

HANSER

Bruce Payette

Windows PowerShell im Einsatz

ISBN-10: 3-446-41239-5

ISBN-13: 978-3-446-41239-2

Inhaltsverzeichnis

Weitere Informationen oder Bestellungen unter
<http://www.hanser.de/978-3-446-41239-2>
sowie im Buchhandel

Inhaltsverzeichnis

Geleitwort	15
Vorwort	17
Danksagung	19
Über dieses Buch	21

Teil I: PowerShell lernen

1 Willkommen bei PowerShell	29
1.1 Was ist PowerShell?	31
1.1.1 Shells, Befehlszeilen und Skriptsprachen	32
1.1.2 Warum eine neue Shell? Warum gerade jetzt?	33
1.1.3 Das Problem der letzten Meile	34
1.2 Seele einer neuen Sprache	34
1.2.1 Aus der Geschichte lernen	35
1.2.2 Unterstützung von .NET	35
1.3 OOP-Auffrischkurs	37
1.3.1 Objektorientiertes Programmieren	37
1.3.2 Objekte in PowerShell	39
1.4 Heh! Wo bleibt der Code?	39
1.4.1 Installieren und starten von PowerShell	40
1.4.2 Editieren von Befehlen	42
1.4.3 Befehlsvervollständigung	43
1.4.4 Auswerten von elementaren Ausdrücken	44
1.4.5 Verarbeiten von Daten	45
1.5 Zusammenfassung	51

2	Grundlagen	53
2.1	Befehlskonzepte und Terminologie	54
2.1.1	Befehle und Commandlets	55
2.1.2	Befehlstypen	58
2.1.3	Aliase und flexible Syntax	62
2.2	Parsing und PowerShell	64
2.2.1	Wie PowerShell ein Skript zerlegt	65
2.2.2	Quoting	66
2.2.3	Parsing im Ausdrucks- und im Befehlsmodus	69
2.2.4	Terminieren von Statements	71
2.3	Pipelines und Befehle	73
2.3.1	Pipelines und Streamingverhalten	74
2.3.2	Parameter und Parameterbindung	75
2.4	Formatierung und Ausgabe	77
2.4.1	Die Format-Commandlets	78
2.4.2	Die Output-Commandlets	80
2.5	Zusammenfassung	83
3	Arbeiten mit Typen	85
3.1	Typenmanagement wie im wilden Westen	85
3.1.1	PowerShell: eine typ-promiskuitive Sprache	86
3.1.2	Typ-System und Typ-Adaption	88
3.2	Basistypen und Literale	90
3.2.1	Strings	90
3.2.2	Zahlen und numerische Literale	95
3.2.3	Collections: Dictionaries und Hashtabellen	97
3.2.4	Collections: Arrays und Sequenzen	102
3.2.5	Typliterale	107
3.3	Typumwandlungen	110
3.3.1	Wie die Typumwandlung funktioniert	111
3.3.2	PowerShells-Algorithmus zur Typumwandlung	113
3.3.3	Spezielle Typumwandlungen beim Binden von Parametern	116
3.4	Zusammenfassung	118
4	Operatoren und Ausdrücke	119
4.1	Arithmetische Operatoren	121
4.1.1	Der Additions-Operator	121

4.1.2	Der Multiplikations-Operator	124
4.1.3	Subtraktion, Division und der Modulo-Operator	126
4.2	Die Zuweisungsoperatoren	128
4.2.1	Mehrfache Zuweisungen	129
4.2.2	Mehrfache Zuweisungen mit Typqualifizierern	130
4.2.3	Zuweisungsoperationen als Wertausdrücke	132
4.3	Vergleichsoperatoren	133
4.3.1	Skalare Vergleiche	134
4.3.2	Vergleichsoperatoren und Auflistungen	138
4.4	Operatoren zum Musterabgleich	140
4.4.1	Wildcard Pattern	140
4.4.2	Reguläre Ausdrücke	141
4.5	Logische und Bit-Operatoren	146
4.6	Zusammenfassung	147
5	Erweiterte Operatoren und Variablen	149
5.1	Operatoren für die Arbeit mit Typen	149
5.2	Die unären Operatoren	152
5.3	Gruppieren, Unterausdrücke und Array-Unterausdrücke	153
5.4	Array-Operatoren	158
5.4.1	Der Komma-Operator ","	158
5.4.2	Der Bereichsoperator	161
5.4.3	Indizieren von Arrays	162
5.5	Operatoren für Eigenschaften und Methoden	168
5.5.1	Der "."-Operator	169
5.5.2	Statische Methoden und der "::"-Operator	171
5.6	Der PowerShell Format-Operator -F	172
5.7	Umleitung und Umleitungsoperatoren	174
5.8	Variablen	177
5.9	Zusammenfassung	182
6	Flusskontrolle in Skripten	183
6.1	Das if/elseif/else-Statement	184
6.2	Die while-Schleife	187
6.3	Die do/while-Schleife	188
6.4	Die for-Schleife	189
6.5	Die foreach-Schleife	191

6.6	Labels, break und continue	195
6.7	Das switch-Statement	197
6.7.1	Grundlegendes zum switch-Statement	198
6.7.2	Wildcard Pattern und switch-Statement	199
6.7.3	Reguläre Ausdrücke mit dem switch-Statement	200
6.7.4	Dateiverarbeitung mit dem switch-Statement	204
6.7.5	Der \$switch-Schleifenenumerator im switch-Statement	205
6.8	Ablaufsteuerung mit Commandlets	206
6.8.1	Das Foreach-Object-Commandlet	207
6.8.2	Das Where-Object-Commandlet	210
6.9	Der Wert von Statements	212
6.10	Zusammenfassung	213
7	Funktionen und Skripte	215
7.1	Grundlagen von Funktionen	216
7.2	Formale Parameter und das param-Statement	219
7.2.1	Spezifizieren von Parametertypen	221
7.2.2	Umgang mit einer variablen Anzahl von Argumenten	223
7.2.3	Initialisieren von Funktionsparametern	224
7.2.4	Verwendung von switch-Parametern zum Definieren von Flags	226
7.2.5	Lebenszeit von Variablen und Sichtbarkeits-Regeln	228
7.2.6	Bereichsmodifizierer	231
7.3	Rückgabewerte von Funktionen	231
7.3.1	Debugging des Funktionsoutputs	234
7.3.2	Das return-Statement	236
7.4	Funktionen in einer Pipeline	238
7.4.1	Filter und Funktionen	239
7.4.2	Funktionen als Commandlets	241
7.5	Verwalten von Funktionen	242
7.6	Zu guter Letzt – Skripte	244
7.6.1	Übergabe von Argumenten an Skripte	245
7.6.2	Das param-Statement	246
7.6.3	Bereiche und Skripte	247
7.6.4	Skripte beenden und exit-Statement	248
7.6.5	Dotting von Skripten und Funktionen	250
7.7	Zusammenfassung	252

8	Skriptblöcke und Objekte	253
8.1	Skriptblock-Grundlagen	254
8.1.1	Aufruf von Befehlen	255
8.1.2	Zugriff auf CommandInfo-Objekte	255
8.1.3	Das Skriptblock Literal	258
8.1.4	Funktionen zur Laufzeit definieren	259
8.2	Erzeugen und Manipulieren von Objekten	261
8.2.1	Blick auf die Mitglieder	261
8.2.2	Synthetische Mitglieder	262
8.2.3	Objekte mit Add-Member erweitern	264
8.2.4	Das Select-Object-Commandlet	270
8.3	Arbeiten am Typsystem	273
8.3.1	Hinzufügen einer Eigenschaft	275
8.3.2	Shadowing vorhandener Eigenschaften	276
8.4	Die PowerShell-Sprache erweitern	277
8.4.1	Kleine Sprachen	277
8.4.2	Hinzufügen eines CustomClass-Schlüsselworts	278
8.5	Typenerweiterung	283
8.6	Code zur Laufzeit generieren	285
8.6.1	Das Invoke-Expression-Commandlet	286
8.6.2	Die ExecutionContext-Variable	286
8.6.3	Erzeugen von Funktionen über das function: Laufwerk	289
8.7	Zusammenfassung	290
9	Fehler, Ausnahmen und Skript-Debugging	293
9.1	Fehlerbehandlung	294
9.1.1	ErrorRecords und ErrorStream	295
9.1.2	Die \$error-Variable und der -ErrorVariable-Parameter	298
9.1.3	Die Variablen \$? und \$LASTEXITCODE	301
9.1.4	\$ErrorActionPreference und der -ErrorAction-Parameter	304
9.2	Umgang mit Fehlern die die Ausführung beenden	307
9.2.1	Das trap-Statement	307
9.2.2	Das throw-Statement	310
9.3	Skript-Debugging	312
9.3.1	Debugging mit den Host-APIs	312
9.3.2	Das Set-PSDebug-Commandlet	314
9.3.3	Überwachen der Befehlsausführung	314

9.3.4	Schrittweise Überwachung der Ausführung von Anweisungen	317
9.3.5	Überwachen undefinierter Variablen im Strict-Modus	319
9.4	Eingebettete Prompts und Breakpoints	320
9.4.1	Abbrechen eines Skripts während des Schritt-Modus	320
9.4.2	Erstellen eines breakpoint-Befehls	322
9.4.3	Skript ruft Stack oder "Wie kam ich hierher?"	324
9.5	Low-Level-Überwachung	326
9.5.1	Das Trace-Command-Commandlet	327
9.5.2	Verfolgen von Typkonvertierungen	328
9.5.3	Überwachen der Parameterbindung	330
9.6	Das Ereignisprotokoll von PowerShell	334
9.6.1	Das Ereignisprotokoll untersuchen	334
9.6.2	Exchange 2007 und das PowerShell-Ereignisprotokoll	336
9.7	Zusammenfassung	337

Teil II: PowerShell anwenden

10	Verarbeitung von Texten, Dateien und XML	341
10.1	Verarbeiten von unstrukturiertem Text	341
10.1.1	Verwenden von System.String für das Arbeiten mit Text	341
10.1.2	Verwendung regulärer Ausdrücke zur Textmanipulation	348
10.2	Dateiverarbeitung	349
10.2.1	Arbeiten mit PSDrives	351
10.2.2	Arbeiten mit Pfaden die Wildcards enthalten	353
10.2.3	Lesen und Schreiben von Dateien	357
10.2.4	Suche in Dateien mit dem Select-String-Commandlet	364
10.3	XML-Verarbeitung	366
10.3.1	Verwendung von XML-Objekten	367
10.3.2	Laden und Speichern von XML-Dateien	371
10.3.3	XML-Dokumente in einer Pipeline verarbeiten	378
10.3.4	XPath verwenden	379
10.3.5	Die Import-Clixml- und Export-Clixml-Commandlets	384
10.4	Zusammenfassung	387
11	.NET und WinForms	389
11.1	.NET in PowerShell verwenden	389

11.1.1	.NET-Grundlagen	390
11.1.2	Arbeiten mit Assemblies	390
11.1.3	Auffinden von Typen	393
11.1.4	Instanzen von Typen erzeugen	395
11.1.5	PowerShell ist nicht C# – eine Moralgeschichte	398
11.1.6	Arbeiten mit generischen Typen	402
11.2	PowerShell und das Internet	406
11.2.1	Beispiel: Herunterladen einer Webseite	406
11.2.2	Beispiel: Ein RSS-Feed verarbeiten	407
11.2.3	Beispiel: Einen Webserver mit PowerShell schreiben	409
11.3	PowerShell und grafische Benutzerschnittstellen	416
11.3.1	WinForms-Grundlagen	417
11.3.2	Beispiel: "Mein erstes Formular"	418
11.3.3	Beispiel: Einfacher Dialog	420
11.3.4	Beispiel: Eine WinForms-Bibliothek	423
11.3.5	Beispiel: Ein einfacher Rechner	426
11.3.6	Beispiel: Daten anzeigen	431
11.3.7	Beispiel: Verwendung von GDI+ zur Grafikausgabe	434
11.4	Zusammenfassung	438
12	Windows Objekte: COM und WMI	439
12.1	COM in PowerShell verwenden	440
12.1.1	Windows-Automatisierung mit COM	442
12.1.2	Netzwerkzugriff, Office-Applikationen und Spielzeuge	454
12.1.3	Einsatz des ScriptControl-Objekts	465
12.1.4	Probleme mit COM	467
12.2	WMI und PowerShell	471
12.2.1	Was sind die WMI und was müssen wir beachten?	471
12.2.2	Das Get-WmiObject-Commandlet	472
12.2.3	Der WMI-Objektadapter	473
12.2.4	WMI-Elfmeterschießen – VBScript versus PowerShell	474
12.2.5	Die Shortcuts für WMI-Typen	479
12.2.6	Arbeiten mit WMI-Methoden	481
12.2.7	Arbeiten mit WMI-Events	483
12.2.8	Modifizierte WMI-Objekte zurückgeben	484
12.3	Welches Objektmodell sollte man verwenden?	486
12.4	Zusammenfassung	487

13 Sicherheit, Sicherheit, Sicherheit	489
13.1 Einführung in die Sicherheit	490
13.1.1 Was Sicherheit ist	490
13.1.2 Was Sicherheit nicht ist	490
13.1.3 Sicherheitsempfinden	490
13.2 Sicherheitsmodellierung	492
13.2.1 Einführung in das Modellieren von Bedrohungen	492
13.2.2 Klassifizieren von Bedrohungen mit dem STRIDE-Modell	493
13.2.3 Bedrohungen, Angriffsobjekte und Risiko-Minimierung	494
13.3 Absichern der PowerShell-Umgebung	498
13.3.1 Standardmäßige Sicherheit	498
13.3.2 Befehlsfad verwalten	499
13.3.3 Wahl der Ausführungs-Richtlinie für Skripte	500
13.4 Signieren von Skripten	502
13.4.1 Wie Public Key-Verschlüsselung und Hashing funktionieren	502
13.4.2 Signierende Autoritäten und Zertifikate	503
13.4.3 Erstellen eines selbst-signierten Zertifikats	504
13.4.4 Skripte mit einem Zertifikat signieren	508
13.4.5 Freigabe des Strong Private Key-Schutzes für Ihr Zertifikat	512
13.5 Sichere Skripte schreiben	515
13.5.1 Verwenden der SecureString-Klasse	516
13.5.2 Arbeiten mit Credentials	518
13.5.3 Vermeiden Sie Invoke-Expression	521
13.6 Zusammenfassung	524
A PowerShell im Vergleich mit anderen Sprachen	527
A.1 PowerShell und CMD.EXE	527
A.1.1 Grundlegendes Navigieren und Dateioperationen	527
A.1.2 Variablen und Substitution	529
A.1.3 Befehle ausführen	531
A.1.4 Syntaktische Unterschiede	532
A.1.5 Textsuche: findstr und Select-String	533
A.1.6 Äquivalente für die For-Schleife	533
A.1.7 Batchdateien und Subroutinen	534
A.1.8 Den Prompt einstellen	536
A.1.9 Verwenden von doskey in PowerShell	537
A.1.10 Verwenden von cmd.exe in PowerShell	539

A.2	PowerShell und UNIX-Shells	540
A.2.1	Beispiel: Alle Prozesse anhalten	540
A.2.2	Beispiel: Stoppen einer gefilterten Prozessliste	541
A.2.3	Beispiel: Berechnen der Größe eines Verzeichnisses	541
A.2.4	Beispiel: Arbeiten mit dynamischen Werten	542
A.2.5	Beispiel: Die Lebenszeit eines Prozesses überwachen	543
A.2.6	Beispiel: Prüfen auf Pre-Release-Binärcode	543
A.2.7	Beispiel: Einen String in Großbuchstaben verwandeln	544
A.2.8	Beispiel: Text in einen String einfügen	544
A.3	PowerShell und Pearl	545
A.4	PowerShell und C#	546
A.4.1	Aufruf von Funktionen und Befehlen	546
A.4.2	Methodenaufrufe	547
A.4.3	Rückgabe von Werten	547
A.4.4	Sichtbarkeit von Variablen	548
A.5	PowerShell und VBScript	548

B	Beispiele zum Administrieren	551
B.1	Active Directory-Infos abrufen	551
B.2	Installierte Software anzeigen	552
B.3	Terminal Server-Einstellungen bestimmen	553
B.4	Auflisten aller Hotfixes des aktuellen Computers	554
B.5	Computer mit fehlenden Hotfixes suchen	555
B.6	Ereignisprotokoll verwenden	558
B.6.1	Ein spezifisches EventLog-Objekt ermitteln	559
B.6.2	Das Ereignisprotokoll als Live-Objekt	559
B.6.3	Remote-Zugriff auf Ereignisprotokolle	560
B.6.4	Speichern von Ereignisprotokollen	561
B.6.5	Ereignisse schreiben	562
B.7	Arbeiten mit vorhandenen Dienstprogrammen	563
B.8	Arbeiten mit Active Directory und ADSI	565
B.8.1	Zugriff auf den Active Directory-Service	565
B.8.2	Benutzer hinzufügen	566
B.8.3	Benutzergruppe hinzufügen	567
B.8.4	User-Eigenschaften aktualisieren	569
B.8.5	User entfernen	570
B.9	Zusammenführen von zwei Datenmengen	571

C Die PowerShell-Grammatik	573
C.1 Statement-Liste	573
C.2 Statement	574
C.2.1 Pipeline	574
C.2.2 Das if Statement	575
C.2.3 Das switch-Statement	576
C.2.4 Das foreach-Statement	576
C.2.5 Die for- und while-Statements	577
C.2.6 Die do/while- und do/until-Statements	577
C.2.7 Das trap-Statement	578
C.2.8 Das finally-Statement	578
C.2.9 Anweisungen zur Flusskontrolle	578
C.2.10 Funktionsdeklarationen	579
C.2.11 Parameterdeklarationen	579
C.3 Ausdruck	580
C.4 Wert	581
C.5 Tokenizer-Regeln	582
Index	585