Powershell

Variablen

```
$variable = Wert
${variable} = Wert
[int] $x = 5
```

Datentypen: `[byte]`, `[int]`, `[long]`, `[single]`, `[double]`, `[decimal]`, `[bool]`, `[char]`, `[string]`

Arrays

```
[int[]] $feld = 7, 12, -5
$feld[0] # 7
$feld[-1] # -5
```

Schlüssel	Wert
0	7
1	12
2	-5

Hash-Tables

```
$lehrer = @{ vorname = "Franz"; nachname = "Kohnle"; alter = 45 }
$lehrer["vorname"] # Franz
$lehrer.nachname # Kohnle
```

Schlüssel	Wert
vorname	Franz
nachname	Kohnle
alter	45

Operatoren

- arithmetisch: `+ - * / %`
- Zuweisung: `= += -= *= /= %= ++ --`
- logisch: `-and -or -xor -not`
- Vergleich: `-eq -ne -ge -gt -lt -le`

Ausdrücke

- `"Hallo"` → `Hallo`
- `4 + 1.5 * 3` → `8,5`
- `4 + "2"` → `6`
- `"4"+ 2` → `42`
- `"Hallo" * 3` → `HalloHalloHallo`
- `1 -lt 2` → `True`

Kontrollstrukturen

```
if switch do while for foreach
```

Skriptdateien

- Dateiendung: `.ps1`
- Skriptausführung zulassen: `Set-ExecutionPolicy Unrestricted` (als Administrator)
- Integrated Scripting Environment: `ISE`

Cmdlets (Commandlets)

Grundstruktur

```
Verb-Substantiv -Parameter Parameterwert -Schalter
```

Beispiele

Cmdlet	Beispiel	Bedeutung
`Show-Command`		Fenster mit Liste aller Cmdlets
`Get-Command`	`Get-Command get*`	Liste aller Cmdlets
`Get-Help`	`Get-Help Get-Command -full`	Hilfe
`Get-Member`	`Get-Process | Get-Member`	laufende Prozesse anzeigen
`Get-ChildItem`	`Get-ChildItem c:\Users`	Inhalt des Verzeichnisses
`Get-History`		Zuletzt eingegebene Befehle
`Invoke-History`	`Invoke-History 5`	Befehl mit Id 5 ausführen

Komplette Hilfe in einer Textdatei speichern

1. Hilfe aktualisieren

als Administrator: `Update-Help`

2. Hilfe in Datei umleiten

```
$AlleThemen = Get-Help *
$Allethemen | Get-Help -Full | Out-File hilfe.txt
```

Aliase

Liste mit Aliasen anzeigen: `Get-Alias`

Alias	Cmdlet
`cat`, `gc`, `type`	`Get-Content`
`cd`, `chdir`, `sl`	`Set-Location`
`cp`, `copy`, `cpi`	`Copy-Item`
`echo`, `write`	`Write-Output`
`kill`	`Stop-Process`
`ls`, `dir`, `gci`	`Get-ChildItem`
`mv`, `move`, `mi`	`Move-Item`
`rm`, `del`, `rmi`	`Remove-Item`

Pipeline mit |

```
Cmdlet1 | Cmdlet2
```

Es werden im Gegensatz zur Linux-Konsole nicht nur Texte, sondern typisierte .NET-Objekte weitergeleitet.

Sortieren

```
| Sort-Object -Property xyz
```

Filtern

```
| Select-Object -First 10
| where { $_.Extension -eq ".txt"}
```

Ausgabe in Datei

```
> datei.txt
| Out-File datei.txt
| Export-Csv datei.csv
| Export-Clixml datei.xml
```

Provider

Provider liefern Zugriff auf Daten

- Liste anzeigen: `Get-PSProvider` bzw. `Get-PSDrive`
- Laufwerk wechseln: `cd` *`Provider`*`:`
- Inhalt anzeigen: `Get-ChildItem`

Zugriff auf .NET-Klassen

`[Klasse]::Methode(...)`	statische Methode
`$objekt.Methode(...)`	Instanzmethode
`$objekt.Eigenschaft`	Instanzeigenschaft

Methoden und Eigenschaften ermitteln

```
[Klasse] | Get-Member
$objekt | Get-Member
```