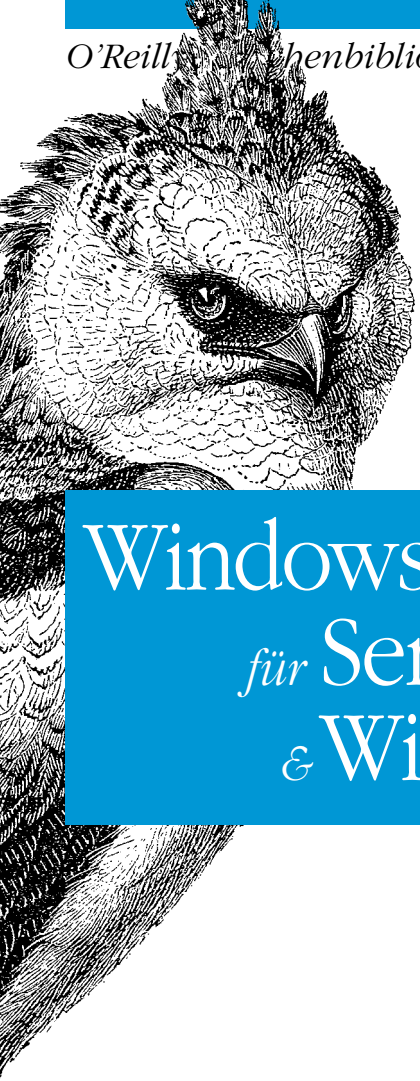
O'Reilly Taschenbibliothek

5. Auflage
Für die gängigen Windows-Betriebssysteme



# Windows-Befehle *für* Server 2012 *&* Windows 8

*kurz & gut*

*Æleen Frisch*
*Helge Klein & Olaf Engelke*

O'REILLY®

5. AUFLAGE

# Windows-Befehle für Server 2012 & Windows 8

*kurz & gut*

*Æleen Frisch, Helge Klein & Olaf Engelke*

O'REILLY®

Beijing · Cambridge · Farnham · Köln · Sebastopol · Tokyo

Die Informationen in diesem Buch wurden mit größter Sorgfalt erarbeitet. Dennoch können Fehler nicht vollständig ausgeschlossen werden. Verlag, Autoren und Übersetzer übernehmen keine juristische Verantwortung oder irgendeine Haftung für eventuell verbliebene fehlerhafte Angaben und deren Folgen.
Alle Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt und sind möglicherweise eingetragene Warenzeichen. Der Verlag richtet sich im Wesentlichen nach den Schreibweisen der Hersteller. Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Alle Rechte vorbehalten einschließlich der Vervielfältigung, Übersetzung, Mikroverfilmung sowie Einspeicherung und Verarbeitung in elektronischen Systemen.

Kommentare und Fragen können Sie gerne an uns richten:
O'Reilly Verlag
Balthasarstr. 81
50670 Köln
E-Mail: kommentar@oreilly.de

Copyright der deutschen Ausgabe:
© 2013 by O'Reilly Verlag GmbH & Co. KG
1. Auflage 2005
2. Auflage 2007
3. Auflage 2008
4. Auflage 2011
5. Auflage 2013

Die Darstellung eines Würgadlers im Zusammenhang mit dem Thema Windows 8 und Windows Server 2012 ist ein Warenzeichen von O'Reilly Media, Inc.

Bibliografische Information Der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über *http://dnb.ddb.de* abrufbar.

Bearbeitung der vierten und fünften Auflage: Olaf Engelke, Hagenow
Bearbeitung der ersten bis dritten Auflage: Helge Klein, Köln
Lektorat: Alexandra Follenius & Susanne Gerbert, Köln
Korrektorat: Eike Nitz, Köln
Satz: III-Satz, Husby, www.drei-satz.de
Umschlaggestaltung: Ellie Volckhausen, Boston & Michael Oreal, Köln
Produktion: Andrea Miß, Köln
Druck: fgb freiburger graphische betriebe; www.fgb.de

ISBN 978-3-95561-061-6

Dieses Buch ist auf 100% chlorfrei gebleichtem Papier gedruckt.

# Inhalt


**Einführung** ..... **7**

**Die Eingabeaufforderung** ..... **10**

**Allgemeine Befehle** ..... **22**

**Dateien und Verzeichnisse** ..... **37**

**Dateisysteme, Volumes und Festplatten** ..... **65**

**Drucker und Warteschlangen** ..... **92**

**Registrierung** ..... **101**

**Prozesse** ..... **107**

**Dienste** ..... **112**

**Berechtigungen und Rechte** ..... **118**

**Systemdiagnose und -information** ..... **127**

**Systemkonfiguration** ..... **145**

**Netzwerk** ..... **160**

**Internet Information Server** ..... **187**

**Benutzer und Gruppen** ..... **190**

**Active-Directory-Verzeichnisdienst** ..... **194**

**Remotedesktopdienste** ..... **230**

**Installation und Deployment** ..... **235**

**Skripten und Batchdateien** ..... **240**

**Zertifikate** ..... **249**

**Die Wiederherstellungsumgebung** ..... **254**

**Konstrukte in Batchdateien** ..... **254**

**Windows PowerShell – Grundlagen** ..... **256**

**LDAP-Suchfilter** ..... **259**

**Windows-GUI – Tipps und Tricks** ..... **260**

**Windows im WWW** ..... **264**

**Index** ..... **267**

# Windows-Befehle für Server 2012 & Windows 8

## Einführung

Am Anfang war die Dunkelheit. So könnte man ganz kurz die Frühzeit der Personalcomputer beschreiben. Jener meist schwarze Bildschirm mit weiß, bernsteingelb oder grün leuchtender Befehlszeile, in der man dem Computer mit mehr oder weniger mühsam erlernten Befehlen und Tastenkombinationen sagen konnte, was er zu tun hatte, war die dominierende Schnittstelle für Administratoren und Anwender. Es sollte einige Zeit dauern, bis grafische Benutzeroberflächen, geeignet auch für Mausbedienung, die Eingabeaufforderung ablösten.

Bei modernen Windows-Versionen wie Windows 8/Server 2012 tritt diese Eingabeaufforderung kaum noch in Erscheinung, sondern wird mehr denn je vor dem Auge des Anwenders verborgen. Das liegt nicht daran, dass sie plötzlich alle Bedeutung verloren hätte. Vielmehr geht Microsoft wohl davon aus, dass die meisten Anwender ohnehin zu den in der grafischen Benutzeroberfläche bereitgestellten Werkzeugen greifen und Profis sich mit der PowerShell arrangieren, die seit Windows 7 integraler Bestandteil des Betriebssystems geworden ist.

### Für wen dieses Buch gedacht ist

Diese Referenz beschreibt die Befehle der Windows-Eingabeaufforderung (die oft auch als Befehlszeile, Kommandozeile, Konsole oder DOS-Prompt bezeichnet wird) in den aktuellen Versionen von Windows. Sie ist nicht nur für Systemadministratoren gedacht,

sondern auch für andere Windows-Anwender. Enthalten sind die meisten Befehle der Client- und Serverversionen von Windows seit Windows 7/Server 2008 R2. Etliche der Befehle haben eine weiter zurückreichende Geschichte und sind in derselben Form oder mit abgewandelter Syntax auch für frühere Versionen des Windows-Betriebssystems gültig.

## Was dieses Buch nicht enthält

Aufgrund des kompakten Formats der Reihe »kurz & gut« wurden selten genutzte und sehr spezielle Befehle teilweise nicht in diese Referenz aufgenommen. Einige weitere Befehle werden nicht behandelt, weil entweder ihre Funktion bereits von einem anderen Befehl übernommen wurde oder sie schlicht veraltet sind. Daher sind Befehle, die es ausschließlich in Windows Vista, Server 2008, Server 2003, XP, anderen früheren Versionen, Resource Kits oder Support Tools gab, nur noch in Ausnahmen in dieser Ausgabe enthalten. Zudem wurden die Informationen zu Befehlen älterer Betriebssystemversionen als Windows 8/Server 2012 gekürzt, um Raum für die neuen Befehle zu schaffen und das Buch dabei weiterhin möglichst kompakt zu halten. In einigen Fällen fehlen auch solche Befehle, die nur dann erscheinen, wenn spezielle Rollen oder Funktionen des Betriebssystems installiert werden.

Ebenfalls nicht enthalten sind Programme mit grafischer Benutzeroberfläche, da sich dieses Buch auf Befehle beschränkt, die in der Eingabeaufforderung bzw. in Skripten nutzbar sind. Ausnahmen gibt es lediglich für solche Programme, die sowohl über eine grafische Oberfläche verfügen als auch komplett oder in wesentlichen Teilen aus der Eingabeaufforderung heraus gesteuert werden können.

## Aufbau

Die Befehle sind nach Funktionsgruppen geordnet und innerhalb dieser Gruppen alphabetisch sortiert. Einen bestimmten Befehl finden Sie am einfachsten über den Index. Die Optionen der Befehle sind zunächst nach ihren Funktionen und dann weitestgehend

nach Wichtigkeit angeordnet. Weniger wichtige Optionen werden gelegentlich nicht aufgeführt. Einige Befehle bieten so viele Optionen, dass es aufgrund des kompakten Formats nicht möglich ist, alle aufzuführen.

Viele der in diesem Buch beschriebenen Befehle lassen sich unter allen gegenwärtig eingesetzten Windows-Versionen verwenden, auch wenn sie nur mit einem Vorgänger oder Nachfolger des von Ihnen eingesetzten Betriebssystems mitgeliefert werden. Einige Tools benötigen jedoch zwingend ein bestimmtes Serverbetriebssystem oder eine Mindestversion des Betriebssystems, weil sie API-Funktionen verwenden, die von Microsoft nicht für alle Versionen des Betriebssystems bereitgestellt wurden. Andere Befehle wurden von Microsoft immer weiter verbessert und verändert, so dass z. B. die Syntax und der Funktionsumfang des mit Windows 8 mitgelieferten Tools anders sind als jene des gleichnamigen Tools in einer Vorgängerversion. In solchen Fällen wird die aktuellste Version beschrieben.

## Konventionen

**`Fett`**
: Kennzeichnet Windows-Befehle und -Optionen.

**`GROSSBUCHSTABEN & FETT`**
: Kennzeichnet interne Befehle des Kommandozeileninterpreters cmd.exe. Diese Befehle sind bei Nutzung einer alternativen Shell, z. B. der PowerShell, nicht unmittelbar verfügbar.

*`Kursiv`*
: Kennzeichnet Parameter, die Sie selbst eingeben müssen.

**`[...]`**
: Kennzeichnet optionale Befehlsteile.

*a* | *b*
: Bedeutet, dass entweder *a* oder *b* eingesetzt werden kann.

*{a | b}*
: Bedeutet, dass entweder *a* oder *b* eingesetzt werden muss.

*HKLM*
*HKCU*
: Kennzeichnen die Registrierungsbäume (Hives) HKEY_LOCAL_MACHINE und HKEY_CURRENT_USER.

**Menüname → Menüname**
: Der Pfeil (→) in Verbindung mit halbfetter Schrift beschreibt die Navigation innerhalb eines Menüs.

Folgende Abkürzungen werden zur Kennzeichnung der Herkunft der Befehle verwendet:

- **AllOS:** in allen Betriebssystemen enthalten
- **7:** Windows 7
- **W2k8R2:** Windows Server 2008 R2
- **8:** Windows 8
- **WS2012:** Windows Server 2012
- **WADK:** Windows Assessment and Deployment Kit
- **WAIK:** Windows Automated Installation Kit
- **WWW:** im Internet frei erhältliches Tool

# Die Eingabeaufforderung

## Starten unter Windows 8/Server 2012

Die neue Benutzeroberfläche begeistert oder verunsichert Anwender der neuen Windows-Version. Sie spiegelt die Absicht von Microsoft wider, den Desktop und damit klassische Windows-Anwendungen abzuschaffen, indem er nur noch als eine App unter vielen angeboten und als »legacy« deklariert ist, also als veraltet. Das Schicksal des Desktops teilt auch die Eingabeaufforderung, so dass es sich bei dieser Auflage sehr wohl um die letzte dieser Reihe handeln könnte. Dennoch ist in den aktuellen Versionen von Windows-Client und -Server die Eingabeaufforderung in ihrer vollen Funktionalität enthalten, allerdings versteckt. Am schnellsten finden Sie sie, indem Sie im Startbildschirm von Windows 8 oder Server 2012 einfach drauf-

lostippen und entweder `cmd` oder `Eingabeaufforderung` eingeben. Sobald Windows fündig geworden ist, können Sie das Symbol zur unmittelbaren Ausführung anklicken. Diese Variante hat einen in vielen Situationen entscheidenden Nachteil, denn die Eingabeaufforderung wird in diesem Fall im Benutzerkontext aufgerufen. Da aber viele der in diesem Buch beschriebenen Befehle Administratorrechte benötigen (und seit Windows Vista/Server 2008 zudem bei standardmäßig aktivierter Benutzerkontenkontrolle die hohe Verbindlichkeitsstufe), reicht der Benutzerkontext oft nicht.

Klicken Sie also stattdessen das gefundene Symbol der Eingabeaufforderung mit der rechten Maustaste an und wählen Sie in der dann im unteren Bildschirmbereich angezeigten Symbolleiste *Als Admin ausführen*, um ein Fenster mit der Eingabeaufforderung mit erhöhten Rechten zu öffnen. Alternativ funktioniert nach wie vor die Tastenkombination *STRG+Umschalt+ENTER* bei ausgewähltem Symbol sowie zum Bestätigen der Sicherheitsabfrage *ALT+J*.

Soll häufig auf die Eingabeaufforderung zurückgegriffen werden, ist diese Methode nicht sonderlich praktikabel. Es empfiehlt sich das Anlegen einer Verknüpfung. Und wo sollte diese liegen? Windows 8 bietet bei Rechtsklick die Erstellung einer Verknüpfung im Startmenü (*An Startmenü anheften*) oder in der Taskleiste des Desktops (*An Taskleiste anheften*) an, wobei Letzteres erfahrungsgemäß die praktischere Variante ist.« Durch Rechtsklick auf das angeheftete Symbol, nochmaligen Rechtsklick auf den Menüeintrag *Eingabeaufforderung* und im folgenden Kontextmenü Anklicken von *Eigenschaften* können Sie einige Einstellungen für verbesserten Komfort anpassen:

*Ausführen in:*
: Hier können Sie einen alternativen Pfad eingeben, in dem sich die Eingabeaufforderung öffnen soll, beispielsweise *%WINDIR%\system32* oder einen anderen Pfad, in dem sich zusätzliche Skriptdateien oder zu bearbeitende Dateien befinden.

*Tastenkombination:*
: Hier können Sie eine Tastenkombination festlegen, mit der die Eingabeaufforderung gestartet werden kann. Verwenden Sie

dafür keine schon anderweitig im System oder in ständig laufenden Anwendungsprogrammen verwendete Kombination, um unerwartete Ergebnisse zu verhindern.

Durch Klick auf die Schaltfläche *Erweitert* öffnet sich das Dialogfeld *Erweiterte Eigenschaften*, in dem Sie durch das Kontrollkästchen *Als Administrator ausführen* festlegen können, dass bei jeder Ausführung der Verknüpfung die Eingabeaufforderung mit erhöhten Rechten geöffnet wird.

Weitere Einstellmöglichkeiten betreffen unter anderem Schriftart und -größe, den verfügbaren Puffer zur Zwischenspeicherung von Befehlen und die Fenstergröße. Eine Eingabeaufforderung im Vollbildmodus steht schon seit mehreren Windows-Versionen nicht mehr zur Verfügung.

Ganz klassisch können Sie eine Verknüpfung auf dem Desktop erzeugen, indem Sie mit der rechten Maustaste auf eine beliebige freie Stelle des Desktops klicken, aus dem Kontextmenü den Eintrag **Neu → Verknüpfung** wählen, als Speicherort des Elements `cmd` eingeben und einen beliebigen Namen festlegen. Nach Beendigung des Assistenten können Sie auch hier die Eigenschaften der Verknüpfung bearbeiten, wie oben für das an die Taskleiste angeheftete Symbol beschrieben wurde.

Schließlich können Sie unter Windows 8/Server 2012 die Tastenkombination *WIN+X* verwenden und aus dem dann auf dem Desktop links unten erscheinenden Menü den Eintrag *Eingabeaufforderung (Administrator)* mit der Maus auswählen.

Eine aus Sicherheitsgründen wenig empfehlenswerte Alternative zur generellen Ausführung der Eingabeaufforderung und aller anderen Anwendungen mit erhöhten Rechten ist die Deaktivierung der Benutzerkontensteuerung des Betriebssystems bei gleichzeitiger Benutzung eines Benutzerkontos der Administratorengruppe.

Sie können jederzeit nachsehen, ob ein Befehlszeilenfenster mit erhöhten Rechten ausgeführt wird: In der Regel beginnt die Titelzeile des Fensters in diesem Fall mit »Administrator:«.

## Eingabe von Befehlen

Befehle gibt man ein, indem man sie in die Befehlszeile tippt und mit der Enter-Taste die Ausführung auslöst. Dabei kann man in der Regel nicht viel falsch machen, abgesehen von Syntaxfehlern, Tippfehlern und der unbeabsichtigten Verwendung von nicht zur Aufgabenstellung passenden Befehlen. Dennoch sind Befehlseingaben in ihren Grundanforderungen nicht immer konsistent, so dass es sich lohnen kann, im Fehlerfall die Hilfe zum jeweiligen Befehl etwas ausführlicher zu lesen:

- Bei der Eingabe von Befehlen ist es Ihnen freigestellt, ob Sie Groß- oder Kleinbuchstaben verwenden.
- Befehle und Befehlsoptionen können normalerweise in Groß- oder Kleinbuchstaben eingegeben werden. Ist das der Fall, werden sie in diesem Buch kleingeschrieben. Nur Optionen, die großgeschrieben werden *müssen*, werden in Großbuchstaben geschrieben.
- Befehlsoptionen werden normalerweise durch einen Schrägstrich eingeleitet: /x. In vielen Fällen kann der Schrägstrich durch ein Minuszeichen ersetzt werden. Einige Befehle akzeptieren nur das Minuszeichen.
- Die Reihenfolge der Optionen ist nicht einheitlich und nicht immer beliebig. Bitte entnehmen Sie sie der Syntax des jeweiligen Befehls.
- Einzelne Parameter werden abhängig vom Befehl durch Leerzeichen, Kommata oder Strichpunkte voneinander getrennt.
- Parameter mit Leerzeichen, beispielsweise Verzeichnispfade und Dateinamen, müssen in der Regel von Anführungszeichen umschlossen werden.
- Regions- und Spracheinstellungen sowie die Sprachversion des Betriebssystems können die Syntax beeinflussen. In diesem Buch wird die deutschsprachige Version des Betriebssystems mit den Einstellungen für Deutschland verwendet.
- Befehle können in der nachfolgenden Zeile fortgesetzt werden, wenn die vorherige Zeile mit dem **^**-Zeichen beendet wurde.

- Durch ein vorangestelltes ^-Zeichen wird auch verhindert, dass der Befehlsinterpreter das folgende Zeichen interpretiert. Solche Zeichen werden auch »Escape-Zeichen« genannt.
- Mehrere Befehle können mit dem &-Zeichen verknüpft werden: `Befehl1 & Befehl2`. Die Befehle werden der Reihe nach ausgeführt.
- Die Ausführung eines Befehls kann davon abhängig gemacht werden, ob der vorangegangene Befehl korrekt ausgeführt werden konnte. Dazu werden die Befehle mit && bzw. || verknüpft:

  `Befehl1 && Befehl2`

  *Befehl2* wird nur ausgeführt, wenn *Befehl1* erfolgreich ausgeführt werden konnte.

  `Befehl1 || Befehl2`

  *Befehl2* wird nur ausgeführt, wenn *Befehl1* nicht erfolgreich ausgeführt werden konnte.

## Umleitung der Ein- und Ausgabe

*<Datei*
: Liest Standardeingabe aus einer Datei statt von der Tastatur.

*>Datei*
*1>Datei*
: Schreibt Standardausgabe in eine Datei statt auf den Bildschirm.

*>>Datei*
*1>>Datei*
: Hängt Standardausgabe an eine Datei an.

*2>Datei*
: Schreibt Standardfehlerausgabe in eine Datei.

*2>>Datei*
: Hängt Standardfehlerausgabe an eine Datei an.

*>Datei 2>&1*
: Schreibt Standardausgabe in eine Datei und leitet Standardfehlerausgabe zur Standardausgabe um. Damit werden Stan-

dardausgabe und Standardfehlerausgabe in dieselbe Datei geschrieben.

*Befehl1 |Befehl2*
*Befehl1 0>Befehl2*
: Stellt eine Verknüpfung zwischen der Standardausgabe des ersten Befehls (*Befehl1*) und der Standardeingabe des zweiten Befehls (*Befehl2*) her.

  In Einzelfällen, beispielsweise für die Wiederverwendung von komplexen Werten wie GUIDs zur Vermeidung von Tippfehlern, ist es hilfreich, den gewünschten Ausschnitt der Eingabeaufforderung in die Zwischenablage zu kopieren und von dort entweder direkt oder nach Bearbeitung im Editor wieder einzufügen. Dafür benutzen Sie in der Titelzeile des Fensters der Eingabeaufforderung das Systemmenü oben links, klicken auf *Bearbeiten/Markieren* und können dann mit der Maus oder mit der Tastatur unter Zuhilfenahme der Pfeiltasten und der gedrückten Umschalttaste den zu kopierenden Bereich in Rechteckform auswählen. Sobald Sie die Auswahl abgeschlossen haben, verwenden Sie die Eingabetaste, um den Text des ausgewählten Bereichs in die Zwischenablage zu kopieren.

  Zum Einfügen aus der Zwischenablage verwenden Sie wiederum den Menüeintrag *Bearbeiten/Einfügen* aus dem Systemmenü.

## Umgebungsvariablen

Dieser Abschnitt erklärt einige wichtige Windows-Umgebungsvariablen. Variablen werden in Eingabeaufforderung und Batchdateien mit dem Prozentzeichen (%) ausgelesen, zum Beispiel zeigt der Befehl `echo %SystemRoot%` den Pfad des Windows-Ordners. Damit können Anwendungen und Skripten auf unterschiedlich eingerichteten Systemen ausgeführt werden, ohne dass eine abweichende Ordnerstruktur für die Aufrufe manuell berücksichtigt werden müsste. Die aktiven System- und Benutzervariablen können mit `set` angezeigt und temporär angepasst sowie mit `setx` dauerhaft geändert werden.

*COMPUTERNAME*
: Enthält den Computernamen der Windows-Installation.

*HOMEDRIVE und HOMEPATH*
: Das Laufwerk und der Pfad auf diesem Laufwerk zum Benutzerprofilordner des angemeldeten Benutzers.

*PATH*
: Mehrere durch Semikolon getrennte Verzeichnisse, die in dieser Reihenfolge nach Befehlen durchsucht werden, wenn diese ohne vorangestellten Pfad aufgerufen werden.

*PATHEXT*
: Mehrere durch Semikolon getrennte Dateierweiterungen, die in dieser Reihenfolge an einen Befehl ohne Erweiterung angehängt werden, um die ausführbare Datei für den Befehl zu finden. Diese Erweiterungen müssen den voranstehenden Punkt beinhalten, also beispielsweise *.exe* oder *.cmd*.

*ProgramFiles*
: Der Verzeichnisname des Programmordners (ab Vista normalerweise *C:\Program Files*).

  **Warnung:** Bei 64-Bit-Versionen des Betriebssystems handelt es sich um den Programmordner für 64-Bit-Software.

*ProgramFiles(x86)*
: Der Verzeichnisname des Programmordners für 32-Bit-Anwendungen in einem 64-Bit-Betriebssystem.

*SystemRoot und windir*
: Der Verzeichnisname des Windows-Systemverzeichnisses (normalerweise *C:\WINDOWS*).

*TEMP und TMP*
: Der komplette Pfad eines Verzeichnisses, das von Anwendungen und vom Betriebssystem zur Ablage temporärer Dateien verwendet wird.

*USERNAME*
: Der Name des angemeldeten Benutzers.

*USERPROFILE*
: Der Pfad zum Profilverzeichnis des angemeldeten Benutzers.

## Installation zusätzlicher Administrationstools

Anders als die Datenträger inzwischen veralteter Windows-Versionen enthalten die Installationsmedien von Windows 8 und Server 2012 keine Support-Tools, da es diese bereits seit Vista nicht mehr gibt. Die Support-Tools von Windows Server 2003 können jedoch (noch) von den Microsoft-Downloadseiten heruntergeladen und trotz der Warnung des Installationsprogramms mit Einschränkungen auch unter aktuellen Windows-Versionen verwendet werden.

Microsoft hat zudem die Praxis aufgegeben, Zusatztools über Resource Kits zugänglich zu machen. Stattdessen werden nützliche Programme direkt veröffentlicht, beispielsweise die Werkzeuge der Sysinternals-Suite. Auf einige dieser Programme gehen wir in diesem Buch ein.

Die Resource Kits älterer Windows-Versionen sind bislang unverändert erhältlich. Nützlich sind insbesondere die Tools des Windows Server 2003 Resource Kit, die Microsoft zum freien Download bereitstellt. Die darin enthaltenen Programme funktionieren trotz der Warnung des Installationsprogramms meist auch unter Windows 8/Server 2012. Gerade bei systemnahen Programmen ist generell nicht auszuschließen, dass diese in Versionen des Betriebssystems, für die sie nicht vorgesehen waren, tatsächlich nicht korrekt funktionieren oder das System sogar beschädigen können. Daher sollten Sie sich vor dem Einsatz solcher Werkzeuge auf produktiven Systemen genauestens über eventuelle Risiken informieren. Zudem sollten Sie Programme, die Sie möglicherweise zukünftig einsetzen wollen, herunterladen und lokal abspeichern. Die Erfahrungen der Vergangenheit haben gezeigt, dass Microsoft sich nicht scheut, beliebte Hilfsprogramme von der Website zu nehmen, wenn sie in Einzelfällen oder bei zu unbedarfter Nutzung unerwünschte Nebenwirkungen zeigen können oder aus Microsoft-Sicht veraltet sind. In der aktuellen Ausgabe dieses Buches werden diese Programme nicht mehr behandelt.

Seit Server 2008 hat Microsoft die Serververwaltungstools (ehemals enthalten in der Datei *adminpak.msi*) umbenannt in »Remoteserver-Verwaltungstools« (Remote Server Administration Tools,

RSAT). Diese dienen der Verwaltung von Servern ab Version 2003. Unter Server 2008 R2 sind sie als Feature im Server Manager installierbar, unter Server 2012 im Dashboard, unter Windows 8 über die Tastenkombination *WIN+X* und dann im Menü *Systemsteuerung→Programme→Windows-Features aktivieren oder deaktivieren*. Allerdings müssen die RSAT für die Nutzung unter Windows 8 und Windows 7 zunächst von Microsoft heruntergeladen und auf herkömmliche Art und Weise installiert werden. Berücksichtigen Sie, dass die Werkzeuge wählerisch beim Clientbetriebssystem sind – so lassen sich die RSAT für Server 2012 beispielsweise nicht unter Windows 7 installieren.

Für die Unterstützung der automatisierten Installation von aktuellen Windows-Versionen in Unternehmen stellt Microsoft das *Windows Assessment and Deployment Kit* zum Herunterladen bereit, das auch einige nützliche Werkzeuge für die Eingabeaufforderung enthält. Für frühere Versionen von Windows hat diese Sammlung die Bezeichnung *Windows Automated Installation Kit* (WAIK).

## Hilfebefehle und -dateien

`help Befehl`
: Zeigt die Hilfe für viele Windows-Standardbefehle an.

***`Befehl`*** `/?`
: Zeigt bei den meisten ausführbaren Dateien einen Hilfetext an.

`net help` ***`Befehl`***
: Zeigt die Hilfe für einen der net-Befehle an. Beispiel: `net help user` erklärt die Optionen des Befehls `net user`.

`net helpmsg` ***`nnnn`***
: Zeigt den Fehlertext zum Windows-Fehler mit der Nummer *nnnn* an.

Das Hilfe- und Supportcenter in Windows 7 und Windows Server 2008 R2 hat eine Befehlszeilenreferenz lediglich als Verweis auf englischsprachige Onlinequellen integriert, unter Windows 8 und Windows Server 2012 fehlt eine solche bislang ganz oder ist sehr

gut verborgen, ein weiteres Indiz dafür, dass die Tage der klassischen Eingabeaufforderung gezählt sein dürften.

### Resource Kits, Support Tools, Windows Assessment and Deployment Kit

Die Resource Kits und Support Tools enthalten weitere, zum Teil sehr umfangreiche Hilfedateien, die Sie am einfachsten über das Startmenü öffnen. Leider können diese Hilfedateien unter Windows 7/2008 R2 und Folgeversionen nicht mehr direkt geöffnet werden, da die Hilfsanwendung *helpctr* in diesen nicht mehr existiert. Sofern erforderlich, kann die Datei *WinHlp32.exe* bei Microsoft in einer dem jeweiligen Betriebssystem angepassten Version heruntergeladen werden. Teilweise sind die Hilfedateien der Zusatzwerkzeuge nur in englischer Sprache verfügbar.

## Windows PowerShell

Seit die Windows PowerShell im Jahr 2007 erstmals veröffentlicht wurde, ist ihre Integration in Microsoft-Betriebssysteme und -anwendungen weit vorangeschritten. In Windows Server 2012 und Windows 8 ist Version 3.0 der PowerShell integriert. Bei der PowerShell handelt es sich um eine .NET-basierte Skriptumgebung, die sowohl die klassische Eingabeaufforderung als auch früher beliebte Skriptsprachen wie VB-Script beerben soll.

Durch die Abhängigkeit vom .NET-Framework ist sie in früheren Windows-Versionen unter Windows PE nicht verfügbar, ebenso wenig in der Installationsvariante Server Core in den Betriebssystemversionen vor Windows Server 2012. Sie ist nicht textbasiert wie andere eingabezeilenorientierte Benutzeroberflächen, sondern gibt Objekte über Pipelines (|) von einem Befehl zum anderen weiter. Die Benennung der Befehle (die als »Cmdlets« bezeichnet werden) folgt dem Schema Verb-Nomen, z.B. `get-command` (Auflistung aller Befehle). Die PowerShell verwendet ein erweiterbares Providermodell, um neben dem Dateisystem folgende Datenspeicher als Laufwerk anzusprechen: Registrierung, Zertifikatsspeicher, Umgebungsvariablen, Aliase, Variablen und Funktionen.

Mit der PowerShell 2.0, die für Windows Server 2008 R2 und Windows 7 schon vorinstalliert ist und für die Betriebssystemversionen ab Windows XP/Server 2003 heruntergeladen werden kann, kamen weitere Möglichkeiten hinzu: Verwaltung von Remotecomputern, Unterstützung von Hintergrundprozessen, ein Provider für Active Directory (das entsprechende Modul lässt sich allerdings nur in Windows Server 2008 R2 und Windows 7 ohne Einschränkungen installieren und einsetzen) sowie ein Editor mit Debuggingfunktionen.

PowerShell 3.0, die in Windows Server 2012 und Windows 8 integriert ist, bringt nochmals eine erhebliche Steigerung des Funktionsumfangs mit sich. PowerShell-Workflows erlauben die Nutzung der Windows Workflow Foundation. Die Unterstützung des .NET-Framework 4 wurde integriert. Windows Server 2012 bietet mit PowerShell Web Access die Möglichkeit, PowerShell-Befehle und -Skriptdateien in einer webbasierten Konsole auszuführen. Das CIM (Common Information Model) wurde als Methode zur Kommunikation integriert. Vorhandene Module werden automatisch importiert, wenn ein in ihnen enthaltenes Cmdlet aufgerufen wird. Intellisense und Tab-Vervollständigung helfen bei der Eingabe von Befehlen. Eine Hilfe, die online aktualisiert werden kann, steuert zur Funktionalität bei. Das sind nur einige der Vorteile, die zeigen, dass die PowerShell mittlerweile »erwachsen geworden« ist.

Im Gegensatz zu früheren Versionen lässt sich die PowerShell 3.0 auch in Windows PE (Windows Preinstallation Environment) integrieren. Zudem wurden zahlreiche neue Befehle hinzugefügt und bestehende erweitert. Für Windows 7 erhalten Sie die PowerShell 3.0 als Bestandteil des Windows Management Framework unter *http://www.microsoft.com/en-us/download/details.aspx?id=34595*. Bitte beachten Sie, dass zahlreiche mit PowerShell 3.0 eingeführte Befehle nur unter Windows 8/Server 2012 zur Verfügung stehen.

Um zu ermitteln, welche Version der PowerShell aktuell auf Ihrem Rechner installiert ist, führen Sie den folgenden Befehl aus:

```
if (test-path variable:psversiontable)
{$psversiontable.psversion} else {[version]"1.0.0.0"}
```

Die Ausgabe zeigt in der Spalte *Major* die Versionsnummer an.

Auch wenn es nach wie vor Anwendungsfälle gibt, in denen die PowerShell im Vergleich zu den spezialisierten Befehlen der Eingabeaufforderung zu komplex ist, um mal eben ein bestimmtes Ziel zu erreichen, gibt es kaum eine bislang in den Bereich der Eingabeaufforderung fallende Aufgabe, die sich nicht mit der PowerShell lösen ließe.

Zudem bietet die PowerShell zahllose Möglichkeiten, die mit Bordmitteln in der Eingabeaufforderung nicht und auch in anderen Skriptsprachen nicht trivial umzusetzen sind. Da fast alle Befehle der Eingabeaufforderung – mit Ausnahme der Parameterübergabe für interne Befehle – unmittelbar auch in der PowerShell-Konsole funktionieren, besteht eigentlich kaum mehr ein Grund, noch aus der Eingabeaufforderung heraus zu arbeiten. Lediglich die ISE (Integrated Scripting Environment, integrierte Skriptumgebung) weist bei der Unterstützung von einzelnen Befehlen der Eingabeaufforderung noch Mankos auf. Die dort integrierte umfassende Hilfe unterstützt dafür den Anwender dabei, die anfangs steile Lernkurve beim Erlernen der PowerShell-Skriptsprache zu bewältigen.

Im weiteren Verlauf dieses Buches werden, sofern es sich anbietet, Möglichkeiten des Einsatzes der PowerShell oder einer Kombination aus PowerShell und Befehlszeile gezeigt. Die praxisnahen Beispiele werden zeigen, dass die PowerShell ihre Stärken auch in Alltagssituationen voll zur Geltung bringen kann. Auf die PowerShell bezogene Abschnitte werden als PowerShell-Tipp gekennzeichnet. Die aufgeführten Beispiele und Hinweise sollen weniger perfekt sein als vielmehr die vielfältigen Möglichkeiten der Problemlösung mit PowerShell nahebringen oder beim Finden dafür relevanter Cmdlets helfen.

# Allgemeine Befehle

## clip

AllOS

**Befehl | clip**
**clip < *Datei***

Der Befehl clip ermöglicht die Umlenkung der Bildschirmausgabe eines Befehls in die Zwischenablage, um deren Inhalt in eine Windows-Anwendung einzufügen.

**PowerShell-Tipp:** Mit clip lassen sich auch PowerShell-Bildschirmausgaben einfach als Text in die Zwischenablage kopieren.

## cmd

AllOS

**cmd** [***Optionen***] [[**/c** | **/k**] [**/s**] ***Befehl***]

Startet eine neue Instanz des Windows-Befehlsinterpreters. Wurde ein Befehl angegeben, wird er ausgeführt. Verwenden Sie die mit cmd /? aufrufbare ausführliche Hilfe, um Informationen zu weiteren Features zu erhalten (z.B. der automatischen Vervollständigung von Pfaden und Befehlen oder der verzögerten Expansion von Variablen). Über die Eigenschaften eines Kommandozeilenfensters (Klick auf das *Systemmenü* in der linken Ecke des Titelfensters/*Eigenschaften*) können Sie sein Aussehen und Verhalten in weiten Bereichen beeinflussen, wobei diese Anpassung im Gegensatz zu den Eigenschaften der Verknüpfung nur für das aktive Fenster gilt. Insbesondere die Vergrößerung des Fensterpuffers, der das Scrollen in den nicht mehr am Bildschirm sichtbaren Zeilen ermöglicht, ist oft sinnvoll.

Der Befehl exit beendet den Kommandozeileninterpreter.

### Optionen

**[/c | /k] [/s]**
: Der Interpreter führt den angegebenen Befehl aus und bleibt nach Beendigung des Befehls aktiv (/k) oder beendet sich (/c). Die Option /s veranlasst den Befehlsinterpreter, den Befehl umschließende Anführungszeichen vor Ausführung des Befehls zu entfernen (normalerweise werden diese beibehalten). Das gilt jeweils in Verbindung mit /c oder /k.

**/q**
: Schaltet die Befehlsausgabe ab (siehe echo off).

**/e:{*on* | *off*}**
: Aktiviert oder deaktiviert die Erweiterungen des Befehlsinterpreters. Der Standardwert wird durch den Registrierungswert *\Software\Microsoft\Command Processor\Enable Extensions* unter *HKCU* oder *HKLM* bestimmt. Im Auslieferungszustand sind die Erweiterungen aktiviert.

**/a | /u**
: Die Ausgabe von internen Befehlen erfolgt im ANSI-(Standard-) bzw. Unicode-Format.

**/d**
: Deaktiviert die Autorun-Einträge in der Registrierung unter *\Software\Microsoft\Command Processor\Autorun* in *HKLM* und *HKCU*.

**/f:{*on* | *off*}**
: Aktiviert bzw. deaktiviert die Ergänzung von Datei- und Verzeichnisnamen mit der Tabulator-Taste.

Weitere Parameter entnehmen Sie dem Aufruf von `cmd /?` in einem offenen Befehlszeilenfenster.

```
cmd /u /? /c > c:\temp\help.txt
```

Leitet unter Verwendung des Unicode-Formats die Hilfeinformationen zum Befehlszeileninterpreter in die Datei *help.txt* um, womit deutsche Sonderzeichen in der Datei in Windows-Anwendungen wie Notepad korrekt dargestellt werden.

**PowerShell-Tipp:** Da interne Befehle des Befehlszeileninterpreters wie beispielsweise `DIR` in der PowerShell nur nachgebildet sind und diese Nachbildungen gebräuchliche Parameter nicht unterstützen, kann durch expliziten Aufruf von `cmd` in der Powershell-Konsole auf das Original des Befehls zurückgegriffen werden:

```
cmd /c DIR /p
```

Gibt den Inhalt des aktuellen Verzeichnisses in einer PowerShell-Konsolensitzung seitenweise am Bildschirm aus.

## command

AllOS (nur 32-Bit)

**`command [[Laufwerk:]Pfad][Gerät][e:nnnnn] [[/p | /c *Befehl*] [/MSG]`**

Startet eine neue Instanz der MS-DOS-Eingabeaufforderung. Da es sich dabei um eine 16-Bit-Instanz handelt, ist diese Möglichkeit nicht Bestandteil der 64-Bit-Versionen von Windows. Der 16-Bit-Befehlsinter-

preter ist nur aus Gründen der Kompatibilität zu sehr alten MS-DOS-Anwendungen noch enthalten.

**Tipp:** Um das Tastaturlayout in einer solchen 16-Bit-MS-DOS-Eingabeaufforderung auf Deutsch umzustellen, verwenden Sie den Befehl `kb16 gr`, den Sie bei Bedarf in einer Batchdatei vor dem Aufruf des eigentlichen Programms einfügen.

## COLOR

AllOS

**`color` *`Farbcode1 Farbcode2`***

Stellt die Hintergrundfarbe und die Schriftfarbe in der aktuell geöffneten Konsole um. Die Farbattribute sind als hexadezimale Werte anzugeben, die durch die Hilfe angezeigt werden. So stellt `color 1f` den Hintergrund auf blau und die Schrift auf weiß ein. Ohne Angabe von Farbcodes wird die Standardanzeige wiederhergestellt.

**PowerShell-Tipp:** Auch die Farbgebung der PowerShell-Konsole lässt sich mit dem Befehl `cmd /c color xy` anpassen. Alternativ lassen sich die folgen nativen PowerShell-Befehle verwenden:

```
$HOST.UI.RawUI.BackgroundColor = "DarkBlue"
$HOST.UI.RawUI.ForegroundColor = "White"
```

## date

AllOS

**`date [`*`tt.mm.`*`[jj]jj] [/t]`**

Stellt das angegebene Datum ein oder fragt danach, wenn es nicht angegeben wurde. Mit der Option `/t` wird das Datum angezeigt, ohne es zu ändern.

**PowerShell-Tipp:** `Get-Date -displayhint date` zeigt das aktuelle Systemdatum an. Um das Datum in Kurzform angezeigt zu bekommen, verwenden Sie `Get-Date -format dd.MM.yyyy`.

Das Cmdlet *Set-Date* erlaubt hingegen, ein neues Systemdatum zu setzen. Beachten Sie, dass dabei die Uhrzeit auf 00:00 gesetzt wird, sofern sie nicht explizit angegeben wird.

## doskey

AllOS

**`doskey [`*`Optionen`*`]`**

Erlaubt den Zugriff auf bereits ausgeführte Befehle mit den Tasten *Pfeil hoch* und *Pfeil herunter* sowie die Erstellung von Makros (Alias-Definitionen).

### Befehlshistorie und Editieroptionen

**`/history`**
: Zeigt die vollständige Befehlshistorie an. Durch Ausgabeumleitung mit > gefolgt von einem Dateinamen können Sie diese Liste für die künftige Verwendung als Makrodatei abspeichern.

**`/listsize=n`**
: Stellt die Größe der Befehlshistorie auf *n* Einträge ein.

Da die Funktionstasten von doskey standardmäßig verfügbar sind, können Sie sich mit *F7* jederzeit die Befehlshistorie anzeigen lassen und mit den Pfeil nach oben/nach unten-Tasten einen Befehl auswählen. Mit *ALT+F7* können Sie diese Historie löschen, mit *F9* wählen Sie eine bestimmte Befehlsnummer aus, um diesen Befehl zu starten.

**`/insert | /overstrike`**
: Stellt den Bearbeitungsmodus für aus der Historie abgerufene Befehle auf *Einfügen* bzw. auf *Überschreiben*. Die Standardeinstellung ist *Einfügen*. Durch Betätigen der Taste *EINFG* können Sie den Modus ebenfalls wechseln.

### Makro-Optionen

**`Makroname=Befehl`**
: Definiert ein Makro. Innerhalb des Befehls können die folgenden Variablen verwendet werden: `$T` fügt ein Trennzeichen ein, `$1` bis `$9` erlauben den Zugriff auf einzelne Parameter, `$*` fügt alle eingegebenen Parameter ein. `$G` ersetzt das Zeichen für die Ausgabeumleitung.

  Beispiel:

  ```
  doskey liste=dir /b $1 $g dateien.txt $t notepad.exe
  dateien.txt
  ```

  Erzeugt ein Makro mit dem Namen *liste*, das die Ausgabe des Befehls `dir /b` für den als Parameter angegebenen Verzeichnispfad in die Datei *dateien.txt* im aktuellen Verzeichnis umlenkt, und öffnet die erzeugte Datei anschließend im Editor zur weiteren Verarbeitung. Das Makro wird durch den Befehl `liste Verzeichnisname` und Betätigung der Eingabetaste aufgerufen.

**`/macros`**
: Zeigt alle vorhandenen Makros an.

**`/macros:all`**
: Zeigt zusätzlich die ausführbaren Dateien zugewiesenen doskey-Makros an.

**/macrofile=*Datei***
: Aktiviert alle in der angegebenen Datei enthaltenen Makros.

**/exename=*Exe-Datei***
: Erlaubt die Zuordnung einer ausführbaren Datei zu dem Makro, das soeben definiert wird.

**/macros:Exe-Datei**
: Zeigt alle vorhandenen Makros an, die der angegebenen ausführbaren Datei zugeordnet sind.

Die Tastenkombination *ALT+F10* löscht alle definierten Makros.

**PowerShell-Tipp:** Standardmäßig werden doskey-Makros dem Befehlszeileninterpreter *cmd.exe* zugeordnet. Durch Verwendung des Parameters `/exename=powershell.exe` können Sie doskey-Makrodateien für die Ausführung von PowerShell-Befehlen verwenden. Dabei beachten Sie bitte, dass die Syntax der Befehle den Anforderungen der PowerShell entsprechen muss – so wäre z. B. die Variable `%WINDIR%` durch den PowerShell-Ausdruck `$env:WINDIR` zu ersetzen.

## find

AllOS

**find [*Optionen*] "*Zeichenfolge*" [*Dateien*]**

Sucht in den angegebenen Dateien, in einem über die Tastatur eingegebenen Text oder über eine Pipeline in der Standardeingabe nach der von Anführungszeichen umschlossenen Zeichenfolge und gibt Zeilen aus, die die Zeichenfolge enthalten (oder deren Anzahl).

Eine beliebte Anwendung von `find` ist die Filterung der Ausgabe eines anderen Befehls. Das folgende Beispiel gibt unter Windows 8 diejenigen Verzeichnisse innerhalb des Benutzerprofils aus, die als Junction auf ein anderes Verzeichnis zeigen:

```
dir /ad %userprofile% | find /i "junction"
```

### Optionen

**/v**
: Zeigt nur die Zeilen an, in denen die Zeichenfolge nicht vorkommt.

**/i**
: Ignoriert Groß-/Kleinschreibung beim Vergleich.

**/c**
: Zeigt nur die Anzahl der übereinstimmenden Zeilen an.

**/n**
: Zeigt vor jeder Zeile die Zeilennummer an.

**/offline**
Schließt bei der Suche in Dateien auch Offlinedateien ein.

## findstr

AllOS

**findstr [*Optionen*] [/c:*Zeichenfolge* | /g:*Datei* | *Zeichenfolgen*] [*Dateien*]**

Sucht in den angegebenen Dateien nach einer oder mehreren Zeichenfolgen oder regulären Ausdrücken und gibt übereinstimmende Zeilen aus. Wenn keine Dateien angegeben wurden, wird die Standardeingabe durchsucht. Falls Sie mehrere Suchbegriffe verwenden wollen, müssen Sie diese in Anführungszeichen einschließen.

### Optionen

**/r**
Interpretiert die Zeichenfolge als regulären Ausdruck.

**/l**
Interpretiert die Zeichenfolge buchstabengetreu.

**/c:***Zeichenfolge*
Kennzeichnet die angegebene Zeichenfolge als Suchbegriff. Diese Option ist dann besonders hilfreich, wenn der Suchbegriff Leerzeichen enthält, da ohne die Option nach jedem einzelnen Wort in der Zeichenfolge gesucht wird.

**/g:***Datei*
Liest die Suchausdrücke aus der angegebenen Datei. Ein Schrägstrich anstelle des Dateinamens bedeutet, dass der Dateiname an der Eingabeaufforderung abgefragt wird.

**/b | /e**
Übereinstimmende Zeilen werden nur dann ausgegeben, wenn die Übereinstimmung am Anfang (/b) oder am Ende (/e) der Zeile auftritt. Es kann nur eine dieser beiden Optionen verwendet werden.

**/i**
Vergleicht ohne Berücksichtigung von Groß-/Kleinschreibung.

**/v**
Zeigt nicht übereinstimmende Zeilen an.

**/x**
Zeigt nur exakt übereinstimmende Zeilen an.

**/n | /o**
Zeigt die Zeilennummer (/n) oder die Anzahl der Zeichen vom Dateianfang bis zur Übereinstimmung (/o) für jede Fundstelle an.

**/m**
Zeigt nur die Namen derjenigen Dateien an, in denen eine Übereinstimmung gefunden wurde.

**/s**
Sucht in den angegebenen Dateien im aktuellen Verzeichnis und allen Unterverzeichnissen.

**/f:***Datei*
Liest die Dateiliste aus der angegebenen Datei. Ein Schrägstrich anstelle des Dateinamens bedeutet, dass der Dateiname bei der Eingabeaufforderung abgefragt wird.

**/d:***Verzeichnisliste*
Durchsucht die Dateien in der angegebenen Verzeichnisliste. Die einzelnen Verzeichnisse werden durch Kommata voneinander getrennt.

**/p**
Dateien, die nicht druckbare Zeichen enthalten, werden übersprungen.

**/offline**
Schließt bei der Suche in Dateien auch Offlinedateien ein.

### Bestandteile von regulären Ausdrücken

**.**
Ein beliebiges Zeichen.

**^ | $**
Der Anfang bzw. das Ende einer Zeile.

**\< | \>**
Der Anfang bzw. das Ende eines Worts.

*****x*
Zeichen *x* verwenden, auch wenn es ein Metazeichen ist (z.B. bedeutet \$ – normalerweise zur Suche am Ende einer Zeile eingesetzt –, dass das Dollarzeichen als Suchbegriff verwendet wird).

**[***Zeichenklasse***]**
Ein beliebiges Zeichen aus einem Zeichensatz.

**[^***Zeichenklasse***]**
Ein beliebiges Zeichen, das nicht im Zeichensatz enthalten ist.

**[*a–z*]**
Ein beliebiges Zeichen aus dem angegebenen Bereich. Es können mehrere Bereiche und Listen von Zeichen in den Klammern angegeben werden.