Using Common Windows PowerShell Operators

Jeff Hicks
Author | Teacher

https://jdhitsolutions.github.io

Windows PowerShell Operators

PowerShell is about *doing*

PowerShell operators are critical parts of the syntax

You can use them interactively at the console

More likely to use in PowerShell scripting

Use Windows PowerShell Help

I am covering common operators

Other operators covered later in the course

Read the help documentation

- Help about_*operators

Learning PowerShell operators is practically self-explanatory

Basic demonstrations are very simple

Operators always start with a dash

You need to try them

Comparison Operators

Compare values

Result is True or False

Windows PowerShell is not case-sensitive*

```
PS C:\> 5 -gt 2
True
PS C:\> 2 -ge 2
True
PS C:\> 5 -lt 2
False
PS C:\> 2 -le 2
True
```

◀ Greater than

◀ Greater than or equal to

◀ Less than

◀ Less than or equal to

```
PS C:\> $i = 7
PS C:\> $i -eq 7
True
PS C:\> $i -ne 7
False
PS C:\> "jeff" -eq "JEFF"
True
PS C:\> "jeff" -ceq "JEFF"
False
```

- This is the assignment operator
- Is $i equal to 7

- Is $i *not* equal to 7?
- String comparisons are not case-sensitive

- But they can be

```
PS C:\> "foo" -like "f*"
True
PS C:\> "bar" -notlike "B*"
False
PS C:\> "bar" -clike "B*"
False
```

- Use wild card comparisons with -Like
- Comparisons are not case-sensitive
- But they can be

  -cnotlike

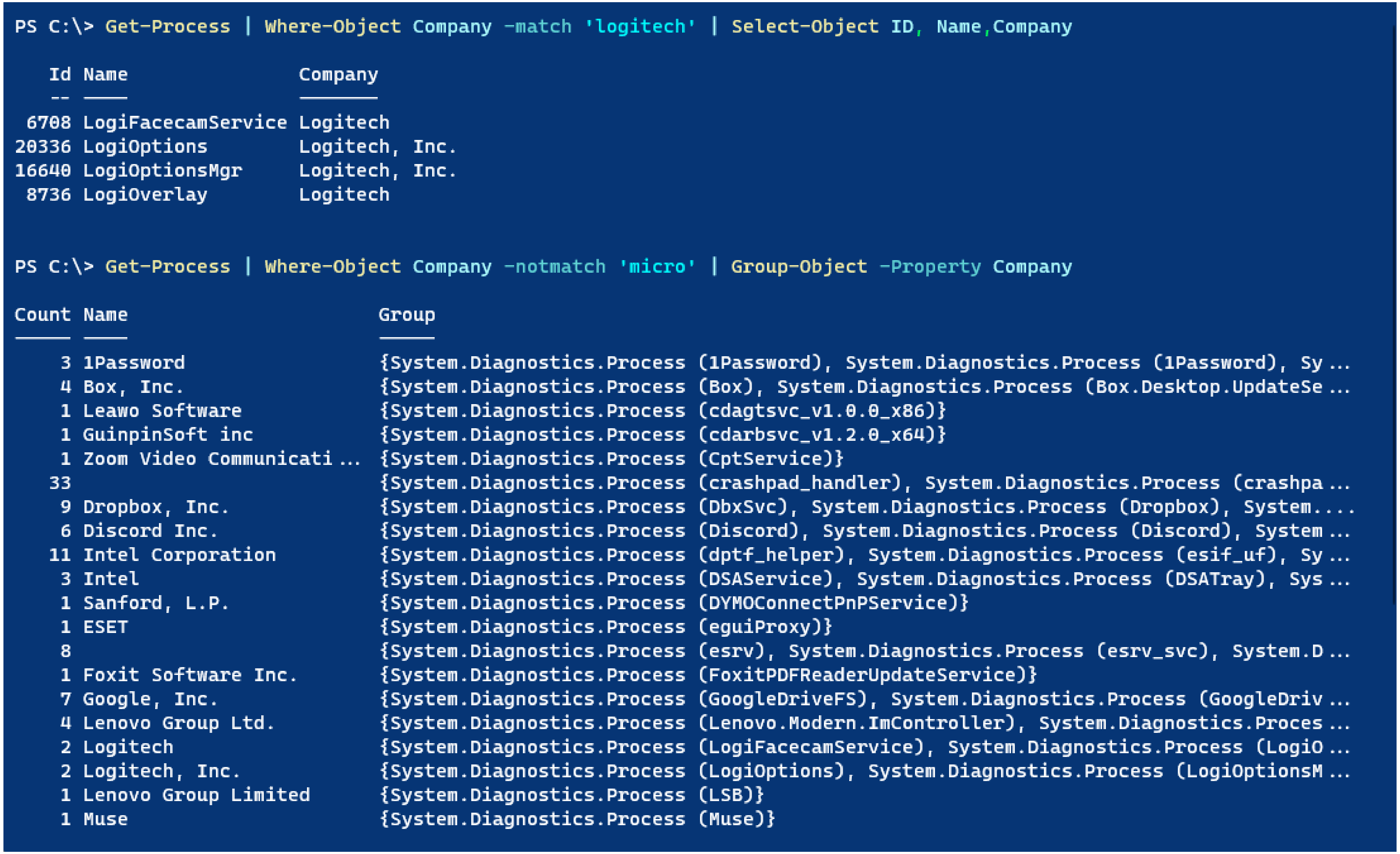
```
PS C:\> "abc-1234" -match "\d+"
True
PS C:\> $matches
Name                               Value
----                               -----
0                                  1234
PS C:\> help about_regular_expressions
PS C:\> "1234" -notmatch "\d+"
False
```

◀ **Use the regular expression operator**

◀ **Built-in variable captures the match**

◀ **This is an advanced topic**

◀ **Easy way to test a non-match**



```
PS C:\> Get-Process | Where-Object Company -match 'logitech' | Select-Object ID, Name,Company

   Id Name                Company
   -- ----                -------
 6708 LogiFacecamService Logitech
20336 LogiOptions         Logitech, Inc.
16640 LogiOptionsMgr      Logitech, Inc.
 8736 LogiOverlay         Logitech


PS C:\> Get-Process | Where-Object Company -notmatch 'micro' | Group-Object -Property Company

Count Name                      Group
----- ----                      -----
    3 1Password                 {System.Diagnostics.Process (1Password), System.Diagnostics.Process (1Password), Sy …
    4 Box, Inc.                 {System.Diagnostics.Process (Box), System.Diagnostics.Process (Box.Desktop.UpdateSe …
    1 Leawo Software            {System.Diagnostics.Process (cdagtsvc_v1.0.0_x86)}
    1 GuinpinSoft inc           {System.Diagnostics.Process (cdarbsvc_v1.2.0_x64)}
    1 Zoom Video Communicati …  {System.Diagnostics.Process (CptService)}
   33                           {System.Diagnostics.Process (crashpad_handler), System.Diagnostics.Process (crashpa …
    9 Dropbox, Inc.             {System.Diagnostics.Process (DbxSvc), System.Diagnostics.Process (Dropbox), System.…
    6 Discord Inc.              {System.Diagnostics.Process (Discord), System.Diagnostics.Process (Discord), System …
   11 Intel Corporation         {System.Diagnostics.Process (dptf_helper), System.Diagnostics.Process (esif_uf), Sy …
    3 Intel                     {System.Diagnostics.Process (DSAService), System.Diagnostics.Process (DSATray), Sys …
    1 Sanford, L.P.             {System.Diagnostics.Process (DYMOConnectPnPService)}
    1 ESET                      {System.Diagnostics.Process (eguiProxy)}
    8                           {System.Diagnostics.Process (esrv), System.Diagnostics.Process (esrv_svc), System.D …
    1 Foxit Software Inc.       {System.Diagnostics.Process (FoxitPDFReaderUpdateService)}
    7 Google, Inc.              {System.Diagnostics.Process (GoogleDriveFS), System.Diagnostics.Process (GoogleDriv …
    4 Lenovo Group Ltd.         {System.Diagnostics.Process (Lenovo.Modern.ImController), System.Diagnostics.Proces …
    2 Logitech                  {System.Diagnostics.Process (LogiFacecamService), System.Diagnostics.Process (LogiO …
    2 Logitech, Inc.            {System.Diagnostics.Process (LogiOptions), System.Diagnostics.Process (LogiOptionsM …
    1 Lenovo Group Limited      {System.Diagnostics.Process (LSB)}
    1 Muse                      {System.Diagnostics.Process (Muse)}
```

```
PS C:\> $i = 4
```

Logical Operators

Sometimes you have complex expressions

```
PS C:\> $i = 4
PS C:\> ($i -le 10) -AND ($PSVersionTable.PSVersion.Major -eq 5)
```

Logical Operators

Sometimes you have complex expressions
This expression AND that expression must be BOTH be True

```
PS C:\> $i = 4
PS C:\> ($i -le 10) -AND ($PSVersionTable.PSVersion.Major -eq 5)
True
```

Logical Operators

Sometimes you have complex expressions
This expression AND that expression must be BOTH be True
The entire expression is True

```
PS C:\> $i = 20
PS C:\> ($i -le 10) -AND ($PSVersionTable.PSVersion.Major -eq 5)
False
```

Logical Operators

One expression is False so the entire expression is False

```
PS C:\> $i = 20
PS C:\> ($i -le 10) -OR ($PSVersionTable.PSVersion.Major -eq 5)
True
```

Logical Operators

If either expression is True, the entire expression is True

```
PS C:\> $i = 20
PS C:\> $name = "jeff"
PS C:\> ($i -ge 20) -AND (($name -eq "Jeff") -OR ($PSEdition -eq "core"))
True
```

Logical Operators

Combine expressions
Parentheses very helpful

```
PS C:\> $i = 20
PS C:\> $name = "jeff"
PS C:\> ($i -ge 20) -AND (($name -eq "Jeff") -OR ($PSEdition -eq "core"))
True
```

Logical Operators

Combine expressions
Parentheses very helpful

```
PS C:\> Test-Path c:\windows\notepad.exe
True
```

Logical Operators

Normal result

```
PS C:\> Test-Path c:\windows\notepad.exe
True
PS C:\> -Not (Test-Path c:\windows\notepad.exe)
False
```

Logical Operators

Reverse the Boolean

```
PS C:\> Test-Path c:\windows\notepad.exe
True
PS C:\> -Not (Test-Path c:\windows\notepad.exe)
False
PS C:\> !(Test-Path c:\windows\notepad.exe)
False
```

Logical Operators

Reverse the Boolean
You can also use !
Expect to use more often in scripting

Other Operators

Math

Range

Unary and Assignment

```
PS C:\> 5+8
13
PS C:\> 9/3
3
PS C:\> 2*3*4
24
PS C:\> 10-6
4
PS C:\> (((5+3)/2)*7)-1
27
```

- Addition
- Division
- Multiplication
- Subtraction
- Control precedence

```
PS C:\> 1..5
1
2
3
4
5
PS C:\> 10..7
10
9
8
7
```

- Range operator
- Get numbers from start to finish
- Get numbers in reverse

```
PS C:\> $a = 1
PS C:\> $a = $a+2
PS C:\> $a
3
PS C:\> $a++
PS C:\> $a
4
PS C:\> $a--
PS C:\> $a
3
PS C:\> $a+=5
PS C:\> $a
8
PS C:\> $a*=2
PS C:\> $a
16
PS C:\> $a/=4
PS C:\> $a
4
```

◂ Assign a value to a variable

◂ Increase by 2

◂ The unary operator – increase value by 1

◂ Decrease value by 1

◂ Increase value by 5

◂ Multiply value by 2

◂ Divide value by 4

Key Take-Aways

Operators are key elements of the Windows PowerShell language

You can use them interactively in the console

… Or when scripting

Group with parentheses for clarity

Read the help!