**PowerShell Cheat Sheet**

**Clear Screen**

> cls

**Command Help**

> findstr -?

**Find Commands**

List all commands…

> gcm

for Get-Command.

List commands with a specific noun or verb

> gcm -noun process

> gcm -verb disable

**Find Alias**

> alias | findstr “Process”

**Tab Expansion**

Type tab or shift-tab to cycle through options.

> cd ..\Folder

**Multi-Line Entry**

Enter a blank line to finish.

> “Hello

>> world.”

>>

Hello

World.

**Aliases**

get-alias

**Pipe**

Search for a string on a lines piped from other commands:

dir | findstr “.txt”

**Output to a File**

{stuff} > “output.txt”

**Read from a File**

gc output.txt

Get-Content output.txt

**Filter (grep functionality)**

Use findstr (tutorial [here](http://www.mkyong.com/linux/grep-for-windows-findstr-example/)).

Ignore case: dir | findstr –i “program”

Look for multiple strings: dir | findstr –i “prog intel”

Look in a file: findstr result “RippleEffect Info Log.txt”

Or use ?, the alias for Where-Object…

dir | ? {$\_ -like “\*.txt”}

**Pretty Print Tables**

Pipe to Format-List or fl

> dir | fl

**View Environment Variables**

> gci env:

gci and ls are aliases for Get-ChildItem

You can also switch to the environment as if it’s a drive.

> cd env:

> dirls

> ls path

**View PATH**

> (ls env:path).value.split(“;”)

**Set an Environment Variables**

You have to use the .Net framework to do this, since PowerShell doesn’t have an appropriate command.

> [Environment]::SetEnvironmentVariable("Test", "Test value.", "User")

It won’t show up in listings until you restart PowerShell, but you can check it like this.

> [Environment]::GetEnvironmentVariable("Test", "User")

Instead of "User", you can also do “Process” or “Machine”.

**Variables**

> $x = 5

> $x

5

More elaborately…

> dir | set x

Then…

> Get-Variable x

Or...

> (Get-Variable x).value

**Check for Null**

$args[0] -eq $null

**List All Variables**

> ls variable:\*

**Delete a Variable**

Use Remove-Variable or its alias, rv.

> rv x

**Lists**

> $names = “Tom”, “Dick”, “Harry”

> $names[1]

Dick

> $names -contains “Harry”

True

**String Formatting**

> “Hello $x.”

Hello 5.

(echo is unnecessary.)

The escape character is ` (in the upper left of the keyboard).

> “`$x”

$x

**See History**

> history

**If, Then, Else**

> If (10 -gt 15) {“Yes”} Else {“No”}

Lower case is fine. There's also an elseif.

**Do While**

> do {“x is $x”; $x = $x+1} while {$x -le 5}

**For Each**

> foreach ($n in $names) {echo “Name: $n”}

**Run a Script**

In the current directory, prefix with ./

> ./Hello (for a .ps1 file)

**Script Arguments**

*PrintArg.ps1:*

$x = $args[0]

$y = $args[1]

"First is $x, second is $y."

> ./PrintArg 5 9

First is 5, second is 9.

**Match Text**

> “Mark Frymire 1975” -Match "Fry"

True

With Like, you specify a regex, so this fails…

> “Mark Frymire 1975” -Like "Fry"

False

…but this works…

> “Mark Frymire 1975” -Like "\*Fry\*"

False

**Replace Text**

> “Hello world”.replace(“Hello”, “Hi”)

Hi world.

**List Processes**

> ps

Or gps or Get-Process.

Get a specific process like this…

> ps chrome

Or multiple processes

> ps chrome,explorer,\*word\*

**See Member Variables**

Pipe it to Get-Member or gm

> Get-Process | gm

**Sort**

> ps | sort id

**Get Process Information**

Get a specific process like this…

> ps chrome | select id,name,path,productversion,starttime

**Kill a Process**

Stop-Process or kill

> kill 3526

or

> kill -processname notepad