It’s all about Progress: Using the PowerShell Write-Progress CmdLet

If you are like me and you like to know how your task is going in a PowerShell process, this is a great little tip for you.

I’ve got a number of long running scripts that perform actions against a collection or query. The ideal thing to have for these is some kind of progress bar to be able to find out how far along your script is.

There are a number of scripts out there which help you to create a full graphical progress bar, but the challenge with those is that it engages lots of external libraries and you very quickly find yourself in a heavy duty yak shaving exercise spending more time getting the progress bar working than working on your actual script.

This is where **Write-Progress** will be a quick and friendly way to get you what you need. The simple CmdLet allows you to write a progress indicator which is overlaid on the screen as the script runs:

I’ve always been a big fan of using Growl for services but adding Growl integration is a bit heavy handed for many small scripts. Write-Progress lets you show a percentage completed by measuring against a counter that you set.

Progress by Count

Here is a simple example. We will write a number of items into a variable and then output the contents while showing a progress bar to indicate where in the output stream you are.

```
$comms = Get-Command

$i=0

ForEach ($comm in $comms) {

$i++

Write-Host $comm

Write-Progress -activity “Listing Commands” -status “Status: ” -PercentComplete (($i / $comms.count)*100)

}
```

It’s a rudimentary example to show you how to add counter logic into a script. As the collection counter rises, it is measured against the sum of objects in the collection which gives you the percentage.

Here is what you see on the screen:

Another sample use case is that you want to perform numerous actions against a number of objects such as users. Remember that to use the Write-Progress feature you simply need to add a counter in your collection and calculate the progress of your counter against the collection total for display.

For the full detail of the CmdLet options, run the ***Get-Help Write-Progress -Detailed*** for the full help display.

So to break down the sample, here is how to add the Write-Progress feature into a script:

- Create your collection: ***$comms = Get-Command***
- Clear your counter: ***$I = 0***
- Loop through the collection: ***ForEach ($comm in $comms) {***
- Increment your counter: ***$I++***
- Write your output as per usual: ***Write-Host $comm***
- Write your progress bar (see below)
- End your loop: ***}***

The parameters we use for the Progress bar are -activity, -Status and **-PercentageComplete** which appear in your output as shown here:

It’s just that easy! All you have to do is use this same method and apply it to your particular collection. It’s a little thing, but can be very helpful with long running processes and scripts to give feedback to the operator, particularly if you don’t write other output to the screen.