PowerShell Object pipeline shell

CONCEPTS

Commands pass .NET objects, not text: filter and sort by property, no column parsing.

Get-ChildItem file objects
↓ |
Where-Object keep some
↓ |
Sort-Object reorder
↓ |
Select-Object shape
↓ |
Format / Export text at the end

Objects flow through the pipeline; $_ is the current one

Cmdlets are named Verb-Noun. Names and parameters are case-insensitive and Tab-complete.

DISCOVER

Get-Help *<cmd>*
Show help for a command

Get-Help *<cmd>* **-Examples**
Show usage examples

Get-Command ***<word>***
Find commands by name

<cmd> **| Get-Member**
List an object's type, properties, methods

PIPELINE

Where-Object *<prop>* **-gt** *<val>*
Keep objects matching a test

Where-Object { $_.Length -gt 1MB }
Filter with a script block

Select-Object *<prop>*, *<prop>*
Keep only these properties

Select-Object -First *<n>*
Take the first n objects

Select-Object -ExpandProperty *<prop>*
Output the bare values

Sort-Object *<prop>* **-Descending**
Sort by a property

ForEach-Object { $_.Name }
Run a block per object

PIPELINE (cont.)

Format-List *
Show every property

FILES & PROVIDERS

Get-ChildItem
List items here

Get-ChildItem -Recurse -Filter *<glob>*
Find files recursively

Set-Location *<path>*
Change directory

Get-Content *<file>*
Read a file line by line

New-Item *<dir>* **-ItemType Directory**
Create a directory

Test-Path *<path>*
Check if a path exists

Select-String *<pattern>* *<file>*
Search text, like grep

VARIABLES & TYPES

$name = *<value>*
Assign a variable

$env:*<NAME>*
Read or set an environment variable

$a = @(1, 2, 3)
Array; $a[-1] is the last item

$h = @{ Key = 'val' }
Hashtable; $h.Key or $h['Key']

[pscustomobject]@{ A = 1; B = 2 }
Build an object with properties

STRINGS & OPERATORS

"Hi $name, $($obj.Prop)"
Double quotes expand variables

'literal $text'
Single quotes do not expand

-eq -ne -gt -lt -ge -le
Compare (case-insensitive)

-like '*<glob>******'
Wildcard match

STRINGS & OPERATORS (cont.)

-match '*<regex>***'**
Regex match; groups in $Matches

-replace '*<regex>***', '***<new>***'**
Regex replace

DATA FORMATS & WEB

Get-Content *<file>* **-Raw | ConvertFrom-Json**
Parse a JSON file

ConvertTo-Json -Depth *<n>*
Serialise objects to JSON

Import-Csv *<file>*
Read CSV as objects

Invoke-RestMethod *<url>*
Call an API, parse the reply

SCRIPTS & FUNCTIONS

function *<Name>* **{ param($x) ... }**
Define a function

param([Parameter(Mandatory)] [string]$Name)
Required, typed parameter

.*<script>***.ps1**
Run a script

if (*<cond>***) { } elseif () { } else { }**
Conditional

foreach ($i in $items) { }
Loop over a collection

ERRORS

try { } catch { $_ } finally { }
Handle terminating errors

-ErrorAction Stop
Make one command's errors catchable

$ErrorActionPreference = 'Stop'
Stop on every error in scope

PSREADLINE KEYS

Tab / Shift + Tab
Cycle completions