)



Other scripts

<https://github.com/MikeFal>

- Powershell repository
(all my scripts, including WIP)



Most of all, USE IT

Find tasks to automate

Manage the file system **ONLY** through Powershell

Rewrite a T-SQL or other task using Powershell

And so on...



Questions



mike@mikefal.net



www.mikefal.net



[@Mike_Fal](https://twitter.com/Mike_Fal)