Using PowerShell's Formatting System

Jeff Hicks

Author/Teacher

@jeffhicks | https://jdhitsolutions.com/blog

PowerShell Output Formatting

PowerShell displays objects at the end of the pipeline

PowerShell has formatting guidelines and rules

You can override the defaults

PowerShell Output Formatting

PS1XML

Objects matter

Default format definition files

- Stored as ps1xml files
- $PSHome
- Do not modify
- You can create your own

PowerShell Output Formatting

Get-Process

Out-Default

Out-Host

PowerShell Output Formatting

Is there a defined default view?

Is there a defined set of properties?

Five or fewer properties display a table

Otherwise display a list

Format Cmdlets

Format-Table

Format-List

Format-Wide

Format-Custom

```
Get-Process
```

Default Formatting Process

Run your initial command

```
Get-Process | Out-Default
```

Default Formatting Process

PowerShell implicitly sends to Out-Default
Out-Default determines what to display

```
Get-Process | Out-Default | Format-Table
```

Default Formatting Process

PowerShell invokes the necessary formatting cmdlet
Could be Format-List

```
Get-Process | Out-Default | Format-Table | Out-Host
```

Default Formatting Process

Results are sent to the hosting application to display

You can override the default formatting

Specify the formatting cmdlet

Leverage formatting options

- Grouping
- Custom properties
- Alternate views

```
Get-Process p* | Select-Object name,id,ws,vm,path | Format-Table
```

Format-Table

Has an alias of ft
Supports grouping
Force spacing with -Autosize
Force long lines to wrap
Makes a “best effort”

```
Get-Service bits | Format-List
```

Format-List

Has an alias of fl
Supports grouping
Good for lengthy property values
Some objects may have a default list view

```
Get-Process | Where-Object { $_.ws -ge 100MB} | Format-Wide
```

Format-Wide

Has an alias of fw
Uses a default display property
Or specify your own
You can specify the number of columns

Formatting Views

Some objects have alternative views

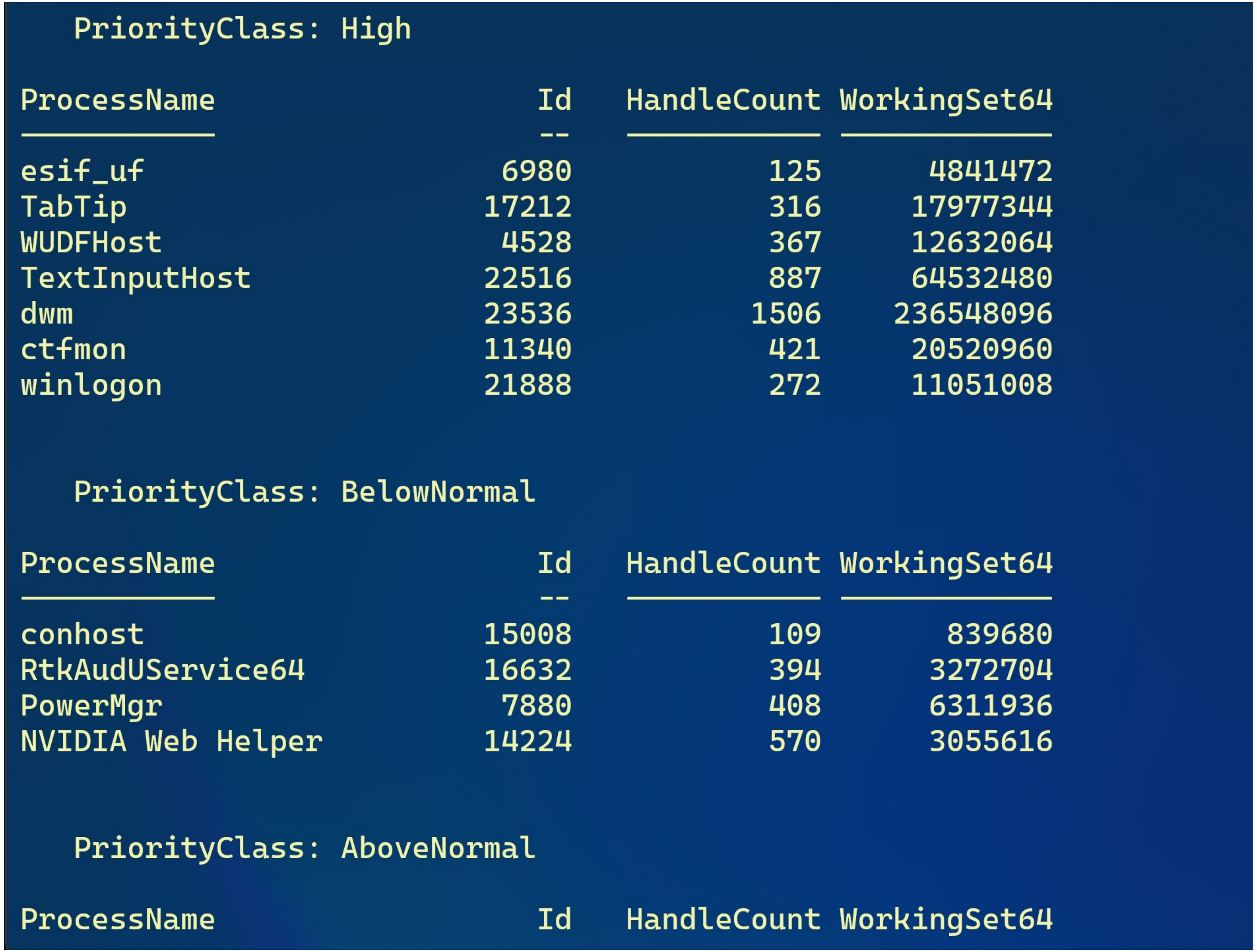
Defined in the ps1xml file

Not easily discoverable

```
Get-Process | Sort-Object priorityclass | Format-Table -view priority
```

Formatting views

May require pre-sorting



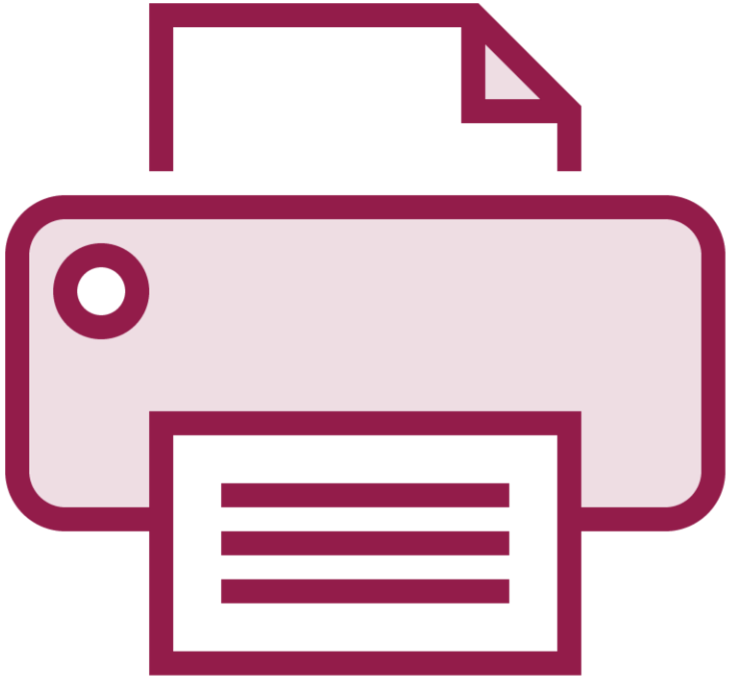
```
   PriorityClass: High

ProcessName                  Id   HandleCount WorkingSet64
-----------                  --   ----------- ------------
esif_uf                    6980           125      4841472
TabTip                    17212           316     17977344
WUDFHost                   4528           367     12632064
TextInputHost             22516           887     64532480
dwm                       23536          1506    236548096
ctfmon                    11340           421     20520960
winlogon                  21888           272     11051008


   PriorityClass: BelowNormal

ProcessName                  Id   HandleCount WorkingSet64
-----------                  --   ----------- ------------
conhost                   15008           109       839680
RtkAudUService64          16632           394      3272704
PowerMgr                   7880           408      6311936
NVIDIA Web Helper         14224           570      3055616


   PriorityClass: AboveNormal

ProcessName                  Id   HandleCount WorkingSet64
```

Formatting should be at the end of your expression. Think, “Format Right”

Demo

Formatting Cmdlets

Other Formatting and Output Options



You *can* pipe format output to another command

Out-* cmdlets will accept formatted objects

- Out-File
- Out-Printer
- Out-String

Exception to the exception

- Out-Gridview

```
PS C:\> Get-WinEvent -FilterHashtable @{Logname='system';Level=2} -MaxEvents 100 |
Format-Table Timecreated,ProviderName,ID,Message -AutoSize -Wrap | Out-File
c:\work\SysErr.txt
```

Out-File

Pay attention to page widths

- Set page width
- Wrap formatted data

```
PS C:\> Get-WinEvent -FilterHashtable @{Logname='system';Level=2} -MaxEvents 100 |
Format-Table Timecreated,ProviderName,ID,Message -AutoSize -Wrap | Out-Printer
```

Out-Printer

Same guidelines as saving to a file
Use Get-Printer to discover printer names
Great way to generate PDF documentation

```
PS C:\> $s = get-service -computername DOM1 | format-list -Property
Name,Displayname,State,StartType,MachineName | Out-String
```

Out-String

Special use case where you need to treat output as a string

```
PS C:\> $s = get-service -computername DOM1 | format-list -Property
Name,Displayname,State,StartType,MachineName | Out-String
PS C:\> $s -replace "DOM1","********"
```

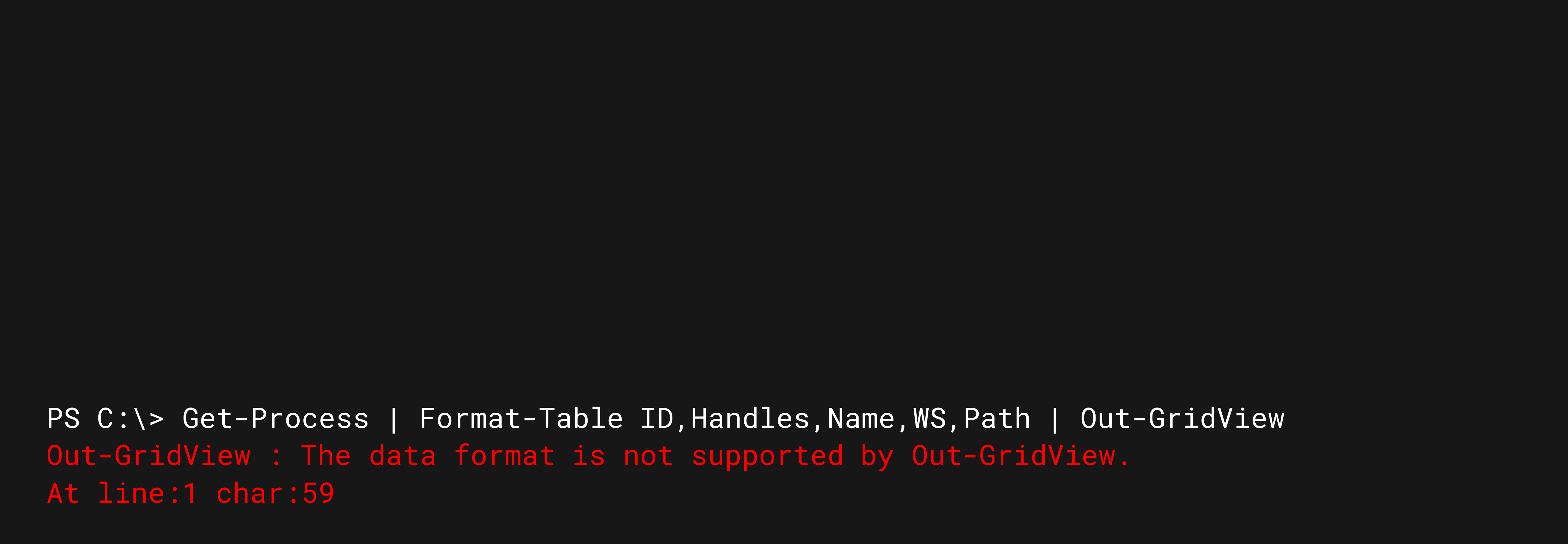
Out-String

Special use case where you need to treat output as a string

```
Name        : ADWS
DisplayName : Active Directory Web Services
StartType   : Automatic
MachineName : ********

Name        : AppHostSvc
DisplayName : Application Host Helper Service
StartType   : Automatic
MachineName : ********
...
```

Out-String

Special use case where you need to treat output as a string



```
PS C:\> Get-Process | Format-Table ID,Handles,Name,WS,Path | Out-GridView
Out-GridView : The data format is not supported by Out-GridView.
At line:1 char:59
```

Out-Gridview

The exception to the exception

```
PS C:\> Get-Process | Select-Object ID,Handles,Name,WS,StartPath | Out-GridView
```

Out-Gridview

Has options to filter and passthru objects

Demo

Other Formatting Options