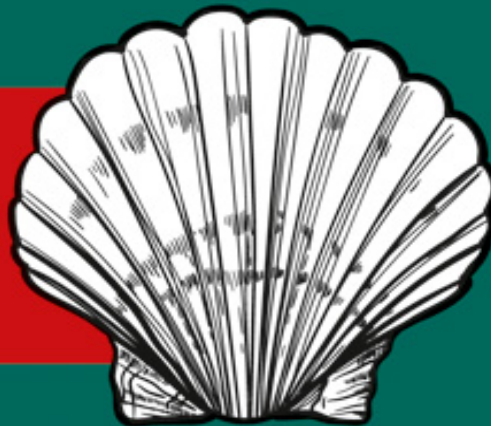


Holger SCHWICHTENBERG

Für  
Windows,  
Linux und  
macOS

# PowerShell 7 und Windows PowerShell 5



## DAS PRAXISBUCH

5. Auflage



Im Internet: Codebeispiele  
und PowerShell-Kurzreferenz

HANSER

[www.IT-Visions.de](http://www.IT-Visions.de)  
Dr. Holger Schwichtenberg

HANSER

Holger Schwichtenberg

**PowerShell 7 und Windows  
PowerShell 5**

Das Praxisbuch

5., aktualisierte Auflage

Alle in diesem Buch 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 Buch enthaltenen Informationen mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autoren 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 übernehmen Autoren und Verlag keine Gewähr dafür, dass beschriebene Verfahren usw. frei von Schutzrechten Dritter sind. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Buch berechtigt deshalb 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.

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. 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.

© 2022 Carl Hanser Verlag München, [www.hanser-fachbuch.de](http://www.hanser-fachbuch.de)

Lektorat: Sylvia Hasselbach

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

Umschlagdesign: Marc Müller-Bremer, [www.rebranding.de](http://www.rebranding.de), München

Umschlagrealisation: Max Kostopoulos

Titelmotiv: © [shutterstock.com/Irina](https://www.shutterstock.com/Irina) Kolesnichenko

Print-ISBN: 978-3-446-47296-9

E-Book-ISBN: 978-3-446-47446-8

E-Pub-ISBN: 978-3-446-47574-8

# Inhalt

## Titelei

## Impressum

## Inhalt

## Vorwort

## Über den Autor

## Teil A: PowerShell-Basiswissen

### 1 Fakten zur PowerShell

#### 1.1 Was ist die PowerShell?

#### 1.2 Geschichte der PowerShell

## 1.3 Welche Varianten und Versionen der PowerShell gibt es?

## 1.4 Windows PowerShell versus PowerShell Core versus PowerShell 7.x

## 1.5 Motivation zur PowerShell

## 1.6 Betriebssysteme mit vorinstallierter PowerShell

## 1.7 Support der PowerShell

## 1.8 Einflussfaktoren auf die Entwicklung der PowerShell

## 1.9 Anbindung an Klassenbibliotheken

## 1.10 PowerShell versus WSH

# 2 Erste Schritte mit der PowerShell

## 2.1 Windows PowerShell herunterladen und auf anderen Windows-Betriebssystemen installieren

## 2.2 Die Windows PowerShell testen

## 2.3 Woher kommen die PowerShell-Befehle?

[2.4 PowerShell Community Extensions \(PSCX\)  
herunterladen und installieren](#)

[2.5 Den Windows PowerShell-Editor „ISE“ verwenden](#)

[2.6 PowerShell 7 installieren und testen](#)

## [3 Einzelbefehle der PowerShell](#)

[3.1 Commandlets](#)

[3.2 Aliase](#)

[3.3 Ausdrücke](#)

[3.4 Externe Befehle \(klassische  
Kommandozeilenbefehle\)](#)

[3.5 Dateinamen](#)

## [4 Hilfsfunktionen](#)

[4.1 Auflisten der verfügbaren Befehle](#)

[4.2 Praxistipp: Den Standort eines  
Kommandozeilenbefehls suchen](#)

### **4.3 Anzahl der Befehle**

### **4.4 Volltextsuche**

### **4.5 Erläuterungen zu den Befehlen**

### **4.6 Hilfe zu Parametern**

### **4.7 Hilfe mit Show-Command**

### **4.8 Hilfefenster**

### **4.9 Allgemeine Hilfetexte**

### **4.10 Aktualisieren der Hilfedateien**

### **4.11 Online-Hilfe**

### **4.12 Fehlende Hilfetexte**

### **4.13 Dokumentation der .NETKlassen**

## **5 Objektorientiertes Pipelining**

### **5.1 Befehlsübersicht**

### **5.2 Pipeline-Operator**

### **5.3 . NET-Objekte in der Pipeline**

### **5.4 Pipeline Processor**

### **5.5 Pipelining von Parametern**

### **5.6 Pipelining von klassischen Befehlen**

### **5.7 Zeilenumbrüche in Pipelines**

### **5.8 Schleifen**

### **5.9 Zugriff auf einzelne Objekte aus einer Menge**

### **5.10 Zugriff auf einzelne Werte in einem Objekt**

### **5.11 Methoden ausführen**

### **5.12 Analyse des Pipeline-Inhalts**

### **5.13 Filtern**

### **5.14 Zusammenfassung von Pipeline-Inhalten**

### **5.15 „Kastrierung“ von Objekten in der Pipeline**

### **5.16 Sortieren**



**5.17 Duplikate entfernen**

**5.18 Gruppierung**

**5.19 Objekte verbinden mit Join-String**

**5.20 Berechnungen**

**5.21 Zwischenschritte in der Pipeline mit Variablen**

**5.22 Verzweigungen in der Pipeline**

**5.23 Vergleiche zwischen Objekten**

**5.24 Weitere Praxislösungen**

## **6 PowerShell-Skripte**

**6.1 Skriptdateien**

**6.2 Start eines Skripts**

**6.3 Aliase für Skripte verwenden**

**6.4 Parameter für Skripte**

**6.5 Skripte dauerhaft einbinden (Dot Sourcing)**

## 6.6 Das aktuelle Skriptverzeichnis

## 6.7 Sicherheitsfunktionen für PowerShell-Skripte

## 6.8 Skripte mit vollen Rechten (Elevation)

## 6.9 Blockierte PowerShell-Skripte

## 6.10 PowerShell-Skripte im Kontextmenü des Windows Explorers

## 6.11 Anforderungsdefinitionen von Skripten

## 6.12 Skripte anhalten

## 6.13 Versionierung und Versionsverwaltung von Skripten

# 7 PowerShell-Skriptsprache

## 7.1 Hilfe zur PowerShell-Skriptsprache

## 7.2 Befehlstrennung

## 7.3 Kommentare

## 7.4 Variablen

## 7.5 Variablenbedingungen

## 7.6 Zahlen

## 7.7 Zeichenketten (Strings)

## 7.8 Reguläre Ausdrücke

## 7.9 Datum und Uhrzeit

## 7.10 Objekte

## 7.11 Arrays

## 7.12 ArrayList

## 7.13 Assoziative Arrays (Hash-Tabellen)

## 7.14 Operatoren

## 7.15 Überblick über die Kontrollkonstrukte

## 7.16 Bedingungen

## 7.17 Unterroutinen (Prozedur/Funktionen)

## 7.18 Eingebaute Funktionen

**7.19 Fehlerausgabe**

**7.20 Fehlerbehandlung**

**7.21 Laufzeitfehler erzeugen**

**7.22 Objektorientiertes Programmieren mit Klassen**

## **8 Ausgaben**

**8.1 Ausgabe-Commandlets**

**8.2 Benutzerdefinierte Tabellenformatierung**

**8.3 Benutzerdefinierte Listenausgabe**

**8.4 Mehrspaltige Ausgabe**

**8.5 Out-GridView**

**8.6 Standardausgabe**

**8.7 Einschränkung der Ausgabe**

**8.8 Seitenweise Ausgabe**

**8.9 Ausgabe einzelner Werte**

## 8.10 Details zum Ausgabeoperator

## 8.11 Ausgabe von Methodenergebnissen und Unterobjekten in Pipelines

## 8.12 Ausgabe von Methodenergebnissen und Unterobjekten in Zeichenketten

## 8.13 Unterdrückung der Ausgabe

## 8.14 Ausgaben an Drucker

## 8.15 Ausgaben in Dateien

## 8.16 Umleitungen (Redirection)

## 8.17 Fortschrittsanzeige

## 8.18 Sprachausgabe

# 9 Das PowerShell-Navigationsmodell (PowerShell Provider)

## 9.1 Einführungsbeispiel: Navigation in der Registrierungsdatenbank

## 9.2 Provider und Laufwerke

### 9.3 Navigationsbefehle

### 9.4 Pfadangaben

### 9.5 Beispiel

### 9.6 Eigene Laufwerke definieren

## 10 Fernausführung (Remoting)

### 10.1 RPC-Fernabfrage ohne WS-Management

### 10.2 Anforderungen an PowerShell Remoting

### 10.3 Rechte für PowerShell-Remoting

### 10.4 Einrichten von PowerShell Remoting

### 10.5 Überblick über die Fernausführungs- Commandlets

### 10.6 Interaktive Fernverbindungen im Telnet-Stil

### 10.7 Fernausführung von Befehlen

### 10.8 Parameterübergabe an die Fernausführung

**10.9 Fernausführung von Skripten**

**10.10 Ausführung auf mehreren Computern**

**10.11 Sitzungen**

**10.12 Implizites Remoting**

**10.13 Zugriff auf entfernte Computer außerhalb der eigenen Domäne**

**10.14 Verwaltung des WS-Management-Dienstes**

**10.15 PowerShell Direct für Hyper-V**

**10.16 Praxislösung zu PowerShell Direct**

## **11 PowerShell-Werkzeuge**

**11.1 PowerShell-Standardkonsole**

**11.2 Windows Terminal**

**11.3 Erweiterung der Konsolen**

**11.4 PowerShell Integrated Scripting Environment (ISE)**

**11.5 PowerShell Script Analyzer**

**11.6 PowerShell Analyzer**

**11.7 PowerShell Tools for Visual Studio**

**11.8 PowerShell Pro Tools for Visual Studio**

**11.9 Visual Studio Developer PowerShell**

**11.10 NuGet Package Manager Console (PMC)**

**11.11 Visual Studio Code mit PowerShell-Erweiterung**

**11.12 PowerShell-Erweiterungen für andere Editoren**

**11.13 PowerShell Web Access (PSWA)**

**11.14 Azure Cloud Shell**

**11.15 ISE Steroids**

**11.16 PowerShellPlus**

**11.17 PoshConsole**

**11.18 PowerGUI**



[11.19 PrimalScript](#)

[11.20 CIM Explorer for PowerShell ISE](#)

## [12 Windows PowerShell Core 5.1 in Windows Nano Server](#)

[12.1 Installation](#)

[12.2 PowerShell-Skriptsprache](#)

[12.3 Werkzeuge](#)

[12.4 Fehlende Funktionen](#)

## [13 PowerShell 7 für Windows, Linux und macOS](#)

[13.1 Motivation für den Einsatz der PowerShell 7 auf Linux und macOS](#)

[13.2 Basis der PowerShell 7](#)

[13.3 Identifizierung der PowerShell 7](#)

[13.4 Funktionsumfang der PowerShell 7](#)

[13.5 Entfallene Befehle in PowerShell 7](#)

[13.6 Erweiterungsmodule nutzen in PowerShell 7](#)

[13.7 Geänderte Funktionen in PowerShell 7](#)

[13.8 Neue Funktionen der PowerShell 7](#)

[13.9 PowerShell 7-Konsole](#)

[13.10 Praxislösung: Fallunterscheidung für PowerShell-Varianten](#)

[13.11 VSCode-PowerShell als Editor für PowerShell 7](#)

[13.12 Verwendung von PowerShell 7 auf Linux und macOS](#)

[13.13 PowerShell-Remoting via SSH](#)

[13.14 Performance-Vorteile der PowerShell 7](#)

[13.15 Dokumentation zur PowerShell 7](#)

[13.16 Quellcode zur PowerShell 7](#)

[Teil B: PowerShell-Aufbauwissen](#)

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

## **14.1 .NET versus .NET Core**

## **14.2 Ermitteln der verwendeten .NET-Version**

## **14.3 .NET-Bibliotheken**

## **14.4 Microsoft Docs**

## **14.5 Überblick über die Verwendung von .NET-Klassen**

## **14.6 Erzeugen von Instanzen**

## **14.7 Parameterbehaftete Konstruktoren**

## **14.8 Initialisierung von Objekten**

## **14.9 Nutzung von Attributen und Methoden**

## **14.10 Statische Mitglieder in .NET-Klassen und statische .NET-Klassen**

## **14.11 Generische Klassen nutzen**

## **14.12 Zugriff auf bestehende Objekte**

[14.13 Laden von Assemblies](#)

[14.14 Liste der geladenen Assemblies](#)

[14.15 Verwenden von NuGet-Assemblies](#)

[14.16 Objektanalyse](#)

[14.17 Aufzählungstypen  
\(Auflistungen/Enumerationen\)](#)

## [15 Verwendung von COM-Klassen](#)

[15.1 Unterschiede zwischen COM und .NET](#)

[15.2 Erzeugen von COM-Instanzen](#)

[15.3 Abruf der Metadaten](#)

[15.4 Nutzung von Attributen und Methoden](#)

[15.5 Liste aller COM-Klassen](#)

[15.6 Holen bestehender COM-Instanzen](#)

[15.7 Distributed COM \(DCOM\)](#)

# **16 Zugriff auf die Windows Management Instrumentation (WMI)**

## **16.1 Einführung in WMI**

## **16.2 WMI in der PowerShell**

## **16.3 Open Management Infrastructure (OMI)**

## **16.4 Abruf von WMI-Objektmenngen**

## **16.5 Fernzugriffe**

## **16.6 Filtern und Abfragen**

## **16.7 Liste aller WMI-Klassen**

## **16.8 Hintergrundwissen: WMI-Klassenprojektion mit dem PowerShell-WMI-Objektadapter**

## **16.9 Beschränkung der Ausgabeliste bei WMI-Objekten**

## **16.10 Zugriff auf einzelne Mitglieder von WMI-Klassen**

## **16.11 Werte setzen in WMI-Objekten**

## 16.12 Umgang mit WMI-Datumsangaben

## 16.13 Methodenaufrufe

## 16.14 Neue WMI-Instanzen erzeugen

## 16.15 Instanzen entfernen

## 16.16 Commandlet Definition XML-Datei (CDXML)

# 17 Dynamische Objekte

## 17.1 Erweitern bestehender Objekte

## 17.2 Komplet dynamische Objekte

# 18 Einbinden von C# und Visual Basic .NET

# 19 Win32-API-Aufrufe

# 20 Benutzereingaben

## 20.1 Read-Host

## 20.2 Benutzerauswahl

## 20.3 Grafischer Eingabedialog

## 20.4 Dialogfenster

## 20.5 Authentifizierungsdialog

## 20.6 Zwischenablage (Clipboard)

# 21 Fehlersuche

## 21.1 Detailinformationen

## 21.2 Einzelschrittmodus

## 21.3 Zeitmessung

## 21.4 Ablaufverfolgung (Tracing)

## 21.5 Erweiterte Protokollierung aktivieren

## 21.6 Script-Debugging in der ISE

## 21.7 Kommandozeilenbasiertes Script-Debugging

# 22 Transaktionen

## 22.1 Commandlets für Transaktionen

## 22.2 Start und Ende einer Transaktion

## 22.3 Zurücksetzen der Transaktion

## 22.4 Mehrere Transaktionen

# 23 Standardeinstellungen ändern mit Profilskripten

## 23.1 Profilpfade

## 23.2 Ausführungsreihenfolge

## 23.3 Beispiel für eine Profildatei

## 23.4 Starten der PowerShell ohne Profilskripte

# 24 Digitale Signaturen für PowerShell-Skripte

## 24.1 Zertifikat erstellen

## 24.2 Skripte signieren

## 24.3 Verwenden signierter Skripte



## 24.4 Mögliche Fehlerquellen

# 25 Hintergrundaufträge („Jobs“)

## 25.1 Voraussetzungen

## 25.2 Architektur

## 25.3 Starten eines Hintergrundauftrags

## 25.4 Hintergrundaufträge abfragen

## 25.5 Warten auf einen Hintergrundauftrag

## 25.6 Abbrechen und Löschen von Aufträgen

## 25.7 Analyse von Fehlermeldungen

## 25.8 Fernausführung von Hintergrundaufträgen

## 25.9 Praxislösung: Einen Job auf mehreren Computern starten

# 26 Geplante Aufgaben und zeitgesteuerte Jobs

## 26.1 Geplante Aufgaben (Scheduled Tasks)

## 26.2 Zeitgesteuerte Jobs

# 27 PowerShell-Workflows

## 27.1 Ein erstes Beispiel

## 27.2 Unterschiede zu einer Function bzw. einem Skript

## 27.3 Einschränkungen bei Workflows

## 27.4 Workflows in der Praxis

## 27.5 Workflows in Visual Studio erstellen

# 28 Ereignissystem

## 28.1 WMI-Ereignisse

## 28.2 WMI-Ereignisabfragen

## 28.3 WMI-Ereignisse seit PowerShell 1.0

## 28.4 Registrieren von WMI Ereignisquellen seit PowerShell 2.0

[28.5 Auslesen der Ereignisliste](#)

[28.6 Reagieren auf Ereignisse](#)

[28.7 WMI-Ereignisse seit PowerShell-Version 3.0](#)

[28.8 Registrieren von .NET-Ereignissen](#)

[28.9 Erzeugen von Ereignissen](#)

## [29 Datenbereiche und Datendateien](#)

[29.1 Datenbereiche](#)

[29.2 Datendateien](#)

[29.3 Mehrsprachigkeit/Lokalisierung](#)

## [30 Desired State Configuration \(DSC\)](#)

[30.1 Grundprinzipien](#)

[30.2 DSC für PowerShell 7](#)

[30.3 Ressourcen](#)

[30.4 Verfügbare DSC-Ressourcen](#)

## **30.5 Eigenschaften einer Ressource**

## **30.6 Aufbau eines DSC-Dokuments**

## **30.7 Commandlets für die Arbeit mit DSC**

## **30.8 Ein erstes DSC-Beispiel**

## **30.9 Kompilieren und Anwendung eines DSC-Dokuments**

## **30.10 Variablen in DSC-Dateien**

## **30.11 Parameter für DSC-Dateien**

## **30.12 Konfigurationsdaten**

## **30.13 Entfernen einer DSC-Konfiguration**

## **30.14 DSC Pull Server**

## **30.15 DSC-Praxislösung 1: IIS installieren**

## **30.16 DSC-Praxislösung 2: Software installieren**

## **30.17 DSC-Praxislösung 3: Software deinstallieren**

[30.18 Realisierung einer DSC-Ressource](#)

[30.19 Weitere Möglichkeiten](#)

## [31 PowerShell-Snap-Ins](#)

[31.1 Einbinden von Snap-Ins](#)

[31.2 Liste der Commandlets](#)

## [32 PowerShell-Module](#)

[32.1 Überblick über die Commandlets](#)

[32.2 Modularchitektur](#)

[32.3 Aufbau eines Moduls](#)

[32.4 Module aus dem Netz herunterladen und installieren mit PowerShellGet](#)

[32.5 Module manuell installieren](#)

[32.6 Doppeldeutige Namen](#)

[32.7 Auflisten der verfügbaren Module](#)

[32.8 Importieren von Modulen](#)

[32.9 Entfernen von Modulen](#)

## [33 Ausgewählte PowerShell-Erweiterungen](#)

[33.1 PowerShell-Module in Windows 8.0 und Windows Server 2012](#)

[33.2 PowerShell-Module in Windows 8.1 und Windows Server 2012 R2](#)

[33.3 PowerShell-Module in Windows 10 und Windows Server 2019](#)

[33.4 PowerShell Community Extensions \(PSCX\)](#)

[33.5 PowerShellPack](#)

[33.6 \[www.IT-Visions.de\]\(http://www.IT-Visions.de\): PowerShell Extensions](#)

[33.7 Quest Management Shell for Active Directory](#)

[33.8 Microsoft Exchange Server](#)

[33.9 System Center Virtual Machine Manager](#)