# Windows PowerShell Quick Reference

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Getting Started To create a variable, use the $ sign.  $var1 = “Hello world”  To write text to the console, use the Write-Host.cmdlet.  $var1 = “Hello world”  Write-Host $var1  To add a comment, use the # sign.  # display message to user  $var1 = “Hello world”  Write-Host $var1 Working with Arrays To create an array, use []  $beatles = [ “john”, “paul”, “george”, “ringo” ]  To access an element, use [] notation  $beatles = [ “john”, “paul”, “george”, “ringo” ]  $drummer = $beatle[3];  To enumerate through an array, use For-Each  $beatles = [ “john”, “paul”, “george”, “ringo” ]  For-Each Accessing Command-line Arguments To access command-line arguments, use the built-in array variable named $args.  $x = $args[0] # get value of first argument  $y = $args[1] # get value of second argument  If no arguments are passed, $args has a value of null.  You can reference the last item in a collection by using the index number –1:  $args[-1] HHow to use Colored TextHow to Insert a Paragraph Return How to Insert Comments  How to insert a pa | Load the SharePoint PowerShell ProviderWriting If Statements Each of these operators can be made case sensitive by adding a **c** immediately after the hyphen. For example, **-ceq** represents the case-sensitive equals operator; **-clt** is the case-sensitive less than operator.   |  |  | | --- | --- | | **-lt** | Less than | | **-le** | Less than or equal to | | **-gt** | Greater than | | **-ge** | Greater than or equal to | | **-eq** | Equal to | | **-ne** | Not equal to | | **-like** | Like (uses wildcards for matching) | | **-notlike** | Not like (uses wildcards for matching) | |