Getting Started with Powershell 3.0 Jumpstart

Module 1: Don’t fear the shell

* Created for automation
* Can be used to solve problems that can’t be solved in gui
* Make sure to run as Administrator
* Use an easily readable font
* Cmdlets are Verb-Noun
* Can also use unix commands or command line commands

Module 2: The help system

* Learning how to figure things out is more important than memorizing commands
* Update the help system with update-help cmdlet
* Get-help, help, and man can all be used to view help system
* -showwindow opens help page in new window]
* Right click to copy and paste in powershell

Module 3: The Pipeline: Getting Connected & Module 4: Extending the Shell

* | (pipe) can be used to take output from one command and use it as input for another command
* Use ‘| out-file’ to save output of a command to a file
* -whatif shows what a command will do without actually doing anything
* -confirm will show a prompt for you to confirm that you want to run the command before running it
* Powershell will dynamically load modules for you

Module 4: Objects for the Admin

* An object has properties (things it has) and methods (things it can do)
* To learn more about an object, pipe to get member ( | gm )
* Learn about ‘where’ command
* Powershell is mostly case insensitive

Module 5: The Pipeline: Deeper

* Use get-help on a command to see which parameters accept pipeline input
* Pipe a command to gm (get member) to see what kind of object will be sent down the pipeline
* ByPropertyName: Object being sent must have a property that is exactly the same as the parameter name
* When you use Select -Property on a property that doesn’t exist it will create it
* To create a property called ComputerName that has the same data as an existing property called Name use the hashtable @{name=’ComputerName’ ;expression={$\_.name}}
* I a parameter supports pipeline input but the property being sent has the wrong name you can change the name in order to be able to use the parameter
* Wmi is useful for grabbing information

Module 6: The Power in the Shell - Remoting

* Universal Code Execution
* Remoting allows you to execute commands securely on other machines on your domain
* Enable remoting in group policy
* Use remoting with the cmdlet Enter-PSSession
* Can use remoting to execute a command on multiple machines at once
* Powershell Web Access allows you to access powershell and execute commands on a machine remotely from a web browser

Module 7: Getting Prepared for Automation

* By default, Powershell does not run scripts
* Scripts can be signed and you can decide whose scripts you will allow to run based on signature
* Powershell will detect script code in signature block and prevent it from running
* Remotesigned execution policy will only require a signature to run scripts downloaded from the internet
* Allsigned execution policy will require a signature to run any script
* Variable syntax: $VarName=Variable contents
* Variables can contain commands and files

Module 8: Automation in Scale: Remoting

* Use sessions to keep powershell running
* You can use powershell to quickly deploy web servers and websites
* Use powershell to install things to multiple servers remotely at the same time
* Implicit Remoting

Module 9: Scripting and Toolmaking

* Powershell ISE console has completion suggestions and color syntax
* Use ctrl-space to get suggestions for completing wmi namespaces
* Use ISE to write and save Powershell scripts
* Use param() in a script to allow for input for a certain variable when running the script
* Add [CmdletBinding()] to script to enable parameter attributes