1

Labs

Windows PowerShell 3

Day 1

Chapter 3

Lab

Using the Help System

Lab Time: 30

1. First, run Update-Help and ensure it completes without errors. That will get a copy of the help on your local computer. This requires an Internet connection, and requires that the shell be running under elevated privileges (which means it must say "Administrator" in the shell's window title bar).

2. Can you find any cmdlets capable of converting other cmdlets' output into HTML?

3. Are there any cmdlets that can redirect output into a file, or to a printer?

4. How many cmdlets are available for working with processes? (Hint: remember that cmdlets all use a singular noun.)

5. What cmdlet might you use to write to an event log?

6. You've learned that aliases are nicknames for cmdlets; what cmdlets are available to create, modify, export, or import aliases?

7. Is there a way to keep a transcript of everything you type in the shell, and save that transcript to a text file?

8. It can take a long time to retrieve all of the entries from the Security event log. How can you get just the 100 most recent entries?

9. Is there a way to retrieve a list of the services that are installed on a remote computer?

10. Is there a way to see what processes are running on a remote computer?

11. Examine the help file for the Out-File cmdlet. The files created by this cmdlet default to a width of how many characters? Is there a parameter that would enable you to change that width?

12. By default, Out-File will overwrite any existing file that has the same filename as what you specify. Is there a parameter that would prevent the cmdlet from overwriting an existing file?

13. How could you see a list of all aliases defined in PowerShell?

14. Using both an alias and abbreviated parameter names, what is the shortest command line you could type to retrieve a list of running processes from a computer named Server1?

15. How many cmdlets are available that can deal with generic objects? (Hint: remember to use a singular noun like "object" rather than a plural one like "objects").

16. This chapter briefly mentioned arrays. What help topic could tell you more about them?

17. The Help command can also search the contents of a help file. Are there any topics that might explain any breaking changes between PowerShell v1 and PowerShell v2?

Chapter

4

Lab

Running Commands

Lab Time: 20

Using just what you learned in this chapter, and in the previous chapter on using the help system, complete the following tasks in Windows PowerShell:

1. Display a list of running processes.

2. Display the 100 most recent entries from the Application event log (don't use Get-WinEvent for this – We've shown you another command that will do this task).

3. Display a list of all commands that are of the "cmdlet" type (this is tricky – we've shown you Get-Command, but you're going to have to read the help to find out how to narrow down the list as we've asked).

4. Display a list of all aliases.

5. Make a new alias, so that you can run "d" to get a directory listing.

6. Display a list of services that begin with the letter "M." Again, read the help for the necessary command – and don't forget that "*" is a near-universal wildcard in PowerShell.

7. Display a list of all Windows Firewall rules. You'll need to use Help or Get-Command to discover the necessary cmdlet!

8. Display a list only of inbound Windows Firewall rules. Same cmdlet as above, but you'll need to read its help to discover the necessary parameter and its allowable values

Chapter 5

Lab

Working with Providers

Lab Time: 15

Complete the following tasks:

1. In the registry, go to HKEY_CURRENT_USER\software\microsoft\Windows\currentversion\explorer. Locate the Advanced key, and set its DontPrettyPath property to 1.

2. Create a zero-length file named C:\Test.txt (use New-Item).

3. Is it possible to use Set-Item to change the contents of C:\Test.txt to TESTING? Or do you get an error? If you get an error, why?

4. What are the differences between the –Filter, -Include, and –Exclude parameters of Get-ChildItem?

Chapter 6

Lab

The Pipeline: Connecting Commands

Lab Time: 30

1. Create two similar, but different, text files. Try comparing them using Diff. To do so, run something like this: Diff -reference (Get-Content File1.txt) -difference (Get-Content File2.txt). If the files have only one line of text that's different, the command should work.

2. What happens if you run Get-Service | Export-CSV services.csv | Out-File from the console? Why does that happen?

3. Apart from getting one or more services and piping them to Stop-Service, what other means does Stop-Service provide for you to specify the service or services you want to stop? Is it possible to stop a service without using Get-Service at all?

4. What if you wanted to create a pipe-delimited file instead of a comma-separated file? You would still use the Export-CSV command, but what parameters would you specify?

5. Is there a way to eliminate the # comment line from the top of an exported CSV file? That line normally contains type information, but what if you wanted to omit that from a particular file?

6. Export-CliXML and Export-CSV both modify the system, because they can create and overwrite files. What parameter would prevent them from overwriting an existing file? What parameter would ask you if you were sure before proceeding to write the output file?

7. Windows maintains several regional settings, which include a default list separator. On U.S. systems, that separator is a comma. How can you tell Export-CSV to use the system's default separator, rather than a comma?

Chapter 7

Lab

Adding Commands

Lab Time: 10

For this lab, you only have one task: run the Networking troubleshooting pack. When you successfully do so, you'll be asked for an "Instance ID;" just hit Enter. Then run a Web Connectivity check and ask for help connecting to a specific Web page. Use http://videotraining.interfacett.com as your test URL. Hopefully, you'll get a "No problems were detected" report, meaning you ran the check successfully.

To accomplish this task, you'll need to discover a command capable of getting a troubleshooting pack, and one capable of executing a troubleshooting pack. You'll also need to discover where the packs are located and how they're named. Everything you need to know is in PowerShell and the help system will find it for you.

That's all the help you get!

Chapter 8

Lab

Objects: Just Data by Another Name

Lab Time: 30

This chapter has probably covered more, and more difficult, new concepts than any chapter so far. Hopefully we were able to make it all make sense, but these exercises should help you cement everything. See if you can complete them all, and remember that there are companion videos and sample solutions at MoreLunches.com. Some of these tasks will draw on skills you learned in previous chapters, as a way of refreshing your memory and keeping you sharp.

1. Identify a cmdlet that will produce a random number.

2. Identify a cmdlet that will display the current date and time.

3. What type of object does the cmdlet from task #2 produce? (What is the type name of the object produced by the cmdlet?)

4. Using the cmdlet from task #2 and Select-Object, display only the current day of the week in a table like this (caution: The output will right-align, so make sure your PowerShell window doesn't have a horizontal scroll bar):

5. Identify a cmdlet that will display information about installed hotfixes.

6. Using the cmdlet from task #5, display a list of installed hotfixes. Sort the list by the installation date, and display only the installation date, the user who installed the hotfix, and the hotfix ID.

7. Repeat task #6, but this time sort the results by the hotfix description, and include the description, the hotfix ID, and the installation date. Put the results into an HTML file.

8. Display a list of the 50 newest entries from the Security event log (you can use a different log, such as System or Application if your Security log is empty). Sort the list so that the oldest entries appear first and so that entries made at the same time are sorted by their index. Display the index, time, and source for each entry. Put this information into a text file (not an HTML file, just a plain text file). You may be tempted to use Select-Object and its –first or –last parameters to achieve this; don't. There's a better way. Also, avoid using Get-WinEvent for now – there's a better cmdlet to work with for this particular task.

Appendix

A

Lab

Review Labs

Review Lab 1 (Based on Chapters 1-6)

Task 1

Run a command that will display the newest 100 entries from the Application event log. Note: Do not use Get-WinEvent.

Task 2

Write a command line that displays only the 5 top processes based on virtual memory (VM) usage.

Task 3

Create a CSV file that contains all services, including only the service names and status. Have Running services listed BEFORE Stopped services.

Task 4

Write a command line that changes the startup type of the BITS service to Manual.

Task 5

Display a list of all files named win*.* on your computer. Start in the C:\ folder. Note: You may need to experiment and use some new parameters of a cmdlet in order to complete this task.

Task 6

Get a directory listing for C:\Program Files. Include all subfolders, and have the directory listing go into a text file named C:\Dir.txt (remember to use the > redirector, or the Out-File cmdlet).

Task 7

Get a list of the most recent 20 entries from the Security event log, and convert the information to XML. Do not create a file on disk: Have the XML display in the console window.

Note that the XML may display as a single top-level object, rather than as raw XML data – that's fine. That's just how PowerShell displays XML. You can pipe the XML object to Format-Custom to see it expanded out into an object hierarchy, if you like.

Task 8

Get a list of services, and export the data to a CSV file named C:\services.csv.

Task 9

Get a list of services. Keep only the services' names, display names, and status, and send that information to an HTML file. Have the phrase "Installed services" displayed in the HTML file before the table of service information.

Task 10

Create a new alias named D, which runs Get-ChildItem. Export just that alias to a file. Now, close the shell and open a new console window. Import that alias into the shell. Make sure you can run D and get a directory listing.

Task 11

Display a list of event logs that are available on your system.

Task 12

Run a command that will display the current directory that the shell is in.

Task 13

Run a command that will display the most recent commands that you have run in the shell. Locate the command that you ran for Task 11. Using two commands connected by a pipeline, re-run the command from Task 11.

In other words, if "Get-Something" is the command that retrieves historical commands, if "5" is the ID number of the command from Task 11, and "Do-Something" is the command that runs historical commands, run this:

Get-Something –id 5 | Do-Something

Of course, those aren't the correct cmdlet names – you'll need to find those. Hint: Both commands that you need have the same noun.

Task 14

Run a command that modifies the Security event log to overwrite old events as needed.

Task 15

Use the New-Item cmdlet to make a new directory named C:\Review. This is not the same as running Mkdir; the New-Item cmdlet will need to know what kind of new item you want to create. Read the help for the cmdlet.

Task 16

Display the contents of this registry key:

HKCU:\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders

Note: "User Shell Folders" is not exactly like a directory. If you CD into it, you won't see anything in a directory listing. User Shell Folders is an item, and what it contains are item properties. There's a cmdlet capable of displaying item properties (although cmdlets use singular nouns, not plural).

Task 17

Find (but please do not run) cmdlets that can…

- Restart a computer
- Shut down a computer
- Remove a computer from a workgroup or domain
- Restore a computer's System Restore checkpoint

Task 18

What command do you think could change a registry value? Hint: It's the same noun as the cmdlet you found for Task 16.