Referenz

0.8

PowerShell-Grundlagen

Elementares

➔ *Groß- und Kleinschreibung ist beliebig!*

Cursortasten:	letzte Befehle anzeigen/bearbeiten
→ı	[Tab] Befehl/Dateinamen/Optionen ergänzen
^C	Abbruch von laufenden Programmen
`	Befehls-Fortsetzung bei Zeilenumbruch
[ret][ret]	Abschluss für manuelle Texteingabe
#	Kommentar
;	Befehlstrenner
|	Pipe
Out-Host -Paging	In einer Pipe: Seitenweises Anzeigen
get-command	Alle Commandlets anzeigen
Get-History	Letzten Befehle anzeigen
F7	Pop-Up: Befehl wiederholen

Hilfen

-?	Als Parameter: Zeigt Hilfe an
get-help *	Zeigt alle Hilfethemen an.
-detailed	Erweiterte Anzeige mit Beispielen
-full	Komplette technische Hilfe
get-help get-	Zeigt die Themen an, die mit "get-" beginnen.
get-help *Objekt*	Zeigt Themen mit "Objekt" im Namen
get-help about*	Hilfe zu Powershell-Anweisungen ("foreach")
get-help about_wildcard	Hilfe zu Wildcards
get-alias	Alle Alias-Namen anzeigen ("get-alias g*")

Informationen zu Kommandos

Get-command get-*	Parameter mit Wildcards
Get-Command *process*	Alle mit "process" im Namen
Get-Command -Verb get	Alle, die mit "get" beginnen
Get-Command -Noun Service	Alle, die mit "Service" enden

Internet-Tipps

joeware.net	Ausgereifte Tools
blogs.msdn.com/Powershell	Microsoft Entwickler

Profile

profile.ps1	Autostart-Datei
notepad $Profile	Profile-Datei bearbeiten

Objekt-Zugriff

Get-Member -MemberType Properties	Methoden + Eigenschaften anzeigen (kurz: "gm")
$_.	Zugriff auf aktuelles Objekt
$objekt	Anzeige auf Bildschirm
(Get-Host).privatedata	Zugriff auf Eigenschaften eines Objekts
(Get-Host).GetType()	Zugriff auf Methoden eínes Objekts

Ausgabe

Format-List *	Alle Eigenschaften eines Objectes anzeigen
Format-Table -AutoSize	Tabellarisch anzeigen
out-host -paging	Seitenweises Ausgeben
Out-File out.txt	Ausgabe in Datei speichern
Out-Printer	Directes Ausdrucken

Konvertierung von Pipe-Daten

ConvertTo-Html	Ausgeben als HTML
ConvertTo-SecureString	Verschlüsselt ausgeben
Export-Csv	Ausgeben als CSV
Export-Clixml	Ausgeben als XML

Provider

Get-PSDrive	Anzeige aller Laufwerke
Get-PSProvider	Alle Provider anzeigen (Drives beachten!)
Bsp: cd HKLM:	In Registry-Ordner HKLM wechseln
Remove-PSDrive	Laufwerk entfernen

Erweiterungen der PowerShell

Get-PSSnapin	Powershell-Tools anzeigen
Get-PSSnapin -registered	Registrierte Tools anzeigen
Add-PSSnapin	Tool aktivieren

Infos

Get-Acl H: | format-list *	Acl-Liste bei Dateien anzeigen lassen
Get-Alias	Alias-Namen anzeigen
get-authenticodesignature	Authentikation für Datei anzeigen
Get-ChildItem	Directory-Listing
Get-Credential	Anmeldefenster für Powershell
Get-Culture	Ländereinstellung anzeigen
Get-Date | ft *	Alles zum aktuellen Datum
Get-EventLog -logname application -newest 10	Neueste 10 Ereignisse anzeigen lassen
Get-ExecutionPolicy	Ausführungseinstellung für Shell-Script
Get-History	Eingegebene Befehle anzeigen lassen
Get-Host	Informationen zur Konsole
Get-Item c:	Directory anzeigen
Get-Location	Aktuelles Verzeichnis anzeigen
Get-Process	Aktuelle Prozesse anzeigen
Get-PSDrive	Alle Laufwerke anzeigen
Get-Service	Alle Dienste anzeigen
Get-TraceSource	Ablaufverfolgung von Scripten
Get-Variable	Alle Variablen anzeigen

Variablen

$a = "Stefan Menne"	Variable neu anlegen
[int] $a = 32	Typvorgabe
($a).Length	Länge ausgeben
$a	Variable anzeigen
$b = 1, 'zwei', 'drei', 4	Array definieren
$b[2]	Array-Element anzeigen
Set-Variable -name pi -value 3.14159265 -option constant	Konstante definieren

Terminal

clear-host	Terminal löschen
get-host	Infos anzeigen
$a = Read-Host "Name"	Eine Zeile lesen (Opt: [-asSecureString])
Write-Host $a -f "green"	Text ausgeben (-f = -foregroundcolor)
Write-Error	Text als Fehler ausgeben

Fenster-Eigenschaften anpassen:

Fenster-Eigenschaften: Quick-Edit-Modus einschalten, dann Rechtsklick!

mode con cols=132	Breiteres Fenster einstellen
get-host | gm	
(get-host).privatedata |gm	
(get-host).PrivateData.ErrorForegroundColor="yellow"	Farbe für Fehler ändern (Tab-Taste für Eingabe nutzen!)

Powershell-Tools

Select-Object

Select-Object	
[-property] ID,Name	Eigenschaften auswählen
-first 7	die ersten 7 Datensätze
-last 12	die letzten 12 Datensätze
-unique	gleiche Datensätze unterdrücken

Bsp: get-Process | Select-Object -property ws,cpu,name -first 5

Sort-Object (Alias: "sort")

Sort-Object	
[-property] Name	Sortierung nach der Eigenschaft
-unique	gleiche Datensätze unterdrücken
-descending	absteigend sortieren
-caseSensitive	Groß-/Kleinschreibung beachten!

Bsp: get-childitem | sort-object -property length -descending

Measure-Object

Measure-Object	
-property …	
-average|sum|minimum|maximum	Zahlen
-line|word|character	Text

Bsp: get-childitem | measure-object -property length -min -max -average

Where-Object (Alias: "?", "where")

Where-Object { *powershell-script* }
Script in einer Pipe ausführen

Bsp: get-service | where-object { $_.Status -eq "Stopped"}
get-process | where-object {$_.workingset -gt 10MB}

Foreach-Object (Alias: "%", "foreach")

Foreach-Object { *powershell-script* }
Für jedes Objekt etwas ausführen

Bsp: @(30,40,80,200) | foreach { $_ / 10 }
400,600,800,100 | ForEach-Object { [Console]::Beep($_,200) }
dir c:\ | % { $n+=1; Write-Host $n ": " $_ }

Operatoren

Standard-Operatoren:

\+ - * / % = += -= *= /= %=

Vergleiche:

-eq (==) -ne (!=) -gt (>) -ge (>=) -lt (<) -le (<=)

String-Vergleiche:

-like (wildcard vergl. "Tom" –like "T*") -notlike (Gegenteil hierzu)
-match (Reg.Expr.-Vergl. "lauf" –match "[abc]") -notmatch

Logische Operatoren:

-and -or -not !

Zahlformatierung

Bsp: "{0:N2}" -f 554.22272 oder "{1:N1}" -f 554.22272,749.24234
Format-Codes: {Index:Code[Genauigkeit]}
Codes: N (Zahl) C (Währung), D (Dezimal), P (Prozent), X (Hex.)
String-Operatoren:
\+ (Zusammenhängen) * 50 (Zeichenkette wiederholen)