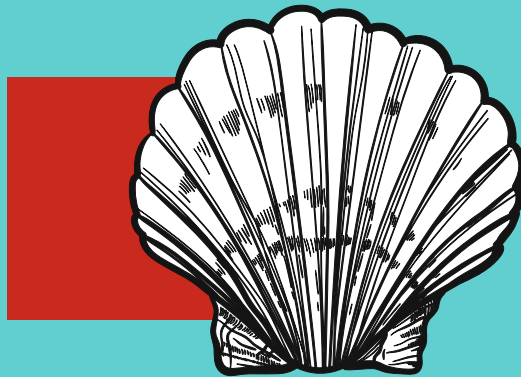


Für
Windows,
Linux und
macOS

Holger SCHWICHTENBERG

PowerShell 7 und Windows PowerShell 5



DAS PRAXISBUCH

6. Auflage



Im Internet: Codebeispiele
und PowerShell-Kurzreferenz

HANSER

www.IT-Visions.de
Dr. Holger Schwichtenberg

Schwichtenberg

PowerShell 7 und Windows PowerShell 5 – das Praxisbuch



Bleiben Sie auf dem Laufenden!

Unser **Computerbuch-Newsletter** informiert Sie monatlich über neue Bücher und Termine. Profitieren Sie auch von Gewinnspielen und exklusiven Leseproben. Gleich anmelden unter:

www.hanser-fachbuch.de/newsletter



Holger Schwichtenberg

PowerShell 7 und Windows PowerShell 5

Das Praxisbuch

6., aktualisierte Auflage

HANSER

Alle in diesem Werk enthaltenen Informationen, Verfahren und Darstellungen wurden nach bestem Wissen zusammengestellt und mit Sorgfalt getestet. Dennoch sind Fehler nicht ganz auszuschließen. Aus diesem Grund sind die im vorliegenden Werk enthaltenen Informationen mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autor und Verlag übernehmen infolgedessen keine juristische Verantwortung und werden keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Art aus der Benutzung dieser Informationen – oder Teilen davon – entsteht. Ebenso wenig übernehmen Autor und Verlag die Gewähr dafür, dass die beschriebenen Verfahren usw. frei von Schutzrechten Dritter sind. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt also auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Die endgültige Entscheidung über die Eignung der Informationen für die vorgesehene Verwendung in einer bestimmten Anwendung liegt in der alleinigen Verantwortung des Nutzers.

Aus Gründen der besseren Lesbarkeit wird auf die gleichzeitige Verwendung der Sprachformen männlich, weiblich und divers (m/w/d) verzichtet. Sämtliche Personenbezeichnungen gelten gleichermaßen für alle Geschlechter.

Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Dieses Werk ist urheberrechtlich geschützt.

Alle Rechte, auch die der Übersetzung, des Nachdruckes und der Vervielfältigung des Buches, oder Teilen daraus, vorbehalten. Wir behalten uns auch eine Nutzung des Werks für Zwecke des Text- und Data Mining nach § 44b UrhG ausdrücklich vor. Kein Teil des Werkes darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren) – auch nicht für Zwecke der Unterrichtsgestaltung – reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

© 2024 Carl Hanser Verlag München, www.hanser-fachbuch.de

Lektorat: Sylvia Hasselbach

Copy editing: Matthias Bloch, Bochum, und Sandra Gottmann, Wasserburg

Umschlagdesign: Marc Müller-Bremer, www.rebranding.de, München

Umschlagrealisation: Max Kostopoulos

Titelmotiv: © shutterstock.com/Irina Kolesnichenko

Satz: Eberl & Koesel Studio, Kempten

Druck und Bindung: Friedrich Pustet GmbH & Co. KG, Regensburg

Printed in Germany

Print-ISBN: 978-3-446-48195-4

E-Book-ISBN: 978-3-446-48196-1

E-Pub-ISBN: 978-3-446-48244-9

Inhalt

Vorwort	XXIV
Über den Autor	XXXII
Teil A: PowerShell-Basiswissen	1
1 Fakten zur PowerShell	3
1.1 Was ist die PowerShell?	3
1.2 Geschichte der PowerShell	4
1.3 Welche Varianten und Versionen der PowerShell gibt es?	6
1.4 Windows PowerShell versus PowerShell Core versus PowerShell 7.x	6
1.5 Motivation zur PowerShell	8
1.6 Betriebssysteme mit vorinstallierter PowerShell	11
1.7 Support der PowerShell	13
1.8 Einflussfaktoren auf die Entwicklung der PowerShell	15
1.9 Anbindung an Klassenbibliotheken	16
1.10 PowerShell versus WSH	17
2 Erste Schritte mit der PowerShell	20
2.1 Windows PowerShell herunterladen und auf anderen Windows-Betriebssystemen installieren	20
2.2 Die Windows PowerShell testen	24
2.3 Woher kommen die PowerShell-Befehle?	33
2.4 PowerShell Community Extensions (PSCX) herunterladen und installieren ...	34
2.5 Den Windows PowerShell-Editor „ISE“ verwenden	41
2.6 PowerShell 7 installieren und testen	45
3 Einzelbefehle der PowerShell	57
3.1 Commandlets	57
3.2 Aliase	70
3.3 Ausdrücke	78
3.4 Externe Befehle (klassische Kommandozeilenbefehle)	79
3.5 Dateinamen	81

4	Hilfefunktionen	82
4.1	Auflisten der verfügbaren Befehle	82
4.2	Praxistipp: Den Standort eines Kommandozeilenbefehls suchen	83
4.3	Anzahl der Befehle	84
4.4	Volltextsuche	86
4.5	Erläuterungen zu den Befehlen	86
4.6	Hilfe zu Parametern	87
4.7	Hilfe mit Show-Command	89
4.8	Hilfefenster	90
4.9	Allgemeine Hilfetexte	92
4.10	Aktualisieren der Hilfedateien	92
4.11	Online-Hilfe	94
4.12	Fehlende Hilfetexte	95
4.13	Dokumentation der .NET-Klassen	96
5	Objektorientiertes Pipelining	98
5.1	Befehlsübersicht	98
5.2	Pipeline-Operator	99
5.3	.NET-Objekte in der Pipeline	100
5.4	Pipeline Processor	101
5.5	Pipelining von Parametern	103
5.6	Pipelining von klassischen Befehlen	105
5.7	Zeilenumbrüche in Pipelines	107
5.8	Schleifen	108
5.9	Zugriff auf einzelne Objekte aus einer Menge	111
5.10	Zugriff auf einzelne Werte in einem Objekt	112
5.11	Methoden ausführen	114
5.12	Analyse des Pipeline-Inhalts	116
5.13	Filtern	131
5.14	Zusammenfassung von Pipeline-Inhalten	136
5.15	„Kastrierung“ von Objekten in der Pipeline	136
5.16	Sortieren	137
5.17	Duplikate entfernen	138
5.18	Gruppierung	139
5.19	Objekte verbinden mit Join-String	145
5.20	Berechnungen	146
5.21	Zwischenschritte in der Pipeline mit Variablen	146
5.22	Verzweigungen in der Pipeline	147
5.23	Vergleiche zwischen Objekten	149
5.24	Weitere Praxislösungen	150

6	PowerShell-Skripte	152
6.1	Skriptdateien	152
6.2	Start eines Skripts	154
6.3	Aliase für Skripte verwenden	155
6.4	Parameter für Skripte	156
6.5	Skripte dauerhaft einbinden (Dot Sourcing)	157
6.6	Das aktuelle Skriptverzeichnis	158
6.7	Sicherheitsfunktionen für PowerShell-Skripte	158
6.8	Skripte mit vollen Rechten (Elevation)	160
6.9	Blockierte PowerShell-Skripte	161
6.10	PowerShell-Skripte im Kontextmenü des Windows Explorers	162
6.11	Anforderungsdefinitionen von Skripten	164
6.12	Skripte anhalten	165
6.13	Versionierung und Versionsverwaltung von Skripten	165
7	PowerShell-Skriptsprache	168
7.1	Hilfe zur PowerShell-Skriptsprache	168
7.2	Befehlstrennung	168
7.3	Kommentare	169
7.4	Variablen	170
7.5	Variablenbedingungen	182
7.6	Zahlen	183
7.7	Zeichenketten (Strings)	187
7.8	Reguläre Ausdrücke	197
7.9	Datum und Uhrzeit	203
7.10	Objekte	204
7.11	Arrays	205
7.12	ArrayList	208
7.13	Assoziative Arrays (Hash-Tabellen)	209
7.14	Operatoren	210
7.15	Überblick über die Kontrollkonstrukte	214
7.16	Bedingungen	219
7.17	Unterroutinen (Prozedur/Funktionen)	222
7.18	Eingebaute Funktionen	228
7.19	Fehlerausgabe	229
7.20	Fehlerbehandlung	231
7.21	Laufzeitfehler erzeugen	243
7.22	Objektorientiertes Programmieren mit Klassen	243

8	Ausgaben	247
8.1	Ausgabe-Commandlets	247
8.2	Benutzerdefinierte Tabellenformatierung	250
8.3	Benutzerdefinierte Listenausgabe	252
8.4	Mehrspaltige Ausgabe	252
8.5	Out-GridView	253
8.6	Standardausgabe	255
8.7	Einschränkung der Ausgabe	257
8.8	Seitenweise Ausgabe	258
8.9	Ausgabe einzelner Werte	259
8.10	Details zum Ausgabeoperator	262
8.11	Ausgabe von Methodenergebnissen und Unterobjekten in Pipelines	265
8.12	Ausgabe von Methodenergebnissen und Unterobjekten in Zeichenketten	266
8.13	Unterdrückung der Ausgabe	267
8.14	Ausgaben an Drucker	267
8.15	Ausgaben in Dateien	268
8.16	Umleitungen (Redirection)	268
8.17	Fortschrittsanzeige	269
8.18	Sprachausgabe	269
9	Das PowerShell-Navigationsmodell (PowerShell Provider)	271
9.1	Einführungsbeispiel: Navigation in der Registrierungsdatenbank	271
9.2	Provider und Laufwerke	272
9.3	Navigationsbefehle	274
9.4	Pfadangaben	275
9.5	Beispiel	277
9.6	Eigene Laufwerke definieren	278
10	Fernausführung (Remoting)	279
10.1	RPC-Fernabfrage ohne WS-Management	280
10.2	Anforderungen an PowerShell Remoting	281
10.3	Rechte für PowerShell-Remoting	282
10.4	Einrichten von PowerShell Remoting	282
10.5	Überblick über die Fernausführungs-Commandlets	285
10.6	Interaktive Fernverbindungen im Telnet-Stil	285
10.7	Fernausführung von Befehlen	287
10.8	Parameterübergabe an die Fernausführung	291
10.9	Fernausführung von Skripten	292
10.10	Ausführung auf mehreren Computern	293
10.11	Sitzungen	294
10.12	Implizites Remoting	299

10.13	Zugriff auf entfernte Computer außerhalb der eigenen Domäne	300
10.14	Verwaltung des WS-Management-Dienstes	303
10.15	PowerShell Direct für Hyper-V	304
10.16	Praxislösung zu PowerShell Direct	306
11	PowerShell-Werkzeuge	309
11.1	PowerShell-Standardkonsole	309
11.2	Windows Terminal	324
11.3	Erweiterung der Konsolen	329
11.4	PowerShell Integrated Scripting Environment (ISE)	331
11.5	PowerShell Script Analyzer	342
11.6	PowerShell Analyzer	347
11.7	PowerShell Tools for Visual Studio	348
11.8	PowerShell Pro Tools for Visual Studio	350
11.9	Visual Studio Developer PowerShell	350
11.10	NuGet Package Manager Console (PMC)	353
11.11	Visual Studio Code mit PowerShell-Erweiterung	354
11.12	PowerShell-Erweiterungen für andere Editoren	356
11.13	PowerShell Web Access (PSWA)	357
11.14	Azure Cloud Shell	362
11.15	ISE Steroids	362
11.16	PowerShellPlus	363
11.17	PoshConsole	366
11.18	PowerGUI	367
11.19	PrimalScript	367
11.20	CIM Explorer for PowerShell ISE	369
12	Windows PowerShell Core 5.1 in Windows Nano Server	371
12.1	Installation	371
12.2	PowerShell-Skriptsprache	371
12.3	Werkzeuge	371
12.4	Fehlende Funktionen	372
13	PowerShell 7 für Windows, Linux und macOS	373
13.1	Motivation für den Einsatz der PowerShell 7 auf Linux und macOS	373
13.2	Basis der PowerShell 7	374
13.3	Identifizierung der PowerShell 7	375
13.4	Funktionsumfang der PowerShell 7	375
13.5	Entfallene Befehle in PowerShell 7	378
13.6	Erweiterungsmodule nutzen in PowerShell 7	384
13.7	Geänderte Funktionen in PowerShell 7	389

13.8	Neue Funktionen der PowerShell 7	391
13.9	PowerShell 7-Konsole	394
13.10	Praxislösung: Fallunterscheidung für PowerShell-Varianten	395
13.11	VSCoDe-PowerShell als Editor für PowerShell 7	396
13.12	Verwendung von PowerShell 7 auf Linux und macOS	400
13.13	PowerShell-Remoting via SSH	406
13.14	Performance-Vorteile der PowerShell 7	409
13.15	Dokumentation zur PowerShell 7	410
13.16	Quellcode zur PowerShell 7	412

Teil B: PowerShell-Aufbauwissen **415**

14 Verwendung von .NET-Klassen **417**

14.1	.NET versus .NET Core	417
14.2	Ermitteln der verwendeten .NET-Version	418
14.3	.NET-Bibliotheken	419
14.4	Microsoft Docs	421
14.5	Überblick über die Verwendung von .NET-Klassen	422
14.6	Erzeugen von Instanzen	422
14.7	Parameterbehaftete Konstruktoren	424
14.8	Initialisierung von Objekten	425
14.9	Nutzung von Attributen und Methoden	426
14.10	Statische Mitglieder in .NET-Klassen und statische .NET-Klassen	428
14.11	Generische Klassen nutzen	431
14.12	Zugriff auf bestehende Objekte	433
14.13	Laden von Assemblies	433
14.14	Liste der geladenen Assemblies	435
14.15	Verwenden von NuGet-Assemblies	436
14.16	Objektanalyse	438
14.17	Aufzählungstypen (Auflistungen/Enumerationen)	439

15 Verwendung von COM-Klassen **443**

15.1	Unterschiede zwischen COM und .NET	443
15.2	Erzeugen von COM-Instanzen	444
15.3	Abruf der Metadaten	444
15.4	Nutzung von Attributen und Methoden	445
15.5	Liste aller COM-Klassen	446
15.6	Holen bestehender COM-Instanzen	447
15.7	Distributed COM (DCOM)	447

16	Zugriff auf die Windows Management Instrumentation (WMI) ..	449
16.1	Einführung in WMI	449
16.2	WMI in der PowerShell	476
16.3	Open Management Infrastructure (OMI)	478
16.4	Abruf von WMI-Objektmengen	478
16.5	Fernzugriffe	479
16.6	Filtern und Abfragen	480
16.7	Liste aller WMI-Klassen	483
16.8	Hintergrundwissen: WMI-Klassenprojektion mit dem PowerShell-WMI-Objektadapter	484
16.9	Beschränkung der Ausgabeliste bei WMI-Objekten	488
16.10	Zugriff auf einzelne Mitglieder von WMI-Klassen	490
16.11	Werte setzen in WMI-Objekten	490
16.12	Umgang mit WMI-Datumsangaben	492
16.13	Methodenaufrufe	493
16.14	Neue WMI-Instanzen erzeugen	494
16.15	Instanzen entfernen	495
16.16	Commandlet Definition XML-Datei (CDXML)	495
17	Dynamische Objekte	499
17.1	Erweitern bestehender Objekte	499
17.2	Komplett dynamische Objekte	501
18	Einbinden von C# und Visual Basic .NET	503
19	Win32-API-Aufrufe	505
20	Benutzereingaben	508
20.1	Read-Host	508
20.2	Benutzerauswahl	509
20.3	Grafischer Eingabedialog	510
20.4	Dialogfenster	511
20.5	Authentifizierungsdiallog	511
20.6	Zwischenablage (Clipboard)	513
21	Fehlersuche	514
21.1	Detailinformationen	514
21.2	Einzelschrittmodus	515
21.3	Zeitmessung	516
21.4	Ablaufverfolgung (Tracing)	517
21.5	Erweiterte Protokollierung aktivieren	519
21.6	Script-Debugging in der ISE	520
21.7	Kommandozeilenbasiertes Script-Debugging	520

22	Transaktionen	522
22.1	Commandlets für Transaktionen	522
22.2	Start und Ende einer Transaktion	523
22.3	Zurücksetzen der Transaktion	524
22.4	Mehrere Transaktionen	525
23	Standardeinstellungen ändern mit Profilskripten	526
23.1	Profilpfade	526
23.2	Ausführungsreihenfolge	528
23.3	Beispiel für eine Profildatei	528
23.4	Starten der PowerShell ohne Profilskripte	530
24	Digitale Signaturen für PowerShell-Skripte	531
24.1	Zertifikat erstellen	531
24.2	Skripte signieren	533
24.3	Verwenden signierter Skripte	535
24.4	Mögliche Fehlerquellen	535
25	Hintergrundaufträge („Jobs“)	536
25.1	Voraussetzungen	536
25.2	Architektur	536
25.3	Starten eines Hintergrundauftrags	537
25.4	Hintergrundaufträge abfragen	538
25.5	Warten auf einen Hintergrundauftrag	539
25.6	Abbrechen und Löschen von Aufträgen	539
25.7	Analyse von Fehlermeldungen	539
25.8	Fernausführung von Hintergrundaufträgen	540
25.9	Praxislösung: Einen Job auf mehreren Computern starten	540
26	Geplante Aufgaben und zeitgesteuerte Jobs	542
26.1	Geplante Aufgaben (Scheduled Tasks)	542
26.2	Zeitgesteuerte Jobs	546
27	PowerShell-Workflows	552
27.1	Ein erstes Beispiel	552
27.2	Unterschiede zu einer Function bzw. einem Skript	556
27.3	Einschränkungen bei Workflows	557
27.4	Workflows in der Praxis	558
27.5	Workflows in Visual Studio erstellen	566

28	Ereignissystem	584
28.1	WMI-Ereignisse	584
28.2	WMI-Ereignisabfragen	584
28.3	WMI-Ereignisse seit PowerShell 1.0	586
28.4	Registrieren von WMI Ereignisquellen seit PowerShell 2.0	587
28.5	Auslesen der Ereignisliste	588
28.6	Reagieren auf Ereignisse	590
28.7	WMI-Ereignisse seit PowerShell-Version 3.0	592
28.8	Registrieren von .NET-Ereignissen	592
28.9	Erzeugen von Ereignissen	593
29	Datenbereiche und Datendateien	595
29.1	Datenbereiche	595
29.2	Datendateien	597
29.3	Mehrsprachigkeit/Lokalisierung	598
30	Desired State Configuration (DSC)	601
30.1	Grundprinzipien	602
30.2	DSC für PowerShell 7	602
30.3	Ressourcen	603
30.4	Verfügbare DSC-Ressourcen	604
30.5	Eigenschaften einer Ressource	607
30.6	Aufbau eines DSC-Dokuments	607
30.7	Commandlets für die Arbeit mit DSC	608
30.8	Ein erstes DSC-Beispiel	608
30.9	Kompilieren und Anwendung eines DSC-Dokuments	609
30.10	Variablen in DSC-Dateien	611
30.11	Parameter für DSC-Dateien	612
30.12	Konfigurationsdaten	613
30.13	Entfernen einer DSC-Konfiguration	616
30.14	DSC Pull Server	619
30.15	DSC-Praxislösung 1: IIS installieren	626
30.16	DSC-Praxislösung 2: Software installieren	628
30.17	DSC-Praxislösung 3: Software deinstallieren	630
30.18	Realisierung einer DSC-Ressource	631
30.19	Weitere Möglichkeiten	631
31	PowerShell-Snap-Ins	632
31.1	Einbinden von Snap-Ins	632
31.2	Liste der Commandlets	636

32 PowerShell-Module	637
32.1 Überblick über die Commandlets	637
32.2 Modularchitektur	638
32.3 Aufbau eines Moduls	639
32.4 Module aus dem Netz herunterladen und installieren mit PowerShellGet	640
32.5 Module manuell installieren	646
32.6 Doppeldeutige Namen	646
32.7 Auflisten der verfügbaren Module	648
32.8 Importieren von Modulen	649
32.9 Entfernen von Modulen	652
33 Ausgewählte PowerShell-Erweiterungen	653
33.1 PowerShell-Module in Windows 8.0 und Windows Server 2012	654
33.2 PowerShell-Module in Windows 8.1 und Windows Server 2012 R2	656
33.3 PowerShell-Module in Windows 10 und Windows Server 2019	658
33.4 PowerShell Community Extensions (PSCX)	662
33.5 PowerShellPack	666
33.6 www.IT-Visions.de: PowerShell Extensions	668
33.7 Quest Management Shell for Active Directory	668
33.8 Microsoft Exchange Server	670
33.9 System Center Virtual Machine Manager	671
33.10 PowerShell Management Library for Hyper-V (pshyperv)	671
33.11 PowerShell Configurator (PSConfig)	672
34 Delegierte Administration/Just Enough Administration (JEA) ..	674
34.1 JEA-Konzept	674
34.2 PowerShell-Sitzungskonfiguration erstellen	674
34.3 Sitzungskonfiguration nutzen	678
34.4 Delegierte Administration per Webseite	679
35 Tipps und Tricks zur PowerShell	680
35.1 Alle Anzeigen löschen	680
35.2 Befehlsgeschichte	680
35.3 System- und Hostinformationen	681
35.4 Anpassen der Eingabeaufforderung (Prompt)	682
35.5 PowerShell-Befehle aus anderen Anwendungen heraus starten	683
35.6 ISE erweitern	684
35.7 PowerShell für Gruppenrichtlinienskripte	685
35.8 Einblicke in die Interna der Pipeline-Verarbeitung	688

Teil C: PowerShell im Praxiseinsatz	689
36 Dateisystem	691
36.1 Laufwerke	692
36.2 Ordnerinhalte	697
36.3 Dateieigenschaften verändern	704
36.4 Eigenschaften ausführbarer Dateien	705
36.5 Kurznamen	707
36.6 Lange Pfade	707
36.7 Dateisystemoperationen	708
36.8 Praxislösung: Dateien umorganisieren	708
36.9 Praxislösung: Zufällige Dateisystemstruktur erzeugen	710
36.10 Praxislösung: Leere Ordner löschen	711
36.11 Praxislösung: Geschwindigkeitsmessung des Dateisystems (beim Kopieren von Dateien)	713
36.12 Einsatz von Robocopy in der PowerShell	714
36.13 NTFS-Komprimierung	717
36.14 Dateisystemkataloge	718
36.15 Papierkorb leeren	718
36.16 Dateieigenschaften lesen	719
36.17 Praxislösung: Fotos nach Aufnahmedatum sortieren	719
36.18 Datei-Hash	720
36.19 Finden von Duplikaten	721
36.20 Verknüpfungen im Dateisystem	723
36.21 Komprimierung	728
36.22 Dateisystemfreigaben	732
36.23 Überwachung des Dateisystems	743
36.24 Dateiversionsverlauf	744
36.25 Windows Explorer öffnen	745
36.26 Windows Server Backup	745
37 Festplattenverschlüsselung mit BitLocker	747
37.1 Übersicht über das BitLocker-Modul	748
37.2 Verschlüsseln eines Laufwerks	749
38 Dokumente	750
38.1 Textdateien	750
38.2 CSV-Dateien	752
38.3 Analysieren von Textdateien	755
38.4 INI-Dateien	759
38.5 XML-Dateien	759

38.6	HTML- und Markdown-Dateien	771
38.7	JSON-Dateien	774
38.8	Binärdateien	785
38.9	Praxislösung: Grafikdateien verändern	786
38.10	Praxislösung: Drucken vieler Dateien	787
39	Microsoft Office	788
39.1	Allgemeine Informationen zur Office-Automatisierung per PowerShell	788
39.2	Praxislösung: Terminserien aus Textdateien anlegen in Outlook	789
39.3	Praxislösung: Outlook-Termine anhand von Suchkriterien löschen	791
39.4	Praxislösung: Grafiken aus einem Word-Dokument (DOCX) extrahieren	792
40	Datenbanken	795
40.1	ADO.NET-Grundlagen	795
40.2	Beispieldatenbank	801
40.3	Datenzugriff mit den Bordmitteln der PowerShell	802
40.4	Hilfsroutinen für den Datenbankzugriff (DBUtil.ps1)	815
40.5	Datenzugriff mit den PowerShell-Erweiterungen	818
40.6	Datenbankzugriff mit SQLPS	822
40.7	Datenbankzugriff mit SQLPSX	822
41	Microsoft-SQL-Server-Administration	823
41.1	PowerShell-Integration im SQL Server Management Studio	824
41.2	SQL-Server-Laufwerk „SQLSERVER:“	825
41.3	Die SQLPS-Commandlets	828
41.4	Die SQL Server Management Objects (SMO)	830
41.5	SQLPSX	833
41.6	Microsoft-SQL-Server-Administration mit der PowerShell in der Praxis	840
42	ODBC-Datenquellen	846
42.1	ODBC-Treiber und -Datenquellen auflisten	847
42.2	Anlegen einer ODBC-Datenquelle	848
42.3	Zugriff auf eine ODBC-Datenquelle	849
43	Registrierungsdatenbank (Registry)	851
43.1	Schlüssel auslesen	851
43.2	Schlüssel anlegen und löschen	852
43.3	Laufwerke definieren	852
43.4	Werte anlegen und löschen	853
43.5	Werte auslesen	854
43.6	Praxislösung: Windows-Explorer-Einstellungen	855
43.7	Praxislösung: Massenanlegen von Registry-Schlüsseln	855

44	Computer- und Betriebssystemverwaltung	857
44.1	Computerinformationen	857
44.2	Versionsnummer des Betriebssystems	859
44.3	Zeitdauer seit dem letzten Start des Betriebssystems	859
44.4	BIOS- und Startinformationen	860
44.5	Windows-Produktaktivierung	861
44.6	Umgebungsvariablen	861
44.7	Schriftarten	865
44.8	Computername und Domäne	865
44.9	Herunterfahren und Neustarten	866
44.10	Windows Updates installieren	867
44.11	Wiederherstellungspunkte verwalten	871
45	Windows Defender	872
46	Hardwareverwaltung	873
46.1	Hardwarebausteine	873
46.2	Plug-and-Play-Geräte	875
46.3	Druckerverwaltung (ältere Betriebssysteme)	875
46.4	Druckerverwaltung (seit Windows 8 und Windows Server 2012)	877
47	Softwareverwaltung	879
47.1	Softwareinventarisierung	879
47.2	Installation von Anwendungen	882
47.3	Deinstallation von Anwendungen	883
47.4	Praxislösung: Installationstest	884
47.5	Praxislösung: Installierte .NET SDKs aufräumen	885
47.6	Windows 10 Apps verwalten	889
47.7	Installationen mit PowerShell Package Management („OneGet“)	892
47.8	Versionsnummer ermitteln	895
47.9	Servermanager	896
47.10	Windows-Features installieren auf Windows-Clientbetriebssystemen	907
47.11	Praxislösung: IIS-Installation	909
47.12	Softwareeinschränkungen mit dem PowerShell-Modul „AppLocker“	911
48	Prozessverwaltung	917
48.1	Prozesse auflisten	917
48.2	Prozesse starten	918
48.3	Prozesse mit vollen Administratorrechten starten	919
48.4	Prozesse unter einem anderen Benutzerkonto starten	920
48.5	Prozesse beenden	921
48.6	Warten auf das Beenden einer Anwendung	922

49	Windows-Systemdienste	923
49.1	Dienste auflisten	923
49.2	Dienstzustand ändern	926
49.3	Diensteigenschaften ändern	926
49.4	Dienste hinzufügen	927
49.5	Dienste entfernen	928
50	Netzwerk	929
50.1	Netzwerkkonfiguration	929
50.2	DNS-Client-Konfiguration	934
50.3	DNS-Namensauflösung	938
50.4	Erreichbarkeit prüfen (Ping)	939
50.5	Windows Firewall	940
50.6	Remote Desktop (RDP) einrichten	947
50.7	E-Mails senden (SMTP)	948
50.8	Auseinandernehmen von E-Mail-Adressen	949
50.9	Abruf von Daten von einem HTTP-Server	949
50.10	Praxislösung: Linkprüfer für eine Website	956
50.11	Aufrufe von SOAP-Webdiensten	959
50.12	Aufruf von REST-Diensten	962
50.13	File Transfer Protocol (FTP)	964
50.14	Hintergrunddatentransfer mit BITS	965
51	Ereignisprotokolle (Event Log)	969
51.1	Protokolleinträge auslesen	969
51.2	Ereignisprotokolle erzeugen	971
51.3	Protokolleinträge erzeugen	971
51.4	Protokollgröße festlegen	971
51.5	Protokolleinträge löschen	971
52	Leistungsdaten (Performance Counter)	972
52.1	Zugriff auf Leistungsindikatoren über WMI	972
52.2	Get-Counter	973
53	Sicherheitseinstellungen	975
53.1	Aktueller Benutzer	975
53.2	Grundlagen	976
53.3	Zugriffsrechtelisten auslesen	981
53.4	Einzelne Rechteinträge auslesen	982
53.5	Besitzer auslesen	984
53.6	Benutzer und SID	984

53.7	Hinzufügen eines Rechteeintrags zu einer Zugriffsrechteliste	988
53.8	Entfernen eines Rechteeintrags aus einer Zugriffsrechteliste	990
53.9	Zugriffsrechteliste übertragen	992
53.10	Zugriffsrechteliste über SDDL setzen	993
53.11	Zertifikate verwalten	994
54	Optimierungen und Problemlösungen	997
54.1	PowerShell-Modul „TroubleshootingPack“	997
54.2	PowerShell-Modul „Best Practices“	1001
55	Active Directory	1003
55.1	Benutzer- und Gruppenverwaltung mit WMI	1005
55.2	Einführung in System.DirectoryServices	1005
55.3	Basiseigenschaften	1017
55.4	Benutzer- und Gruppenverwaltung im Active Directory	1019
55.5	Verwaltung der Organisationseinheiten	1027
55.6	Suche im Active Directory	1028
55.7	Navigation im Active Directory mit den PowerShell Extensions	1035
55.8	Verwendung der Active-Directory-Erweiterungen von <i>www.IT-Visions.de</i>	1036
55.9	PowerShell-Modul „Active Directory“ (ADPowerShell)	1038
55.10	PowerShell-Modul „ADDSDeployment“	1067
55.11	Informationen über die Active Directory-Struktur	1070
56	Gruppenrichtlinien	1073
56.1	Verwaltung der Gruppenrichtlinien	1073
56.2	Verknüpfung der Gruppenrichtlinien	1075
56.3	Gruppenrichtlinienberichte	1077
56.4	Gruppenrichtlinienvererbung	1079
56.5	Weitere Möglichkeiten	1080
57	Lokale Benutzer und Gruppen	1081
57.1	Modul „Microsoft.PowerShell.LocalAccounts“	1081
57.2	Lokale Benutzerverwaltung in älteren PowerShell-Versionen	1082
58	Microsoft Exchange Server	1085
58.1	Daten abrufen	1085
58.2	Postfächer verwalten	1086
58.3	Öffentliche Ordner verwalten	1087
59	Internet Information Services (IIS)	1088
59.1	Überblick	1088
59.2	Navigationsprovider	1090

59.3	Anlegen von Websites	1092
59.4	Praxislösung: Massenanlegen von Websites	1093
59.5	Ändern von Website-Eigenschaften	1095
59.6	Anwendungspool anlegen	1096
59.7	Virtuelle Verzeichnisse und IIS-Anwendungen	1097
59.8	Website-Zustand ändern	1097
59.9	Anwendungspools starten und stoppen	1098
59.10	Löschen von Websites	1098
60	Virtuelle Systeme mit Hyper-V	1099
60.1	Das Hyper-V-Modul von Microsoft	1100
60.2	Die ersten Schritte mit dem Hyper-V-Modul	1102
60.3	Virtuelle Maschinen anlegen	1106
60.4	Umgang mit virtuellen Festplatten	1112
60.5	Konfiguration virtueller Maschinen	1115
60.6	Praxislösungen: Ressourcennutzung überwachen	1119
60.7	Dateien kopieren in virtuelle Systeme	1121
60.8	PowerShell Management Library for Hyper-V (für ältere Betriebssysteme)	1122
61	Windows Nano Server	1125
61.1	Das Konzept von Nano Server	1125
61.2	Einschränkungen von Nano Server	1127
61.3	Varianten des Nano Servers	1129
61.4	Installation eines Nano Servers	1129
61.5	Docker-Image	1130
61.6	Fernverwaltung mit PowerShell	1131
61.7	Windows Update auf einem Nano Server	1133
61.8	Nachträgliche Paketinstallation	1133
61.9	Abgespeckter IIS unter Nano Server	1135
61.10	Nano-Serververwaltung aus der Cloud heraus	1136
62	Docker-Container	1137
62.1	Container-Varianten für Windows	1137
62.2	Docker-Installation auf aktuellem Windows 10 und Windows 11	1141
62.3	Docker-Installation auf älteren Windows 10-Clients	1149
62.4	Docker-Installation auf Windows Server	1151
62.5	Docker PowerShell installieren	1153
62.6	Docker-Basiswissen	1154
62.7	Container mit modernem .NET	1157
62.8	Container mit IIS-Webserver und klassischem ASP.NET	1166
62.9	Container mit Linux und PowerShell 7	1175

62.10	Container mit Linux und Microsoft SQL Server	1177
62.11	Docker-Container mit Visual Studio	1179
62.12	Weitere Container-Befehle	1184
63	Microsoft Azure	1190
63.1	Azure-Konzepte	1190
63.2	Kommandozeilenwerkzeuge für die Azure-Verwaltung	1192
63.3	Benutzeranmeldung und Informationsabfrage	1195
63.4	Azure Ressourcen-Gruppen	1196
63.5	Azure Web-Apps	1196
63.6	Azure SQL Server	1198
63.7	Azure Kubernetes Services (AKS)	1199
63.8	Azure DevOps (ADO)	1223
64	Grafische Benutzeroberflächen (GUI)	1244
64.1	Einfache Nachfragedialoge	1244
64.2	Einfache Eingabe mit Inputbox	1245
64.3	Komplexere Eingabemasken	1246
64.4	Universelle Objektdarstellung	1248
64.5	WPF PowerShell Kit (WPK)	1249
64.6	Direkte Verwendung von WPF	1257
Teil D: Profiwissen – Erweitern der PowerShell		1259
65	Unit Tests mit Pester	1261
65.1	Einführung in das Konzept des Unit Testing	1261
65.2	Pester installieren	1262
65.3	Befehle in Pester	1262
65.4	Testen einer PowerShell-Funktion	1263
65.5	Testgenerierung	1264
65.6	Tests starten	1264
65.7	Prüf-Operationen	1266
65.8	Mock-Objekte	1266
65.9	Test von Dateisystemoperationen	1267
66	Entwicklung von Commandlets in der PowerShell- Skriptsprache	1269
66.1	Aufbau eines skriptbasierten Commandlets	1269
66.2	Verwendung per Dot Sourcing	1271
66.3	Parameterfestlegung	1272
66.4	Fortgeschrittene Funktion (Advanced Function)	1278

66.5	Mehrere Parameter und Parametersätze	1281
66.6	Unterstützung für Sicherheitsabfragen (-whatif und -confirm)	1283
66.7	Kaufmännisches Beispiel: Test-CustomerID	1285
66.8	Erweitern bestehender Commandlets durch Proxy-Commandlets	1288
66.9	Dokumentation	1294
67	Entwicklung eigener Commandlets mit C#	1298
67.1	Technische Voraussetzungen	1299
67.2	Grundkonzept der .NET-basierten Commandlets	1301
67.3	Schrittweise Erstellung eines minimalen Commandlets	1303
67.4	Erstellung eines Commandlets mit einem Rückgabeobjekt	1311
67.5	Erstellung eines Commandlets mit mehreren Rückgabeobjekten	1313
67.6	Erstellen eines Commandlets mit Parametern	1317
67.7	Verarbeiten von Pipeline-Eingaben	1319
67.8	Verkettung von Commandlets	1322
67.9	Fehlersuche in Commandlets	1326
67.10	Statusinformationen	1329
67.11	Unterstützung für Sicherheitsabfragen (-whatif und -confirm)	1334
67.12	Festlegung der Hilfeinformationen	1336
67.13	Erstellung von Commandlets für den Zugriff auf eine Geschäftsanwendung ...	1341
67.14	Konventionen für Commandlets	1342
67.15	Weitere Möglichkeiten	1344
68	PowerShell-Module erstellen	1345
68.1	Erstellen eines Skriptmoduls	1345
68.2	Praxislösung: Umwandlung einer Skriptdatei in ein Modul	1347
68.3	Erstellen eines Moduls mit Binärdateien	1347
68.4	Erstellen eines Moduls mit Manifest	1348
68.5	Erstellung eines Manifest-Moduls mit Visual Studio	1355
69	Hosting der PowerShell	1357
69.1	Voraussetzungen für das Hosting	1358
69.2	Hosting mit PSHost	1359
69.3	Vereinfachtes Hosting seit PowerShell 2.0	1362
Anhang A: Crashkurs Objektorientierung		1365
Anhang B: Crashkurs .NET		1373
B.1	Was ist das .NET Framework?	1376
B.2	Was ist .NET Core/.NET?	1377
B.3	Eigenschaften von .NET	1378

B.4	.NET-Klassen	1379
B.5	Namensgebung von .NET-Klassen (Namensräume)	1379
B.6	Namensräume und Softwarekomponenten	1381
B.7	Bestandteile einer .NET-Klasse	1382
B.8	Vererbung	1383
B.9	Schnittstellen	1383
Anhang C: Weitere Informationen im Internet		1384
Anhang D: Abkürzungsverzeichnis		1385
Stichwortverzeichnis		1409

Vorwort

Liebe Leserin, lieber Leser,

willkommen zur aktuellen Auflage meines PowerShell-Buchs! Es handelt sich hierbei um die sechste Auflage des Windows PowerShell 5-Buches und die zehnte Auflage des PowerShell-Buches insgesamt, das erstmalig 2007 bei Addison-Wesley erschienen ist.

Was ist das Thema dieses Buchs?

Das vor Ihnen liegende Fachbuch behandelt die Windows PowerShell in der Version 5.1 sowie die plattformneutrale PowerShell 7.4 von Microsoft wie auch ergänzende Werkzeuge von Microsoft und Drittanbietern (z. B. PowerShell Community Extensions).

Das Buch ist aber auch für Sie geeignet, wenn Sie noch eine ältere Version der PowerShell einsetzen. Welche Funktionen neu hinzugekommen sind, wird jeweils in diesem Buch erwähnt.

Wer bin ich?

Mein Name ist Holger Schwichtenberg, ich bin derzeit 51 Jahre alt und habe im Fachgebiet Wirtschaftsinformatik promoviert. Ich lebe (in Essen, im Herzen des Ruhrgebiets) davon, dass mein Team und ich im Rahmen unserer Firma www.IT-Visions.de anderen Unternehmen bei der Entwicklung von .NET-, Web- und PowerShell-Anwendungen beratend und schulend zur Seite stehen. Zudem entwickeln wir Software im Auftrag von Kunden in zahlreichen Branchen.

Es ist nur ein Hobby, IT-Fachbücher zu schreiben, denn damit kann man als Autor kaum Geld verdienen. Dieses Buch ist, unter Mitzählung aller nennenswerten Neuauflagen, das 92. Buch, das ich allein oder mit Co-Autoren geschrieben habe. Meine weiteren Hobbys sind Mountain Biking, Fotografie und Reisen.

Natürlich verstehe ich das Bücherschreiben auch als Werbung für die Arbeit unserer Unternehmen, und wir hoffen, dass der ein oder andere von Ihnen uns beauftragen wird, Ihre Organisation durch Beratung, Schulung und Auftragsentwicklung zu unterstützen.

Wer sind Sie?

Damit Sie den optimalen Nutzen aus diesem Buch ziehen können, möchte ich – so genau es mir möglich ist – beschreiben, an wen sich dieses Buch richtet. Hierzu habe ich einen Fragebogen ausgearbeitet, mit dem Sie schnell erkennen können, ob das Buch für Sie geeignet ist.

Sind Sie Systemadministrator in einem Windows-Netzwerk?	<input type="radio"/> Ja	<input type="radio"/> Nein
Laufen die für Sie relevanten Computer mit den von PowerShell unterstützten Betriebssystemen? (Windows 7/8/8.1/10/11, Windows Server 2008/2008 R2/2012/2012 R2/2016/2019/2022, macOS, Linux)	<input type="radio"/> Ja	<input type="radio"/> Nein
Sie besitzen zumindest rudimentäre Grundkenntnisse im Bereich des (objektorientierten) Programmierens?	<input type="radio"/> Ja	<input type="radio"/> Nein
Wünschen Sie einen kompakten Überblick über die Architektur, Konzepte und Anwendungsfälle der PowerShell?	<input type="radio"/> Ja	<input type="radio"/> Nein
Sie können auf Schritt-für-Schritt-Anleitungen verzichten?	<input type="radio"/> Ja	<input type="radio"/> Nein
Sie können auf formale Syntaxbeschreibungen verzichten und lernen lieber an aussagekräftigen Beispielen?	<input type="radio"/> Ja	<input type="radio"/> Nein
Sie erwarten nicht, dass in diesem Buch alle Möglichkeiten der PowerShell detailliert beschrieben werden?	<input type="radio"/> Ja	<input type="radio"/> Nein
Sind Sie, nachdem Sie ein Grundverständnis durch dieses Buch gewonnen haben, bereit, Detailfragen in der Dokumentation der PowerShell, von .NET und WMI nachzuschlagen, da das Buch auf rund 1400 Seiten nicht alle Details erläutern, sondern – in dem Sinn „Hilfe zur Selbsthilfe“ – nur ausgewählte Aspekte darstellen kann, anhand deren Sie dann Ihre eigenen Lösungen für Ihre spezifischen Szenarien entwickeln?	<input type="radio"/> Ja	<input type="radio"/> Nein

Wenn Sie alle obigen Fragen mit „Ja“ beantwortet haben, ist dieses Fachbuch richtig für Sie. In anderen Fällen sollten Sie sich erst mit einführender Literatur beschäftigen.

Was ist neu in diesem Buch?

Die vorliegende Auflage wurde auf PowerShell Version 7.4 aktualisiert und bestehende Inhalte des Buchs an vielen Stellen optimiert. Zudem wurde das Feedback einiger Leser eingearbeitet, um Beispiele und Texte weiter zu verbessern.

Sind in diesem Buch alle Features der PowerShell beschrieben?

Die PowerShell umfasst mittlerweile mehrere Tausend Commandlets mit jeweils zahlreichen Optionen. Zudem gibt es unzählige Erweiterungen mit vielen Hundert weiteren Commandlets. Außerdem existieren zahlreiche Zusatzwerkzeuge. Es ist allein schon aufgrund der Vorgaben des Verlags für den Umfang des Buchs nicht möglich, alle Commandlets und Parameter hier auch nur zu erwähnen. Zudem habe ich – obwohl ich selbst fast jede Woche mit der PowerShell in der Praxis arbeite – immer noch nicht alle Commandlets und alle Parameter jemals selbst eingesetzt.

Ich beschreibe in diesem Buch, was ich selbst in der Praxis, in meinen Schulungen und bei Kundeneinsätzen verwende. Es macht auch keinen Sinn, hier jedes Detail der PowerShell zu dokumentieren. Stattdessen gebe ich Ihnen **Hilfe zur Selbsthilfe**, damit Sie die Konzepte gut verstehen und sich dann Ihre spezifischen Lösungen anhand der Dokumentation selbst erarbeiten können.

Wie aktuell ist dieses Buch?

Die Informationstechnik hat sich immer schon schnell verändert. Seit aber auch Microsoft die Themen „Agilität“ und „Open Source“ für sich entdeckt hat, ist die Entwicklung nicht mehr nur schnell, sondern zum Teil rasant:

- Es erscheinen in kurzer Abfolge immer neue Produkte.
- Produkte erscheinen schon in frühen Produktstadien als „Preview“ mit Versionsnummern wie 0.1.
- Produkte ändern sich sehr häufig, teilweise im Abstand von drei Wochen (z. B. Visual Studio und Azure DevOps).
- Aufwärts- und Abwärtskompatibilität ist kein Ziel bei Microsoft mehr. Es wird erwartet, dass Sie Ihre Lösungen ständig den neuen Gegebenheiten anpassen.
- Produkte werden nicht mehr so ausführlich dokumentiert wie früher. Teilweise erscheint die Dokumentation erst deutlich nach dem Erscheinen der Software. Oft bleibt die Dokumentation auch dauerhaft lückenhaft.
- Produkte werden schnell auch wieder abgekündigt, wenn sie sich aus der Sicht der Hersteller bzw. aufgrund des Nutzerfeedbacks nicht bewährt haben.



HINWEIS: Nicht nur Microsoft geht so vor, sondern viele andere Softwarehersteller (z. B. Google) agieren genauso.

Unter diesen neuen Einflussströmen steht natürlich auch dieses etablierte Fachbuch. Leider kann man ein gedrucktes Buch nicht so schnell ändern wie Software. Verlage definieren nicht unerhebliche Mindestauflagen, die abverkauft werden müssen, bevor neu gedruckt werden darf. Das E-Book ist keine Alternative. Die Verkaufszahlen zeigen, dass nur eine kleine Menge von Lesern technischer Literatur ein E-Book statt eines gedruckten Buchs kauft. Das E-Book wird offenbar nur gerne als Ergänzung genommen. Das kann ich gut verstehen, denn ich selbst lese auch lieber gedruckte Bücher und nutze E-Books nur für eine Volltextsuche.

Daher kann es passieren, dass – auch schon kurz nach dem Erscheinen dieses Buchs – einzelne Informationen in diesem Buch nicht mehr zu neueren Versionen passen. Wenn Sie so einen Fall feststellen, schreiben Sie bitte eine Nachricht an mich (siehe unten). Ich werde dies dann in Neuauflagen des Buchs berücksichtigen.

Zudem ist zu beachten, dass zwischen Abgabe des Manuskripts beim Verlag und Auslieferung des Buchs aus der Druckerei an den Buchhandel meist vier bis fünf Monate liegen.

Welche PowerShell-Versionen werden besprochen?

Das Buch bespricht sowohl die Windows PowerShell 5.1 als auch die PowerShell 7.4.

- Bei der Windows PowerShell 5.1 wird die RTM-Version besprochen, die Microsoft in der aktuellen Version von Windows 10/11 bzw. Windows Server 2019/2022 mitliefert.
- Bei PowerShell 7.4 wird die RTM-Version vom 16. November 2023 behandelt.

Warum behandelt das Buch auch noch Version 5.1 und nicht nur Version 7.4?

Windows PowerShell 5.1 ist heute in den Unternehmen in Deutschland der Standard, denn diese Version der PowerShell wird mit Windows 10/11 und Windows Server 2016, Windows Server 2019 sowie Windows Server 1709, Windows Server 1909 und Windows Server 2022 ausgeliefert.

Die PowerShell 7.4 wird bisher mit keinem einzigen Betriebssystem ausgeliefert, sondern muss getrennt heruntergeladen und installiert werden. Eine Zusatzinstallation ist in vielen Unternehmen mit stark abgeschotteten Systemen gar nicht möglich.

Ein zweites Argument für die Beibehaltung der Version 5.1 in diesem Fachbuch ist, dass die PowerShell 7.4 der Windows PowerShell 5.1 funktional immer noch nicht ganz ebenbürtig ist. Einige Befehle sind weiterhin nur in der Windows PowerShell verfügbar.

Daher wird die Windows PowerShell 5.1 auch weiterhin eine große Bedeutung haben und in diesem Buch auch weiterhin behandelt.

Welche Betriebssysteme werden besprochen?

Der Schwerpunkt des Buchs liegt auf der Nutzung der PowerShell unter Windows. Es gibt Hinweise und Beispiele für die Nutzung der PowerShell unter Linux (am Beispiel Ubuntu) und macOS.

Bei Windows gibt es Hinweise auf Unterschiede zwischen verschiedenen Windows-Varianten (Client/Server) und Windows-Versionen.

Auch wenn Windows 11 bereits erschienen ist, ist Windows 10 das im professionellen Einsatz vorherrschende Betriebssystem. Das Buch geht auf existierende kleinere Unterschiede zwischen Windows 10 und Windows 11 ein, die meisten Screenshots sind aber mit Windows 10 gemacht. Einige Screenshots sind mit älteren Windows-Versionen geschossen, was aber kein Problem ist, denn inhaltlich hat sich nichts geändert (nur optisch an der Titelleiste und der Schriftart).

Woher bekommt man die Beispiele aus diesem Buch?

Unter <http://www.powershell-doktor.de/leser> biete ich ein **ehrenamtlich betriebenes** Webportal für Leser meiner Bücher an. Bei der Erstregistrierung müssen Sie das Lösungswort **Sektion31** angeben. Nach erfolgter Registrierung erhalten Sie dann ein persönliches Zugangskennwort per E-Mail.

In diesem Portal können Sie

- die Codebeispiele aus diesem Buch in einem Archiv herunterladen,
- eine PowerShell-Kurzreferenz „Cheat Sheet“ (zwei DIN-A4-Seiten als Hilfe für die tägliche Arbeit) kostenlos herunterladen sowie
- Feedback zu diesem Buch geben (Bewertung abgeben und Fehler melden).

Kurzreferenz ("Cheat Sheet") Windows PowerShell

Autor: Dr. Holger Schwichtenberg (www.IT-Visions.de) v1.5.2 / 22.03.2018

Hilfe

Alle installierten Module
Get-Module -ListAvailable | ? Name, ModulTyp, ExportedCommands
Alle Befehle mit 'Get-'
Get-Command Get-*

Wichtige Navigations-Commandlets

Mit den Navigations-Commandlets kann man nicht nur in Dateien, sondern auch andere Dateien und hierarchischen Mengen arbeiten.

Dir HKLM:\Software\NewItems HKLM:\Software\ITVisions ID HKLM:\Software\ITVisions

Table with 2 columns: Commandlet Name and Description. Includes Get-PSDrive, Get-Location, Get-Item, etc.

Active Directory-Commandlets

Diese Commandlets erfordern das Active Directory-PowerShell-Modul auf dem Client und ADWS (Active Directory WebServices) auf dem AD-Server:

Table with 2 columns: Commandlet Name and Description. Includes Get-ADObject, Get-ADUser, Get-ADGroup, etc.

Weitere wichtige Commandlets

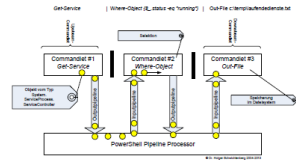
Table with 2 columns: Commandlet Name and Description. Includes Get-Date, Get-Service, Get-Process, etc.

Pipelining-Grundkonzept

Beliebig viele Commandlets können mit dem Pipe-Symbol | verkett werden.

Alternativ kann man Zwischenergebnisse in Variablen, die mit \$ beginnen, ablegen.

Die Pipeline befördert .NET-Objekte. Die Beförderung ist synchron (außer bei eigenen „blockierenden“ Commandlets wie Set-Content).



Wichtige Pipelining-Commandlets

Table with 2 columns: Commandlet Name and Description. Includes Where-Object, Select-Object, Sort-Object, etc.

Vergleichsoperatoren

Da die Zeichen < und > für Umkehrungen der Ausgabemenge verwendet werden, können PowerShell eher ungewöhnliche Operatoren zum Einsatz:

Table with 3 columns: Operator, Comparison, and Meaning. Includes -lt, -gt, -eq, -ne, etc.

Vorderseite der PowerShell-Kurzreferenz

Kurzreferenz ("Cheat Sheet") Windows PowerShell

Autor: Dr. Holger Schwichtenberg (www.IT-Visions.de) v1.5.2 / 22.03.2018

Table with 2 columns: Operator and Description. Includes -notmatch, -is, -in, etc.

Für die logische Verknüpfung werden -and und -or sowie -not (alles !). verwendet.

Ein- und Ausgabe-Commandlets

Table with 2 columns: Commandlet Name and Description. Includes Format-Table, Get-Format, etc.

Benutzerdefinierte Tabellenausgabe
Get-Process | @([Label]::New -Expression={\$_.ID}; Width=5),

Zeichenketten und Ausdrücke

Einbringen einer Variablen in eine Zeichenkette
'Der Befehl ist \$Befehl!'
Hier muss {} zur Abgrenzung vom Doppelpunkt eingesetzt werden

Objektorientierter Zugriff auf Pipeline-Objekte

Anzahl der Objekte in der Pipeline
(Get-Service | where {\$_.status -eq 'running'}) | Count

Einzige Eigenschaften der Pipeline-Objekte ausgeben
(Get-Dir) | Out-GridView
(Get-Process) | Name

Methodenausführung in allen Pipeline-Objekten
(Get-Process explore | sort ws -desc) | Kill()

PowerShell-Datentypen

Table with 2 columns: PowerShell Type and .NET Type. Includes [char], [int], [bool], etc.

PowerShell-Skriptsprache

Bedingungen
if ((Get-Date).Year -le 2014) { 'AP' } else { 'New' }
Schleifen
for(\$i = 1; \$i -le 10; \$i++) { \$i }

Unterfunktionen mit Pflichtparameter und optionalen Parameter
function Get-DLL([Parameter(Mandatory=\$)]\$Name, [string]\$Dir = '')

.NET Framework-Klassen

PowerShell kann alle auf dem lokalen System vorhandenen .NET-Klassen auch direkt (d.h. ohne Einsatz von Commandlets) verwenden.

Zugriff auf statische Mitglieder
[System.Environment]::MachineName
[System.Console]::Beep(500, 500)

Instanzierung und Zugriff auf Instanzmitglieder
\$S = New-Object System.DirectoryServices.DirectoryEntry("WinNT://Server/HZ")

Zusätzliche Assembly laden und nutzen
[System.Reflection.Assembly]::LoadWithPartialName("Microsoft.VisualBasic")

Component Object Model (COM)

PowerShell kann alle installierten COM-Komponenten verwenden.
\$ie = New-Object -com "InternetExplorer.Application"

Windows Management Instrumentation (WMI)

PowerShell kann alle lokalen oder entfernten WMI-Klassen verwenden.
Get-CimInstance -Namespace root/cimv2 Computer MyServer

Liste aller WMI-Klassen aus einem Nomenstrraum von einem Computer
Get-CimClass -Namespace root/cimv2 Computer MyServer

Liste aller Instanzen einer WMI-Klasse auf einem Computer
Get-CimInstance Win32_LogicalDisk -Namespace root/cimv2 Computer MyServer

WQL-Abfrage auf einem Computer
Get-CimInstance -Query "Select * from Win32_NetAdapter where adapterType like '802.11'" Computer MyServer

Zugriff auf eine Instanz und Änderung der Instanz
\$C = Get-CimInstance Win32_LogicalDisk -Namespace root/cimv2 -Filter "DeviceID='C:'" Computer MyServer

Alternativ mit allen WMI-Commandlets
\$C = (WMI) "\MyServer\root\cimv2:Win32_LogicalDisk.DeviceID='C:'" \$C.VolumeName = "System"

Aufruf einer WMI-Methode
Invoke-CimMethod -Path "MyServer\root\cimv2:Win32_ComputerSystem.Name='MyServer' -Name 'Rename' -ArgumentList 'MyNewServer'

Links

technet.microsoft.com/scriptcenter
blogs.msdn.com/powershell
www.powershell.com
www.gotwrench.com
www.it-visions.de/scripting/powershell

Über den Autor

Dr. Holger Schwichtenberg gehört zu den bekanntesten Experten für die Programmierung mit Microsoft-Produkten in Deutschland. Er hat zahlreiche Bücher zu .NET und PowerShell veröffentlicht und spricht regelmäßig auf Fachkonferenzen.

Rückseite der PowerShell-Kurzreferenz