

Windows Scripting lernen

Von Windows Script Host und Visual Basic Script bis zur Windows PowerShell

Bearbeitet von
www.IT-Visions.de

1. Auflage 2016. Buch. 554 S.
ISBN 978 3 446 44800 1
Format (B x L): 18 x 24,5 cm
Gewicht: 1145 g

[Weitere Fachgebiete > EDV, Informatik > Programmiersprachen: Methoden > Microsoft Programmierung](#)

schnell und portofrei erhältlich bei


DIE FACHBUCHHANDLUNG

Die Online-Fachbuchhandlung beck-shop.de ist spezialisiert auf Fachbücher, insbesondere Recht, Steuern und Wirtschaft. Im Sortiment finden Sie alle Medien (Bücher, Zeitschriften, CDs, eBooks, etc.) aller Verlage. Ergänzt wird das Programm durch Services wie Neuerscheinungsdienst oder Zusammenstellungen von Büchern zu Sonderpreisen. Der Shop führt mehr als 8 Millionen Produkte.



Leseprobe

Holger Schwichtenberg

Windows Scripting lernen

Von Windows Script Host und Visual Basic Script bis zur Windows
PowerShell

ISBN (Buch): 978-3-446-44800-1

ISBN (E-Book): 978-3-446-44944-2

Weitere Informationen oder Bestellungen unter

<http://www.hanser-fachbuch.de/978-3-446-44800-1>

sowie im Buchhandel.

Inhalt

Vorwort	XVII
Website für Leser	XXII
Über den Autor Dr. Holger Schwichtenberg	XXIII
Hinweise für den Leser	XXV
Schreibweisen in diesem Buch	XXV
Hinweise zu den Listings	XXVI
1 Einführung in den Windows Script Host (WSH)	1
1.1 Der Windows Script Host	1
1.2 Scripting versus Programmierung	2
1.3 Voraussetzungen	2
1.4 Die Sprache Visual Basic Script	4
1.5 Das erste Skript	5
1.6 Scripting im Kommandozeilenfenster	8
1.7 Das zweite Skript: Versionsnummern ermitteln	11
1.8 Ein Wort zur Sicherheit	12
1.9 Wie geht es weiter?	13
1.10 Fragen und Aufgaben	14
2 Scripting-Werkzeuge	15
2.1 Nur zur Not: Notepad	16
2.2 Einer für alles: PrimalScript	18
2.3 Der WSH-Spezialist: SystemScripter	20
2.3.1 Fehlerarten	23
2.3.2 Start des Debuggers	24
2.3.3 Funktionen des Microsoft Script Debuggers	26
2.4 Fragen und Aufgaben	27

3	Scripting und die Benutzerkontensteuerung	29
3.1	Benutzerkontensteuerung	29
3.2	WSH-Skripte arbeiten nicht mit der Benutzerkontensteuerung zusammen	31
3.3	Lösungen des Problems	32
3.4	Start aus dem Admin-Konsolenfenster heraus	33
3.5	Anlegen einer Verknüpfung zu wscript.exe oder cscript.exe	35
3.6	Benutzerkontensteuerung deaktivieren	36
3.7	Änderungen der Benutzerkontensteuerung in Windows 8.x und Windows 10 sowie Windows Server 2012 sowie Windows Server 2016	37
4	Programmieren mit VBScript	39
4.1	Die Visual-Basic-Sprachfamilie	39
4.2	Allgemeines zum Arbeiten mit VBScript	40
4.3	Kommentare	42
4.4	Literale	42
4.5	Konstanten	44
4.5.1	Vordefinierte Konstanten	45
4.5.2	Definieren eigener Konstanten	46
4.5.3	Verwenden von Konstanten	47
4.6	Variablen	47
4.6.1	Verwendung von Variablen	48
4.6.2	Datentypen	50
4.7	Operatoren	51
4.7.1	Arithmetische Operatoren	52
4.7.2	Vergleichsoperatoren	53
4.7.3	Logische Operatoren	54
4.7.4	Bitweise Operationen	55
4.8	Bedingungen	56
4.8.1	If ... Then	57
4.8.2	Select Case	58
4.8.3	Bedingte Ausgaben zur Fehlersuche	59
4.9	Schleifen	61
4.9.1	For ... Next	61
4.9.2	Do ... Loop	62
4.10	Arrays (Variablenmengen)	65
4.10.1	Eindimensionale Arrays	66
4.10.2	Mehrdimensionale Arrays	68
4.11	Eingebaute Funktionen	69
4.11.1	Eingabehilfen	69
4.11.2	Ein- und Ausgabefunktionen	69

4.11.3	Konvertierungsfunktionen	74
4.11.4	Abs() und Int()	76
4.11.5	Rnd()	77
4.11.6	InStr()	78
4.11.7	Left(), Right() und Mid()	78
4.11.8	Replace() und Trim()	79
4.11.9	UCase() und LCase()	80
4.11.10	Split() und Join()	81
4.11.11	Date(), Time() und Now()	82
4.11.12	DateAdd() und DateDiff()	82
4.11.13	Hour(), Minute(), Second(), Day(), Month(), Year() und WeekDay()	84
4.11.14	Format(), FormatNumber() und FormatDateTime()	85
4.11.15	IsDate(), IsNumeric(), IsArray()	87
4.12	Unterroutinen	88
4.12.1	Unterroutine ohne Rückgabewert (Prozedur)	89
4.12.2	Unterroutine mit Rückgabewert (Funktion)	90
4.13	Benutzerdefinierte Fehlerbehandlung	91
4.14	Fragen und Aufgaben	93
5	Programmieren mit Objekten	95
5.1	Was ist ein Objekt?	95
5.2	Was ist eine Klasse?	96
5.3	Objekte haben Beziehungen	99
5.4	Was ist eine Komponente?	100
5.5	Wie arbeitet man mit Objekten?	101
5.5.1	Objektvariablen	102
5.5.2	Instanziierung eines Objekts aus einer Klasse	102
5.5.3	Auslesen des Werts eines Attributs	104
5.5.4	Setzen des Werts eines Attributs	104
5.5.5	Aufruf einer Methode	104
5.5.6	Reagieren auf ein Ereignis	105
5.5.7	Löschen eines Objekts	105
5.5.8	Duplizieren eines Objekts	106
5.5.9	Vergleich zweier Objekte	107
5.5.10	Ermitteln der Klasse, zu der ein Objekt gehört	107
5.6	Eingabehilfen für Objekte	107
5.7	Wie erfahre ich, welche Objekte es überhaupt gibt?	108
5.8	Was passiert, wenn ein Objekt nicht da ist?	110
5.9	Was ist eine Objektmenge?	110
5.9.1	For Each ... Next	111

5.9.2	Zugriff auf einzelne Objekte in einer Objektmenge	112
5.9.3	Verändern einer Objektmenge	113
5.10	Fragen und Aufgaben	114
6	Komponenten für das Scripting	115
6.1	WSH Runtime (WSHRun)	115
6.1.1	Installation	116
6.1.2	Klassen	116
6.1.3	Beispiele	118
6.2	Scripting Runtime (SCRRun)	118
6.2.1	Installation	119
6.2.2	Klassen	119
6.2.3	Objektauswahl	121
6.2.4	Beispiele	121
6.3	ActiveX Data Objects (ADO)	121
6.3.1	Installation	122
6.3.2	Klassen	122
6.3.3	Objektauswahl	123
6.3.4	Beispiele	125
6.4	Active Directory Service Interface (ADSI)	125
6.4.1	Installation	126
6.4.2	Klassen	127
6.4.3	Hilfsmittel	132
6.5	Group Policy Management-Komponente (GPMC Objects)	133
6.5.1	Installation	135
6.5.2	Klassen	135
6.5.3	Hilfsmittel	138
6.5.4	Beispiele	142
6.6	Windows Management Instrumentation (WMI)	142
6.6.1	Installation	143
6.6.2	WMI-Klassen	144
6.6.3	Scripting-Hilfsklassen für WMI	145
6.6.4	Objektauswahl	147
6.6.5	Hilfsmittel	150
6.7	Microsoft XML (MSXML)	151
6.7.1	XML-Grundlagen	151
6.7.2	Installation	154
6.7.3	Klassen	154
6.8	Fragen und Aufgaben	155

7	Datenübergabe und Datenausgabe	157
7.1	Kommandozeilenparameter	158
7.1.1	Komplexere Parameter	159
7.1.2	Kommandozeilenparameter des WSH	161
7.2	Zugriff auf Datendateien	162
7.2.1	Zugriff auf CSV-Dateien	163
7.2.2	Zugriff auf INI-Dateien	166
7.2.3	Zugriff auf Access-Datenbanken	170
7.2.4	Zugriff auf XML-Dateien	175
7.3	Fragen und Aufgaben	180
8	Scripting des Dateisystems	183
8.1	Dateien	183
8.1.1	Auflisten von Dateien	183
8.1.2	Dateieigenschaften bestimmen	184
8.1.3	Dateieigenschaften ändern	186
8.1.4	Anlegen einer Textdatei	187
8.1.5	Lesen einer Textdatei	188
8.1.6	Schreiben von Dateien	191
8.1.7	Umbenennen einer Datei	192
8.1.8	Kopieren einer Datei	192
8.1.9	Verschieben einer Datei	193
8.1.10	Dateien suchen	194
8.1.11	Suchen in Dateiinhalten	196
8.1.12	Dateien löschen	197
8.2	Verzeichnisse	198
8.2.1	Auflisten eines einzelnen Verzeichnisses	198
8.2.2	Auflisten eines Verzeichnisbaums	199
8.2.3	Anlegen eines Verzeichnisses	200
8.2.4	Verzeichnisattribute bestimmen	200
8.2.5	Umbenennen eines Verzeichnisses	202
8.2.6	Löschen von Verzeichnissen	203
8.2.7	Kopieren von Verzeichnissen	204
8.2.8	Verschieben von Verzeichnissen	204
8.2.9	Verzeichnis suchen	205
8.2.10	Eine Verzeichnisstruktur gemäß einer XML-Datei anlegen	207
8.2.11	Eine Verzeichnisstruktur in einer XML-Datei dokumentieren	210
8.3	Papierkorb leeren	213
8.4	Rechte auf Dateien und Verzeichnisse vergeben	215
8.5	Laufwerke	215
8.5.1	Auflisten von Laufwerken	215

8.5.2	Laufwerkstyp bestimmen	217
8.5.3	Dateisystemtyp ermitteln	218
8.5.4	Speicherplatzbelegung anzeigen	219
8.5.5	Mit einem Netzlaufwerk verbinden	222
8.5.6	Netzwerkverbindung trennen	223
8.5.7	Festplattenprüfung (CheckDisk)	223
8.6	Freigaben	224
8.6.1	Anlegen von Freigaben	225
8.6.2	Löschen von Freigaben	226
8.6.3	Rechte auf Freigaben	226
8.7	Fragen und Aufgaben	226
9	Scripting der Benutzerverwaltung	229
9.1	Benutzerverwaltung für lokale Benutzerkonten	230
9.1.1	Anlegen eines Benutzerkontos	230
9.1.2	Umbenennen eines Benutzers	233
9.1.3	Kennwort eines Benutzers ändern	234
9.1.4	Anlegen einer Benutzergruppe	235
9.1.5	Hinzufügen eines Benutzers zu einer Gruppe	237
9.1.6	Entfernen eines Benutzers aus einer Gruppe	238
9.1.7	Deaktivieren eines Benutzerkontos	238
9.1.8	Löschen einer Gruppe	239
9.1.9	Löschen eines Benutzers	240
9.2	Active-Directory-Benutzerverwaltung unter Windows Server	241
9.2.1	Anlegen einer Organisationseinheit	241
9.2.2	Anlegen eines Organisationseinheitenbaums im Active Directory	243
9.2.3	Anlegen eines Benutzerkontos	245
9.2.4	Anlegen von Benutzern aus einer Access-Datenbank	246
9.2.5	Anlegen einer Benutzergruppe	248
9.2.6	Hinzufügen eines Benutzers einer Gruppe	249
9.2.7	Ändern des Kennworts	251
9.2.8	Umbenennen eines Benutzers	251
9.2.9	Ändern der Benutzerdaten	252
9.2.10	Deaktivieren eines Benutzerkontos	253
9.2.11	Entfernen eines Benutzers aus einer Gruppe	254
9.2.12	Löschen eines Benutzerkontos	256
9.2.13	Löschen einer Organisationseinheit	257
9.3	Fragen und Aufgaben	258

10	Scripting der Computerverwaltung	259
10.1	Computer auflisten	259
10.2	Leistung eines Computers ermitteln	261
10.3	Computerkonto erstellen	263
10.4	Computerkonto löschen	264
10.5	Computer zu Domäne hinzufügen	265
10.6	Computer umbenennen	266
10.7	Einen Computer herunterfahren/neu starten	268
10.8	Fragen und Aufgaben	269
11	Scripting der Ereignisprotokolle	271
11.1	Protokolleinträge lesen	272
11.2	Protokolleinträge schreiben	273
11.3	Protokolleinträge auswerten	276
11.4	Datensicherung des Ereignisprotokolls	278
11.5	Ereignisprotokoll anlegen	279
11.6	Ereignisprotokoll löschen	280
11.7	Ereignisprotokoll leeren	281
11.8	Überwachung von Einträgen	282
11.9	Fragen und Aufgaben	283
12	Scripting der Systemdienste	285
12.1	Auflisten aller Dienste	285
12.2	Auflisten aller laufenden Dienste	287
12.3	Status ermitteln	287
12.4	Starten	288
12.5	Beenden eines Dienstes	289
12.6	Neustart eines Dienstes auf mehreren Computern gemäß einer Textdatei	290
12.7	Anhalten eines Dienstes	292
12.8	Fortsetzen eines Dienstes	293
12.9	Daten ändern	294
12.10	Dienste überwachen	296
12.11	Fragen und Aufgaben	297
13	Scripting des Desktops	299
13.1	Desktop verändern	299
13.2	Startmenü verändern	300
13.3	Fragen und Aufgaben	302

14	Scripting der Registrierungsdatenbank	303
14.1	Eintrag lesen	305
14.1.1	Zugriff mit WSHRun	305
14.1.2	Zugriff mit WMI	306
14.2	Wert schreiben	308
14.2.1	Alternative: WMI	309
14.3	Eintrag anlegen	311
14.4	Eintrag löschen	311
14.4.1	Alternative 1: Löschen mit der WSHRun-Komponente	311
14.4.2	Alternative 2: Löschen mit der WMI-Komponente	312
14.5	Unterschlüssel auflisten	312
14.6	Schlüssel anlegen	313
14.6.1	Alternative: WMI	314
14.7	Schlüssel löschen	315
14.7.1	Alternative: Löschen mit der WSHRun-Komponente	316
14.8	Berechtigungen vergeben	316
14.9	Fragen und Aufgaben	317
15	Scripting der Netzwerkkonfiguration	319
15.1	Festlegen einer statischen IP-Adresse	320
15.1.1	Besonderheiten	322
15.2	Standard-Gateway festlegen	322
15.3	DNS-Server festlegen	324
15.4	WINS-Server festlegen	325
15.5	Auf DHCP umstellen	326
15.6	Fragen und Aufgaben	327
16	Scripting der Softwareverwaltung	329
16.1	Installierte Software auflisten (Softwareinventarisierung)	329
16.2	Software (entfernt) installieren	334
16.3	Software auf mehreren Computern installieren (gemäß einer XML-Datei)	335
16.4	Software deinstallieren	337
16.5	Fragen und Aufgaben	337
17	Scripting der Prozessverwaltung	339
17.1	Prozesse auflisten	339
17.2	Prozesse (entfernt) starten	341
17.2.1	Prozesse starten mit WScript.Shell	342
17.2.2	Prozesse starten mit Win32_Process	343

17.3	Prozesse (entfernt) beenden	346
17.3.1	Prozesse beenden mit WScript.Shell	346
17.3.2	Prozesse beenden mit Win32_Process	348
17.4	Fragen und Aufgaben	349
18	Scripting der Gruppenrichtlinien	351
18.1	Informationen über ein einzelnes Gruppenrichtlinienobjekt	351
18.1.1	Suche nach einem GPO	351
18.1.2	Informationen über ein GPO	352
18.1.3	Verknüpfungen auflisten	352
18.1.4	Das komplette Skript	353
18.2	Alle Gruppenrichtlinien und ihre Verknüpfungen auflisten	355
18.3	Eine Gruppenrichtlinie für einen Container auflisten	358
18.4	Eine Gruppenrichtlinie mit einem AD-Container verknüpfen	360
18.5	Eine Gruppenrichtlinienverknüpfung löschen	362
18.6	Eine Gruppenrichtlinie löschen	364
18.7	Sicherungskopien von Gruppenrichtlinien anlegen	366
18.8	Sicherungskopien einer Gruppenrichtlinie auflisten	368
18.9	Wiederherstellung von Gruppenrichtlinien	369
18.10	Weitere Möglichkeiten	371
18.11	Fragen und Aufgaben	371
19	Scripting-Sicherheit	373
19.1	Bedrohungen durch WSH-Skripte	373
19.2	Schutz vor bösen Skripten	374
19.2.1	Globale WSH-Deaktivierung	374
19.2.2	Sperrung auf Skriptdateiebene	375
19.2.3	WSH-Skripte signieren	375
19.2.4	Skriptkontrolle durch Richtlinien für Softwareeinschränkungen ..	386
19.3	Schutz vor dem Einblick in den Quellcode	388
19.4	Ein Skript unter einem anderen Benutzerkontext starten	390
19.4.1	Benutzerwechsel für ein komplettes Skript	390
19.4.2	Benutzerwechsel im Skriptablauf	392
19.5	Fragen und Aufgaben	398
20	Windows PowerShell (WPS) 5.0	399
20.1	Vergleich zwischen WSH und PowerShell	399
20.2	Voraussetzungen und Installation	401
20.3	PowerShell-Werkzeuge	401
20.4	PowerShell-Commandlets	404
20.5	PowerShell-Pipelines	405

20.6	Ausgaben	408
20.7	Navigation in Containern	410
20.8	Hilfe zur PowerShell	412
20.9	PowerShell-Skripte	414
20.9.1	PowerShell-Skript-Editoren	415
20.9.2	Ein Beispiel	416
20.9.3	Sprachkonstrukte	417
20.9.4	Skripte ausführen	419
20.10	Fernausführung von Befehlen (Remoting)	422
20.11	Zusatzkomponenten und Klassen nutzen	424
20.12	Zusätzliche PowerShell-Module mit weiteren Commandlets	424
20.12.1	Module manuell installieren	424
20.12.2	Module automatisch herunterladen und installieren (ab PowerShell 3.0)	425
20.12.3	Module auflisten	431
20.12.4	Module laden	432
20.13	COM-Komponenten, die man auch im WSH mit VBScript nutzen kann	433
20.14	.NET-Klassen	434
20.15	WMI-Klassen	435
20.15.1	Abruf von WMI-Objektmengen	436
20.15.2	Fernzugriffe	437
20.15.3	Filtern und abfragen	437
20.15.4	Filtern mit Get-WmiObject	438
20.15.5	Zugriff auf einzelne WMI-Objekte	438
20.15.6	WQL-Abfragen	440
20.15.7	Ermittlung der Mitglieder des WMI-Objekts	441
20.15.8	Umgang mit WMI-Datumsangaben	443
20.15.9	Zugriff auf Mitglieder von WMI-Klassen	443
20.15.10	Statische Klassenmitglieder	445
20.15.11	Werte setzen in WMI-Objekten	445
20.15.12	Methodenaufrufe mit Invoke-WmiMethod	446
20.15.13	Liste aller WMI-Klassen	446
20.15.14	Neue WMI-Instanzen erzeugen	447
20.15.15	Weitere Möglichkeiten	448
20.16	PowerShell-Commandlets in Aktion	448
20.17	PowerShell-Skripte aus der Praxis	453
20.17.1	Leere Ordner löschen	453
20.17.2	Fotos nach Aufnahmedatum sortieren	454
20.17.3	Papierkorb leeren	456
20.17.4	Freigaben anlegen	456

20.17.5	Netzwerkconfiguration	466
20.17.6	Massenanlegen von Active-Directory-Benutzerkonten	468
20.17.7	Massenanlegen von IIS-Websites	472
20.17.8	Massenanlegen von Registry-Schlüsseln	473
20.17.9	Softwareinstallation	475
20.17.10	Virtuelles System in Hyper-V anlegen	476
21	Wie geht es weiter?	479
Anhang A: Eingebaute Funktionen in VBScript		481
A.1	Numerische Funktionen	481
A.2	Formatierungsfunktionen	482
A.3	Zeichenkettenfunktionen	482
A.4	Datums-/Uhrzeitfunktionen	484
A.5	Array-Funktionen	485
A.6	Funktionen zur Arbeit mit COM-Klassen	485
A.7	Systemfunktionen und Ein-/Ausgabe	486
A.8	Typprüfung und -umwandlung	486
A.9	Sonstige Funktionen	487
Anhang B: Lösungen zu den Übungsaufgaben in diesem Buch		489
B.1	Lösungen zu Kapitel 1	489
B.2	Lösungen zu Kapitel 2	490
B.3	Lösungen zu Kapitel 3	491
B.4	Lösungen zu Kapitel 4	492
B.5	Lösungen zu Kapitel 5	493
B.6	Lösungen zu Kapitel 6	493
B.7	Lösungen zu Kapitel 7	494
B.8	Lösungen zu Kapitel 8	495
B.9	Lösungen zu Kapitel 9	496
B.10	Lösungen zu Kapitel 10	498
B.11	Lösungen zu Kapitel 11	498
B.12	Lösungen zu Kapitel 12	499
B.13	Lösungen zu Kapitel 13	499
B.14	Lösungen zu Kapitel 14	500
B.15	Lösungen zu Kapitel 15	500
B.16	Lösungen zu Kapitel 16	502
B.17	Lösungen zu Kapitel 17	502
B.18	Lösungen zu Kapitel 18	504
B.19	Lösungen zu Kapitel 20	504

Anhang C: Abkürzungsverzeichnis	507
Anhang D: Quellen und weiterführende Literatur	515
Stichwortverzeichnis	519

Vorwort

Vorwort zur sechsten Auflage (2016)

Liebe Leserinnen, liebe Leser,

mittlerweile gibt es Windows 10 und Windows Server 2016. Aber den Verlag Addison-Wesley, bei dem dieses Buch zehn Jahre lang in fünf Auflagen erschienen ist, gibt es nicht mehr. Nun hat das Buch im Carl Hanser Verlag eine schöne neue Heimat gefunden.

Neu in dieser Auflage ist die Version 5.0 der Windows PowerShell. Zur PowerShell gibt es viele neue Skripting-Beispiele. Ebenso wurde das Buch auf Windows 10 und Windows Server 2016 aktualisiert. Bewusst sind aber nicht alle Bildschirmabbildungen mit diesen neuesten Betriebssystemen gemacht, da das Buch sich weiterhin als von der Betriebssystemversion unabhängiges Werk versteht, das auch Leser anspricht, die nicht die neueste Betriebssystemversion verwenden können oder wollen.

Viel Erfolg mit diesem Buch wünscht Ihnen

Dr. Holger Schwichtenberg

Essen, im Januar 2016

Vorwort zur fünften Auflage (2012)

Liebe Leserinnen, liebe Leser,

das Erscheinen von Windows 8 und Windows Server 2012 nehmen wir zum Anlass für eine erneute Aktualisierung dieses Buchs. Auch hier ist der Windows Script Host weiterhin enthalten und ein wichtiges Werkzeug für die automatisierte Systemadministration.

„Windows Scripting Lernen“ ist das letzte verbliebene Buch zum Windows Scripting Host (WSH) auf dem deutschen Markt.

Neben einigen Aktualisierungen zum WSH (insbesondere hinsichtlich der restriktiven Vergabe administrativer Rechte in Windows 8) bietet diese 5. Auflage Ihnen vor allem mehr Inhalte zum Thema Windows PowerShell. Behandelt wird die PowerShell 3.0, die in Windows 8 und Windows Server 2012 enthalten ist und auf Windows 7 und Windows Server 2008 (inkl. R2) als Zusatz installierbar ist.

Ich danke Ihnen für Ihre Treue zu diesem Buch.

Dr. Holger Schwichtenberg

Essen, im September 2012

Vorwort zur vierten Auflage (2009)

Liebe Leserinnen, liebe Leser,

das Erscheinen von Windows 7 und Windows Server 2008 R2 nehmen wir zum Anlass für eine Aktualisierung des Buchs. Die weiterhin hohen Verkaufszahlen im Zeitalter von Vista und PowerShell zeigen, dass der Windows Scripting Host (WSH) in den Unternehmen noch aktiv genutzt wird, selbst wenn er in den letzten Jahren keine großen Veränderungen mehr erfahren hat.

Natürlich ist der Markt kleiner geworden. In Deutschland gab es einmal sechs Bücher zum WSH. Von diesen sind nur noch „Windows Scripting Lernen“ und der große Bruder „Windows Scripting“ übrig geblieben.

Mittelfristig wird die PowerShell größere Bedeutung als der WSH erlangen. In diesem Buch gebe ich Ihnen einen Ausblick auf die PowerShell 2.0. Aber dieser Ausblick ist hier bewusst kurz gewählt. In meinen Büchern „Windows Scripting (6. Auflage)“ und „Windows PowerShell 2.0 – Das Praxishandbuch“ (beide bei Addison-Wesley erschienen) gehe ich genauer auf die PowerShell ein.

Ich danke Ihnen für Ihre Treue zu diesem Buch.

Dr. Holger Schwichtenberg

Essen, im Oktober 2009

Vorwort zur dritten Auflage (2007)

Liebe Leserinnen, liebe Leser,

auch mit Erscheinen von Windows Vista und der Windows PowerShell ist der Windows Script Host (WSH) noch aktuell, und er wird es auch für die nächsten Jahre noch bleiben. Windows Vista basiert entgegen früheren Ankündigungen noch nicht auf dem .NET Framework, sondern weiterhin komplett auf dem Component Object Model (COM) und noch älteren C/C++-Techniken. Der WSH und seine COM-basierten Scripting-Komponenten sind also das primäre Instrument für die automatisierte Systemadministration unter Vista.

Die Windows PowerShell, die Microsoft als gemeinsamen Nachfolger von WSH und Windows-Kommandozeilen-Shell ansieht, ist zwar mächtig und einfach, steht aber hinsichtlich des direkt nutzbaren Funktionsumfangs (in Form der Commandlets) noch weit hinter dem WSH zurück. Noch muss man hier in vielen Fällen in die Tiefen von .NET einsteigen. Erst mit kommenden Windows-Versionen und anderen Microsoft-Produkten wird es für die PowerShell einen Funktionsumfang geben, der auch Scripting-Einsteiger anspricht.

Grund genug also, diesem meistverkauften deutschen Scripting-Buch eine neue Auflage zu spendieren. In dieser Neuauflage finden Sie neben vielen kleinen Verbesserungen neue Texte zu folgenden Themen:

- Scripting-Neuerungen in Windows Vista (Kapitel 19)
- Einführung in die Windows PowerShell (Kapitel 20)
- Prozessverwaltung per Skript, insbesondere die Kommunikation zwischen Skripten und Konsolenanwendungen (Kapitel 16)

- Mehr Beispiele zur Verwendung von Text- und XML-Dateien als Ein- und Ausgabeformat für Skripte (in mehreren Kapiteln)

Beim Alten geblieben ist die Website, auf der Sie sich registrieren können für die Foren, Zusatz-Downloads und den Newsletter:

<http://www.Windows-Scripting.de>

Weiterhin viel Spaß beim Skripten wünscht Ihnen

Dr. Holger Schwichtenberg

Essen, im Mai 2007

Vorwort zur zweiten Auflage (2004)

Dass viele Administratoren sich ein Einsteigerbuch zum WSH wünschten, war mir klar, als ich dieses Buch Ende 2002 zusammen mit meinen drei Co-Autoren geschrieben habe. Dass wir damit die Position des Marktführers unter den Scripting-Büchern in Deutschland einnehmen würden, hätte ich nicht erwartet. Natürlich freuen wir uns sehr über die positive Resonanz.

Bereits Ende 2003 ist ein korrigierter Nachdruck erschienen, in dem wir die restlichen kleinen Tippfehler in dem Buch und auf der CD beseitigt haben. Nun liegt eine überarbeitete und erweiterte zweite Auflage vor Ihnen. Komplett neu in diesem Buch sind die Kapitel 16 („Scripting der Gruppenrichtlinien“) und 17 („Sicheres Scripting“).

Herzlich bedanken möchte ich mich bei allen Lesern, die durch ihr Feedback geholfen haben, diese zweite Auflage noch besser zu machen.

Ausdrücklich hinweisen möchte ich Sie auf die Website zu diesem Buch:

<http://www.Windows-Scripting.de>

Aktuell bietet Ihnen diese Website folgende Informationen und Dienste:

- Umfangreiches Windows Scripting-Glossar
- FAQ zum WSH
- Diskussionsforum zum Windows Scripting (Fragen von registrierten Lesern werden von den Autoren dieses Buchs vorrangig beantwortet!)
- Verzeichnis von Scripting-Komponenten
- Klassenreferenz für die Windows Management Instrumentation (WMI)
- Feedback-Fragebogen zu diesem Buch
- Aktualisierungen zu den Skripten in diesem Buch (sofern sich technische Änderungen in Windows ergeben oder Verbesserungen von uns oder den Lesern gefunden werden)
- Skriptarchiv mit über 200 weiteren WSH-Skripten
- Scripting-News und -Newsletter
- Und last, but not least: Informationen zu unseren Scripting-Schulungen und zum Support bei Fragen rund um den WSH.

Ich wünsche Ihnen nun viel Erfolg mit diesem Buch und würde mich freuen, Sie auf meiner Website begrüßen zu dürfen!

Dr. Holger Schwichtenberg

Essen, im Mai 2004

Vorwort zur ersten Auflage (2002)

WSH Zur automatisierten Systemadministration ist der Windows Script Host (WSH) eine sehr mächtige Alternative gegenüber der schon etwas angestaubten Windows-Batch-Programmierung. Unsere Erfahrungen aus Scripting-Schulungen und Beratungsterminen in den letzten vier Jahren haben aber gezeigt, dass es vielen Administratoren schwerfällt, sich in die Welt des Scriptings einzuarbeiten – oft auch gehemmt durch die Tatsache, dass das Scripting zum Bereich Programmierung/Softwareentwicklung gezählt wird.

Zielgruppe

Scripting ohne Vorkenntnisse „Windows Scripting Lernen“ wendet sich an Administratoren ohne Programmierkenntnisse. Dieses Buch enthält eine schrittweise Einführung in die Entwicklung von Skripten. Auch ohne Vorerfahrung in der Programmierung lernen Sie durch dieses Buch die Möglichkeiten zur automatisierten Administration von Unternehmensnetzwerken mit dem Windows Script Host (WSH), Visual Basic Script und verschiedenen sogenannten COM-Komponenten kennen.

Methodik

Didaktischer Aufbau Das Buch hat eine didaktische Struktur mit aufeinander aufbauenden Kapiteln. Bewusst wird darauf verzichtet, detaillierte Hintergründe sowie jede Möglichkeit und jede Option vorzustellen. Dieses Buch fokussiert auf das Wesentliche, um Ihnen einen leichten Einstieg in das Windows Scripting zu ermöglichen.

Einführung Alle grundlegenden Konzepte der Programmierung wie Variablen, Fallunterscheidungen, Schleifen, Fehlerbehandlung und die Arbeit mit Komponenten, Klassen und Objekten werden von Grund auf eingeführt. Außerdem finden Sie eine ausführliche Erklärung zur Installation und Konfiguration der Skripte und Komponenten sowie Hinweise auf mögliche Probleme oder Fehlersituationen.

Die Beispiele sind bewusst einfach gehalten. Dennoch werden Sie lernen, alle wesentlichen Aufgaben der System- und Netzwerkadministration durch Skripte zu lösen. Der deutliche Schwerpunkt dieses Buches liegt nicht auf dem Scripting im Heimeinsatz, sondern auf dem Scripting in Unternehmensnetzwerken. Daher finden Sie hier auch Themen wie das Scripting des Active Directory, der Netzwerkkonfiguration und von Ereignisprotokollen.

Am Ende jedes Kapitels stehen Aufgaben, die Sie einsetzen können, um Ihr Wissen zu vertiefen und praktisch zu üben. Gewisse Wiederholungen sind in diesem Einsteigerbuch übrigens kein Fehler, sondern didaktische Absicht.

Wie Sie dieses Buch lesen sollten

Leseanleitung Aufgrund des didaktischen Konzepts sollten Sie die ersten fünf Kapitel dieses Buches unbedingt sequenziell in der vorgegebenen Reihenfolge lesen. Ab Kapitel 6 werden dann verschiedene Gebiete des Scriptings aufgabenorientiert behandelt. Die Kapitel 6 bis 16 müssen Sie nicht notwendigerweise sequenziell lesen. Hier können Sie durchaus direkt zu den Kapiteln springen, die Sie besonders interessieren. Die von uns gewählte Reihenfolge beinhaltet aber eine Steigerung im Schwierigkeitsgrad.

Am Ende eines jeden Kapitels gibt es einen Aufgabenteil; die passenden Lösungen stehen zusammenhängend im Anhang, sodass das „Spicken“ etwas erschwert wird.

Weitere Unterstützung im WWW

Als Leser dieses Buches haben Sie Zugriff auf einen zugangsbeschränkten Bereich der deutschen Windows Scripting-Website, die Sie unter <http://www.Windows-Scripting.de> finden. Ein Service dieser Website ist, dass Sie den Autoren dieses Buches verbliebene Fragen stellen können.

Website

Der große Bruder „Windows Scripting“

Im Buchhandel werden Sie einen „großen Bruder“ zu diesem Buch finden, der schon zwei Jahre länger auf dem Markt ist: „Windows Scripting“ geht parallel zu diesem „Windows Scripting Lernen“ bei Addison-Wesley in die dritte Auflage. Dieser Titel aus der WinTec-Reihe ist ein umfassendes Nachschlagewerk zu allen Bereichen der Skriptprogrammierung und richtet sich an Entwickler und Administratoren, die bereits Vorkenntnisse in mindestens einer Programmiersprache besitzen. Wenn Sie nach der Lektüre von „Windows Scripting Lernen“ noch Wissensdurst verspüren, dann sollten Sie zum großen Bruder greifen.

Weiter-
führende
Literatur

Dank

Mein herzlicher Dank gilt

- meinen Co-Autoren Sven Conrad, Thomas Gartner und Oliver Scheer, die tatkräftig mitgeholfen haben, den umfangreichen Stoff aus dem „Windows Scripting“-Buch auf die „Lernen“-Reihe herunterzubereiten,
- Ayşe Aruca und Georg Meindl für ihre kritischen Anmerkungen als Testleser dieses Buches,
- meiner Korrektorin Astrid Schürmann, die wieder einmal mit hoher Präzision nicht nur die sprachlichen Fehler aus unserem Text entfernt, sondern uns auch auf inhaltliche Inkonsistenzen hingewiesen hat,
- und meiner Lektorin Sylvia Hasselbach für ihre Geduld bei der doch langwierigen Geburt dieses Werks.

Ich wünsche Ihnen nun viel Erfolg mit diesem Buch.

Holger Schwichtenberg

Essen, im November 2002

■ Website für Leser

Zu diesem Buch gibt es eine eigene Website:

<http://www.windows-scripting.de>

Sie als Leser haben neben den öffentlichen Bereichen auch die Möglichkeit, auf einen geschützten Bereich zuzugreifen, der besondere Informationen enthält:

Downloads: die aktuellen Versionen der in diesem Buch abgedruckten Skripte sowie weitere Skripte und Codebeispiele

Verzeichnis	Inhalt
\Skripte	Alle Skripte aus dem Buch, geordnet nach Kapiteln
\Install	Erweiterungen, Komponenten, Sprachen und Tools für das Windows Scripting (zum Teil als Vollversionen, zum Teil als Demoverversionen)
\Weitere Informationen	Dieses Verzeichnis enthält zusätzliche Dokumentationen zu Visual Basic Script, dem WSH und einigen der besprochenen Komponenten.
\Über den Autor	Informationen über den Autor dieses Buchs

Foren: Wenn Sie Fragen haben oder Ihre Meinung zu einem Thema dieses Buchs äußern möchten, dann können Sie hier auf Reaktionen anderer Nutzer hoffen.

Leser-Bewertung: Vergeben Sie Noten für dieses Buch und lesen Sie nach, was andere Leser von diesem Buch halten.

Bug-Report: Melden Sie hier Fehler, die Sie in diesem Buch gefunden haben! Hier können Sie auch nachlesen, welche Fehler anderen nach Drucklegung aufgefallen sind.

Newsletter: Alle registrierten Leser erhalten in unregelmäßigen Abständen einen Newsletter.



Der URL für den Zugang zum Leser-Portal lautet:

<http://www.windows-scripting.de/leser>

Bei der Anmeldung müssen Sie das Erstzugangskennwort Defiance Skies angeben (Defiance ist eine Science-Fiction-Serie).

Bitte beachten Sie, dass das Leser-Portal eine freiwillige, private Leistung des Autors ist, auf die es keinen Rechtsanspruch gibt.

Über den Autor

Dr. Holger Schwichtenberg



- Studienabschluss Diplom-Wirtschaftsinformatik an der Universität Essen
- Promotion an der Universität Essen im Gebiet komponentenbasierter Softwareentwicklung
- Seit 1996 selbstständig als unabhängiger Berater, Dozent, Softwarearchitekt und Fachjournalist
- Leiter des Berater- und Dozententeams bei *www.IT-Visions.de*
- Leitung der Softwareentwicklung im Bereich Microsoft/.NET bei der 5minds IT-Solutions GmbH & Co. KG (*www.5minds.de*)
- Lehrbeauftragter an den Fachhochschulen in Münster und Graz
- 60 Fachbücher bei zahlreichen Verlagen und mehr als 800 Beiträge in Fachzeitschriften
- Gutachter in den Wettbewerbsverfahren der EU gegen Microsoft (2006 – 2009)
- Ständiger Mitarbeiter der Zeitschriften iX (seit 1999), dotnetpro (seit 2000) und Windows Developer (seit 2010) sowie beim Online-Portal heise.de (seit 2008)
- Regelmäßiger Sprecher auf nationalen und internationalen Fachkonferenzen (z.B. Microsoft TechEd, Microsoft Summit, Microsoft IT Forum, BASTA, BASTA-on-Tour, .NET Architecture Camp, Advanced Developers Conference, Developer Week, OOP, DOTNET Cologne, VS One, NRW.Conf, Net.Object Days, Windows Forum)

- Zertifikate und Auszeichnungen von Microsoft:
 - Microsoft Most Valuable Professional (MVP)
 - Microsoft Certified Solution Developer (MCSD)
- Thematische Schwerpunkte:
 - Microsoft .NET Framework, Visual Studio, C#, Visual Basic
 - .NET-Architektur/Auswahl von .NET-Technologien
 - Einführung von .NET Framework und Visual Studio/Migration auf .NET
 - Webanwendungsentwicklung mit IIS, ASP.NET und JavaScript sowie TypeScript
 - Verteilte Systeme, Webservices, Enterprise .NET
 - Relationale Datenbanken, XML, Datenzugriffsstrategien
 - Objektrelationales Mapping (ORM), insbesondere ADO.NET Entity Framework
 - Windows PowerShell (WPS) und Windows Management Instrumentation (WMI)
- Ehrenamtliche Community-Tätigkeiten:
 - Vortragender für die International .NET Association (INETA)
 - Betrieb diverser Community-Websites www.dotnetframework.de, www.entwicklerlexikon.de, www.windows-scripting.de, www.aspnetdev.de u. a.
- Weblog: www.dotnet-doktor.de
- Kontakt: buero@IT-Visions.de sowie Telefon 02 01 64 95 90-0

9

Scripting der Benutzerverwaltung

Dieses Kapitel versetzt den Administrator in die Lage, selbst komplexere Vorgänge in der Benutzerverwaltung durch das Zusammenführen einzelner Vorgänge zu vereinfachen. Benutzerverwaltung soll hier im weiteren Sinne auch Benutzergruppen und Benutzercontainer umfassen.

Lernziele

Die Verwaltung von Benutzerkonten in Unternehmensnetzen gewinnt immer mehr an Bedeutung. Während das Verwalten einzelner Benutzer durch den Administrator noch in endlicher Zeit erledigt werden kann, gestaltet sich das Verwalten der Benutzerkonten in komplexen Netzwerken sehr aufwendig. Hier verschafft die Skriptprogrammierung dem Administrator die Möglichkeit, lästige Aufgaben durch einfaches Aufrufen von Skripten zu erledigen.

Vereinfachte Administration



HINWEIS: Aufgrund unterschiedlicher Anforderungen und Vorgehensweisen ist dieses Kapitel getrennt in die Benutzerverwaltung für lokale Benutzerkonten und Active-Directory-basierte Systeme. Die Benutzerverwaltung für lokale Benutzer in Windows-Clients und Windows Server erfolgt heute in allen Windows-Betriebssystemen bis hin in Windows 10 und Windows Server 2016 noch auf die gleiche Weise wie einst in Windows NT. Die Active-Directory-Benutzerverwaltung kann wirklich nur auf das Active Directory angewendet werden.

ADSI

Die Benutzerverwaltung basiert auf der Komponente ADSI. Es gibt zwar auch einige Klassen in WMI für die Benutzerverwaltung, die Verwaltung mit ADSI ist jedoch einfacher und vollständiger, sodass sie hier verwendet wird.

ADSI



HINWEIS: Als wichtige Begriffe seien noch einmal wiederholt: Ein Container ist ein Verzeichnisobjekt, das andere Verzeichnisobjekte enthalten darf. Über einen Container kann man mit `For Each` eine Schleife bilden. Ein Blatt ist ein Verzeichnisobjekt, das keine Unterobjekte enthält; somit ist eine `For Each`-Schleife nicht möglich.

■ 9.1 Benutzerverwaltung für lokale Benutzerkonten

Flache
Struk-
turen

Die Frage, ob nicht Active-Directory-basierte Windows-Versionen überhaupt einen Verzeichnisdienst haben, führt gewöhnlich zu hitzigen Diskussionen, da diese Betriebssysteme alle Verzeichnisobjekte in flachen Strukturen verwalten. Es existieren nur einige wenige Container und auch das Anlegen von eigenen Untercontainern wird nicht unterstützt. Aus Gründen der Einfachheit verwenden wir hier jedoch den Begriff Verzeichnisdienst auch für NT4-Domänen.



HINWEIS: Ebenfalls aus Gründen der Vereinfachung wird in diesem Kapitel immer von der NT4-Benutzerverwaltung gesprochen.

Die hier vorgestellten Verfahren gelten für:

- Windows-NT-Domänen (vor Windows 2000),
- Windows-Client ab Version 2000 Professional bis zum heute aktuellen Windows 10,
- Windows Server ab Version 2003 (bis einschließlich des heute aktuellen Windows Server 2016), die nicht Domänencontroller sind.

9.1.1 Anlegen eines Benutzerkontos

Benutzer
erzeugen

Vor dem Anlegen eines neuen NT-Benutzerkontos muss zunächst die Bindung an den übergeordneten Domain- oder an einen Computer-Container hergestellt werden. Dazu wird bei `GetObject()` ein ADSI-Pfad zu einem Computer oder einer Domäne angegeben. Der Pfad ist sehr einfach:

`WinNT://COMPUTERNAME` oder `WinNT://DOMÄNENNAME`



ACHTUNG: Auch wenn dies in Kapitel 5 schon mehrfach erwähnt wurde, sei hier dennoch erneut die Warnung ausgesprochen: WinNT müssen Sie mit großen (W), (N), (T) und kleinem (i), (n) schreiben. Die häufigste Ursache für nicht funktionierende ADSI-Skripte ist die falsche Schreibweise dieses Begriffs. Dieser Fehler tritt so häufig auf, weil die Relevanz der Groß- und Kleinschreibung in der VBScript-Programmierung sehr selten ist.



TIPP: Die zusätzliche Angabe des Klassennamens im ADSI-Pfad beschleunigt den Aufruf, weil ADSI dann genau weiß, wonach es suchen soll. Der Klassenname kann am Ende des Pfads durch ein Komma getrennt angegeben werden:

`WINNT://COMPUTERNAME, Computer` oder `WinNT://DOMÄNENNAME, Domain`

Grundsätzlich wird in ADSI ein Objekt mit der Methode `Create()` angelegt. Bei der Methode `Create()` sind der Klassenname `user` und als zweiter Parameter der gewünschte Benutzername anzugeben. Erst mit dem Aufruf von `SetInfo()` wird der Benutzer tatsächlich angelegt. Create()

Die User-Klasse verlangt keine Pflichtattribute; im Skript werden allerdings die folgenden optionalen Attribute verwendet: Attribute

- **FullName:** kennzeichnet den Anzeigenamen des Benutzers
- **Description:** eine Beschreibung des Benutzers
- **HomeDirectory:** der Pfad zu dem Verzeichnis, in dem der Benutzer seine Daten ablegt
- **AccountExpirationDate:** Datum, an dem das Konto ungültig wird
- **PasswordExpirationDate:** Datum, an dem das Kennwort des Kontos abläuft. `PasswordExpirationDate` kann aber nicht beschrieben werden. Das Ablaufdatum kann nur beeinflusst werden über `MaxPasswordAge` auf Domänen- bzw. Computerebene. Damit der Benutzer nach dem Anmelden sein Kennwort ändern muss, setzt man `Benutzer.PasswordExpired = 1`.
- **LoginScript:** das bei der Anmeldung des Benutzers auszuführende Skript



TIPP: Tipp: Bitte passen Sie in diesem Skript unbedingt den Namen des Containers an, bevor Sie es testen. Tragen Sie in die Konstante `CONTAINERNAME` den Namen eines Computers oder einer Domäne ein, die bei Ihnen erreichbar ist. Selbstverständlich müssen Sie Administratorrechte auf dem Computer bzw. der Domäne besitzen, um das Skript ausführen zu können.

Listing 9.1: /Skripte/Kapitel08/WinNT/BenutzerAnlegen.vbs

```
' BenutzerAnlegen.vbs
' Anlegen eines Benutzerkontos
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Benutzer
Dim Domaene
' Name des Containers, in dem der Benutzer angelegt werden soll
Const CONTAINERNAME = "PC171" ' Computernamen oder Domänenname
Const KLASSE = "Computer" ' oder: Domain
' Zugriff auf Domain-Objekt
Set Domaene = GetObject("WinNT://" & CONTAINERNAME & "," & KLASSE)
' Benutzer anlegen
Set Benutzer = Domaene.Create("user", "WilliWinzig3")
' Setzen von Eigenschaften
' Voller Name
Benutzer.FullName = "Willi Winzig"
' Beschreibung des Benutzers
Benutzer.Description = "Herr Willi Winzig ist unser neuer Mitarbeiter."
' Home-Verzeichnis des Benutzers
Benutzer.HomeDirectory = "e:\homes\winzig"
' Ablaufdatum des Kontos: 1 Jahr
```

```

Benutzer.AccountExpirationDate = Now() + 365
' Verweis auf das Login-Skript
Benutzer.LoginScript = "benutzer.bat"
' Kennwort setzen
Benutzer.SetPassword "SehrGeheim123"
' Kennwortänderung bei erster Anmeldung erzwingen
Benutzer.PasswordExpired = 1
' Festschreiben der Werte
Benutzer.SetInfo
' Meldung ausgeben
WScript.Echo "Benutzer angelegt!"

```

In den folgenden Bildschirmabbildungen werden bewusst verschiedene neuere Betriebssysteme verwendet, um zu beweisen, dass diese Vorgehensweise auch in modernen Betriebssystemen und im Zeitalter des Active Directory noch relevant ist. Viele Administratoren glauben fälschlicherweise, die Benutzerkontenverwaltung in einem Netzwerk mit Active Directory würde komplett über LDAP laufen.

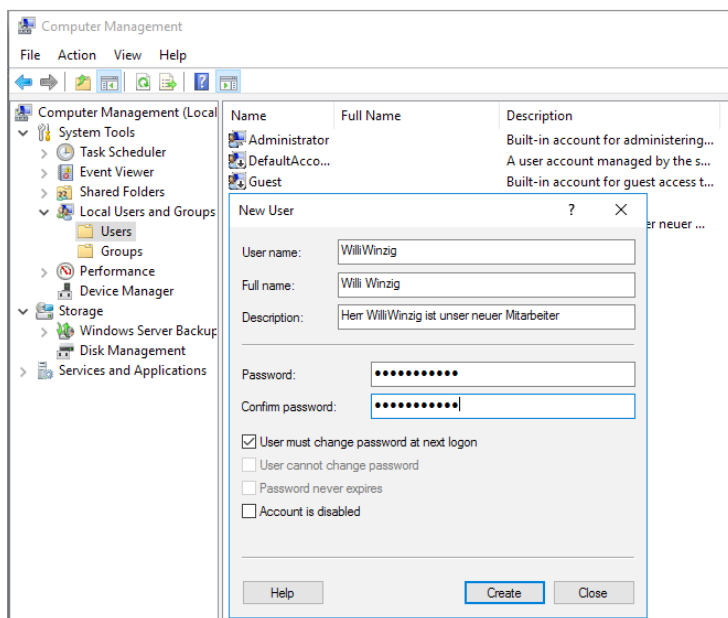


Abbildung 9.1: Anlegen des Benutzers Willi Winzig als lokalen Benutzer

Der neue Benutzer WilliWinzig erscheint aber nicht in der Benutzerkontenverwaltung der Systemsteuerung, weil er nicht Mitglied der Standardgruppe „Benutzer“ ist. Der neu angelegte Benutzer gehört zunächst zu keiner Gruppe und er hat auch noch kein Kennwort. Diese beiden Schritte erfolgen in den nächsten Skripten.

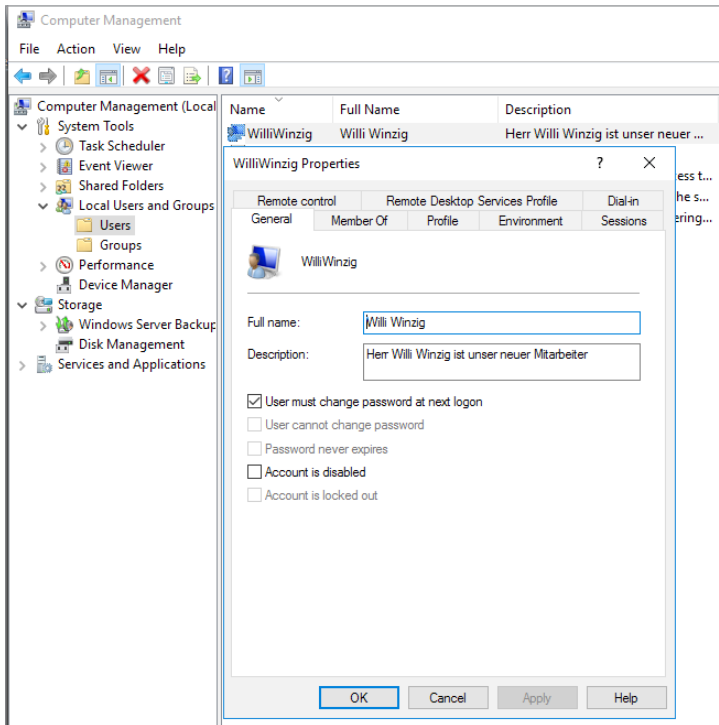


Abbildung 9.2: Anzeige des neuen Benutzerkontos (hier: Windows Server 2016)

9.1.2 Umbenennen eines Benutzers

Der NT4-Verzeichnisdienst erlaubt die Umbenennung eines Benutzerkontos nach dem Anlegen, da für die eindeutige Identifizierung nicht der Kontoname, sondern der Security Identifier (SID) des Kontos maßgeblich ist. Das Konto verliert also nicht seine Gruppenzuordnungen oder Rechte. Die Methode zur Umbenennung heißt in ADSI `MoveHere()`. Diese Methode wird sowohl von der `Computer`- als auch von der `Domain`-Klasse unterstützt.

Namens-
änderung



ACHTUNG: Es ist nicht möglich, ein lokales Benutzerkonto zu verschieben, weil es pro Computer nur einen Container für Benutzer geben kann. Eine Verschiebung zwischen Containern ist nicht möglich.

Das Skript deklariert die benötigten Variablen für die Objekte. Durch den Aufruf von `GetObject()` wird eine Instanz des `Domain`-Objekts erzeugt und der Variablen `Container` zugewiesen. Als Parameter werden der WinNT-Provider und der Name des Computers angegeben, auf dem sich das Benutzerkonto befindet.

Umbenennen durch Verschieben

Der Aufruf der `MoveHere()`-Methode des `Domain`-Objekts mit dem ADSI-Pfad des Benutzers sowie dem neuen Benutzernamen führt die Umbenennung durch. Die erfolgreiche Umbenennung wird durch eine Meldung angezeigt.

Listing 9.2: /Skripte/Kapitel08/WinNT/BenutzerUmbenennen.vbs

```
' BenutzerUmbenennen.vbs
' Umbenennen eines Benutzerkontos
' verwendet: ADSI
' =====
Option Explicit

' Konstanten - bitte anpassen!!!
Const CONTAINERNAME = "PC171" ' Computername oder Domänenname
Const ALTERNAME = "WilliWinzig"
Const NEUERNAME = "WilliWichtig"

' Notwendige Variablen deklarieren
Dim Container
' Zugriff auf Domain-Objekt
Set Container = GetObject("WinNT://" & CONTAINERNAME)
' MoveHere ausführen
Container.MoveHere _
"WinNT://" & CONTAINERNAME & "/" & ALTERNAME, NEUERNAME
' Meldung ausgeben
Wscript.Echo "Benutzer umbenannt!"
```

Wie Sie in nachfolgender Bildschirmabbildung sehen können, wird durch die `MoveHere()`-Methode nur der Benutzername selbst, nicht aber die anderen Attribute wie `FullName` und `Description` beeinflusst.



ACHTUNG: Bitte beachten Sie, dass nach Ausführung dieses Skripts das Benutzerkonto „WilliWinzig“ nicht mehr existiert. Da die nachfolgenden Skripte „WilliWinzig“ verwenden, sollten Sie mithilfe des ersten Skripts in diesem Kapitel „WilliWinzig“ wieder anlegen.

9.1.3 Kennwort eines Benutzers ändern

Grundsätzlich gibt es zwei Möglichkeiten, ein Kennwort mit ADSI zu setzen:

Set
Password()

- Bei `SetPassword()` ist die Angabe des bisherigen Kennworts nicht nötig.
- Bei der Methode `ChangePassword()` muss das bisherige Kennwort angegeben werden.

Change
Password()

`ChangePassword()` sollte angewendet werden, wenn sichergestellt werden soll, dass nur der betreffende Benutzer selbst das Kennwort ändert. Die Methode lässt sich nur ausführen, wenn die Kontorichtlinien dies erlauben (wenn Sie das Skript ausgeführt haben, das die minimale Kennwortdauer auf zehn Tage setzt, dann kann `ChangePassword()` erst nach zehn Tagen zum ersten Mal ausgeführt werden!).



TIPP: Für den Administrator ist die Methode `SetPassword()` gedacht, da das alte Kennwort nicht bekannt sein muss. `SetPassword()` kann nicht nur beim erstmaligen Setzen, sondern zu beliebiger Zeit ausgeführt werden.



ACHTUNG: In älteren Windows-Versionen (vor Windows Server 2003 und Windows XP mit Service Pack 2) konnte das Kennwort beim Anlegen eines neuen Benutzers erst gesetzt werden, nachdem das Anlegen mit `SetInfo()` vollzogen wurde. Damit ist eine potenzielle Sicherheitslücke geschlossen. Für den LDAP-Provider gilt jedoch die Aussage „Erst Konto komplett anlegen, dann Kennwort setzen“ immer noch. Das potenzielle Risiko kann hier dadurch umgangen werden, dass das Konto, das im Standard deaktiviert ist, erst nach der Kennwortvergabe aktiviert wird!

Listing 9.3: /Skripte/Kapitel08/WinNT/KennwortAendern1.vbs

```
' KennwortAendern1.vbs
' Setzen eines Kennworts für ein Benutzerkonto
' verwendet: ADSI
' =====
Dim Benutzer
' Bitte ADSI-Pfad anpassen: WinNT://CONTAINER/BENUTZERNAME
Set Benutzer = GetObject("WinNT://PC171/WilliWinzig,user")
Benutzer.SetPassword "HeImut"
Msgbox "Kennwort gesetzt!"
```

Listing 9.4: /Skripte/Kapitel08/WinNT/KennwortAendern2.vbs

```
' KennwortAendern2.vbs
' Ändern eines Kennworts für ein Benutzerkonto
' verwendet: ADSI
' =====
Dim Benutzer
' Bitte ADSI-Pfad anpassen: WinNT://CONTAINER/BENUTZERNAME
Set Benutzer = GetObject("WinNT://PC171/WilliWinzig,user")
Benutzer.ChangePassword "HeImut", "Gerhard"
Msgbox "Kennwort geändert!"
```



TIPP: Um den Benutzer zu veranlassen, sein Kennwort bei der nächsten Anmeldung zu ändern, wird die Eigenschaft `AccountExpirationDate` auf das aktuelle Datum und die aktuelle Uhrzeit gesetzt.

9.1.4 Anlegen einer Benutzergruppe

Das Einrichten einer Gruppe erfolgt analog zur Erstellung eines User-Objekts. Beachten Sie aber den bei `Create()` anzugebenden Klassennamen `Group`.

Andere
Klasse

Lokal oder
global

GroupType ist ein Pflichtattribut des lokalen Benutzerkontos, das aber automatisch auf den Wert 2 (globale Gruppe) gesetzt wird, wenn der ADSI-Client keinen Wert vorgibt. Das Beispielskript allerdings erzeugt eine lokale Gruppe (Wert 4).

Listing 9.5: /Skripte/Kapitel08/WinNT/GruppeAnlegen.vbs

```
' GruppeAnlegen.vbs
' Anlegen einer lokalen Gruppe
' verwendet: ADSI
' =====

Option Explicit
' Variablen deklarieren
Dim Container
Dim Gruppe
' Konstanten definieren
Const GRUPPENNAME = "Finanzbeamte"
' Name des Containers, in dem der Benutzer angelegt werden soll
Const CONTAINERNAME = "PC171" ' Computername oder Domänenname

' Zugriff auf Domain-Objekt
Set Container = GetObject("WinNT://" & CONTAINERNAME)
' Erzeugen der Gruppe
Set Gruppe = Container.Create("group", GRUPPENNAME)
' Gruppentyp setzen
' 4 = Lokale Gruppe, 2= Globale Gruppe
Gruppe.Put "GroupType", 4
' Beschreibungstext setzen
Gruppe.Description = "Beispielgruppe für das Buch 'Windows Scripting lernen'"
' Festschreiben der Änderungen
Gruppe.SetInfo
' Meldung ausgeben
WScript.Echo "Gruppe " & GRUPPENNAME & " wurde angelegt!"
```

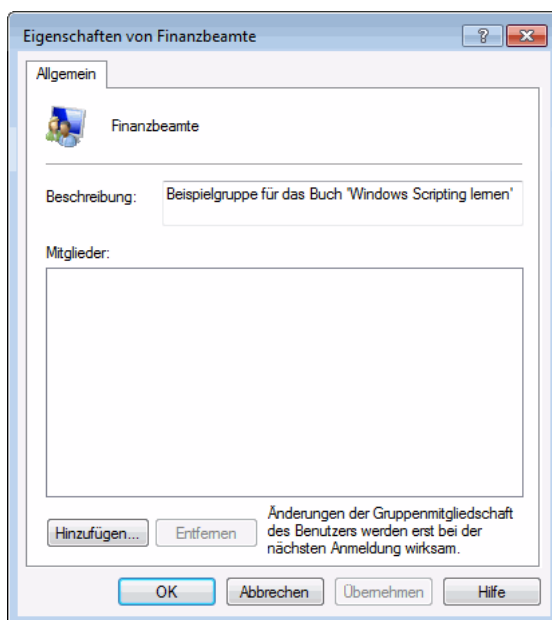


Abbildung 9.3:
Neue Gruppe erstellt

9.1.5 Hinzufügen eines Benutzers zu einer Gruppe

Das folgende Skript ordnet einen bestehenden Benutzer einer existierenden Gruppe zu. Das Skript deklariert die benötigte Variable für das Group-Objekt. Durch den Aufruf von `GetObject()` wird eine Instanz des Group-Objekts erzeugt und der Variablen `Gruppe` zugewiesen. Durch Aufruf der `Add()`-Methode des Group-Objekts wird der als Parameter übergebene Benutzer der Gruppe zugeordnet.

Der Befehl `SetInfo()` ist hier nicht notwendig, die Änderung wird sofort wirksam.



ACHTUNG: Der Benutzer muss bereits vorhanden sein, ansonsten wird die Fehlermeldung „Ein Mitglied konnte in der lokalen Gruppe nicht hinzugefügt oder entfernt werden, da das Mitglied nicht vorhanden ist.“ ausgegeben.

Listing 9.6: /Skripte/Kapitel08/WinNT/BenutzerzuGruppe.vbs

```
' BenutzerzuGruppe.vbs
' Hinzufügen eines Benutzers zu einer Gruppe
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Gruppe
' Zugriff auf das Gruppen-Objekt
Set Gruppe = GetObject("WinNT://PC171/Finanzbeamte,Group")
' Hinzufügen des Benutzer-Objekts zur Gruppe
Gruppe.Add ("WinNT://PC171/WilliWinzig")
' Meldung ausgeben
Wscript.Echo "Benutzer WilliWinzig zur Gruppe 'Finanzbeamte' hinzugefügt!"
```

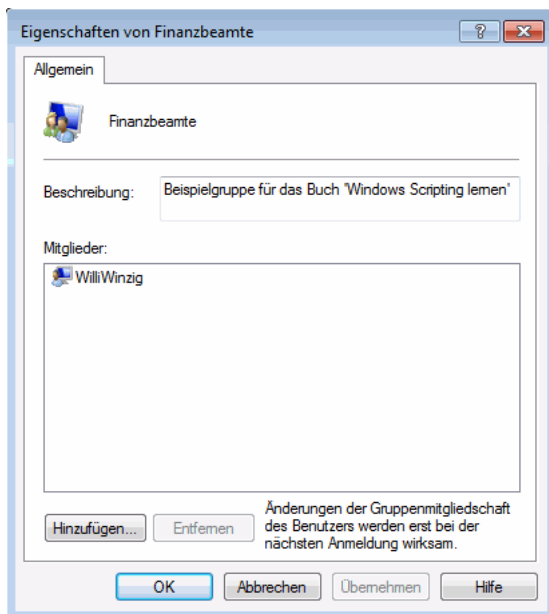


Abbildung 9.4:
Benutzer in die Gruppe eingefügt

9.1.6 Entfernen eines Benutzers aus einer Gruppe

Benutzer
entfernen

Das nachfolgende Skript *LoescheBenutzerausGruppe.vbs* entfernt einen Benutzer aus einer Benutzergruppe. Zentraler Befehl ist die Methode `Remove()`, die auf einem Group-Objekt ausgeführt wird. Als Parameter erwartet `Remove()` den ADSI-Pfad des Benutzers, der aus der Gruppe entfernt werden soll.

Identifi-
kation

Der Variablen `Gruppe` wird durch `GetObject()` ein Verweis auf das Group-Objekt der betreffenden Gruppe zugewiesen. Danach wird `Remove()` ausgeführt. Der Befehl `SetInfo()` ist hier nicht notwendig, die Änderung wird sofort wirksam.

Listing 9.7: /Skripte/Kapitel08/WinNT/LoescheBenutzerausGruppe.vbs

```
' LoescheBenutzerausGruppe.vbs
' Löschen eines Benutzerkontos aus einer Gruppe
' verwendet: ADSI
' =====
Option Explicit
Dim Benutzer
Dim Gruppe
' Zugriff auf das Gruppen-Objekt
Set Gruppe = GetObject("WinNT://PC171/Finanzbeamte,group")
' Benutzer-Objekt aus der Gruppe entfernen
Gruppe.Remove("WinNT://PC171/WilliWinzig,user")
Wscript.Echo "Der Benutzer WilliWinzig wurde aus der " & _
    "Gruppe Finanzbeamte entfernt."
```

9.1.7 Deaktivieren eines Benutzerkontos

Konto
sperrern

Soll einem Benutzer der Zugang zum Netzwerk nur kurzfristig entzogen werden, muss das Konto nicht gelöscht, sondern es kann deaktiviert werden. Ein Konto kann auch gesperrt werden. Dies hat denselben Effekt wie die Deaktivierung.

Das nachfolgende Beispiel zeigt, wie mithilfe des Attributs `AccountDisabled` ein Benutzer deaktiviert wird, sodass er sich nicht mehr am Netz anmelden kann. Dazu wird mit `GetObject()` ein Verweis auf das User-Objekt erstellt und in der Variablen `Benutzer` gespeichert. Durch Setzen der Eigenschaft `AccountDisabled` auf den Wert `True` wird das User-Objekt angewiesen, das Konto zu sperren. Die Sperrung erfolgt erst mit dem Aufruf von `SetInfo()`.

Ent-
sperrung

Die Umkehrung der Aktion ist mit der Zuweisung des booleschen Werts `False` an die Eigenschaft `AccountDisabled` möglich.

Listing 9.8: /Skripte/Kapitel08/WinNT/DeaktiviereKonto.vbs

```
' DeaktiviereKonto.vbs
' Deaktivieren eines Benutzerkontos
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Benutzer
```

```
' Zugriff auf User-Objekt
Set Benutzer= GetObject("WinNT://PC171/WilliWinzig,user")
' Deaktivierung
Benutzer.AccountDisabled = True ' = False zum Reaktivieren!
' Cache schreiben
Benutzer.SetInfo
' Meldung ausgeben
Wscript.Echo "Benutzer WilliWinzig deaktiviert!"
```

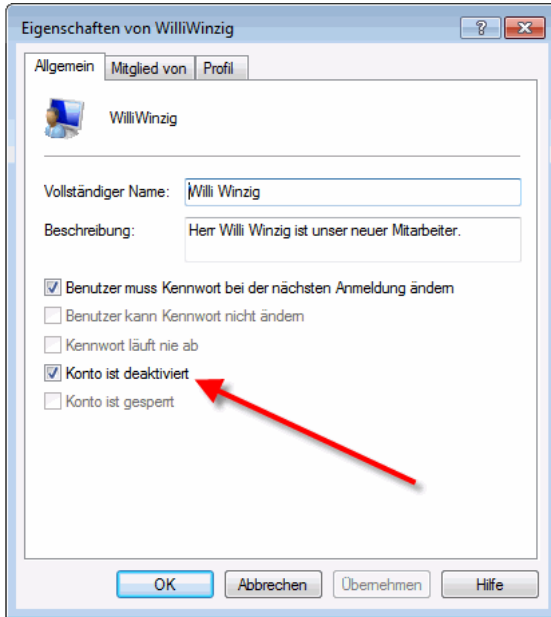


Abbildung 9.5:
Benutzerkonto ist deaktiviert.

9.1.8 Löschen einer Gruppe

Zentraler Befehl beim Löschen eines Objekts ist die Methode `Delete()`, die nur von Container-Objekten (also den Klassen `Domain` und `Computer`) bereitgestellt wird. `Delete()` erwartet nicht nur den Namen des zu löschenden Objekts, sondern zuvor im ersten Parameter auch den Klassennamen. `Delete()`

Im nächsten Beispiel wird das Löschen einer Gruppe demonstriert. Dazu wird nach der Variablendeklaration der Variablen `Container` der Verweis auf das durch `GetObject()` erzeugte `Domain`- oder `Computer`-Objekt zugewiesen. Unter Angabe des Klassennamens `Group` und des Gruppennamens löscht die `Delete()`-Methode die Gruppe aus dem Container.



ACHTUNG: Bitte verwechseln Sie nicht die Methoden `Remove()` und `Delete()`. `Remove()` entfernt einen Benutzer aus einer Gruppe. Eine Gruppe gilt nicht als ein Container-Objekt, weil die Gruppe den Benutzer im engeren Sinne nicht enthält, sondern nur einen Verweis auf den Benutzer speichert.

Ein Objekt kann immer nur in einem Container sein. Wäre die Gruppe ein Container, könnte der Benutzer nur Mitglied einer einzigen Gruppe sein. Nach einem `Remove()` ist das Benutzerkonto immer noch vorhanden. `Delete()` dagegen entfernt einen Benutzer aus einem Container, sodass er permanent gelöscht wird.

Sofortige
Löschung

Ein expliziter Aufruf von `SetInfo()` ist hier nicht notwendig. Die Löschung wird sofort durchgeführt.

Listing 9.9: /Skripte/Kapitel08/WinNT/LoescheGruppe.vbs

```
' LoescheGruppe.vbs
' Löschen einer Gruppe
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Container
' Konstanten definieren
Const GRUPPENNAME ="Finanzbeamte"
' Zugriff auf Domain-Objekt
Set Container = GetObject("WinNT://ServerE02")
' Löschen der Gruppe
Container.Delete "group", GRUPPENNAME
' Meldung ausgeben
Wscript.Echo "Gruppe " & GRUPPENNAME & " wurde gelöscht!"
```

9.1.9 Löschen eines Benutzers

Verwechslungen
möglich

Ein Benutzer wird gelöscht durch den Aufruf der `Delete()`-Methode des Containers, in dem er enthalten ist. Das folgende Beispiel zeigt das Löschen eines Domänenbenutzers. Bei der `Delete()`-Methode ist – wie beim Erzeugen – der Klassenname `User` anzugeben, um Verwechslungen mit eventuell gleichnamigen `Group`-Objekten zu vermeiden. Der Aufruf von `SetInfo()` ist nicht notwendig; `Delete()` wird sofort ausgeführt!

Listing 9.10: /Skripte/Kapitel08/WinNT/LoescheBenutzer.vbs

```
' LoescheBenutzer.vbs
' Löschen eines Benutzerkontos
' verwendet: ADSI
' =====
Option Explicit
' Variable deklarieren
Dim Container
' Zugriff auf Domain-Objekt
Set Container = GetObject("WinNT://ServerE02")
' Benutzer löschen
Container.Delete "user", "WilliWinzig"
' Meldung ausgeben
Wscript.Echo "Benutzer gelöscht!"
```

■ 9.2 Active-Directory-Benutzerverwaltung unter Windows Server

Die folgenden Beispiele demonstrieren den Umgang mit der Benutzerverwaltung in einem Active Directory (kurz: AD) unter Windows Server (ab Version Windows 2000 Server bis einschließlich Windows Server 2016). Alle Beispiele setzen die im ersten Unterkapitel erzeugte Organisationseinheit *WSH-Scripting* voraus.



ACHTUNG: Weil dieser Fehler sehr häufig gemacht wird, seien Sie an dieser Stelle noch einmal gewarnt: Sie können mit den folgenden Skripten wirklich nur die Objekte in einem Active Directory im engeren Sinne verwalten. Benutzer und Gruppen, die lokal auf einem Windows-Client oder Windows-Server, der nicht Domänencontroller ist, existieren, gehören nicht zum Active Directory. Diese Objekte werden wie ein NT4-System verwaltet (vgl. vorheriges Unterkapitel). Gleiches gilt für einen Windows 2000 Server oder einen Windows Server 2003 (inkl. R2) oder einen Windows Server 2008 (inkl. R2), auf dem das Active Directory nicht installiert ist.

Microsoft hat in der Vergangenheit leider viele Benutzer mit einer zu globalen Verwendung des Begriffs Active Directory Service in die Irre geführt. Dies zeigt sich zum Beispiel auch am Namen der Komponente Active Directory Service Interface (ADSI). Wie bereits in Kapitel 5 geschildert, ist diese Komponente keineswegs nur für das Active Directory zuständig.

9.2.1 Anlegen einer Organisationseinheit

Eine hervorstechende Eigenschaft des Active Directory (gegenüber der NT4-Benutzerverwaltung) besteht darin, beliebige Organisationsstrukturen in Form von Containern abbilden zu können. Ein solcher Container heißt im Active Directory *Organizational Unit*.

Ver-schach-telte Ein-heiten

Das Beispielskript erstellt eine Organisationseinheit im Active Directory. Dazu wird nach der Variablendeklaration ein Verweis auf das RootDSE-Objekt erzeugt und in der Variablen *Wurzel* abgelegt. RootDSE kennzeichnet das oberste Element in einem Active Directory.

Oberste Ebene



ACHTUNG: Wichtig ist auch hier, dass Sie LDAP komplett in Großbuchstaben schreiben. Die Schreibweise der nachfolgenden Wörter ist jedoch egal (*rootdse*, *ROOTDSE*, *rootDSE* etc.).

Nun wird der LDAP-Pfad des Wurzelverzeichnisses vom Active Directory durch Abfrage der Eigenschaft *defaultNamingContext* ermittelt und in der Variablen *Domaene* gespeichert.



TIPP: Die Verwendung des RootDSE-Objekts bringt den Vorteil, dass Sie nicht den kompletten LDAP-Pfad zu Ihrem Active Directory wissen müssen. Selbstverständlich können Sie den Pfad aber auch manuell angeben. Dies ist in vielen der folgenden Skripte gezeigt.

Viele
Eigen-
schaften

Beachten Sie beim Anlegen von Organisationseinheiten den Klassennamen (organizationalUnit) im ersten Parameter und das dem eigentlichen Containernamen vorangestellte OU= im zweiten Parameter bei Create().

Die Organisationseinheiten stellen neben vielen Eigenschaften unter anderem die Eigenschaft Description zur Verfügung, die eine Beschreibung des Objekts zulässt. Erst durch den Aufruf von SetInfo() wird das Objekt im Active Directory abgelegt.

Listing 9.11: /Skripte/Kapitel08/LDAP/ErzeugeOU.vbs

```
' ErzeugeOU.vbs
' Erzeugen einer Organisationseinheit im Active Directory
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Wurzel, Domaene, OrgEinheit
' Konstanten definieren
Const OUName="WSH-Scripting"
' Oberstes Element des AD holen
Set Wurzel = GetObject("LDAP://RootDSE")
' Wurzelverzeichnis bestimmen
Set Domaene = GetObject("LDAP://" & Wurzel.Get("defaultNamingContext"))
' OU anlegen
Set OrgEinheit = Domaene.Create("organizationalUnit", "OU=" & OUName)
' Beschreibung setzen
OrgEinheit.Description = "Dies ist eine OU für das Scripting-Buch"
' Werte festschreiben
OrgEinheit.SetInfo
' Ausgabe
WScript.Echo "OU wurde angelegt:" & OrgEinheit.AdsPath
```

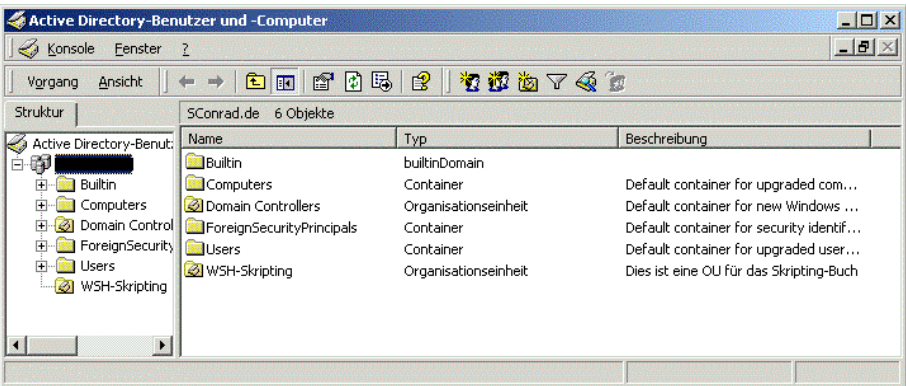


Abbildung 9.6: Die Organisationseinheit WSH-Scripting wurde erzeugt.

9.2.2 Anlegen eines Organisationseinheitenbaums im Active Directory

Durch den hierarchischen Aufbau ist das Active Directory in der Lage, Container-Objekte auf mehreren Ebenen aufzunehmen. Das Beispiel in diesem Abschnitt zeigt, wie man aus einer XML-Datei eine Organisationseinheitenhierarchie anlegt.

Unter-geordnete Organisations-einheit anlegen

Das Finden der richtigen Organisationsstruktur ist oftmals ein Politikum beim Aufbau einer Active-Directory-Infrastruktur. Benötigt wird ein Instrument, mit dem man eine umfangreiche Hierarchie von Organisationseinheiten auf einfache Weise definieren und mit dessen Hilfe man die Organisationseinheiten nachher schnell implementieren kann.

Für hierarchische Daten hat sich der XML-Standard inzwischen durchgesetzt. Die folgende Abbildung zeigt, wie man eine Organisationsstruktur in XML-Form definieren könnte. Wenn man eine solche Hierarchie einmal definiert hat, liegt es nahe, ein Skript zu verwenden, das diese Hierarchie automatisch in das Active Directory einfließen lässt.

```
<?xml version="1.0" encoding="utf-8" ?>
- <OUstruktur>
- <OU Name="www.IT-Visions.de">
  <OU Name="Geschäftsführung" />
  <OU Name="Experten">
    <OU Name="DOTNET" />
    <OU Name="Scripting" />
  - <OU Name="Windows">
    <OU Name="Client" />
    <OU Name="Server" />
  </OU>
</OU>
<OU Name="Verwaltung" />
</OU>
</OUstruktur>
```

Abbildung 9.7:

Eingabedatei für das Anlegen einer AD-OU-Struktur

9.2.2.1 Das Skript

Das Skript ist trotz seiner Mächtigkeit überschaubar. Neben dem Active Directory Service Interface (ADSI) kommt eine weitere Scripting-Komponente, das Microsoft XML Document Object Model (MSXML), zum Einsatz, um die XML-Datei zu lesen. Durch rekursive Programmierung (die Routine ParseXMLDokument() ruft sich immer wieder selbst auf, wenn es noch eine Unterebene gibt) kann man das Skript sehr prägnant halten. Das Ergebnis des Skripts ist in der nachfolgenden Bildschirmabbildung zu sehen.

Listing 9.12: /Skripte/Kapitel08/LDAP/OUs_AusXmlDateiAnlegen.vbs

```
' -----
' Skriptname: OU_SausXmlDateiAnlegen.vbs
' Autor: Dr. Holger Schwichtenberg 2006-2007
' -----
' Dieses Skript erstellt eine OU-Struktur aus einer XML-Datei
' -----
' Verwendet SCRRun, MSXML, ADSI
' -----
```

Option Explicit

```

' Parameter
Const WURZEL = "LDAP://E02/dc=it-visions,dc=local"
Const EINGABEDATEI = "OUStruktur.xml"

' Deklaration der Variablen
Dim Datei, WSHShell
Dim XMLDokument
Dim wurzelknoten

' COM-Objekte erstellen
Set WSHShell = CreateObject("Wscript.shell")
Set XMLDokument = CreateObject("Msxml2.DOMDocument")

' Pfad zur Eingabedatei
Datei = WSHShell.CurrentDirectory & "/" & EINGABEDATEI

' Asynchrones Laden ausschalten
XMLDokument.async = False
' Datei laden
XMLDokument.load(Datei)
WScript.Echo "Dokument geladen"
' Fehler?
If XMLDokument.parseError.reason <> "" Then MsgBox XMLDokument.parseError.reason,
,"Fehler"
' Wurzel-Knoten auswählen
Set wurzelknoten = XMLDokument.selectSingleNode("OUStruktur")
' Durchlauf starten bei Wurzel
ParseXMLDokument 0, wurzelknoten, WURZEL

' === Alle Kind-Knoten durchlaufen
Sub ParseXMLDokument(ByVal ebene, ByVal wurzelknoten, ByVal ouwurzel)
Dim OUKnoten, neuou
ebene = ebene + 1
For Each OUKnoten In wurzelknoten.ChildNodes
WScript.echo "OU gefunden in XML-Datei: " & Space(ebene) & OUKnoten.
getAttribute("Name")
neuou = OUAnlegen(ouwurzel,OUKnoten.getAttribute("Name"))
ParseXMLDokument ebene, OUKnoten, neuou ' Rekursion
Next
End Sub

' === OU anlegen
Function OUAnlegen(Vater,OUName)
Dim Domain, OrgEinheit
' Verweis auf die bestehende OU holen
Set Domain = GetObject(Vater)
' Neue OU anlegen
Set OrgEinheit = Domain.Create("organizationalUnit", "OU=" & OUName)
' Beschreibung setzen
OrgEinheit.Description = "Angelegt mit dem Skript von Holger Schwichtenberg!"
' Werte festschreiben
OrgEinheit.SetInfo
' Ausgabe
WScript.Echo "OU wurde angelegt:" & OrgEinheit.ADsPath
OUAnlegen = OrgEinheit.ADsPath
End Function

```



Abbildung 9.8:
Ergebnis der Skriptausführung

9.2.3 Anlegen eines Benutzerkontos

Ähnlich, aber dennoch nicht identisch zu NT4 ist das Anlegen eines Benutzers im Active Directory. Neben dem Verzeichnisnamen benötigt jeder AD-Benutzer als Pflichtattribut einen SAMAccountName. Da bei LDAP anders als bei NT4 der Attributname des Schlüsselattributs (hier „cn“) Teil des Verzeichnisnamens ist, muss dem neuen Benutzernamen in der Create()-Methode getrennt durch ein Gleichheitszeichen der Attributname vorangestellt werden, der der Identifizierung der Instanzen dieser Klasse dient (also: „cn=“).

Andere
Attribute

Das Beispielskript legt einen neuen Benutzer innerhalb der Organisationseinheit *WSH-Scripting* an. Bitte haben Sie Verständnis dafür, dass nicht alle Attribute besprochen werden können, aber das würde den Rahmen dieses Buches sprengen. Die verwendeten Attribute sind:

Benutzer
in Con-
tainer

- SamAccountName: kennzeichnet den Benutzer
- AccountDisabled: aktiviert oder deaktiviert das Konto des Benutzers

Listing 9.13: /Skripte/Kapitel08/LDAP/BenutzerAnlegen.vbs

```
' BenutzerAnlegen.vbs
' Erzeugen eines Benutzerkontos im AD
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Container
Dim Benutzer
' Konstanten definieren
Const BenutzerName="HugoHastig"
' Bindung an Container
Set Container =
GetObject("LDAP://ServerE02/OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Erzeugen des neuen Benutzers
Set Benutzer = Container.Create("user", "CN=" & BenutzerName)
' Attribute setzen
Benutzer.Put "samAccountName", BenutzerName
' Festschreiben der Werte
Benutzer.SetInfo
' Konto aktivieren
Benutzer.AccountDisabled = False
' Festschreiben der Werte
Benutzer.SetInfo
' Meldung ausgeben
WScript.Echo "Benutzer " & Benutzer.AdsPath & " angelegt"
```

Die folgende Darstellung zeigt den Benutzer innerhalb der Organisationseinheit „WSH-Scripting“.

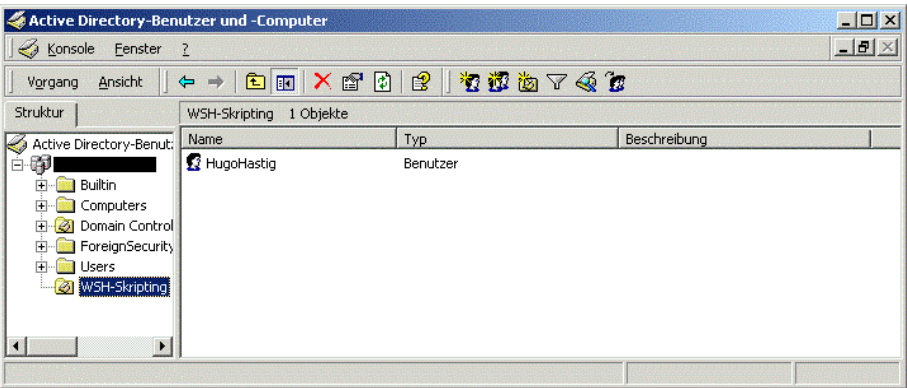


Abbildung 9.9: Der erzeugte Benutzer „HugoHastig“ in der Organisationseinheit „WSH-Scripting“

9.2.4 Anlegen von Benutzern aus einer Access-Datenbank

Große Netzwerke

Das Verwalten einzelner Benutzer kann noch in endlicher Zeit erledigt werden; kritisch wird die Verwaltung der Benutzer in großen Netzwerken. Als Datenbasis für das Anlegen von großen Benutzermengen eignet sich Microsoft Access bestens. Die Daten können über Formulare in einer Access-Anwendung verwaltet werden. Das folgende Skript liest alle in der Tabelle „Benutzer“ in der Datenbank *BenutzerDB.mdb* enthaltenen Benutzer aus und legt diese im Active Directory an.

Wieder-verwend-barer Code

Zum Anlegen der Benutzer wird das Skript aus dem vorherigen Kapitel benutzt. Lediglich die Methode `SetPassword()` wird hier zusätzlich verwendet. Die Methode erlaubt das Setzen des Kennworts beim Anlegen eines Benutzerkontos (vgl. Kapitel 8.2.2).

 **HINWEIS:** Die Verwendung von Access-Tabellen wird in Kapitel 6 beschrieben.

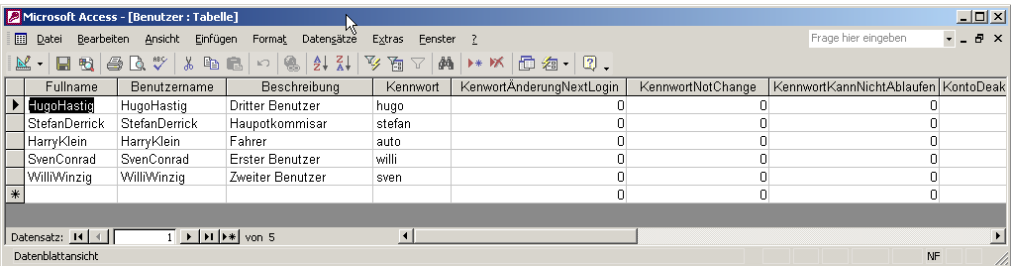


Abbildung 9.10: Die Tabelle „Benutzer“, die als Datenbasis dient

Listing 9.14: /Skripte/Kapitel08/LDAP/BenutzerAnlegenAusDatenbank.vbs

```

' BenutzerAnlegenAusDatenbank.vbs
' Erzeugen von AD-Benutzern aus einer Datenbank
' verwendet: ADSI, ADO
' =====
Option Explicit
' Variablendeklaration
Dim DatenQuelle
Dim DBConnection, SqlString, Ergebnismenge
Dim Container, Benutzer
' Konstanten definieren
Const Verbindung="Provider=Microsoft.Jet.OLEDB.4.0; Data Source=BenutzerDB.MDB;"
' Connection-Objekt erzeugen
Set DBConnection = CreateObject("ADODB.Connection")
' Verbindung öffnen
DBConnection.Open Verbindung
' Alle Benutzer verwenden
SqlString="SELECT * FROM Benutzer"
' SQL-Statement ausführen
Set Ergebnismenge = DBConnection.Execute(SqlString)
On Error Resume Next
' An den Anfang des Abfrageergebnisses springen
Ergebnismenge.MoveFirst
' Bindung an Container
Set Container = GetObject("LDAP://ServerE02/OU=WSH-Scripting,
    DC=IT-Visions,DC=de")
' Durchlaufe gesamte Datenbasis
Do While Not Ergebnismenge.eof
    ' Aufruf der Hilfsroutine
    BenutzerAnlegen Container, Ergebnismenge("Fullname"), _
        Ergebnismenge("Kennwort")
    ' Nächsten Satz aus der Ergebnismenge holen
    Ergebnismenge.MoveNext
Loop
' Schließen der Abfrage
Ergebnismenge.Close
' Schließen der Verbindung
DBConnection.Close

Sub BenutzerAnlegen (Container, Benutzername,Passwort)
' Hilfsroutine: Erzeugen eines Benutzerkontos unter Windows 2000
' Variablen deklarieren

Dim Benutzer
' Erzeugung des neuen Benutzers
Set Benutzer = Container.Create("user", "cn=" & Benutzername)
' Attribute setzen
Benutzer.Put "samAccountName", CStr(Benutzername)
' Festschreiben der Werte
Benutzer.SetInfo
' Konto aktivieren
Benutzer.AccountDisabled = false
Benutzer.SetInfo
' Kennwort des Benutzers setzen
Benutzer.SetPassword Passwort
' Meldung ausgeben
WScript.Echo "Benutzer " & Benutzer.AdsPath & " angelegt"

```

```
' Freigeben der Objekte
End Sub
```

9.2.5 Anlegen einer Benutzergruppe

Um Benutzer in Gruppen verwalten zu können, müssen diese erst angelegt werden. Eine Zuweisung von Benutzern an nicht vorhandene Gruppen erzeugt den Laufzeitfehler „Ein solches Objekt ist auf dem Server nicht vorhanden.“.

Kompati-
bilität

Das nachfolgende Skript generiert nach der Konstantendefinition und Variablendeklaration ein Domain-Objekt durch den Zugriff auf das LDAP-Verzeichnis. Die Create()-Methode erzeugt ein Group-Objekt und legt es in der Variablen Gruppe ab. Dem Namen der Gruppe muss cn= vorangestellt werden.

Im nächsten Schritt werden zwei Eigenschaften des Group-Objekts gesetzt. Die Eigenschaft samAccountName kennzeichnet den Gruppennamen sowohl für die Windows-NT-3.51- bzw. -4.0-Welt als auch für das Active Directory.

Group
Type

Die Eigenschaft GroupType gibt den Gruppentyp an. Für die Gruppentypen existieren nachfolgende Werte, die bitweise verknüpft werden können (OR-Operator):

```
Const GLOBAL_GROUP = 2
Const LOCAL_GROUP = 4
Const UNIVERSAL_GROUP = 8
Const SECURITY_ENABLED = -2147483648
```

Beispiele:

- 4 ist eine lokale Verteilergruppe.
- -2147483644 (=4 or -2147483648) ist eine lokale Sicherheitsgruppe.

Der Aufruf von SetInfo() schreibt die Änderungen im Active Directory fest.

Listing 9.15: /Skripte/Kapitel08/LDAP/GruppeAnlegen.vbs

```
' GruppeAnlegen.vbs
' Erzeugen einer Gruppe im Active Directory
' verwendet: ADSI
' =====
Option Explicit
' Konstantendefinition
Const ADS_GROUP_TYPE_DOMAIN_LOCAL_GROUP = &H4
Const ADS_GROUP_TYPE_SECURITY_ENABLED = &H80000000
Const GruppenName="ScriptingGruppe"
' Variablendeklaration
Dim Gruppe
Dim Domaene
' Domain-Objekt erzeugen
Set Domaene = GetObject _
("LDAP://Server02/OU=WSH-Scripting,DC=IT-Visions,DC=de")
'Erzeugen des Group-Objekts
Set Gruppe = Domaene.Create("group", "CN=" & GruppenName)
' Name für WinNT
Gruppe.Put "samAccountName", GruppenName
```

```
' Gruppentyp setzen
Gruppe.Put "groupType", ADS_GROUP_TYPE_DOMAIN_LOCAL_GROUP _
Or ADS_GROUP_TYPE_SECURITY_ENABLED
' Werte festschreiben
Gruppe.SetInfo
' Ausgabe
WScript.echo "Die Gruppe " & Gruppe.AdsPath & " wurde angelegt."
```

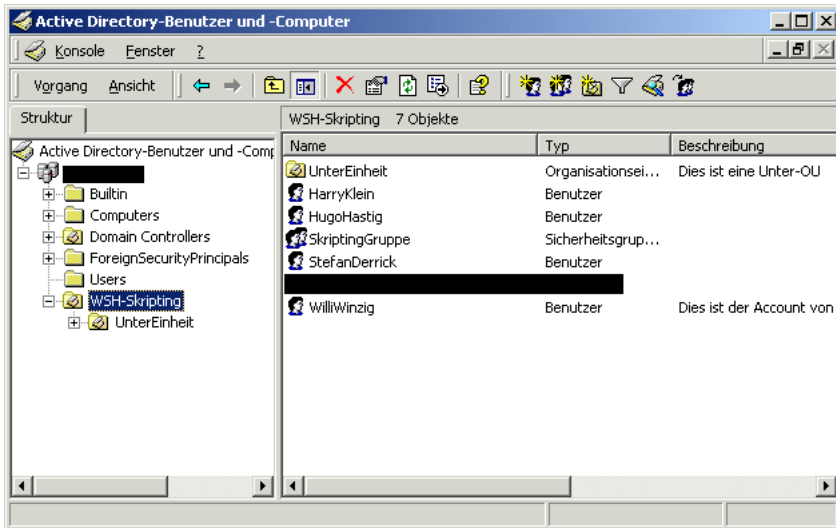


Abbildung 9.11: Die Gruppe „SkriptingGruppe“ wurde angelegt.



TIPP: Das Attribut samAccountName und der Name der Gruppe müssen nicht gleich sein. Hier sind unterschiedliche Zuweisungen möglich.

9.2.6 Hinzufügen eines Benutzers einer Gruppe

Das Verwalten von Benutzern in Gruppen erleichtert dem Administrator die Zuweisung von Rechten an eine Auswahl von Personen. Ein Benutzer kann einer beliebigen Anzahl von Gruppen zugeordnet werden. Das Beispielskript demonstriert eine solche Zuordnung eines Benutzers zu einer Gruppe.

Rechte-
zuweisung

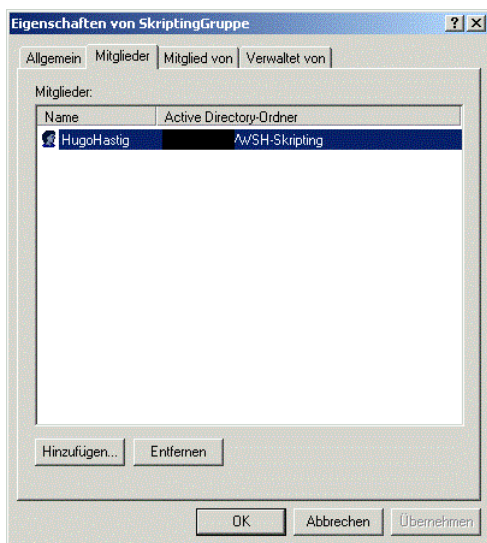


Abbildung 9.12:
Benutzer „HugoHastig“ zur
„ScriptingGruppe“ hinzugefügt

LDAP-
Identifi-
kation

Durch `GetObject()` wird eine Referenz auf das Group-Objekt der angegebenen Gruppe erzeugt und anschließend wird mittels `Add()` der Benutzer der Gruppe zugeordnet. Hierbei ist sowohl die Identifikation der Gruppe als auch die des Benutzers über den kompletten LDAP-Pfad notwendig.

Listing 9.16: /Skripte/Kapitel08/LDAP/BenutzerzuGruppe.vbs

```
' BenutzerzuGruppe.vbs
' Hinzufügen eines Benutzerkontos zu einer Gruppe im AD
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Gruppe
' Konstanten definieren
Const GruppenName="ScriptingGruppe"
Const BenutzerName="HugoHastig"
' Bindung an Gruppen-Container
Set Gruppe = GetObject("LDAP://ServerE02/CN=" & GruppenName & _
",OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Hinzufügen eines Benutzers zur Gruppe
Gruppe.Add ("LDAP://ServerE02/CN=" & BenutzerName & _
",OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Ausgabe
WScript.Echo "Der Benutzer " & BenutzerName & " wurde der Gruppe " _
& GruppenName & " hinzugefügt."
```

9.2.7 Ändern des Kennworts

Das Ändern von Kennwörtern ist eine der am häufigsten vergessenen Aufgaben eines Benutzers. In diesem Beispiel wird dem Administrator ein Skript an die Hand gegeben, mit dem Kennwörter von Benutzern geändert werden können.

Auch hier werden wie bei der Benutzerverwaltung von Windows NT 4.0 zwei Methoden für den Kennwortwechsel angeboten: Wie NT 4.0

- Mit der Methode `ChangePassword()` kann das Kennwort nur unter Angabe des bisherigen Kennworts geändert werden.
- Bei `SetPassword()` ist die Angabe des bisherigen Kennworts nicht erforderlich.



TIPP: Bitte beachten Sie auch die Hinweise zur Kennwortänderung bei der NT4-Benutzerverwaltung (Kapitel 8.2.2).

Im Skript wird durch `GetObject()` ein Verweis auf das Benutzerkonto erzeugt, dessen Kennwort geändert werden soll. Als Parameter ist der komplette LDAP-Pfad auf das User-Objekt anzugeben. Anschließend erfolgt der Aufruf der Methode `SetPassword()`, die das Kennwort auf den neuen Wert setzt. Diese Änderung wird sofort durchgeführt. Sofortige Änderung

Listing 9.17: /Skripte/Kapitel08/LDAP/KennwortAendern.vbs

```
' KennwortAendern.vbs
' Ändern eines Benutzerkennwortes im AD
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Container
Dim Benutzer
' Konstanten definieren
Const BenutzerName="HugoHastig"
Const Kennwort="williw"
' Zugriff auf das Benutzer-Objekt
Set Benutzer = GetObject("LDAP://laptop/CN=" & BenutzerName & _
",OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Kennwort ändern
Benutzer.SetPassword(Kennwort)
' Meldung ausgeben
WScript.Echo "Kennwort für Benutzer " & Benutzer.AdsPath & " wurde geändert"
```

9.2.8 Umbenennen eines Benutzers

Die Umbenennung eines Benutzers wird in diesem Beispiel anhand des folgenden Skripts demonstriert. Dazu wird die Methode `MoveHere()` verwendet. Nicht über Attribute



HINWEIS: Eine Umbenennung über die Zuweisung an die im User-Objekt vorhandenen Attribute ist nicht möglich. So ist es nicht möglich, den Namen, der in der „Active Directory-Benutzer und -Computer“-Konsole angezeigt wird, zu verändern, sondern lediglich die Eigenschaften `givenName`, `samAccountName` und `displayName`. Keines dieser drei Attribute steuert allerdings die Anzeige in besagter Konsole.

Umbenennen durch Verschieben

Um nun einen Benutzer umzubenennen, wird erst ein Verweis auf ein bestehendes Benutzerkonto erzeugt. Dies geschieht durch die Zuweisung des kompletten LDAP-Pfads und den anschließenden Aufruf von `GetObject()`. Nun werden der `MoveHere()`-Methode der LDAP-Pfad zu dem zu ändernden Benutzerkonto sowie der neue Name des Benutzers übergeben.

Listing 9.18: /Skripte/Kapitel08/LDAP/BenutzerUmbenennen.vbs

```
' BenutzerUmbenennen.vbs
' Umbenennen eines Benutzers im Active Directory
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Domaene
' Konstanten definieren
Const NeuerName="WilliRiesig"
Const BenutzerName="HugoHastig"
' Bindung an Domain-Container
Set Domaene = GetObject("LDAP://ServerE02/OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Setzen der Attribute
Domaene.MoveHere "LDAP://ServerE02/CN=" & BenutzerName & _
                 ",OU=WSH-Scripting,DC=IT-Visions,DC=de", "CN=" & NeuerName
Wscript.Echo "Benutzer wurde umbenannt"
```



ACHTUNG: Eine Gesamtübersicht über die Attribute des User-Objekts würde den Rahmen dieses Kapitels sprengen. Um einen Überblick über die einzelnen Attribute von Objekten zu erhalten, hat sich der Microsoft Active Directory Service Browser (ADB) als sehr nützlich erwiesen. Sie finden ihn in den Downloads zu diesem Buch im Verzeichnis */Install/Werkzeuge/ADSI*.

9.2.9 Ändern der Benutzerdaten

Daten-
änderung

Das folgende Skript zeigt, wie man verschiedene Eigenschaften eines vorhandenen Benutzers ändern kann. Die Attributnamen sind sprechend und daher verständlich.

Listing 9.19: /Skripte/Kapitel08/LDAP/BenutzerdatenAendern.vbs

```
' BenutzerdatenAendern.vbs
' Ändern von Benutzerdaten im Active Directory
' verwendet: ADSI
```

```
' =====
Option Explicit
' Variablen deklarieren
Dim Benutzer
' Bindung an Benutzer-Container
Set Benutzer = GetObject("LDAP://ServerE02/CN=WilliRiesig, _
OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Setzen der Attribute
Benutzer.Put "samAccountName", "WilliRiesig"
Benutzer.Put "givenName", "Willi"
Benutzer.Put "sn", "Riesig"
Benutzer.Put "displayName", "WilliRiesig"
Benutzer.Put "physicalDeliveryOfficeName", "Zimmer 4711"
Benutzer.Put "telephoneNumber", "555-789877"
Benutzer.Put "mail", "williRiesig@IT-Visions.de"
Benutzer.Put "description", "Dies ist der Account von Willi Riesig"
Benutzer.Put "WWWHomePage", "http://www.IT-Visions.de"
' Werte werden festgeschrieben
Benutzer.SetInfo
Wscript.Echo "Benutzer " & Benutzer.AdsPath & " wurde geändert!"
```

9.2.10 Deaktivieren eines Benutzerkontos

Soll einem Benutzer der Zugang zum Netzwerk nur kurzfristig entzogen werden, muss man das Konto nicht löschen, sondern kann es kurzfristig deaktivieren. Sperrung

Das nachfolgende Beispiel zeigt, wie mithilfe des Attributs `userAccountControl` ein Benutzer deaktiviert wird, sodass er sich nicht mehr am Netz anmelden kann. Dazu wird mit `GetObject()` ein Verweis auf das User-Objekt erstellt und in der Variablen `Benutzer` gespeichert. Die Referenz auf den Benutzer erfordert den kompletten LDAP-Pfad zur Identifikation.

Nun wird in der Variablen `UserAccountData` der aktuelle Wert der Eigenschaft `userAccountControl` abgelegt. Durch Verknüpfung des aktuellen Status von `userAccountControl` mit dem Wert der Konstanten `ADS_UF_ACCOUNTDISABLE` (2) über den OR-Operator und die Zuweisung des Werts mittels `Put()` wird das Konto zur Sperrung vorbereitet. Die eigentliche Sperrung erfolgt erst mit dem Aufruf von `SetInfo()`. Deakti-
vierung

Listing 9.20: /Skripte/Kapitel08/LDAP/DeaktiviereKonto.vbs

```
' DeaktiviereKonto.vbs
' Deaktivieren eines Benutzerkontos im Active Directory
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Container
Dim Benutzer
Dim UserAccountData
' Konstanten definieren
Const BenutzerName="WilliRiesig"
Const ADS_UF_ACCOUNTDISABLE = 2
' Zugriff auf das Benutzer-Objekt
```

```

Set Benutzer = GetObject("LDAP://ServerE02/CN=" & BenutzerName & _
    ",OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Benutzerdaten ermitteln
UserAccountData = Benutzer.Get("userAccountControl")
' Daten ändern
Benutzer.Put "userAccountControl", UserAccountData OR ADS_UF_ACCOUNTDISABLE
' Änderungen festschreiben
Benutzer.SetInfo
' Meldung ausgeben
WScript.Echo "Benutzerkonto " & Benutzer.AdsPath & " wurde deaktiviert"

```



HINWEIS: Das Aktivieren des gesperrten Kontos ist ebenfalls möglich. Allerdings existiert hierfür keine Konstante. Das Konto kann durch die Verknüpfung der Eigenschaft `userAccountControl` mit dem Wert 4 durch den AND-Operator wieder aktiviert werden. Auch hier ist der Aufruf von `SetInfo()` notwendig.

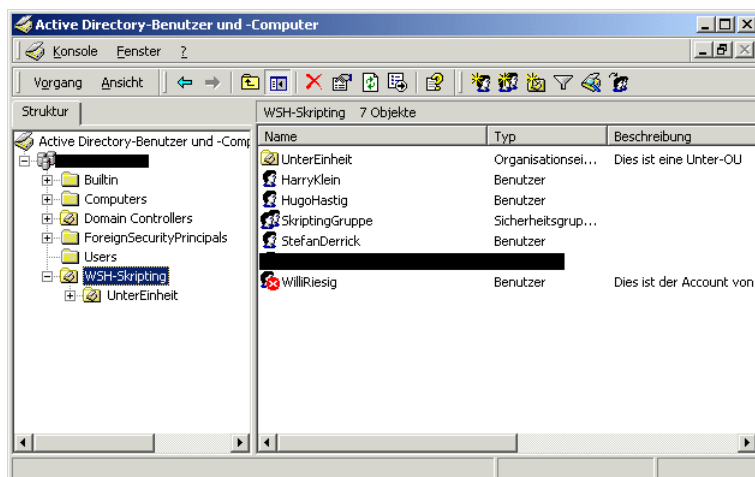


Abbildung 9.13: Benutzerkonto für „WilliRiesig“ ist deaktiviert.

9.2.11 Entfernen eines Benutzers aus einer Gruppe

Rechte-
entzug

Durch Entfernen eines Benutzers aus einer Gruppe können Rechte und Beschränkungen bequem entfernt bzw. hinzugefügt werden. Im Beispiel wird ein Verweis auf das Group-Objekt der angegebenen Gruppe erzeugt und durch die `Remove()`-Methode des Group-Objekts wird der Benutzer aus der Gruppe entfernt. Sowohl die Identifikation des Group-Objekts als auch die des User-Objekts erfolgt über einen gültigen LDAP-Pfad.

Listing 9.21: /Skripte/Kapitel08/LDAP/LoescheBenutzerausGruppe.vbs

```

' LoescheBenutzerausGruppe.vbs
' Löschen eines Benutzerkontos aus einer Gruppe im AD

```

```
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Gruppe
' Konstanten definieren
Const GruppenName="ScriptingGruppe"
Const BenutzerName="HugoHastig"
' Bindung an Gruppen-Container
Set Gruppe = GetObject("LDAP://ServerE02/CN=" & GruppenName & _
",OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Entfernen des Benutzers
Gruppe.Remove ("LDAP://ServerE02/CN=" & BenutzerName & _
",OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Ausgabe
WScript.Echo "Der Benutzer " & BenutzerName & _
" wurde aus der Gruppe " & GruppenName & " entfernt."
Löschen einer Gruppe
```

Werden Gruppen nicht mehr benötigt, weil beispielsweise Abteilungen aufgelöst wurden, lassen sich diese Objekte auch wieder aus dem Active Directory entfernen.

Das Beispiel demonstriert das Vorgehen an der aus den vorherigen Kapiteln bekannten Gruppe „ScriptingGruppe“. Um diese Gruppe zu löschen, wird durch `GetObject()` ein Verweis auf das Users-Objekt angelegt. Ein anschließender Aufruf der `Delete()`-Methode löscht die Gruppe. Als Parameter werden der Klassenname `Group` sowie der Name des Group-Objekts (mit vorangestelltem `CN=`) erwartet.

Gruppen-
löschung



HINWEIS: Die `Delete()`-Methode gibt kein Objekt zurück, und ein Aufruf von `SetInfo()` ist nicht notwendig, da die Aktion sofort ausgeführt wird.

Listing 9.22: /Skripte/Kapitel08/LDAP/LoescheGruppe.vbs

```
' LoescheGruppe.vbs
' Entfernen einer Gruppe im AD
' verwendet: ADSI
' =====
Option Explicit
' Variablendeklaration
Dim Domaene
' Konstanten definieren
Const GruppenName="ScriptingGruppe"
' Domain-Objekt erzeugen
Set Domaene = GetObject _
("LDAP://ServerE02/OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Löschen der Gruppe
Domaene.Delete "group", "CN=" & GruppenName
' Ausgabe
WScript.echo "Die Gruppe " & GruppenName & " wurde entfernt."
```

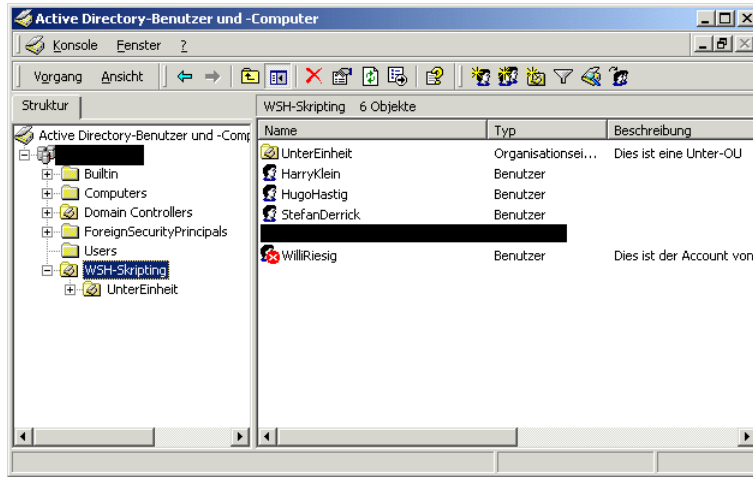


Abbildung 9.14: Die Gruppe wurde gelöscht.

9.2.12 Löschen eines Benutzerkontos

Benutzer
loswerden

Da nicht alle Benutzer im Unternehmen verbleiben, muss gelegentlich auch mal einer gelöscht werden. Um nun einen angelegten Benutzer wieder loszuwerden, nutzt das Beispielskript die `Delete()`-Methode des übergeordneten Containers. Der Verweis auf den Container, in diesem Fall die Organisationseinheit *WSH-Scripting*, wird über den LDAP-Pfad referenziert und durch den Aufruf von `GetObject()` erzeugt. Anschließend wird die `Delete()`-Methode mit dem Klassennamen `User` und dem Namen des Benutzers (mit vorangestelltem `CN=`) aufgerufen. Die `Delete()`-Methode löscht den Benutzer sofort und ohne Nachfrage.

Listing 9.23: /Skripte/Kapitel08/LDAP/LoescheBenutzer.vbs

```
' LoescheBenutzer.vbs
' Löschen eines Benutzerkontos im AD
' verwendet: ADSI
' =====
Option Explicit
' Variablen deklarieren
Dim Container
' Konstanten definieren
Const BenutzerName="HugoHastig"
' Zugriff auf das Container-Objekt
Set Container = GetObject _
("LDAP://ServerE02/OU=WSH-Scripting,DC=IT-Visions,DC=de")
' Benutzer löschen
Container.Delete "User", "CN=" & BenutzerName
' Meldung ausgeben
WScript.Echo "Benutzer " & BenutzerName & " wurde gelöscht"
```

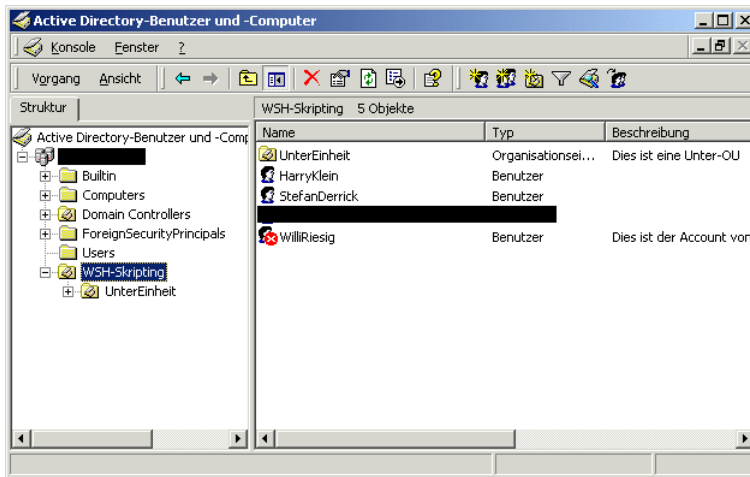


Abbildung 9.15: Der Benutzer HugoHastig wurde gelöscht.

9.2.13 Löschen einer Organisationseinheit

Analog zur Erstellung von Organisationseinheiten lassen sich diese auch wieder aus dem Active Directory entfernen. Die Vorgehensweise ist identisch mit dem Löschen eines Benutzers oder einer Gruppe. Anstelle der Create()-Methode wird zum Löschen die Delete()-Methode aufgerufen.

Container
löschen



ACHTUNG: Die Organisationseinheit kann nur gelöscht werden, wenn sie keine Unterobjekte mehr enthält.

Listing 9.24: /Skripte/Kapitel08/LDAP/LoescheOU.vbs

```
' LoescheOU.vbs
' Löschen einer Organisationseinheit im AD
' verwendet: ADSI
' =====
' Variablendeklaration
Dim Root
Dim Domain
' Konstanten definieren
Const OUName="WSH-Scripting"
' Oberstes Element des AD holen
Set Wurzel = GetObject("LDAP://RootDSE")
' Wurzelverzeichnis bestimmen
Set Domaene = GetObject("LDAP://" & Wurzel.Get("defaultNamingContext"))
' OU löschen
Domaene.Delete "organizationalUnit", "OU=" & OUName
' Ausgabe
WScript.Echo "OU " & OUName & " wurde gelöscht"
```

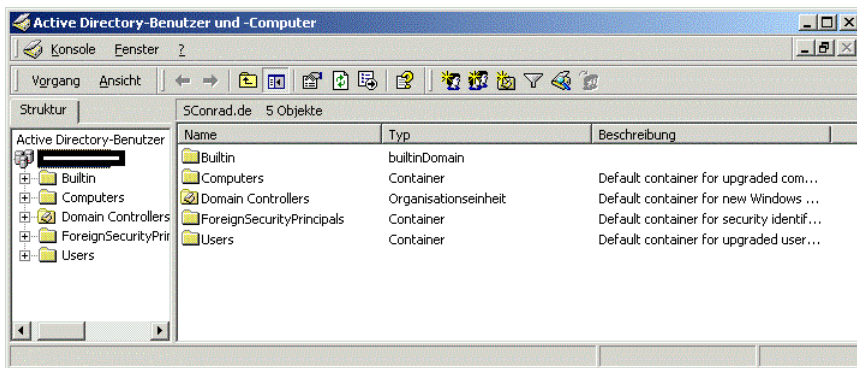


Abbildung 9.16: Nun ist die OU auch wieder weg.

■ 9.3 Fragen und Aufgaben

1. Verliert ein Benutzer nach der Umbenennung seines Kontos seine zugewiesenen Berechtigungen?
2. Wird beim Zuordnen eines Benutzers zu einer Gruppe dieser Benutzer automatisch angelegt, wenn er nicht vorhanden ist?
3. Aktiviert eine Zuweisung des Werts True an die Eigenschaft AccountDisabled das Konto eines Benutzers oder deaktiviert sie es?
4. Können Benutzer in der NT4-Benutzerdatenbank wie im Active Directory auf andere Ebenen verschoben werden?
5. Wenn eine Gruppe gelöscht wird, werden dann automatisch die darin enthaltenen Benutzer gelöscht?
6. Ist eine Organisationseinheit ein Container oder ein Blatt-Objekt?
7. Kann das Kennwort eines Benutzers ohne Kenntnis des alten Kennworts geändert werden? Wenn ja, mit welcher Methode?
8. WaskennzeichnetdernachfolgendeLDAP-Pfadgenau?*LDAP://ServerE02/CN=Trainer, OU=Scripting,DC=IT-Visions,DC=de*
9. Ist die Verschachtelung von Container-Objekten einer Beschränkung unterworfen?
10. Lassen sich Container-Objekte mit einem einfachen Delete() rekursiv löschen?

Stichwortverzeichnis

Symbole

^ 52
- 52
/ 52
\ 52
& 52
+ 52
< 53
<= 53
<> 53
= 53
> 53
>= 53
\$false 418
\$true 418
.dll 100f.
.exe 100f.
.NET 135, 175, 375, 388, 399,
434
.udl 124
.wsc 100f.

A

Abs 76, 481
Absolutwert 481
Absturz 2
AcceptPause 292
AcceptResume 293
AcceptStop 289
Access-Datenbank 162, 170, 246
AccessMask 457
AccountDisabled 238, 245, 258
AccountExpirationDate 231, 235
ActiveConnection 173
Active Directory 126f. 130, 133,
229, 232, 241 ff., 255 ff., 355,
358, 388, 452, 468
Active Scripting 1, 4
ActiveX 385
ActiveX-Komponente 110

ADB 132, 252
Add 113, 177, 237, 250
Add-ADGroupMember 471
Add-Computer 450
Add-Content 411, 454
Addition 52
AddNew 180
Add-Type 435, 455
Add-VMHardDiskDrive 476
Administrator 29, 229
Administratoren (Gruppe) 263
ADO 121 ff., 171, 173, 179, 260
ADsDSOObject 260
ADSI 125 ff., 150, 224, 229 ff.,
234, 238, 241, 243, 259 f., 394,
397 f., 418, 494
- Impersonifizierung 390
- LDAP-Provider 132
- Pfad 127
ADsSecurity 215, 494
ADsSystemInfo 103
Aktualisierungsintervall 141
Alias 404, 448
AND 54, 185 f., 254
Anführungszeichen 8, 43, 45
Anmeldeskript 231, 373, 398,
420
Anmeldung 235
ANSI 482
Anweisungsblock 93
Anwendungsprotokoll 271, 274,
283, 450
Anzeigename 294
AppendChild 177
Application Compatibility Tool-
kit 33
Archiv 201
Argument 196
Arkustangens 481
Array 65 f., 112, 173, 307, 312,
320, 485, 487
Asc 482

ASP 1, 18
Assembly 434
Async 179
AtEndOfStream 165, 189
Atn 481
Attribut 100, 111, 155, 184, 186,
201, 215
- XML 153
Auflistung 110
Aufnahmedatum 454
Ausdruck 47
Ausgabefenster 20
Auszeichnungssprache 151
Authenticode 375, 386
Automatic 294
Auto-Vervollständigen 19

B

Backslash 308
Backup 278
BackupEventLog 278
BASIC 39
Batch 1, 161
Baumstruktur 175
Bedingung 58, 93, 125
- PowerShell 417
Bedrohung 373
Beep 434
Benutzer 93, 111, 125, 157, 162,
222, 232 ff., 245 ff., 252 ff., 272,
299
- Massenanlegen 468
Benutzerdatenbank 126, 258
Benutzergruppe *siehe* Gruppe
Benutzerkontensteuerung 36
Benutzerkontenverwaltung 232
Benutzerkontextwechsel *siehe*
Impersonifizierung
Benutzerkonto 230, 245, 256,
452, 468, 502
Benutzeroberfläche 303

Benutzerschnittstelle 142
 Benutzerverwaltung 229 f., 241, 265
 – Active Directory 241
 – lokal 229 f.
 Benutzerwechsel *siehe* Impersonifizierung
 Berechtigung 258, 316
 Betriebssystem 13, 157, 184, 450
 – installieren 476
 Bezeichner 45, 47, 88, 102
 Beziehung 99
 Binärdatei 451
 Bit 55
 Bitmap 299, 454 f.
 – Datei 299
 Blatt 258
 BMC 142
 bool 418
 Boolean 51, 487
 Boolescher Wert 43
 Boot 294
 Browser 5
 Bug XXII, 22
 byte 418
 Byte 486

C

C# 388, 417
 C++ 14
 cat 411
 CBool 486
 CByte 486
 CCur 486
 cd 411
 CDate 486
 CDbI 486
 ChangeAccess 463
 ChangePassword 234, 251
 char 418
 Chkdsk() 446
 ChkDsk 224
 ChkTrust.exe 384
 Chocolatey 430
 Chocolatey.org 429
 Chr 482
 Chrome 429, 430
 CIM 142, 144
 CimClass 435
 CimInstance 435, 441
 CInt 132, 486
 Cisco 142
 Clear-Eventlog 450
 ClearEventLog() 281
 Clipboard 409
 CLng 487
 Close 163 ff., 188 f., 191

Cmdlet
 – Commandlet 404
 Codezeile 41
 Collection *siehe* Objektmenge
 COM 107, 399, 433, 456, 485
 COM+ 101
 Commandlet 404
 Common Information Model *siehe* CIM
 Common Management Information Protocol 142
 Compare-Object 406
 Compiler 2
 Component Object Model *siehe* COM.
 Computer 99, 109, 128, 147, 210, 215, 223 f., 230 f., 233, 239, 252, 259, 264, 266, 268 f., 290, 320
 – Name 116, 265
 Computerkonto 263 f.
 Computerverwaltung 259
 Connection 122, 171 ff., 180
 ConnectServer 337
 Const 46
 Container 131, 230, 239, 243, 256, 258, 263
 ConvertTo-Html 449
 copy 411
 CopyFile 192
 CopyFolder 204, 227
 Copy-Item 411
 Cos 481
 Count 111 f., 158
 CQL 441
 Create 131, 231, 235, 242, 245, 257, 263, 343
 Create() 447
 CreateElement 177
 CreateFolder 200, 227
 CreateKey 314
 CreateObject 100, 103, 114, 116 f., 120, 163 f., 173, 179, 205, 220, 222, 397, 485
 CreateProcessingInstruction 177
 CreateShortcut 118, 301
 CreateTextFile 163 f., 169, 187, 330, 494
 CreationTime 454
 CryptoAPI 375, 384
 cscript.exe 8, 33, 35, 91
 cscript.exe 158, 162, 195, 374, 390 f., 397
 CSng 487
 CStr 132, 487
 CSV 163, 165, 173, 177, 409, 468, 472 f.

Currency 51, 486
 CursorType 173

D

Date 51, 82, 484, 486
 DateAdd 82, 484
 DateCreated 184, 201
 DateDiff 82, 484
 Datei 103, 119, 121, 183 ff., 409, 451
 – Eigenschaft 184, 201
 – Erweiterung 161, 302
 – Recht 215
 – Replikationsdienst 272
 – Verknüpfung 158
 Dateisystem 19, 111, 119 f., 124, 183, 198, 217, 278, 405, 453
 – Typ 218
 Dateisystemfreigabe 456
 Dateisystemrecht 375
 DateLastAccessed 184, 201
 DateLastModified 184, 201
 Daten 170
 Datenbank 121 f., 171, 260
 Datenbanktabelle 173
 Datendatei 162
 Datenlink 124
 Datensatz 260
 Datensicherung 278
 Datenträger 219, 223
 Datentyp 50, 132, 418, 487
 – ADSI 132
 DatePart 484
 DateSerial 484
 DateTime 418
 DateValue 484
 Datum 82, 84, 93, 484
 Day 84, 484
 DCOM 101, 435
 Debugger 22 f., 161
 Debugging 8, 22, 27
 decode.exe 389
 Default Domain Controllers Policy 356
 Default Domain Policy 356
 defaultNamingContext 241, 260, 265 f., 497
 Deklaration 48
 del 411
 Delete 113, 131, 226 f., 239 f., 255 ff.
 DeleteFile 194, 227
 DeleteKey 316
 DeleteObject 264
 DeleteValue 312
 Description 231, 234, 242
 Designer 125

Desktop 299, 302
 DeviceID 218, 224, 321
 Dezimalsystem 43
 DHCP 319 ff., 466
 Dialogfenster 10, 159
 Dienst 125, 150, 285 ff.
 diff 406
 Dim 48
 dir 411, 450
 Directory 252
 – Service 271
 DirectoryEntry 434
 Disabled 295
 Disjunktion 54
 DisplayName 252, 285
 Division 52
 DLL 116, 119, 127
 DNS 271, 319, 324 ff.
 DnsClient 467
 DNSClient 467
 DNS-Server 467
 DNSServerSearchOrder 324
 do 418
 Do ... Loop 62, 282, 296
 Do...Loop 260
 DOM 153, 175, 485
 Domain 230, 233 f., 239, 248
 Domäne 96, 99, 128 ff., 231, 259,
 265 f., 270
 – Administrator 263
 – hinzufügen 450
 Domänencontroller 140, 266
 DOMDocument 154 f., 177, 179
 DOS 8, 10, 158, 401
 double 418
 Double 51, 486
 Drag&Drop 9, 161
 Drive 120, 215, 217, 219, 227
 DriveLetter 111, 215
 DriveType 215, 217
 Drucker 128, 409, 449
 Druckerverbindung 116
 DVD 476

E

Echo 7, 165
 EditPlus 18
 Eingabeaufforderung 8, 10
 Eintrag 272 ff., 282
 Element 152 f., 155
 Embedded Visual Basic 39
 Empty 51
 EnableDHCP 326 f.
 EnableDNS 324
 EnableLUA 37
 Enable-PSremoting 422
 Enable-PSRemoting 403

EnableStatic 320, 326
 Endlosschleife 282
 Enter-PSSession 423
 Entwicklungsumgebung 109, 479
 EnumerateSubkeys 312
 env 450
 EOF 172
 erase 411
 Ereignis 95, 102, 105, 274, 277,
 296
 Ereignisanzeige 274 f., 295
 Ereignisprotokoll 117, 150, 271 f.,
 278, 281, 283
 Ereignisprotokolldienst 272
 Erfolgsüberwachung 272
 Err 91, 103, 315
 ErrorAction 454
 Eval 487
 Event 145
 EventCode 282
 EventCreate.exe 275, 279
 Excel 103
 Exchange Server 126, 480
 ExecNotificationQuery 282, 296
 ExecQuery 149, 272, 277, 285
 Execute 487
 ExecuteGlobal 487
 EXIF 454
 Exklusion 54
 Exp 481
 Export-CLIXML 409
 Export-Csv 408
 Export-CSV 409

F

False 54, 55
 Fehler 17, 22 f., 91, 272
 Fehlerbehandlung 91
 Fehlercode
 – WMI 344
 Fehlerkontrolle 295
 Fehlermeldung 203 f., 220, 237
 Fehlerstelle 17
 Fehlertyp 294
 Fehlerüberwachung 272
 Fernausführung 422
 Festplatte
 – virtuell 476
 Festplattenprüfung 223 f.
 Field 122
 File 98, 121, 155, 173 f., 183 f., 188,
 196 ff., 202
 FileExist 184, 186, 188
 FileService 226
 FileShare 224
 FileSystem 218 f.

FileSystemObject 98, 120,
 163 ff., 173, 183 f., 186 ff., 192 f.,
 196, 198, 200 ff., 215, 217 f.,
 220, 226 f., 300 f. 329 f.
 File System Objects *siehe* FSO
 Filter 487
 Find-Module 427 f.
 Find-Package 429 f.
 Fix 481
 Flag 55, 184
 Fließkommazahl 43
 Folder 121, 196 ff., 201 f., 205
 FolderExists 203, 301
 FolderName 300
 for 418
 ForAppending 162, 165, 494
 foreach 418
 For Each 112, 131, 149, 184, 198,
 218, 272, 307, 330, 340
 Foreach-Object 406 ff., 417, 449,
 472
 foreach (PowerShell) 505
 Format 74, 85
 FormatCurrency 482
 FormatDateTime 85, 482
 Formatierung 85
 Format-List 409
 FormatNumber 85, 482
 FormatPercent 482
 Format-Table 409, 450, 478
 Format-Volume 476
 Format-Wide 408 f.
 Formel 47
 For ... Next 61
 For ... Next 62, 111
 For...Next 335
 ForReading 162, 191
 ForWriting 162
 Foto
 – sortieren 454
 FQDN 324 f.
 FreeSpace 219, 220
 Freeware 18
 Freigabe 224 ff., 456, 464 f.
 FSO 120
 FullAccess 463
 FullName 231, 234
 function 417 f.
 Funktion 69

G

Ganzzahl 43
 Ganzzahldivision 52
 Gateway 319 ff., 326 f., 467
 GB 419
 gc 411
 Gebietsschema 484

Get-ADComputer 471
 Get-ADGroup 471
 Get-ADGroupMember 471
 Get-ADObject 452, 471
 Get-ADOrganizationalUnit 471
 Get-ADUser 452, 471
 Get-Alias 413, 448
 GetBinaryValue 306
 Get-Childitem 405, 413, 450 f., 505
 Get-ChildItem 411
 Get-CimAssociatedInstance 435
 Get-CimClass 435, 446
 Get-CimInstance 435 ff., 446
 Get-Command 412, 432
 Get-Content 408, 411, 454, 472 f.
 Get-Credential 413
 GetDrive 121, 215, 217, 219 f.
 GetDWORDValue 306
 Get-EventLog 450
 GetEx 132
 GetExpandedStringValue 306
 GetFile 103, 121, 183 f., 186, 198, 329
 GetFolder 121, 183, 197 f., 201, 205, 301
 Get-Help 412 f.
 Get-Item 411
 Get-ItemProperty 411
 Get-Location 411
 Get-Mailbox 452
 Get-Member 406, 441
 Get-Module 431
 GetMultiStringValue 306 f.
 GetNamedItem 155
 Get-NetIPInterface 467
 GetObject 103, 121, 127, 149, 156, 218, 224 f., 230, 233, 237 ff., 250 ff., 394, 397, 485
 Get-Package 430
 Get-PackageProvider 429
 Get-PackageSource 429
 GetPassword 393
 Get-Process 404, 406 ff., 423, 448 f.
 Get-PSDrive 411, 413
 Get-PSRepository 428
 GetRef 485
 Get-Service 423, 449
 Get-SmbShare 458, 462
 Get-SmbShareAccess 464
 GetSpecialFolder 169
 GetStringValue 306
 GetTempName 169
 GetTempName() 433
 Get-WmiObject 408, 448 ff.
 Get-WmiObject 435 ff., 440
 gi 411

GivenName 252
 Gleichheit 53
 Gleichheitszeichen 167
 gm 406
 gp 411
 gpi 411
 GPM 136, 355
 GPMBackup 137, 370
 GPMBackupCollection 137
 GPMBackupDir 137, 368
 GPMC 133, 135, 138, 366
 – Komponente 134, 351
 GPMC Objects 133
 GPMConstants 137
 GPMDomain 136 f., 351, 358, 370
 GPMGPO 136 f., 355, 358, 365 f.
 GPMGPOCollection 137, 355
 GPMGPOLink 137, 358, 362, 365
 GPMGPOLinksCollection 137, 358, 362
 GPMSearchCriteria 137, 368
 GPMSOM 136 f.
 GPMSOMCollection 137
 GPO 134, 138, 355
 gpupdate.exe 138, 140
 Grafikkarte 436
 Grant-SmbShareAccess 464
 Groß- und Kleinschreibung 7, 80, 230, 404
 Group 110, 237, 239, 248, 250, 254 f.
 Group-Object 406, 450
 GroupType 236, 248
 Gruppe 125, 236 ff., 248 f., 254 f., 257, 468 f., 471
 Gruppenmitglieder 471
 Gruppenrichtlinie 133 ff., 139, 351, 358, 364, 370
 – Editor 134, 139
 Gruppenrichtlinieneinstellung 141
 Gruppenrichtlinienergebnis 139
 Gruppenrichtlinienmodellierung 139
 Gruppenrichtlinienobjekt *siehe* GPO
 Gruppenrichtlinienverwaltung 133, 138
 GUID 135, 355, 360, 362

H

Handheld 39
 Hardware 2
 Hash-Tabelle 446
 Hashtable 418
 Hash-Wert 375, 386
 Haskell 4

Herausgeber 422
 Hex 487
 Hexadezimalzahl 43, 135
 Hilfe
 – PowerShell 412
 Hintergrundbild 299
 Hive 305
 HKCR 305
 HKCU 305, 317
 HKEY_CLASSES_ROOT 109
 HKEY_CLASSES_ROOT 305
 HKEY_CURRENT_USER 305
 HKEY_LOCAL_MACHINE 148, 305, 308
 HKLM 305
 Hochkomma 42
 HomeDirectory 231
 Hour 84, 484
 HTML 20, 151, 153, 449, 472
 HTTP 109
 Hyper-V 476

I

IADsUser 98
 if 417
 If ... Then 57
 IIS 1, 126, 411, 472, 480
 – 8.0 472
 Impersonifizierung
 – ADSI 390
 – WMI 397
 – WSH 392
 Import-CLIXML 409
 Import-Csv 472
 Import-CSV 409, 469
 Import-Module 432
 Index 485
 Informationen 272
 Informationsspeicherung 152
 Inhaltsdaten 171
 INI-Datei 166 ff., 303
 InputBox 73, 392, 435
 Insert 113
 InsertBefore 177
 Install 334
 Install-Module 426, 428
 InstanceCreationEvent 282
 InstanceDeletionEvent 145
 InstancesOf 149
 Instanz 96 f., 110
 Instanziierung 97
 Instanziierung 434
 InStr 78, 196, 482
 InStrRev 301, 482
 int 418
 Int 76, 481

Integer 51, 274, 487
 Integrated Scripting Environment
 siehe ISE
 Integrität 375
 IntelliSense 108
 Interface 98
 International .NET Association
 XXIV
 Internet Explorer 23
 – Zone 386
 Internet Information Services
 siehe IIS
 Interpreter 2
 Inventar 331
 Invoke-CimMethod 435, 446
 Invoke-Command 422f.
 Invoke-WmiMethod 435, 446
 IP
 – Adresse 466
 IPAddress 467
 IP-Adresse 150, 319f., 322f.
 IsArray 87, 487
 IsDate 87, 487
 ISE 402, 427
 IsEmpty 487
 IsNull 487
 IsNumeric 87, 487
 ISO 476
 IsObject 487
 IsReady 216, 220
 Item 112, 158
 ItemIndex 149
 iX XXIII, 515

J

Jahr 83f.
 JavaScript 5
 Job 162
 Join 81, 483
 JoinDomainOrWorkgroup 265
 JPEG 454
 JScript 5

K

KB 419
 Kennwort 222, 234f., 246, 251
 – WSH 392
 Kill() 407
 Klasse 98ff., 107, 117, 130, 147,
 188, 223, 231, 485
 – COM 433
 – .NET 434
 – PowerShell 419
 Klassenmitglieder
 – statisch 445
 Knoten 155, 175

– XML 154
 Kodierung 389
 Kommandozeile 17, 195
 Kommandozeilenanwendung
 342
 Kommandozeilenoption 25
 Kommandozeilenparameter 157,
 158, 161
 Kommandozeilen-
 programmierung 1
 Kommentar 42, 418
 – PowerShell 417
 Kompilierung 23, 479
 Kompilierungsfehler 5, 23
 Komponente 13, 100, 117f., 241,
 479
 Konfigurationsspeicher 157
 Konjunktion 54
 Konsole 198, 203
 Konstante 45, 88, 104
 – selbst definiert 45
 – vordefiniert 45
 Konstantendefinition 44
 Konto *siehe* Benutzer
 Konvertierung 74
 Kosinus 481

L

Laufwerk 111, 113, 118, 120, 148,
 150, 183, 194, 201, 215, 218ff.,
 222f.
 Laufwerksliste 411
 Laufwerkstyp 217
 Laufzeitfehler 24
 LBound 485
 LCase 80, 483
 LDAP 7, 109, 126f., 232, 245,
 248, 250ff., 256, 258, 260,
 264, 353, 355, 394
 Leerzeichen 7f., 79
 Left 78, 169, 483
 Leistung 261
 Len 483
 Length 112, 179
 Linie 100
 Liste 409
 Literal 47, 104
 Lizenz 329
 Load 179
 LoadPicture 486
 LockType 173
 Log 481
 Logarithmus 481
 LogEvent 274
 Logische Fehler 24
 Long 51, 306, 314, 487
 Love-Letter-Wurm 373, 375

Is 411
 LTrim 79, 483

M

MachineName 434
 Makrosprache 39
 Managed Object 142
 ManagementClass 435, 437
 ManagementObject 435, 437,
 441, 443
 ManagementObjectSearcher 437
 Manual 295
 MapNetworkDrive 222
 Maschinensprache 2, 40
 MB 419
 md 454f., 473
 Measure-Object 406
 Meta-Objektmodell
 – WMI 145
 Metasprache 151
 Methode 100, 113, 148, 184, 186,
 194, 203, 205, 226, 234, 239f.,
 245f., 268
 Microsoft Access 125, 171, 246
 Microsoft Certified Solution
 Developer XXIV
 Microsoft Excel 469
 Microsoft Exchange Server 453
 Microsoft Office 39, 122
 Microsoft Script Debugger *siehe*
 Script Debugger
 Microsoft Script Encoder 389
 Microsoft Word 433
 Microsoft XML Document Object
 Model 243
 Mid 78, 483
 Migrationstabelle 371
 Mikroprozessor 2, 40
 Minute 83f., 484
 Mitglied
 – WMI 443
 mkdir 411
 Mod 52
 Modul 424
 Module
 – auflisten 431
 – herunterladen 425
 – installieren 424
 – laden 432
 Module Browser 426
 Modulo 52
 Monat 83
 Moniker 97, 485
 Month 84, 484
 MonthName 484
 Most Valuable Professional XXIV
 move 411

Move-ADObject 471
 MoveFile 193
 MoveFirst 172
 MoveFolder 204
 MoveHere 233 f., 251 f., 266
 Move-Item 411, 451
 MoveNext 172
 Msado15.dll 122
 MSFT_SmbShare 462
 MSFT_SmbShareAccess-
 ControlEntry 464
 MsgBox 70, 486
 MSI 274, 449, 475
 msscrdbg.exe 23
 MSXML 151, 153 f., 243, 335
 Msxml2.DOMDocument 335
 Msxml3.dll 154
 MTS 101
 Multiedit 18

N

Nachkommastelle 74, 76
 Nachricht 150
 Namensraum 147 f., 289, 294
 Negation 52, 54
 NetAdapter 467
 NetBIOS 291, 325
 NetTCPIP 467
 NetWare 126
 Netware Directory Service 126
 Netzlaufwerk 120, 222
 Netzwerk 210, 225, 229, 253,
 323
 Netzwerkadapter 449
 Netzwerkkarte 96, 150, 328,
 466
 Netzwerkkonfiguration 319, 466
 Netzwerkmanagement 142
 Netzwerkprotokoll 319, 322
 Netzwerkkumgebung 319
 Netzwerkverbindung 116, 223
 Neustart 267, 297
 New-ADGroup 471
 New-ADOrganizationalUnit 452,
 471
 New-ADUser 452, 471
 New-CimInstance 435, 447
 New-CimSession 437
 New-CimSessionOption 437
 New-EventLog 450
 New-Item 410 f., 473
 New-ItemProperty 473
 New-Mailbox 453
 New-NetIPAddress 467
 New-Object 433 f.
 New-Partition 476
 New-SmbShare 462 f.

New-VM 476
 NextEvent 282, 297
 NoAccess 463
 Node 177
 Not 54, 200
 Notation 100
 Notepad 6, 16, 107, 124
 Nothing 105
 Novell 126
 Now 82, 484
 NTFS 375, 388
 Null 51, 343

O

Objekt 51, 95, 98, 106 f., 177, 197,
 202, 231, 242, 248, 270, 337,
 407
 Objektbaum 99
 Objektmenge 111 f., 285
 Objektmodell 116 f., 120, 123,
 129 f., 147
 – MSXML 154
 Objektorientierung 102, 407
 Objektreferenz 186
 Objekttyp 97
 Objektvariable 106
 Oct 487
 Oder 54
 ogv 409
 Oktalzahl 43, 487
 OLE 101
 OLEDB 123 f.
 oleview.exe 109
 On Error 91, 288
 Open 172 f.
 OpenDSObject 395
 OpenTextFile 165, 173, 188 f.
 OpenTextStream 121
 Operator 55
 – PowerShell 418
 Option Explicit 48 f.
 OR 54, 253
 Ordner 453
 – Dateisystem 118
 – löschen 453
 Organisationseinheit 134, 139,
 241, 243, 245 f., 257 f., 452,
 468 f.
 OrganizationalUnit 241 f.
 Out-Clipboard 409
 Out-File 409
 Out-GridView 409, 413
 Out-Host 409
 Out-Null 409
 Out-Printer 409
 Out-Speech 409

P

PackageManagement 425, 428
 Papierkorb 456
 Parameter 69, 104, 159, 173
 Parameterliste 105
 ParentFolder 201
 Parser 24
 Path 215, 225, 306
 Pathname 330
 PauseService 292
 PB 419
 Perl 4, 417
 Pfad 127, 194, 196, 198, 234, 315,
 330
 Pfadangabe 118
 Pfeilspitze 100
 pick-head 505
 Picture 486
 Ping 408, 416
 Ping.exe 342
 Pipeline 444
 Pipelining 405
 Postfach 453
 Potenzierung 52
 PowerShell 399, 448
 – Ausgaben 408
 – Beispiele 448
 – COM 433
 – Commandlet 404, 448
 – Hilfe 412
 – Installation 401
 – ISE 402
 – Modul 424
 – Navigation 410
 – .NET 434
 – Pipeline 405
 – Remoting 422
 – Skript 414, 453
 – Werkzeug 401
 – WMI 435
 PowerShellCX 516
 PowerShell Gallery 426
 PowerShellGet 425 f.
 PowerShellPlus 415
 PrimalScript 18 f., 108, 415
 PrimalSense 108
 Printer 409
 Programmfehler *siehe* Fehler
 Programmiersprache 5, 14, 50
 Programmierung 2
 Properties 443
 Provider 127, 171, 180, 233, 259 f.
 Prozedur 89
 Prozentzahl 86
 Prozess 339, 343, 346, 449
 Prozessor 448
 Prozessverwaltung 339

PSCX 426
 Put 132, 148, 253
 PutEx 132
 pwd 411
 Python 4

Q

QBasic 39
 Quartal 83
 QueryDialect 441

R

Read 188
 ReadAccess 463
 ReadAll 188, 190, 196
 Read-Host 409
 ReadLine 8, 165, 188 f.
 Reboot 269
 Rechner *siehe* Computer
 Recht 215, 226, 316
 RecordSet 122, 171 ff., 179 f.
 ReDim 67
 Reflection 407 f.
 REG_BINARY 305
 RegDelete 280, 311, 315 f.
 REG_DWORD 305
 REG_EXPAND_SZ 305
 Register-CimIndicationEvent 435
 Register-Packagesource 427, 429
 Registry 117, 148, 157, 166, 280, 303 ff., 316 f., 374, 473
 REG_MULTI_SZ 305
 RegRead 305 f.
 REG_SZ 305
 RegWrite 267, 279, 308, 313
 Rekursion 194, 199
 REM 42
 Remove 113, 238, 239, 254
 Remove-ADGroupMember 471
 Remove-ADObject 471
 Remove-ADOrganizationalUnit 452
 Remove-CimInstance 435
 Remove-Computer 450
 Remove-Item 411, 473, 476
 Remove-NetIPAddress 467
 Remove-NetRoute 467
 RemoveNetWorkDrive 223
 Remove-SmbShare 463
 Remove-VM 476
 Remove-WmiObject 435
 Rename 268
 Rename-ADObject 471
 Rename-Computer 450
 Rename-Item 411, 451

Replace 79, 329, 483
 Restart-Computer 450
 Resultant Set of Policies *siehe* RSoP
 return 417
 Revoke-SmbShareAccess 464
 REXX 4, 18
 Right 78, 167, 301, 483
 rm 411
 rmdir 411
 Rnd 66, 481
 Root 148, 269, 272, 294, 306
 RootDSE 241 f., 260, 263 f., 497
 Round 481
 Router 323
 RSoP 139
 RTrim 79, 483
 Ruby 4
 Rückgabewert 70, 89, 104
 runas.exe 391

S

SAMAccountName 245, 248 f., 252, 263, 497
 Sammlung *siehe* Objektmenge
 Sapien 18, 108
 Save-Module 426
 Schablone 97
 Schleife 48, 61 f., 111, 190
 Schlüssel 303 ff., 315 ff.
 Schlüsselwort 106
 Schnittstelle 98
 Scintilla 18
 Scopes of Management *siehe* SOM
 Screen Scraping 342, 347
 screnc.exe 389
 Script Center 14
 Script Debugger 22, 25
 Script Encoder 389
 ScriptEngine 486
 ScriptEngineBuildVersion 486
 ScriptEngineMajorVersion 486
 ScriptEngineMinorVersion 486
 Scripting 1, 8
 Scripting.FileSystemObject 111, 114, 120 f., 433
 Scripting Runtime *siehe* SCRRun
 ScriptPW.Password 393
 SCRRun 118 f., 180, 218, 290 f., 300 f., 335
 Sccrun.dll 119
 SDDL 457
 secedit.exe 138, 140
 Second 84, 484
 Sekunde 83 f.

SELECT 122, 125, 139, 149 f., 260 f., 268 f., 277, 279, 281 f., 286 ff., 294, 297
 Select Case 58, 217
 SelectNodes 335, 501
 Select-Object 406, 444, 450
 ServiceName 288 f.
 Set-ADAccountPassword 471
 Set-AuthenticodeSignature 421
 SetBinaryValue 309
 Set-CimInstance 435, 445
 Set-Content 411, 449
 Set-DnsClientServerAddress 467
 SetDWORDValue 309
 Set-Executionpolicy 420
 SetExpandedStringValue 309
 SetGateway 323
 SetInfo 225, 235, 237 f., 240, 242, 248, 253 f.
 Set-ItemProperty 411
 SetLocale 484
 Set-Location 410 f.
 SetMultiStringValue 309
 SetNamedItem 155
 Set-NetIPInterface 467
 SetPassword 234 f., 246, 251
 SetStringValue 309
 Set-VMDVdDrive 476
 SetWINSserver 325
 Set-WmiInstance 435, 445
 Sgn 481
 ShareName 215
 Shell.Application 456
 ShortName 201
 Show-Command 413
 Shutdown 269
 Sicherheit 12, 36, 373
 – WMI 146
 Sicherheitsprotokoll 271
 Sicherungskopie 366, 368
 SID 233
 Signatur 375, 386
 signcode.exe 375, 382
 Signcode-Wizard 381
 Signierung 384
 SilentlyContinue 454
 Sin 482
 single 418
 Single 51
 Skript 2, 8
 – MSH 414
 – signiert 375
 – Sperrung 375
 Skripteditor 402
 Skriptkodierung 389
 Skriptsprache 2
 Sleep 291
 Software 329, 334, 337

Softwareeinschränkung *siehe* SRP
 Softwareinstallation 475
 Softwarepaket 429
 Softwarequelle 429
 Software Restriction Policies *siehe* SRP
 Softwareverwaltung 329
 SOM 135, 353, 358
 Sonderordner 119
 Sonderzeichen 158
 Sortieren 405
 Sort-Object 405 f., 449 f., 505
 sp 411
 Space 483
 Spalte 122, 125, 173
 SpecialFolders 300, 499
 Speech 409
 Speicherbereich 102
 Speicherplatz 220
 Speicherplatzbelegung 219
 Speicherverwaltung 105
 Spitzname 97
 Split 81, 165, 173, 290, 483
 Sprachausgabe 409
 Sprache 2, 342
 SQL 122 f., 125, 149 f., 171, 260, 268, 287
 SQL Server 125, 480
 Sqr 482
 SRP 375, 386 f.
 Startmenü 300
 StartMode 294
 Startmodus 288, 294
 Start-Process 406, 448
 StartService 288
 Startverzeichnis 194
 Start-VM 476
 Startzeitpunkt 294
 StdRegProv 303, 306, 309, 311 f., 314, 316
 Stop-Computer 450
 Stop-Process 406 f.
 Stop-VM 476
 StrComp 483
 string 418
 String 51, 483, 487
 StrReverse 483
 Strukturdaten 171
 Stunde 83, 84
 SubFolder 196, 198, 205
 Subnetz-Maske 320, 322 f., 467
 Subtraktion 52
 Suchanfrage 137
 su.exe 390
 SWbemObject 98, 145
 SWbemObjectSet 145
 SWbemService 224, 337

SWbemServices 145
 Syntax 41, 90, 104, 110
 – coloring 18
 System 289, 294
 System32 450
 Systemdatum 484
 System.Diagnostics.Process 407
 Systemdienst 285, 449
 Systemintegrität 289
 Systemmanagement 142
 System.Management.
 ManagementObject 443
 Systemprotokoll 271
 SystemScripter 20, 22
 Systemsteuerung 232, 261
 System.Windows.Forms 454
 Systemzeit 484
 SYSVOL 361

T

Tabelle 122, 125, 173, 175, 246
 Tabellenausgabe 409
 Tag 83 f., 155, 175
 Tan 482
 TargetInstance 282
 Taskmanager 346
 TB 419
 Tcl 18
 TCP/IP 319, 326 ff., 467
 TechNet 14
 Temp 169, 183
 Terminate 348
 Test-Connection 452
 Text 179
 Textbearbeitung 162
 Textdatei 119, 124, 162 f., 187 f., 190, 226
 Textpad 18
 TextStream 121, 162 f., 165, 167, 180, 183, 187 f.
 TIFF 454
 Time 82, 484
 Timer 484
 TimeSerial 484
 TimeValue 484
 TotalSize 219
 Treiber 123
 Trim 79, 185, 291, 483
 True 54 f.
 TrustPolicy 384 f.
 TrustPolicy Editor 385
 type 411
 Type 201
 TypeName 50, 107, 487
 types.ps1xml 444
 Typsystem 2

U

Überwachung 119, 282, 296
 UBound 164, 485
 UCase 80, 196, 483
 Uhrzeit 82
 UltraEdit 18
 Umgebungsvariable 116, 450
 UNC 210, 274
 Ungleichheit 53
 Uninstall 337
 Uninstall-Package 430
 Universal Coordinated Time 443
 Unix 417
 Interroutine 45, 88, 90, 104, 479
 – PowerShell 417
 Unterschlüssel 312
 Unterstrich 11, 45, 58
 Update 173
 User 110, 231, 238, 240, 251 ff.
 User Account Control 29
 UserAccountControl 253 f., 264, 497

V

Variable 45, 65, 88, 93, 102, 104, 189
 – PowerShell 417
 Variant 51, 74, 132
 VarType 487
 VBA 39
 VBScript 4 f., 13, 27, 88, 91, 105 f., 393
 – Funktionen 481
 VBScript.dll 5
 Verbindung 222
 Verbindungszeichenfolge 123
 Vergleich 107
 Vergleichsoperation 52
 Verknüpfung 117, 201
 Verschlüsselung 119
 Versionsnummer 11
 Verzeichnis 101, 121, 194, 196 ff., 202, 204 f., 231
 – Objekt 229
 – Recht 215
 – Struktur 207, 210
 Verzeichnisdienst 125, 127, 230, 236
 Verzeichnisstruktur 207, 210
 VHD 476
 VIM 18
 Virtualisierung 476
 Virtuelle Machine *siehe* VM
 Vista 36
 – Sicherheit 36
 Visual Basic 479

Visual Basic .NET 388
 Visual Basic Script 479
 Visual InterDev 25, 109
 Visual Studio 479
 VM 476
 VolumeName 111

W

W3C 153
 Wahrheit 57
 Wahrheitswert 43
 Währung 85
 Warnung 272
 WBEM 142
 WbemLocator 334
 WebAdministration 472
 Webseite 342
 Webserver 411
 Website 472
 WeekDay 84, 485
 WeekDayName 485
 Werkzeug 109, 133
 – Debugger 22
 Wert 55, 311, 323
 WHERE 287
 Where-Object 406 ff., 449 f.
 while 418
 Win32 144
 Win32_ACE 460
 Win32_CDROMDrive 444
 Win32_ComputerSystem 265, 267 f.
 Win32_Group 452
 Win32_LogicalDisk 148, 215, 218, 223 f., 446, 450
 Win32_NetworkAdapter 321, 449
 Win32_NetworkAdapterConfiguration 150, 320 f., 323, 325 f.
 Win32_NTEventLogFile 278, 281
 Win32_NTLogEvent 150, 272, 277, 282
 Win32_OperatingSystem 269, 450, 497
 Win32_PerfRawData_PerfOS_Processor 448
 Win32_Printer 449
 Win32_PrintJob 449
 Win32_Process 339, 341, 344, 348, 502
 Win32_Processor 448
 Win32_Product 329 f., 334, 408, 449, 475
 Win32_SecurityDescriptor 460
 Win32_Service 150, 285, 287 ff., 292 f., 296, 449
 Win32_Share 458
 Win32_Trustee 460

Win32_UserAccount 452
 Win32_Videocontroller 444
 Win32_VideoController 436
 Win32_WinSAT 262
 Windows 35, 36
 Windows 7 3, 9, 126, 143, 262, 401
 Windows 8 3, 401, 441, 472
 Windows 9x 144, 273
 Windows 10 3, 143, 229 f., 261, 401 f.
 Windows 95 126, 143, 300
 Windows 98 126, 143
 Windows 2000 126 f., 143, 259, 279, 319, 323
 Windows 2000 Server 133
 Windows as a Service 401
 Windows Editor *siehe* Notepad
 Windows Explorer 10, 116
 Windows Management Framework 143, 401
 Windows Management Instrumentation *siehe* WMI
 Windows ME 126, 143 f., 273
 Windows NT 271
 Windows NT 4.0 126, 143, 259, 272
 Windows Remote Management 422
 Windows Script Component *siehe* WSC
 Windows Script File *siehe* WSF
 Windows Server 2003 126 f., 139, 143, 241
 Windows Server 2008 126, 401
 Windows Server 2008 R2 3, 384
 Windows Server 2012 3, 401, 441, 472
 Windows Server 2016 3, 127, 129, 143, 229 f., 241, 401 f.
 Windows-Sicherheit *siehe* Sicherheit
 Windows System Assessment Tool 261
 Windows Vista 126, 233, 261, 401
 Windows XP 3, 126 f., 143, 224, 267 f., 275, 351, 375
 Win.ini 167
 WinMgmt.exe 144
 WinMgmts 147, 149
 WinNT 7, 127, 129, 230, 233, 259
 WinRM 422
 WINS 319, 325 ff.
 WinSafer *siehe* SRP
 WITHIN 296
 WMI 142 f., 148, 150, 218, 226, 229, 265, 267, 269, 273, 277 f., 281 ff., 285, 288 f., 292, 296 f., 305, 310 ff., 316 f., 320, 327, 329, 334, 398, 418, 435, 437, 466
 – Filter 139
 – Impersonifizierung 392
 – Query Language 440
 – Repository 446
 – Scripting API 145
 – Tools 150
 – Version 143
 WMIClass 437, 440
 WMI Object Browser 150, 292, 339, 344
 WMISeacher 449
 WMISEARCHER 437, 440
 Woche 83
 Wochentag 83 f.
 Wohlgeformtheit 153
 Word 119
 WQL 149 f., 218, 268, 272, 282, 287 f., 337, 339, 348, 440 f.
 Write 191
 WriteBlankLines 191
 Write-EventLog 450
 Write-Host 417
 WriteLine 163, 169, 188, 191
 WSC 18
 WScript 7, 95, 117, 158, 165, 267, 291, 305, 308 f., 312, 329
 WScript.Arguments 158
 Wscript.Echo 70
 WScript.Echo 7 f., 14, 43, 95
 wscript.exe 4, 8, 33, 91, 158, 161, 374, 390 f.
 WScript.Network 116, 497
 WScript.Shell 116, 311, 315, 339, 502
 WSF 20, 479
 WSH 1 f., 4, 6, 8, 23, 25, 39, 103, 116 f., 157, 161, 163, 227, 274 f., 277, 283, 303, 374, 388, 393, 399, 480
 – Deaktivierung 374
 – Debugging 22, 161
 – Impersonifizierung 392
 – Kennwortspeicherung 392
 – Kommandozeilenparameter 161
 – Timeout 162
 WSHCollection 118
 WSHEnvironment 118
 WSHExec 118
 WSH.log 273
 WSHNetwork 117, 222 f., 265
 Wshom.ocx 116
 WSHRun 115, 265, 267, 274, 279 f., 283, 299, 305, 308, 311, 314 ff.

WSH Runtime *siehe* WSHRun
WSHShell 117, 273 ff., 279 f.,
299 ff., 303, 309, 313, 316
WSHShortcut 118
WSHURLShortcut 118
WSH-Virus 373
WSMan 411
WS-Management 422 f., 435,
437
Wurzelschlüssel 305, 314, 317

X

XML 20, 151 ff., 157, 162, 175, 179,
181, 207, 243, 409, 418, 464 f.,
479
– Datei 175, 177, 179, 207, 210,
335, 337
– Dokument 153, 176

XMLDOMImplementation 154
XMLDOMNamedNodeMap 155
XMLDOMNode 155
XMLDOMNodeList 154 f., 176 f.,
210
XMLDOMParseError 154
Xor 54

Y

Year 84, 485

Z

Zahl 55, 99, 102, 321, 481 f., 485
Zahlensystem 43
Zeichenkette 93, 102, 151, 173,
196, 266, 274, 315, 483, 485,
487

Zeichenkettenverkettung 52
Zeiger 102
Zeitformat 482
Zeitintervall 484
Zertifikat 375 f., 382, 386, 421
Zertifikatsdatei 382, 385
Zertifikatsmanager 385
Zertifikatsspeicher 382
Zertifikatsvertrauensliste 381
Zertifikatsverwaltung 422
Zertifizierungsstelle 421
Zielhost 322
Zufallszahl 66
Zufallszahlengenerator 77
Zuweisung 48