

Inhaltsverzeichnis

Einführung	17
Wer braucht eigentlich die PowerShell?	18
Moderne Lernkurve	18
Computer – ich / ich – Computer: die PowerShell als Fremdsprache	19
Eine strategische Plattform	20
Plattformübergreifende Technik	21
Persönliche Entwicklung	22
Wie Sie dieses Buch nutzen	22
Noch mehr Unterstützung	23
Teil A	
Einzelne Befehle	25
1 Erste Schritte mit der Windows PowerShell	27
Die PowerShell starten	28
Die PowerShell bequem starten	28
Die PowerShell in die Taskleiste integrieren	29
Die interaktive PowerShell-Konsole	30
Unvollständige und mehrzeilige Eingaben	31
Wichtige Tastenkombinationen	32
Text markieren und einfügen	35
Die Konsole angenehm einrichten	36
Die Konsolen-Eigenschaften öffnen	36
Optionen festlegen	36
Schriftart und Schriftgröße festlegen	37
Fenster- und Puffergröße festlegen	38
Farben auswählen	38
Änderungen direkt in der PowerShell festlegen	38
Änderungen speichern	39
Testen Sie Ihr Wissen!	40
Zusammenfassung	42
2 PowerShell-Cmdlets einsetzen	45
Was sind eigentlich Cmdlets?	46
Tätigkeiten auswählen	46
Alles, was Sie über Parameter wissen müssen	52

Hilfe zu Parametern (und Cmdlets) abrufen	58
Fehlermeldungen verstecken	59
Sonderfall: Common Parameter	60
Hilfe zu allgemeinen PowerShell-Themen abrufen	62
Hands On: Ereignisprotokolle administrieren	67
Testen Sie Ihr Wissen!	70
Nachschlag für Profis	75
Kleine Rätselstunde	75
Weniger Tippen mit abgekürzten Parametern	79
Parameter-Binding sichtbar machen	83
PowerShell-Cmdlets über Stichwörter finden	84
Kindersicherung für Cmdlets	85
Zusammenfassung	87
3 Befehle in der PowerShell-Konsole	91
Sieben Befehlsarten unter einem Dach	92
Rechnen und Parametrisieren	94
Grundrechenarten	94
Mit Zahlensystemen und Einheiten rechnen	95
Nicht-PowerShell-Befehle ausführen	101
Windows-Anwendungen starten	101
Konsolenbasierte Programme	102
Die alte Konsole starten	104
Programme beenden	105
Globale Ordner	106
Befehlsnamen per Alias abkürzen	111
Aliasnamen auflösen	112
Eigene Aliasnamen festlegen	113
Eigene Aliasnamen speichern	114
Funktionen: eigene Befehle erzeugen	116
Funktionen für mehr Bequemlichkeit	116
Funktionen für mehr Kompatibilität	117
Funktionen verändern das Verhalten der PowerShell	118
Eigene neue Cmdlets erstellen	119
Vordefinierte Funktionen erforschen	120
Dateien und Skripts aufrufen	122
Skripts starten	122
PowerShell-Skripts zulassen oder verbieten	127
Profildateien einsetzen	130
PowerShell-Skripts extern starten	131
PowerShell-Skripts automatisiert aufrufen	133
Nachschlag für Profis	138
Kleine Rätselstunde	138
Mehrdeutige Befehle	143
Zusammenfassung	144

4	Provider: virtuelle Laufwerke	149
	Provider und virtuelle PowerShell-Laufwerke	150
	Provider finden	151
	Virtuelle Laufwerke finden	153
	Neue virtuelle Laufwerke anlegen	153
	Virtuelle Laufwerke entfernen	155
	Mit PowerShell-Laufwerken arbeiten	155
	In PowerShell-Laufwerken navigieren	156
	Relative und absolute Pfade	157
	Relative Pfadnamen in absolute Pfadnamen verwandeln	157
	Ordnerpositionen speichern	158
	Laufwerksinhalte auflisten	159
	Platzhalterzeichen verwenden	161
	Auf bestimmte Items zugreifen	161
	Item-Properties auflisten	162
	Neue Laufwerksinhalte anlegen	164
	Hinter den Kulissen: New-Item	165
	Registrierungswerte ändern und anlegen: Item-Property	167
	Inhalte verschieben und kopieren	169
	Inhalte umbenennen	169
	Inhalte löschen	170
	Ganze Containerinhalte löschen	170
	Container samt Inhalt löschen	171
	ItemProperties löschen	171
	Spezielle Laufwerksoptionen	171
	Nachschlag für Profis	173
	Zusammenfassung	176
5	Neue PowerShell-Befehle nachrüsten	179
	Snap-Ins verwenden	180
	Geladene Snap-Ins	180
	... und installierte Snap-Ins	181
	Snap-Ins nachinstallieren	182
	Snap-Ins verwenden	182
	Module verwenden	184
	Bereits geladene Module auflisten	184
	Verfügbare Module anzeigen	184
	Module verwenden	185
	Nachschlag für Profis	187
	Dateisystemhelfer	187
	Bildbearbeitung mit der PowerShell	188
	Expeditionstools für Low-Level-Scripter	189
	Zusammenfassung	191

Teil B	
Pipeline	193
6 Die PowerShell-Pipeline	195
Befehle mit der PowerShell-Pipeline verketteten	196
Befehle verketteten	196
Parameter einsetzen	197
Objekte und die objektorientierte Pipeline	199
Mit Objekteigenschaften arbeiten	200
Versteckte Objekteigenschaften sichtbar machen	202
Datenpipelines: Informationen aufbereiten	203
Ergebnisse mit Where-Object reduzieren	204
Objekteigenschaften mit Select-Object reduzieren	211
Ergebnisse mit Sort-Object in die richtige Reihenfolge bringen	213
Die PowerShell-Pipeline als universelle Datenabfragesprache	214
Konversions-Pipelines: Informationen umwandeln	216
ForEach-Object: Pipelineobjekte einzeln bearbeiten	217
Eingaben bearbeiten und umwandeln	217
Objekte auf Objekteigenschaften reduzieren	218
Ergebnisse gruppieren und messen	220
Aktionspipelines: Operationen durchführen	226
Prozesse oder Dienste stoppen	226
Eingaben berechnen oder aus Quellen lesen	227
Nachschlag für Profis	230
Kleine Rätselstunde	230
Protokolldateien parsen	237
Textersetzungen in Dateien	239
Konsolenbefehle automatisieren	240
Ausblick auf weitere Möglichkeiten	242
Zusammenfassung	243
7 Pipelineergebnisse exportieren	247
Die Ausgabeautomatik der PowerShell verstehen	248
Ausgaben besser formatieren	249
Ergebnisse als Tabelle	249
Ergebnisse als Liste	253
Mehrspaltige Anzeige	254
Ergebnisse als Text ausgeben	254
Das Geheimnis der Textumwandlung	255
Out-Host	257
Out-Null: Ergebnisse verschlucken	258
Out-File: Konsolenausgaben in Textdateien umlenken	258
Out-Printer: Ergebnisse zu Papier bringen	260
Out-String: Textdarstellungen erzwingen	261
Ergebnisse als Objekte exportieren	266
Out-GridView: Objekte im Spreadsheet anzeigen	266
Export-Csv: Export an Excel und andere Programme	267
Ergebnisse als XML serialisieren	269

HTML-Reports erstellen	273
Objekteigenschaften in HTML-Spalten umwandeln	274
HTML im Webbrowser anzeigen	275
HTML-Reports farbig und ansprechend gestalten	276
Mehrere HTML-Tabellen kombinieren	277
Nachschlag für Profis	281
Aufgaben und Rätsel	281
Zusammenfassung	286
Teil C	
Objekte	287
8 Mit Objekten arbeiten	289
Einführung: Objekte und Typen	290
Objekteigenschaften lesen und ändern	292
Objekteigenschaften auslesen	293
Beispiel: Dateiversionen bestimmen	295
Eigenschaften ändern	300
Dynamisch berechnete Objekteigenschaften	305
Objektmethoden aufrufen	310
Praxis: Textmanipulationen	311
Eine Methode mit mehreren Signaturen	313
Selbst definierte Objekte erstellen	314
Simple neue Objekte erzeugen	315
Dynamisch neue Objekte aufbauen	316
Bestehende Objekte erweitern	317
Dauerhafte Erweiterung von Objekten	320
Nachschlag für Profis	321
Kleine Rätselstunde	321
Hashtable und dynamische Eigenschaften	324
Sort-Object	325
Select-Object	326
Format-Wide	327
Format-List	328
Format-Table	328
Group-Object	329
Zusammenfassung	330
9 Typen verwenden	333
Den Typ eines Objekts bestimmen	334
Objekte umwandeln in andere Typen	335
Neue Objekte durch Umwandlung	337
Verkettete Konversionen	338
Vorteil spezifischer Typen	340
Gefahren bei der Umwandlung	342
Beispiel: Arrays mit Extrafunktionen	343

Typen enthalten weitere Befehle	344
Eigenschaften und Methoden eines Typs auflisten	345
Mathematische Funktionen	346
Netzwerkfunktionen	348
Umgebungsvariablen	349
Konsoleneinstellungen	350
Typen suchen und finden	353
Assemblys auflisten	355
Typen mit bestimmten Befehlen finden	356
Praxis: Dialogfelder finden	357
Neue Objekte aus Typen erzeugen	359
Neue Objekte mit New-Object	359
Konstruktoren verwenden	361
Konstruktoren ermitteln	362
Klassische COM-Objekte ansprechen	364
Ein COM-Objekt erzeugen	364
Beispiele	365
Welche COM-Objekte gibt es sonst noch?	371
Traditionelle Skriptsprachen portieren	371
Eigene neue Typen erzeugen	373
Eine neue Klasse erstellen	374
Auf Low-Level-APIs zugreifen	375
Nachschlag für Profis	376
Fragen und Antworten	376
Projekt: selbsterklärende .NET-Objekte	380
WMI-Remotenzugriff mit Anmeldung	381
Kontextmenübefehle verwenden	382
Mit XML arbeiten	382
Anmeldedaten speichern	389
Bilderformate konvertieren	390
Zusammenfassung	391

Teil D

Skripts	395
----------------------	-----

10 Funktionen	397
Neue Funktionen anlegen	398
Funktionen mit der PowerShell ISE erzeugen	399
Lebensdauer der Funktionen	401
Schreibgeschützte Funktionen	401
Argumente an Parameter übergeben	402
Parameter deklarieren	403
Streng typisierte Parameter verwenden	404
Zwingend erforderliche Parameter	405
Vorgabewerte festlegen	406
Aliasnamen für Parameter	406
Common Parameter	407
Switch-Parameter funktionieren wie Schalter	410

Mehrere Parametersets verwenden	410
Parameter <i>by Reference</i>	412
Bedienungsanleitung hinzufügen	413
Kommentarbasierte Hilfe einfügen	414
Rückgabewerte festlegen	417
Ein Rückgabewert oder mehrere?	417
Return-Anweisung	418
Auf die Rückgabewerte zugreifen	418
Unerwünschte Rückgabewerte	419
Pipelinefähige Funktionen	422
\$Input – langsamer sequenzieller Modus	422
Filter: schneller Streamingmodus	424
Echte Pipelinefunktionen entwickeln	425
Pipeline und direkte Argumente kombinieren	426
Unterschiedliche Parameter über die Pipeline ansprechen	427
Parameter über Objekteigenschaften empfangen	429
Nachschlag für Profis	430
Vorhandene Funktionen untersuchen	430
Prompt: eine bessere Eingabeaufforderung	430
Proxyfunktionen: Highlights für Profis	434
<i>ConfirmImpact</i> für eigene Funktionen festlegen	441
Zusammenfassung	442
11 Variablen	443
Eigene Variablen anlegen	444
Variablen in Text einbetten	444
Spezielle Variablennamen	446
Werte zuweisen	446
Prüfen, ob eine Variable existiert	448
Variablen löschen	448
Konstanten anlegen	449
Variablen dokumentieren	450
Umgebungsvariablen und andere Sonderformen	450
Gültigkeitsbereich von Variablen	452
Automatische Abschirmung abschalten	453
Gültigkeit explizit festlegen	454
Arrays verwenden	458
Text-Ergebnisse klassischer Befehle weiterverarbeiten	460
Objektergebnisse moderner Befehle weiterverarbeiten	460
Neue Arrays anlegen	461
Arrayelemente ansprechen	461
Mehrere Elemente aus einem Array auswählen	462
Elemente zu einem Array hinzufügen und daraus entfernen	463
Besondere Formen eines Arrays	464
Nachschlag für Profis	468
Kleine Rätselstunde	468
Manuell auf Gültigkeitsbereiche zugreifen	472
Streng typisierte Arrays	474
Zusammenfassung	475

12	Bedingungen	477
	Bedingungen formulieren	478
	Einen Vergleich durchführen	479
	Unterschiedliche Datentypen miteinander vergleichen	480
	Vergleiche umkehren	482
	Vergleiche kombinieren	483
	Vergleiche mit Arrays	483
	Bedingungen anwenden	486
	Where-Object	486
	if...elseif...else	487
	switch	489
	Nachschlag für Profis	495
	Zusammenfassung	496
13	Schleifen	497
	foreach und ForEach-Object	498
	Vor- und Nachteile beider Ansätze	499
	Der interne foreach-Zähler	500
	Die speziellen Skriptblöcke von ForEach-Object	501
	do und while	501
	Fortsetzungs- und Abbruchbedingungen	502
	Variablen als Fortsetzungskriterium verwenden	503
	Endlosschleifen ohne Fortsetzungskriterium	503
	for-Schleife	504
	for-Schleife: nur Sonderform der while-Schleife	505
	switch-Anweisung	506
	Dateiinhalte zeilenweise bearbeiten	507
	Nachschlag für Profis	508
	Schleifen vorzeitig abbrechen	508
	Schleifendurchläufe mit continue überspringen	510
	Verschachtelte Schleifen und Sprungmarken	510
	Ungewöhnliche Einsatzbereiche für die for-Schleife	511
	Befehlszeilenparameter mit <i>switch</i> auswerten	512
	Zusammenfassung	515
14	Skripts und Module	517
	PowerShell-Skripts verwenden	518
	Skripts wie Befehle aufrufen	519
	Argumente an Skripts übergeben	519
	Richtig gute Skripts erstellen	520
	Parameter und Bedienungsanleitung	520
	Funktionen und Bibliotheken	521
	Pipelineskripts erstellen	522
	Digitale Signaturen für Skripts	523
	Ein Zertifikat finden	524
	PowerShell-Skripts signieren	525
	Signierte PowerShell-Skripts überprüfen	527

Eigene Module erstellen	530
Eine Funktion als Modul verpacken	531
Modul laden, verwenden und wieder entfernen	532
Nachschlag für Profis	534
Kleine Rätselstunde	534
Skripts und Abhängigkeiten managen	536
Codesigning-Zertifikat mit Dialogfeld auswählen	540
Ein neues Zertifikat anlegen	541
Zertifikate als vertrauenswürdig erklären	543
Eine Mini-PKI aufbauen	544
Zusammenfassung	547
15 Fehler finden, abfangen und vermeiden	549
Fehlertoleranz und Standardfehlerhandling	550
<i>ErrorAction</i> festlegen	550
<i>\$ErrorActionPreference</i> verwenden	551
Fehlermeldungen anpassen	551
Fehler selbst behandeln	552
Basisfehlerhandling	552
Fehler in nativen Anwendungen abfangen	552
<i>try...catch</i> -Blöcke verwenden	554
Traps einsetzen	555
ErrorRecords und Exceptions – Details zum Fehler	558
Verschiedene Exceptions untersuchen	558
Wie ErrorRecords aufgebaut sind	560
Wie Exceptions aufgebaut sind	561
Aussagekräftige Fehlermeldungen erzeugen	562
Nur bestimmte Fehlertypen behandeln	563
Typ einer Exception bestimmen	564
Spezifische Fehlerhandler schreiben	564
Fehlerhandler kombinieren	566
Eigene Fehler auslösen	569
Fehler finden: debuggen	571
Tracing: ausgeführte Anweisungen anzeigen lassen	574
Stepping: Code schrittweise ausführen	575
PowerShell-Debugger verwenden	575
Nachschlag für Profis	576
Kleine Rätselstunde	576
Maskierte Fehler behandeln	580
Zusammenfassung	582
Teil E	
Spezielle Techniken	583
16 Windows PowerShell-Remoting	585
Übersicht: Remotingtechnologien	586
Klassische Remotezugriffe	586
Das neue universelle PowerShell-Remoting	590

Remoteausführung von Befehlen	593
Interaktives Remoting	594
PowerShell-Remoting abschalten	595
Mit Remotesessions arbeiten	598
Einmalsessions	599
Wieder verwendbare Sessions	599
Sessions zu mehreren Computern: Fan-Out	600
Sessions verwalten und freigeben	602
Implizites Remoting	603
Sessions konfigurieren und absichern	607
Eingebaute Sitzungskonfigurationen	608
32-Bit-Code auf 64-Bit-Systemen ausführen	609
Eigene Sitzungskonfigurationen	610
Berechtigungen für Sitzungskonfigurationen setzen	614
Einschränkungen und potenzielle Fallen	616
Wie Objekte transportabel werden	616
Fehler finden und beheben	620
RPC-Server nicht verfügbar	620
Zugriff verweigert	621
Kerberos-Fehlermeldung	622
Öffentliche Netzwerke entdeckt	623
Andere Fehler	624
Zusammenfassung	624
17 Hintergrundjobs	627
Aufgaben im Hintergrund ausführen	628
Hintergrundjob anlegen	628
Laufende Hintergrundjobs kontrollieren	628
Ergebnisse eines Hintergrundjobs abrufen	629
Hintergrundjobs abschließen	629
Parallelverarbeitung für mehr Geschwindigkeit	630
Der Parameter <i>-AsJob</i>	631
Allgemeine Parallelverarbeitung	632
Remotejobs und Authentifizierung	633
Hintergrundjobs auf Remotecomputern starten	633
Weitere Remotejobverfahren	635
Was Sie bei Hintergrundjobs bedenken sollten ...	636
PowerShell-Session unter anderem Benutzerkonto ausführen	638
Anmeldeinformationen in Hintergrundjob verwenden	638
Nachschlag für Profis	640
Kleine Rätsel	640
32-Bit-Skripts auf 64-Bit-Systemen	642
Zusammenfassung	643

18	Ereignisverarbeitung	645
	Ereignisse verwenden	646
	Ein Ereignis überwachen	646
	Ereignisüberwachung wieder abschalten	647
	Auf Events warten	647
	Blick hinter die Kulissen	648
	Hintergrundjobs überwachen	649
	Manuelle Überwachung	649
	Automatische Überwachung	649
	Ordner überwachen	651
	Aufgaben regelmäßig durchführen	652
	WMI-Ereignisse empfangen	653
	Details zum Event erfahren	653
	Systemänderungen erkennen	654
	Eigene Ereignisse auslösen	655
	Nachschlag für Profis	655
	Automatische Variablenüberwachung einrichten	656
	Newsticker in Konsolentitleiste	656
	Zusammenfassung	657
19	Texte und reguläre Ausdrücke	659
	Kurze Wiederholung: Textinformationen	660
	Here-Strings: mehrzeiligen Text erfassen	661
	Mit dem Benutzer kommunizieren	662
	Textausgaben formatieren	663
	Textoperatoren	664
	Formatierung von Zahlen festlegen	665
	Werte tabellarisch ausgeben: feste Breite	669
	Methoden des String-Objekts	670
	Textmuster und reguläre Ausdrücke	672
	Muster beschreiben	673
	\$Matches – Informationen finden	678
	Texte ersetzen	680
	Rückverweise verwenden	680
	Mit dem RegEx-Objekt arbeiten	682
	Statische RegEx-Klasse einsetzen	683
	Mehrzeiliger Modus	684
	Nachschlag für Profis	685
	Kleine Rätselstunde	685
	Methoden analysieren: Beispiel Split()	687
	IP-Adressen erkennen	688
	Zusammenfassung	689
	Stichwortverzeichnis	691
	Über den Autor	707