

Visual Studio LIVE!  
EXPERT SOLUTIONS FOR .NET DEVELOPERS

ROCK YOUR CODE  
TOUR 2017

CHICAGO

## Brian.About();



- Co-founder DuoMyth
  - Partner with MCW Technologies
  - Member of Pluralsight's Technical Staff
  - Co-author Pro ALM 2013 from Wrox (<http://bri.gd/bcazba01>)
  - Microsoft MVP for Visual Studio and Development Technologies
- brianr@mcwtech.com | @brianrandell | blog.brianrandell.com





## Why PowerShell

- Save time
- Automate repetitive tasks
- Access APIs more easily than via .NET / native code
- Share the love with the IT Pro in your life
- Available on all modern versions of Windows—no install hassle

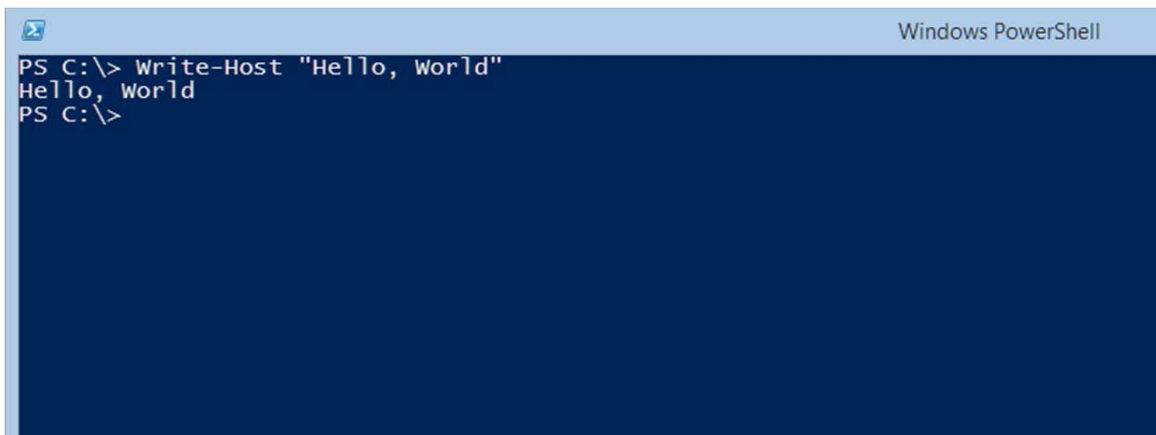


## Why PowerShell

- Save time
- Automate repetitive tasks
- Access APIs more easily than via .NET / native code
- Share the love with the IT Pro in your life
- Available on all modern versions of Windows—no install hassle—and in preview on Windows, macOS, and Linux as PowerShell Core.



## Hello World

A screenshot of a Windows PowerShell terminal window. The title bar is light blue and says "Windows PowerShell". The terminal has a dark blue background. The text shown is:

```
PS C:\> Write-Host "Hello, World"
Hello, World
PS C:\>
```



## What is PowerShell

*PowerShell is a distributed, scalable, heterogeneous configuration, and automation framework, consisting of an interactive command-line shell, a scripting language, and a set of tooling (integrated scripting environment) for Windows, macOS, and Linux.*

– Jeffrey Snover, Microsoft, the “father” of PowerShell



# PowerShell: Getting Started

### Essential Commands

- First use Tab expansion to help complete commands
- \$PSxxx variables provide information
  - \$PSVersionTable gives you the version of PowerShell
- Get-Help to get syntax and examples
- Get-Command lists all commands available current sessions
- Get-PSDrive to system "drives"



### Tools of the Trade

- Built-in to Windows
  - PowerShell Console
  - PowerShell ISE
- Separate Installs
  - Visual Studio Code
    - Install the PowerShell Editor Services extension
    - Ctrl+P, then "ext install PowerShell" and install the extension named "PowerShell"
  - Visual Studio 2012, 2013, 2015, and 2017
    - Install PowerShell support via Installer or via Gallery
    - Microsoft MVP written endorsed by Microsoft



## Security: PowerShell Script Execution

- Microsoft has disabled PowerShell scripts by default
- Use `Set-ExecutionPolicy` to control script security
  - **Restricted**—Only allows interactive commands. PowerShell scripts are not allowed to execute (this is the default).
  - **All Signed**—Only allows scripts signed by a trusted publisher to execute.
  - **Remote Signed**—PowerShell uses the URL Security Zones API to determine where a script came from. It allows local scripts to run. Those from other zones are blocked unless they are signed by a trusted publisher.
  - **Unrestricted**—"Danger Will Robinson!"
    - Uh it does what it says. Consider carefully.
    - Must have Administrative rights to execute
- Use `Get-ExecutionPolicy` to determine current state



## cmdlet vs. script

- "A cmdlet is a lightweight command that is used in the Windows PowerShell environment."<sup>1</sup>
- You generally write cmdlets using languages like Visual Basic or Visual C#
  - However, you can use PowerShell to write "advanced functions"<sup>2</sup>
- A script is a text file with a `.ps1` extension organized to execute one or more PowerShell commands and/or cmdlets



### Error handling

- Bad things happen to good scripters
- Consider using proper error handling
- PowerShell supports Try/Catch/Finally blocks
  - **Try**—do something
  - **Catch**—when an error occurs it's packaged as an exception object—you catch it, examine it, and determine what to do
    - You can have more than one catch block—each one catches a specific type of exception
  - **Finally**—before leaving, do some final clean up—just be careful not to cause another error



### Writing to the Event Log

- Good scripts log what they're doing
- Use the **Write-EventLog** cmdlet
- You must first have an Event Source that you can create with the **New-EventLog** cmdlet

```
New-EventLog -Source "VSLIVE" -LogName Application
Write-EventLog -LogName Application -Source "VSLIVE" -EntryType Information
-EventId 1 -Message "Testing"
```



## Reading a list of items from a File

- When automating a set of “things”, it’s often nice to get the list from a text file
- The **Get-Content** cmdlet is your friend
- Pipeline the content to **Foreach-Object** and you can “process” multiple items easily in one line of code



## Pipelining key in PowerShell

- Feed “objects” or data to a command
- Process groups of items

```
Get-Process | Select-Object -Property Name, VM, CPU
```





### Local vs. Remote Automation

- Microsoft designed PowerShell for remote automation
- **Enable-PSRemoting** may be required
  - Requires Administrative permissions
- **Enter-PSSession** starts a remote session
- Read PowerShell FAQ for more details
  - `Get-Help about_Remote_FAQ` in PowerShell
  - <http://technet.microsoft.com/library/hh847845.aspx>



### Remote Access to Workgroup Machine

- # Check to see if trusted (next two commands requires Administrative rights)
- `Get-Item wsman:\localhost\Client\TrustedHosts`
- # Trust remote
- `Set-Item WSMan:\localhost\Client\TrustedHosts %nameorIP%`
- # Connect using PowerShell Remoting
- `$c = %nameoripd%`
- `$user = %username%`
- `Enter-PSSession -ComputerName $c -Credential $user`
- # Exit Remote Session
- `Exit-PSSession`
- # Reset Trusted List (requires Admin rights)
- `Clear-Item -Path wsman:\localhost\Client\TrustedHosts`



# Example Uses: Hyper-V

## Hyper-V API Support

- All releases of Hyper-V support automation via WMI
- Hyper-V on Windows 8 and Windows Server 2012 added built-in support for PowerShell
- Lots of cmdlets
  - PowerShell command **Get-Command -Module Hyper-V** will give you the current list
  - [http://technet.microsoft.com/en-us/library/hh848559\(v=wps.620\).aspx](http://technet.microsoft.com/en-us/library/hh848559(v=wps.620).aspx)
- What about PowerShell & Hyper-V on Server 2008 R2 SP1?
  - PowerShell Management Library on CodePlex
    - <http://pshyperv.codeplex.com/>
  - Created by a former Microsoft employee, James O'Neill
    - <http://jamesone111.wordpress.com/>



## Security: User Rights for Hyper-V

- Be a local Administrator
  - If UAC is enabled you'll need to run elevated to get full access
- Be a member of the local Hyper-V Administrators group
  - "Members of this group have complete and unrestricted access to all features of Hyper-V."
  - As long as you're a member of this group, you don't have to be a local Administrator
  - **HOWEVER** other operations you may wish to perform might negate the previous point



# Example Uses: Azure

### Working with Azure

- Use Web Platform Installer to Install and Configure
  - <https://azure.microsoft.com/en-us/documentation/articles/powershell-install-configure/>
- Essential way to automate Azure
  - Sometimes only way to access a feature
- Lots of cmdlets
  - PowerShell command **Get-Command -Module Azure** will give you the current list
  - <https://msdn.microsoft.com/en-us/library/azure/jj554330.aspx>
- Two “versions” of PowerShell today
  - Classic mode
    - Traditional imperative model
  - ARM mode (Azure Resource Manager)
    - Declarative model (similar to DSC)



### Code Samples

- I'll be posting my code up on GitHub
  - <http://bri.gd/briangh-powershell>



thank you  
owners  
or developers

contact me

[brian a. randell](#)

partner, mcw technologies

[brianr@mcwtech.com](mailto:brianr@mcwtech.com)

[@brianrandell](#)

[blog.brianrandell.com](http://blog.brianrandell.com)