The Problem

- PowerShell is all about the objects
- Leveraging the pipeline is critical to maximum efficiency
- PowerShell solutions are still arriving
- …legacy command line tools fill the gap
- ….how can we turn them into PowerShell tools?

Requirements

- CLI tool should use StdOut
- CLI tool should offer predictable or formatted output
- Avoid interactive tools
- …unless they offer a “scripted” solution
- These techniques are better suited for scripting

Techniques

- Look for PowerShell data formats (e.g. CSV)
- Parse text output
- Leverage regular expressions
- Re-Assemble as an object with New-Object

Look for PowerShell Data Formats

- Does the CLI command offer data formats?
- Can you save formatted results to a file?
- Some output you can turn directly into objects

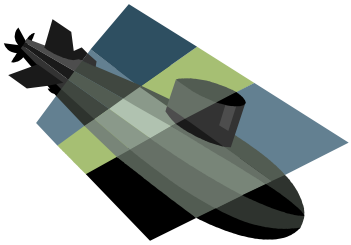
```
PS C:\> driverquery /fo csv | convertFrom-csv
```

- Be careful creating property names with spaces

Parse Text

- Skip lines using Select-Object
- Split lines into arrays
- Use Substring() method to further parse
- Use Regular Expression Patterns
- Trim your parts

Dive! Dive! Dive!



Examples and Demos

- DriverQuery.exe
- Schtasks.exe
- Net User
- Nbtstat.exe
- Get-NbtMac

- *Focus on technique examples not commands*

Future Challenges

- Defining properties of type such as DateTime
- Incorporating type and format extensions
- “Good” property names are always an issue
- Handling requests for StdIn
- Handling command line errors