

# PowerShell Cheat Sheet

## Essential Commands

To get help on any cmdlet use get-help  
Get-Help Get-Service  
To get all available cmdlets use get-command  
Get-Command  
To get all properties and methods for an object use get-member  
Get-Service | Get-Member

## Setting Security Policy

View and change execution policy with Get-Execution and Set-Execution policy  
Get-Executionpolicy  
Set-Executionpolicy remotesigned

## To Execute Script

```
powershell.exe -noexit &"c:\myscript.ps1"
```

## Variables

Must start with \$  
\$a = 32  
Can be typed  
[int]\$a = 32

## Arrays

To initialise  
\$a = 1,2,4,8  
To query  
\$b = \$a[3]

## Functions

Parameters separate by space. Return is optional.  
function sum ([int]\$a,[int]\$b)  
{  
 return \$a + \$b  
}  
sum 4 5

## Constants

Created without \$  
Set-Variable -name b -value 3.142 -option constant  
Referenced with \$  
\$b

## Creating Objects

To create an instance of a com object  
New-Object -comobject <ProgID>  
\$a = New-Object -comobject "wscript.network"  
\$a.username

To create an instance of a .Net Framework object. Parameters can be passed if required  
New-Object -type <.Net Object>  
\$d = New-Object -Type System.DateTime 2006,12,25  
\$d.get\_DayOfWeek()

## Writing to Console

Variable Name  
\$a  
or  
Write-Host \$a -foregroundcolor "green"

## Capture User Input

Use Read-Host to get user input  
\$a = Read-Host "Enter your name"  
Write-Host "Hello" \$a

## Passing Command Line Arguments

Passed to script with spaces  
myscript.ps1 server1 benp  
Accessed in script by \$args array  
\$servername = \$args[0]  
\$username = \$args[1]

## Miscellaneous

Line Break `  
Get-Process | Select-Object `name, ID  
Comments #  
# code here not executed  
Merging lines ;  
\$a=1;\$b=3;\$c=9  
Pipe the output to another command |  
Get-Service | Get-Member

## Do While Loop

Can repeat a set of commands while a condition is met

```
$a=1  
Do {$a; $a++}  
While ($a -lt 10)
```

## Do Until Loop

Can repeat a set of commands until a condition is met

```
$a=1  
Do {$a; $a++}  
Until ($a -gt 10)
```

## For Loop

Repeat the same steps a specific number of times

```
For ($a=1; $a -le 10; $a++)  
{ $a }
```

## ForEach - Loop Through Collection of Objects

Loop through a collection of objects

```
Foreach ($i in Get-Childitem c:\windows)  
{ $i.name; $i.creationtime }
```

## If Statement

Run a specific set of code given specific conditions

```
$a = "white"  
if ($a -eq "red")  
    {"The colour is red"}  
elseif ($a -eq "white")  
    {"The colour is white"}  
else  
    {"Another colour"}
```

## Switch Statement

Another method to run a specific set of code given specific conditions

```
$a = "red"  
switch ($a)  
{  
    "red" {"The colour is red"}  
    "white" {"The colour is white"}  
    default {"Another colour"}  
}
```

## Reading From a File

Use Get-Content to create an array of lines. Then loop through array

```
$a = Get-Content "c:\servers.txt"  
foreach ($i in $a)  
{ $i }
```

## Writing to a Simple File

Use Out-File or > for a simple text file

```
$a = "Hello world"  
$a | out-file test.txt  
Or use > to output script results to file  
. \test.ps1 > test.txt
```

## Writing to an Html File

Use ConvertTo-Html and >

```
$a = Get-Process  
$a | Convertto-Html -property Name,Path,Company > test.htm
```

## Writing to a CSV File

Use Export-Csv and Select-Object to filter output

```
$a = Get-Process  
$a | Select-Object Name,Path,Company | Export-Csv -path test.csv
```