Powershell for Beginners

The questions and answers were developed straight from the Microsoft Technet pages and I have included links for each topic.

The purpose of this crossword was to hopefully introduce some reluctant Administrators to Powershell. Powershell will be a mandatory requirement in the very near future, if it is not already in your area.

If you find that Powershell can (will) make your life much easier, then I would highly suggest these links to open your mind and your world to Powershell.

The Lonely Administrator - Jeffery Hicks
I am certified Microsoft professional, a Microsoft MVP and an IT veteran with almost 20 years of experience, much of it spent as an IT consultant specializing in Windows server technologies.
http://jdhitsolutions.com/blog/

Concentrated Technology is the brainchild of IT gurus Don Jones and Greg Shields. We offer a number of different products and services, depending on which part of the IT industry you work in. http://www.concentratedtech.com/

Claus Nielsen
http://xipher.dk

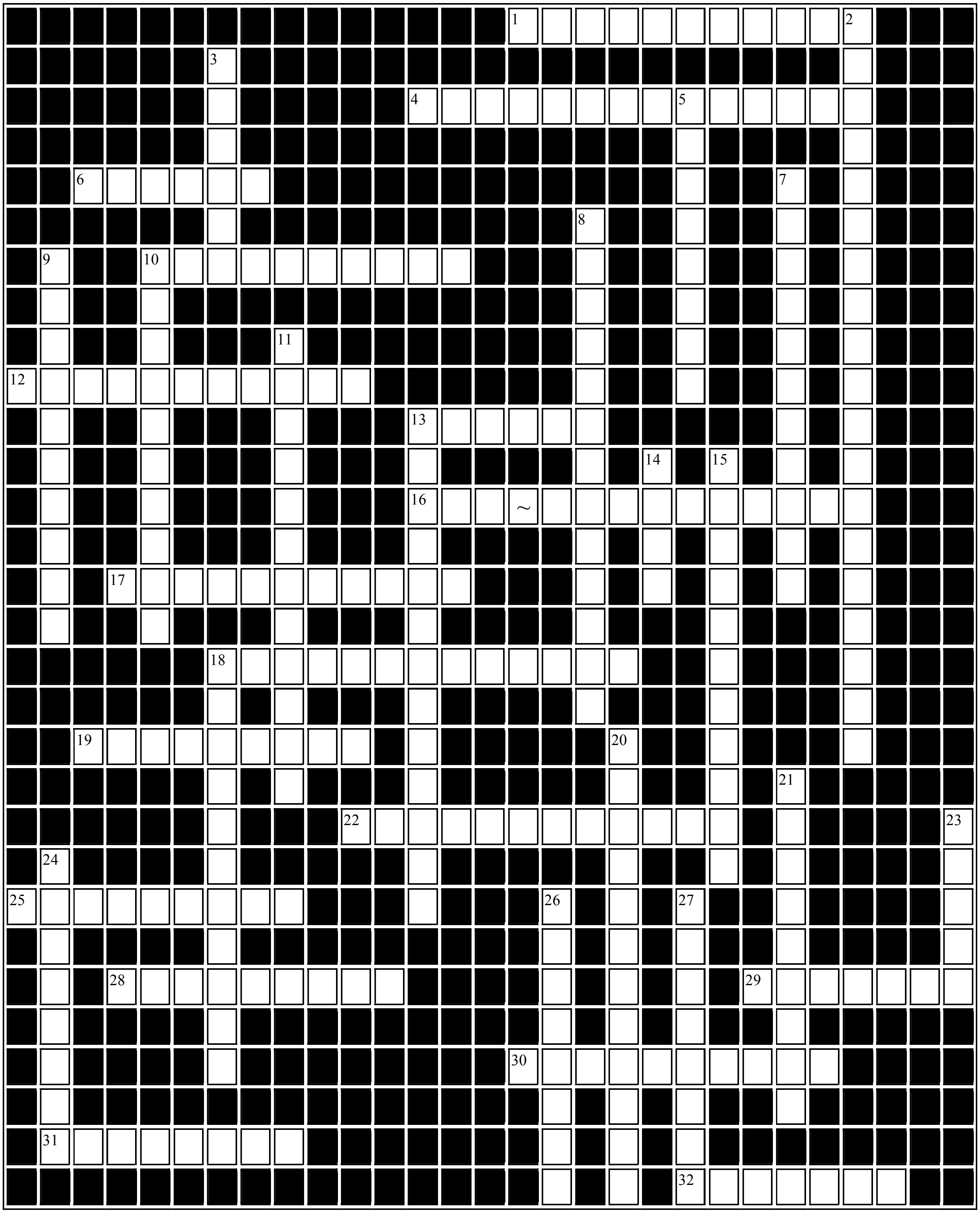
vSphere PowerCLI
The best automation tool for VMware vSphere
Intertube Serial #913714318 · http://www.vmware.com/go/powercli

Powershell - Bruce Payette ql Function
http://learningpcs.blogspot.com/2011/01/powershell-bruce-payette-ql-function.html

Dmitry Sotnikov My background includes various positions in Software Development, Technical Communications, Technical Support (was setting up 24×7 phone support in Aelita Software – that was a fun job!), Product Management, and now New Product Research.
http://dmitrysotnikov.wordpress.com/about/

Powershell For Beginners

Tim Bolton



Across

1. In Windows PowerShell, there are two types of errors: BLANK and Non-terminating.
4. BLANK-BLANK cmdlet uses script blocks and the $_ descriptor for the current pipeline object to let you run a command on each object in the pipeline.
6. A BLANK is a package of commands and other items that you can use in Windows PowerShell.
10. Using vSphere PowerCLI to list all the vSphere Servers (VMHost) on the connected vSphere Server and some of their properties BLANK
12. We can organize displayed data to make it easier to scan by using the BLANK-BLANK cmdlet.
13. Using vSphere PowerCLI to list all the snapshots for all virtual machines BLANK | Get-Snapshot
16. BLANK BLANK provides automatic Help when you are typing in the Script Pane or in the Command Pane.
17. You can get the services on a local or remote computer by using the BLANK-BLANK cmdlet.
18. BLANK-BLANK is the most important cmdlet for general system management tasks.
19. BLANK are a type of command in Windows PowerShell. The best thing about BLANK is that they are really easy to write.
22. To determine the path of your current directory location, enter the BLANK-BLANK command.
25. Cmdlets Use BLANK-BLANK Names to Reduce Command Memorization.
28. BLANK act like a series of connected segments of pipe. Items moving along the pipeline pass through each segment.
29. Although you might not realize it at first, when you work in Windows PowerShell, you are working with .NET Framework BLANK.
30. The default execution policy, BLANK, is the most secure of the execution policies.
31. vSphere BLANK is a set of snapins based on Windows PowerShell that provide administration and automation for VMware vSphere.
32. A Windows PowerShell BLANK is a dynamic link library (.dll) that implements cmdlets and providers.

Down

2. To find the execution policy on your system, type: BLANK-BLANK
3. A BLANK is a single-feature command that manipulates objects in Windows PowerShell.
5. BLANK-BLANK formats file contents to look like console output. This causes the output to be truncated just as it is in a console window in most circumstances.
7. To see a list of Windows PowerShell drives, type: BLANK-BLANK
8. BLANK-BLANK, is one of the most useful for doing real work via WMI.
9. BLANK-BLANK converts objects into a series of comma-separated (CSV) strings and saves the strings in a CSV file.
10. One of the most useful cmdlets is BLANK-BLANK, which displays information about the .NET Framework object.
11. BLANK-BLANK gets the events in an event log, or a list of the event logs, on the local or remote computers.
13. The BLANK-BLANK cmdlet is designed specifically to return all items found within a container such as a folder.
14. In addition to its interactive interface, Windows PowerShell fully supports scripting. In Windows PowerShell, script files have a BLANK file name extension.
15. To get a list of available Cmdlets in your Powershell console key in BLANK-BLANK | MORE This will let us step through and view the pages full of Content.
18. To get the processes running on the local computer, run a BLANK-BLANK with no parameters.
20. The BLANK-BLANK command allows you to specify your current directory location.
21. PowerShell lets you create BLANK - essentially named objects - to preserve output to use later.
23. Windows PowerShell supports the concept of an BLANK, that is, an alternate name for a command.
24. To get Help about Windows PowerShell cmdlets, use the BLANK-BLANK cmdlet.
26. BLANK contains the enumerator of a foreach-object loop.
27. You can create one or more Windows PowerShell BLANK for Windows PowerShell ISE and use them to add the variables, aliases, commands and functions that you use frequently.

ACROSS

1) Errors will occur occasionally when you use the Windows PowerShell, such as when you are trying to set your location to a nonexistent directory or trying to remove a file without the required privileges.   In Windows PowerShell, there are two types of errors: **Terminating** errors: Errors that halt the execution of the command. & **Non-terminating** errors: Errors that do not halt the execution of the command. http://technet.microsoft.com/en-us/library/bb648602(VS.85).aspx

4) The **ForEach-Object** cmdlet uses script blocks and the $_ descriptor for the current pipeline object to let you run a command on each object in the pipeline. This can be used to perform some complicated tasks. http://technet.microsoft.com/en-us/library/dd347717.aspx

6) A **module** is a package of commands and other items that you can use in Windows PowerShell. After you run the setup program or save the module to disk, you can import the module into your Windows PowerShell session and use the commands and items. You can also use modules to organize the cmdlets, providers, functions, aliases, and other commands that you create, and share them with others. http://technet.microsoft.com/en-us/library/dd745031(VS.85).aspx

10) Using vSphere PowerCLI to list all the vSphere Servers (VMHost) on the connected vSphere Server andsome of their properties **Get-VMHost** http://communities.vmware.com/docs/DOC-13700

12) We can organize displayed data to make it easier to scan by using the **Sort-Object** cmdlet. http://technet.microsoft.com/en-us/library/dd347718.aspx

13) Using vSphere PowerCLI to list all the snapshots for all virtual machines **Get-VM** | Get-Snapshot http://communities.vmware.com/docs/DOC-13700

16) **Tab completion** provides automatic Help when you are typing in the Script Pane or in the Command Pane. http://technet.microsoft.com/en-us/library/dd819457.aspx

17) You can get the services on a local or remote computer by using the **Get-Service** cmdlet. http://technet.microsoft.com/en-us/library/dd315239.aspx

18) **Get-WmiObject** is the most important cmdlet for general system management tasks. All critical subsystem settings are exposed through WMI. Furthermore, WMI treats data asobjects that are in collections of one or more items. http://technet.microsoft.com/en-us/library/dd315240.aspx

19) **Functions** are a type of command in Windows PowerShell. The best thing about **functions** is that they are really easy to write. Unlike cmdlets, which are written in C#, **functions** are just a named grouping of Windows PowerShell commands and expressions. If you can type commands in Windows PowerShell, you can write **functions**. http://technet.microsoft.com/en-us/library/dd745030(VS.85).aspx

22) To determine the path of your current directory location, enter the **Get-Location** command http://technet.microsoft.com/en-us/library/dd315262.aspx

25) Cmdlets Use **Verb-Noun** Names to Reduce Command Memorization. http://technet.microsoft.com/en-us/library/dd315315.aspx

28) **Pipelines** act like a series of connected segments of pipe. Items moving along the **pipeline** pass through each segment. To create a pipeline in Windows PowerShell, you connect commands together with the pipe operator "|". The output of each command is used as input to the next command. http://technet.microsoft.com/en-us/library/dd347655.aspx

29) Although you might not realize it at first, when you work in Windows PowerShell, you are working with .NET Framework **objects**. As you gain experience, the power of object processing becomes more evident, and you'll find yourself using the **objects** and even thinking in objects. http://technet.microsoft.com/en-us/library/bb613487(VS.85).aspx

30) The default execution policy, **Restricted**, is the most secure of the execution policies. It does not permit any scripts to run, and it does not permit any configuration files, including a Windows PowerShell profile, to be loaded. You can still use Windows PowerShell interactively. http://technet.microsoft.com/en-us/library/bb648601(VS.85).aspx

31) vSphere **PowerCLI** is a set of snapins based on Windows PowerShell that provide administration and automation for VMware vSphere. vSphere PowerCLI ships with over 200 commandlets (pre-built commands) to help administrators manage vSphere. http://communities.vmware.com/community/vmtn/server/vsphere/automationtools/powercli

32) A Windows PowerShell **snap-in** (PSSnapin) is a dynamic link library (.dll) that implements cmdlets and providers. When you receive a snap-in, you need to install it, and then you can add the cmdlets and providers in the snap-in to your Windows PowerShell session. http://technet.microsoft.com/en-us/library/dd745031(VS.85).aspx

DOWN

2) To find the execution policy on your system, type: **get-executionpolicy**
http://technet.microsoft.com/en-us/library/bb648601(VS.85).aspx

3) A **Cmdlet** is a single-feature command that manipulates objects in Windows PowerShell.
http://technet.microsoft.com/en-us/library/bb648597(VS.85).aspx

5) **Out-file** formats file contents to look like console output. This causes the output to be truncated just as it is in a console window in most circumstances.
http://technet.microsoft.com/en-us/library/dd347585.aspx

7) Windows PowerShell comes with several useful drives which are supported by Windows PowerShell providers. To see a list of Windows PowerShell drives, type: **get-psdrive**
http://technet.microsoft.com/en-us/library/ms714417(VS.85).aspx

8) Windows Management Instrumentation (WMI) is a core technology for Windows system administration because it exposes a wide range of information in a uniform manner. Because of how much WMI makes possible, the Windows PowerShell cmdlet for accessing WMI objects, **Get-WmiObject**, is one of the most useful for doing real work.
http://technet.microsoft.com/en-us/library/dd315379.aspx

9) **Export-CSV** Converts objects into a series of comma-separated (CSV) strings and saves the strings in a CSV file. http://technet.microsoft.com/en-us/library/dd347724.aspx

10) One of the most useful cmdlets is **Get-Member**, which displays information about the .NET Framework object that a command returns. The information includes the type, properties, and methods of the object. http://technet.microsoft.com/en-us/library/bb613480(VS.85).aspx

11) **Get-EventLog** Gets the events in an event log, or a list of the event logs, on the local or remote computers. http://technet.microsoft.com/en-us/library/dd315250.aspx

13) The **Get-ChildItem** cmdlet is designed specifically to return all items found within a container such as a folder. http://technet.microsoft.com/en-us/library/dd315381.aspx

14) In addition to its interactive interface, Windows PowerShell fully supports scripting. In Windows PowerShell, script files have a **.ps1** file name extension.
http://technet.microsoft.com/en-us/library/bb613481(VS.85).aspx

15) To get a list of available Cmdlets in your Powershell console key in **Get- COMMAND** | MORE This will let us step through and view the pages full of Content.
http://technet.microsoft.com/en-us/library/dd347696.aspx

18) To get the processes running on the local computer, run a **Get-Process** with no parameters.
http://technet.microsoft.com/en-us/library/dd347650.aspx

20) The **Set-Location** command allows you to specify your current directory location. http://technet.microsoft.com/en-us/library/ee176962.aspx

21) Windows PowerShell works with objects. Windows PowerShell lets you create **variables** - essentially named objects - to preserve output to use later. http://technet.microsoft.com/en-us/library/dd315305.aspx

23) Cmdlet names can be cumbersome to type. To minimize typing, and to make it easier for users accustomed to other shells to use Windows PowerShell, Windows PowerShell supports the concept of an **alias**, that is, an alternate name for a command. You can create an **alias** for a cmdlet name, function name, or the name of an executable file, and then type the alias instead of the name in any command. http://technet.microsoft.com/en-us/library/bb648603(VS.85).aspx

24) To get Help about Windows PowerShell cmdlets, use the **Get-Help** cmdlet. http://technet.microsoft.com/en-us/library/dd347689.aspx

26) **$ForEach** Contains the enumerator of a foreach-object loop. http://technet.microsoft.com/en-us/library/dd347675.aspx

27) You can create one or more Windows PowerShell **profiles** for Windows PowerShell ISE and use them to add the variables, aliases, commands and functions that you use frequently. A **profile** affects every Windows PowerShell ISE session that you start. You can also use a **profile** to preserve the items that you customize in the Windows PowerShell ISE environment, such as color and font preferences. http://technet.microsoft.com/en-us/library/dd819434.aspx