

THE ADDISON-WESLEY MICROSOFT TECHNOLOGY SERIES



ESSENTIAL POWERSHELL

HOLGER
SCHWICHTENBERG

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States please contact:

International Sales
international@pearson.com

Visit us on the Web: www.informit.com/aw

Library of Congress Cataloging-in-Publication Data

Schwichtenberg, Holger.

Essential PowerShell / Holger Schwichtenberg.

p. cm.

ISBN 978-0-672-32966-1

1. Windows PowerShell (Computer programming language) 2. Command languages (Computer science) 3. Scripting languages (Computer science) 4. Systems programming (Computer science) 5. Microsoft Windows (Computer file) I. Title.

QA76.73.W56S39 2008

005.42—dc22

2008020010

Copyright © 2008 by Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, write to:

Pearson Education, Inc
Rights and Contracts Department
501 Boylston Street, Suite 900
Boston, MA 02116
Fax (617) 671 3447

ISBN-13: 978-0-672-32966-1

ISBN-10: 0-672-2966-2

Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana. First printing June 2008

This Book Is Safari Enabled

The Safari® Enabled icon on the cover of your favorite technology book means the book is available through Safari Bookshelf. When you buy this book, you get free access to the online edition for 45 days.

Safari Bookshelf is an electronic reference library that lets you easily search thousands of technical books, find code samples, download chapters, and access technical information whenever and wherever you need it.

To gain 45-day Safari Enabled access to this book:

- Go to <http://www.informit.com/onlineedition>
- Complete the brief registration form
- Enter the coupon code LJRM-BWZP-1HUA-KCPF-YA2S

If you have difficulty registering on Safari Bookshelf or accessing the online edition, please e-mail customer-service@safaribooksonline.com.

Editor-in-Chief

Karen Gettman

Executive Editor

Neil Rowe

Development Editor

Mark Renfrow

Managing Editor

Kristy Hart

Project Editor

Betsy Harris

Copy Editor

Keith Cline

Indexer

Publishing Works,
Inc.

Proofreader

Paula Lowell

Technical Editor

Tony Bradley

Publishing Coordinator

Cindy Teeters

Cover Designer

Gary Adair

Compositor

Nonie Ratchiff

PREFACE

Windows PowerShell is one of the most amazing products Microsoft has released in recent years, because it brings console-based system administration and scripting to the next level of abstraction. PowerShell is an excellent replacement for classic Windows shell commands and for Windows Script Host (WSH). PowerShell copies a lot of good features from UNIX shells and combines them with the power of the .NET Framework. In contrast to WSH, PowerShell enables consistent, straightforward, command-line system administration that does not require much software development knowledge.

Unfortunately, in the first version of PowerShell, the number of high-level commands is limited. For many tasks, lower-level concepts are required, especially the .NET Framework and Windows Management Instrumentation (WMI).

What Does This Book Cover?

This book covers the standard PowerShell commandlets, additional free commandlets (for example, PowerShell Community Extensions), and the direct use of classes from the .NET Framework, the Component Object Model (COM), WMI, and the Active Directory Service Interface (ADSI).

Because PowerShell is an extensive topic, this book cannot provide an exhaustive reference of all PowerShell commands and solutions for all possible administrative tasks. However, you will find a concise introduction to the most common command and scenarios. For more detailed information about PowerShell, refer to the Microsoft documentation for PowerShell, WMI, ADSI, and the .NET Framework (approximately 100,000 pages) as an additional source.

Who Should Read This Book?

The primary target audience comprises Windows administrators seeking a method of automated system administration that is more powerful than the classic Windows Shell but less complex than WSH and the associated COM components. After reading this book, administrators will be able to use PowerShell as their day-to-day command-line interface for all administrative tasks.

As a prerequisite, aside a good knowledge of the Windows operation system, you should have a basic understanding of object-oriented programming languages. Basic concepts of object orientation such as classes, objects, attributes, and methods are not explained in this book.

How This Book Is Structured

This book is organized into 24 chapters, some of which, based on your previous experience and knowledge of certain concepts, you might find easier to understand than others. The 24 chapters are split into two parts:

- **Part I: Getting Started with PowerShell.** Part I introduces the PowerShell architecture, all basic concepts (such as pipelining and navigation), the PowerShell Script Language, and the tools you should know.
- **Part II: Windows PowerShell in Action.** Part II covers PowerShell script solutions for day-to-day administrative tasks related to Windows services and Windows application, such as file system, processes, event logs, registry, networking, printers, documents, databases, Active Directory, and software installation. Each chapter contains dozens of self-contained examples.

The appendixes contain a list of all commandlets from PowerShell 1.0, the PowerShell Community Extensions 1.1.1, and the www.IT-Visions.de PowerShell Extensions 2.0. You will also find a short preview of the next version of Windows PowerShell (Version 2.0).

Throughout the text, you will find codes that match up to codes in Appendix C, “Bibliography.” These codes are encased in brackets (for example, [MS01]). The appendix lists the code, the correlating subject, and

a link that will provide you with more information.

Occasionally, when a line of code is too long to fit on one line in the printed text, a code-continuation character has been used to show that the line continues. For example

```
"{0} can be reached at {1}.  
↳This information is dated: {2:D}." -f $a, $b, $c
```

This Book's Website

Many of the scripts are available for download from its website, www.Windows-Scripting.com. This website also contains errata for this book and the option to offer feedback to the author.

FIRST STEPS WITH WINDOWS POWERSHELL

In this chapter:

What Is Windows PowerShell?	3
Downloading and Installing PowerShell Community Extensions	16
Testing the PowerShell Extensions	18
Downloading and Installing the PowerShellPlus	19
Testing the PowerShell Editor	20

This chapter introduces Windows PowerShell and helps you set up your environment. In addition, the chapter provides a few easy examples that demonstrate how to use PowerShell.

What Is Windows PowerShell?

Windows PowerShell (WPS) is a new .NET-based environment for console-based system administration and scripting on Windows platforms. It includes the following key features:

- A set of commands called *commandlets*
- Access to all system and application objects provided by Component Object Model (COM) libraries, the .NET Framework, and Windows Management Instrumentation (WMI)
- Robust interaction between commandlets through pipelining based on typed objects

- A common navigation paradigm for different hierarchical or flat information stores (for example, file system, registry, certificates, Active Directory, and environment variables)
- An easy-to-learn, but powerful scripting language with weak and strong variable typing
- A security model that prevents the execution of unwanted scripts
- Tracing and debugging capabilities
- The ability to host WPS in any application

This book includes syntax and examples for these features, except the last one, which is an advanced topic that requires in-depth knowledge of a .NET language such as C#, C++/CLI, or Visual Basic .NET.

A Little Bit of History

The DOS-like command-line window survived many Windows versions in almost unchanged form. With WPS, Microsoft now provides a successor that does not just compete with UNIX shells, it surpasses them in robustness and elegance. WPS could be called an adaptation of the concept of UNIX shells on Windows using the .NET Framework, with connections to WMI.

Active Scripting with Windows Script Host (WSH, pronounced “wish”) is much too complex for many administrators because it presupposes much knowledge about object-oriented programming and COM. The many exceptions and inconsistencies in COM make WSH and the associated component libraries hard to learn.

Even during the development of Windows Server 2003, Microsoft admitted that it had asked UNIX administrators how they administer their operating system. The short-term result was a large number of additional command-line tools included in Windows Server 2003. However, the long-term goal was to replace the DOS-like command-line window of Windows with a new, much more powerful shell.

Upon the release of the Microsoft .NET Framework in 2002, many people were expecting a “WSH.NET.” However, Microsoft stopped the development of a new WSH for the .NET Framework because it foresaw that using .NET-based programming languages such as C# and Visual Basic .NET would require administrators to know even more about object-oriented software development.

Microsoft recognized the popularity of and satisfaction with UNIX shells and decided to merge the pipelining concept of UNIX shells with the .NET Framework. The goal was to develop a new shell that was simple to use but nearly as robust as a .NET program. The result: WPS.

In the first beta version, the new shell was presented under the code name Monad at the Professional Developer Conference (PDC) in October 2003 in Los Angeles. After the intermediate names Microsoft Shell (MSH) and Microsoft Command Shell, the shell received its final name, PowerShell, in May 2006. The final version of WPS 1.0 was released on November 11, 2006 at TechEd Europe 2006.

NOTE The main architect of WPS 1.0 was Jeffrey Snover. He is always willing to discuss his “baby” and answer questions. At large international Microsoft technical conferences, such as the Professional Developer Conference (PDC) and TechEd, you can easily find him; he is the only person at the Microsoft booths wearing a tie.

Why Use WPS?

If you need a reason to use WPS, here it comes. Just consider the following solution for one common administrative task in both the *old* WSH and the *new* WPS.

An inventory script for software is to be provided that will read the installed MSI packages using WMI. The script will get the information from several computers and summarize the results in a CSV file (*softwareinventory.csv*). The names (or IP addresses) of the computers to be queried are read from a TXT file (*computers.txt*).

The solution with WSH (Listing 1.1) requires 90 lines of code (including comments and parameterizing). In WPS, you can do the same thing in just 13 lines (Listing 1.2). If you do not want to include comments and parameterizing, you need just one line (Listing 1.3).

Listing 1.1 Software Inventory Solution 1: WSH

```
Option Explicit

' --- Settings
Const InputFileName = "computers.txt"
Const OutputFileName = "softwareinventory.csv"
```

(continues)

Listing 1.1 Software Inventory Solution 1: WSH *(continued)*

```

Const Query = "SELECT * FROM Win32_Product where not
↳Vendor like '%Microsoft%'"

Dim objFSO                ' Filesystem Object
Dim objTX                 ' Textfile object
Dim i                     ' Counter
Dim Computer              ' Current Computer Name
Dim InputFilePath        ' Path for InputFile
Dim OutputFilePath       ' Path of OutputFile

' --- Create objects
Set objFSO = CreateObject("Scripting.FileSystemObject")

' --- Get paths
InputFilePath = GetCurrentPath & "\" & InputFileName
OutputFilePath = GetCurrentPath & "\" & OutputFileName

' --- Create headlines
Print          "Computer" & ";" & _
              "Name" & ";" & _
              "Description" & ";" & _
              "Identifying Number" & ";" & _
              "Install Date" & ";" & _
              "Install Directory" & ";" & _
              "State" & ";" & _
              "SKU Number" & ";" & _
              "Vendor" & ";" & _
              "Version"

' --- Read computer list
Set objTX = objFSO.OpenTextFile(InputFilePath)

' --- Loop over all computers
Do While Not objTX.AtEndOfStream
    Computer = objTX.ReadLine
    i = i + 1
    WScript.Echo "=== Computer #" & i & ": " & Computer
    GetInventory Computer
Loop

' --- Close Input File

```

```
objTX.Close

' === Get Software inventory for one computer
Sub GetInventory(Computer)

Dim objProducts
Dim objProduct
Dim objWMIService

' --- Access WMI
Set objWMIService = GetObject("winmgmts:" & _
    "{impersonationLevel=impersonate}!\" & Computer & _
    "\root\cimv2")
' --- Execute WQL query
Set objProducts = objWMIService.ExecQuery(Query)
' --- Loop
For Each objProduct In objProducts
    Print _
    Computer & ";" & _
    objProduct.Name & ";" & _
    objProduct.Description & ";" & _
    objProduct.IdentifyingNumber & ";" & _
    objProduct.InstallDate & ";" & _
    objProduct.InstallLocation & ";" & _
    objProduct.InstallState & ";" & _
    objProduct.SKUNumber & ";" & _
    objProduct.Vendor & ";" & _
    objProduct.Version
Next
End Sub

' === Print
Sub Print(s)
Dim objTextFile
Set objTextFile = objFSO.OpenTextFile(OutputFilePath, 8, True)
objTextFile.WriteLine s
objTextFile.Close
End Sub

' === Get Path to this script
Function GetCurrentPath
GetCurrentPath = objFSO.GetFile (WScript.ScriptFullName).ParentFolder
End Function
```

Listing 1.2 Software Inventory Solution 2: WPS Script

```
# Settings
$InputFileName = "computers.txt"
$OutputFileName = "softwareinventory.csv"
$query = "SELECT * FROM Win32_Product where not
↳Vendor like '%Microsoft%'"

# Read computer list
$Computers = Get-Content $InputFileName

# Loop over all computers and read WMI information
$Software = $Computers | foreach { get-wmiobject -query $Query -
computername $_ }

# Export to CSV
$Software | select Name, Description, IdentifyingNumber, InstallDate,
↳InstallLocation, InstallState, SKUNumber, Vendor, Version |
↳export-csv $OutputFileName -notypeinformation
```

Listing 1.3 Software Inventory Solution 3: WPS Pipeline Command

```
Get-Content "computers.txt" | Foreach {Get-WmiObject -computername
↳$_ -query "SELECT * FROM Win32_Product where not
↳Vendor like '%Microsoft%'" } | Export-Csv "Softwareinventory.csv"
↳-notypeinformation
```

Downloading and Installing WPS

Windows Server 2008 is the first operating system that includes WPS on the DVD. However, it is an additional feature that can be installed through Add Feature in the Windows Server 2008 Server Manager.

WPS can be downloaded (see Figure 1.1) and installed as an add-on to the following operating systems:

- Windows XP for x86 with Service Pack 2
- Windows XP for x64 with Service Pack 2
- Windows Server 2003 for x86 with Service Pack 1

- Windows Server 2003 for x64 with Service Pack 1
- Windows Server 2003 for Itanium with Service Pack 1
- Windows Vista for x86
- Windows Vista for x64

Note that WPS is not included in Windows Vista, although Vista and WPS were released on the same day. Microsoft decided not to ship any .NET-based applications with Vista. Only the .NET Framework itself is part of Vista.

POWERSHELL DOWNLOAD PAGE www.microsoft.com/windowsserver2003/technologies/management/powershell/download.aspx

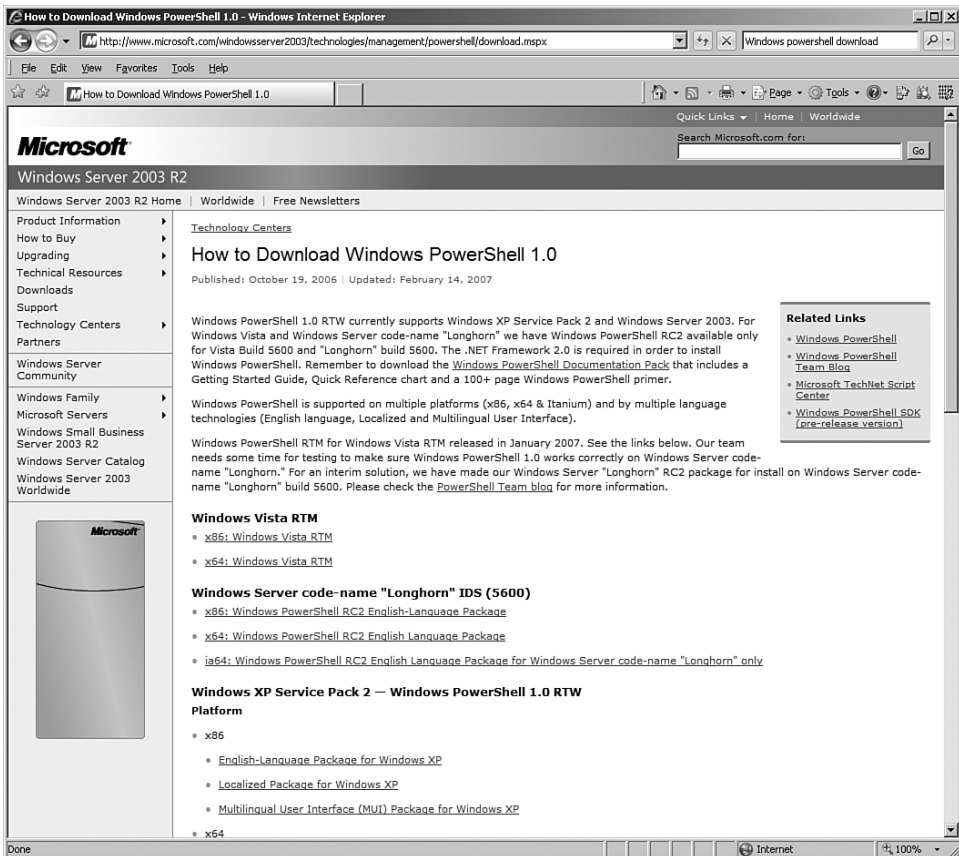


Figure 1.1 WPS download website

WPS requires that .NET Framework 2.0 or later be installed before running WPS setup. Because Vista ships with .NET Framework 3.0 (which is a true superset of 2.0), no .NET installation is required for it. However, on Windows XP and Windows Server, you must install .NET Framework 2.0, 3.0, or 3.5 first (if they are not already installed by another application).

MICROSOFT .NET FRAMEWORK 3.0 REDISTRIBUTABLE PACKAGE

www.microsoft.com/downloads/details.aspx?FamilyId=10CC340B-F857-4A14-83F5-25634C3BF043&displaylang=en

The setup routine installs WPS to the directory *%systemroot%\system32\WindowsPowerShell\V1.0* (on 32-bit systems) or *%systemroot%\Syswow64\WindowsPowerShell\V1.0* (for 64-bit systems). You cannot change this folder during setup.

TIP If for any reason you want to uninstall WPS, note that WPS is considered a software update to the Windows operating system (that is, not a normal application). Therefore, in the Add or Remove Programs control panel applet, it is not listed as a program; instead, it is listed as an update called Hotfix for Windows (KB x). The Knowledge Base (KB) number varies on different operating systems. However, you can identify WPS installation in the list by its icon (see Figure 1.2). On Windows XP and Windows Server 2003, you must check the Show Updates check box to see the WPS installation.

Taking WPS for a Test Run

This section includes some commands to enable you to try out a few WPS features. WPS has two modes, interactive mode and script mode, which are covered separately.

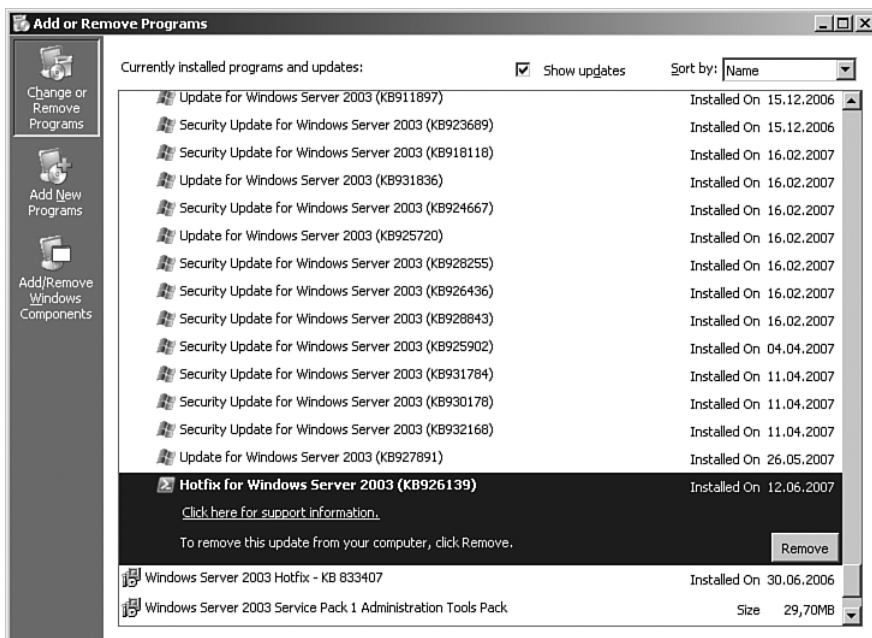


Figure 1.2 The uninstall option for WPS is difficult to find. (This screenshot is from Windows Server 2003.)

WPS in Interactive Mode

First, you'll use WPS in interactive mode.

Start WPS. An empty WPS console window will display (see Figure 1.3). At first glance, you might not see much difference between it and the traditional Windows console. However, there is much more power in WPS, as you will soon see.

At the command prompt, type **get-process** and then press the Return key. A list of all running processes on your local computer will display (see Figure 1.4). This was your first use of a simple WPS commandlet.

NOTE Note that the letter case does not matter. WPS does not distinguish between uppercase and lowercase letters in commandlet names.

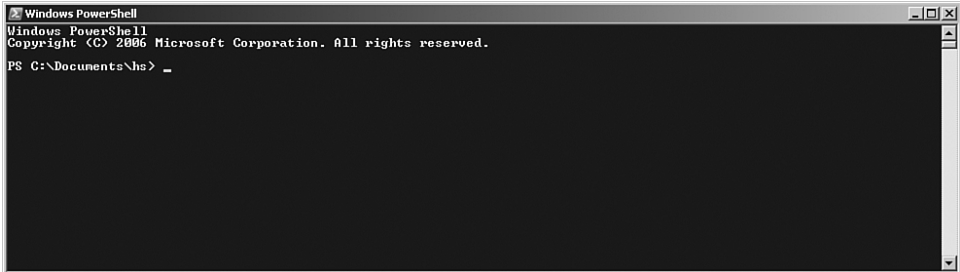


Figure 1.3 Empty WPS console window

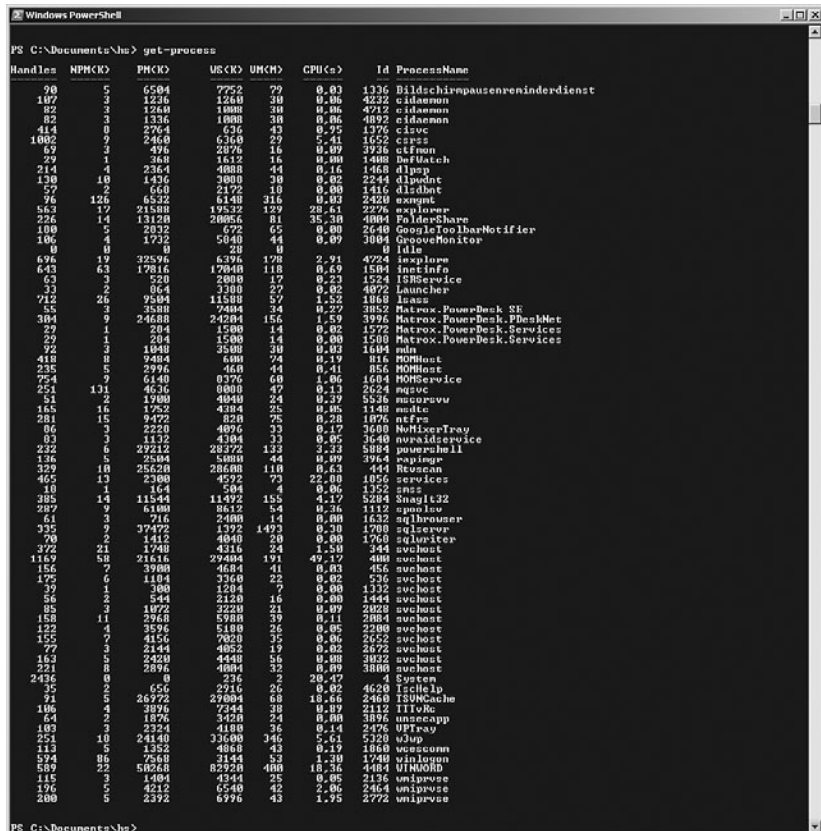
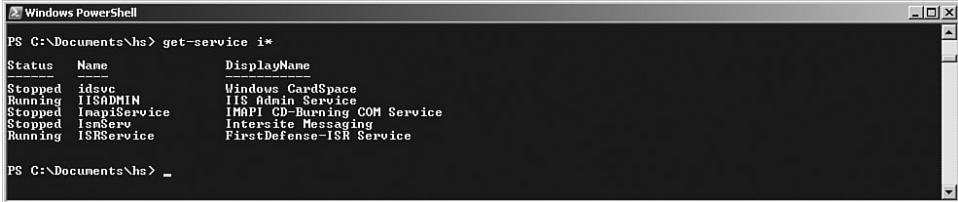


Figure 1.4 The Get-Process commandlet output

At the command prompt, type `get-service i*`. A list of all installed services with a name that begins with the letter *I* on your computer will

display (see Figure 1.5). This was your first use of a commandlet with parameters.



```

Windows PowerShell
PS C:\Documents\hs> get-service i*

```

Status	Name	DisplayName
Stopped	idsvcs	Windows CardSpace
Running	IISADMIN	IIS Admin Service
Stopped	InetService	Internet Publishing and WWW Service
Stopped	IISRServ	Internet Information Services (IIS) World Wide Web Publishing Service
Running	ISRSERVICE	Internet Information Services (IIS) Management Console (ISM) Service

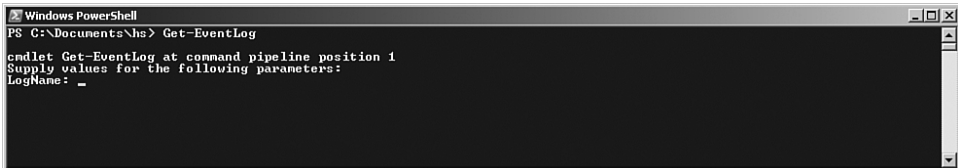
```

PS C:\Documents\hs> _

```

Figure 1.5 A filtered list of Windows services

Type **get-** and then press the Tab key several times. You will see WPS cycling through all commandlets that start with the verb *get*. Microsoft calls this feature *tab completion*. Stop at `Get-EventLog`. When you press Enter, WPS prompts for a parameter called `LogName` (see Figure 1.6). `LogName` is a required parameter. After typing **Application** and pressing Return, you will see a long list of the current entries in your Application event log.



```

Windows PowerShell
PS C:\Documents\hs> Get-EventLog
cmdlet Get-EventLog at command pipeline position 1
Supply values for the following parameters:
LogName: _

```

Figure 1.6 WPS prompts for a required parameter.

The last example in this section introduces you to the pipeline features of WPS. Again, we want to list entries from a Windows event log, but this time we want to get only some entries. The task is to get the most recent ten events that apply to printing. Enter the following command, which consists of three commandlets connected via pipes (see Figure 1.7):

```

Get-EventLog system | Where-Object { $_.source -eq "print" }
➔ | Select-Object -first 10

```

Note that WPS seems to get stuck for a few seconds after printing the first ten entries. This is the correct behavior because the first commandlet

(`Get-EventLog`) will receive all entries. The filtering is done by the subsequent commandlets (`Where-Object` and `Select-Object`). Unfortunately, `Get-EventLog` has no included filter mechanism.

```

Windows PowerShell
PS C:\Documents\hs>
PS C:\Documents\hs> Get-EventLog system | where-object { $_.source -eq "print" } | select-object -first 10

```

Index	Time	Type	Source	EventID	Message
...	27 Jun 12 19:02	Info	Print	10	Document 2, Microsoft Word - TY25_1.doc owned by hs was print...
...	48 Jun 12 14:24	Info	Print	10	Document 26, infas_angebot_A-27715088.pdf owned by hs was pri...
...	46 Jun 12 14:19	Info	Print	10	Document 25, http://reiseauskunft.bahn.de/bin/query.exe/dn?ld...
...	45 Jun 12 14:17	Info	Print	10	Document 24, http://reiseauskunft.bahn.de/bin/query.exe/dn?ld...
...	42 Jun 12 13:57	Info	Print	10	Document 23, http://reiseauskunft.bahn.de/bin/query.exe/dn?ld...
...	41 Jun 12 13:48	Info	Print	10	Document 22, Microsoft Office Outlook - Memoformat owned by h...
...	39 Jun 12 13:47	Info	Print	10	Document 21, Microsoft Office Outlook - Memoformat owned by h...
...	33 Jun 12 13:10	Info	Print	10	Document 20, Microsoft Word - 2422 Fachlichtbox.doc owned by ...
...	32 Jun 12 13:10	Info	Print	10	Document 19, Microsoft Office Outlook - Memoformat owned by h...
...	31 Jun 12 13:09	Info	Print	10	Document 18, Microsoft Office Outlook - Memoformat owned by h...

Figure 1.7 Filtering event log entries

WPS in Script Mode

Now it's time to try out PowerShell in script mode and incorporate a WPS script. A WPS script is a text file that includes commandlets/elements of PowerShell Script Language (PSL). The script in this example creates a new user account on your local computer.

Open Windows Notepad (or any other text editor) and enter the following lines of script code (which consists of comments, variable declarations, COM library calls, and shell output):

Listing 1.4 Create a User Account

```

### PowerShell Script
### Create local User Account

# Variables
$Name = "Dr. Holger Schwichtenberg"
$Accountname = "HolgerSchwichtenberg"
$Description = "Author of this book / Website: www.windows-scripting.com"
$Password = "secret+123"
$Computer = "localhost"

"Creating User on Computer $Computer"

```

```
# Access to Container using the COM library
↳ "Active Directory Service Interface (ADSI)"
$Container = [ADSI] "WinNT://$Computer"

# Create User
$objUser = $Container.Create("user", $Accountname)
$objUser.Put("Fullname", $Name)
$objUser.Put("Description", $Description)
# Set Password
$objUser.SetPassword($Password)
# Save Changes
$objUser.SetInfo()

"User created: $Name"
```

Save the text file with the name **createuser.ps1** into the directory *c:\temp*. Note that the file extension must be *.ps1*.

Now start WPS. Try to start the script by typing **c:\temp\createuser.ps1**. (You can use tab completion for the directory and file-names.) This attempt will fail because script execution is, by default, not allowed in WPS (see Figure 1.8). This is not a bug; it is a security feature. (Remember the Love Letter worm for WSH?)

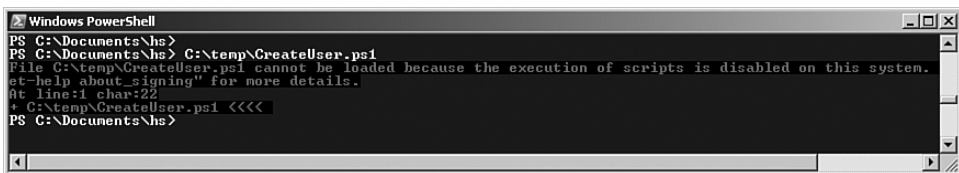


Figure 1.8 Script execution is prohibited by default.

For our first test, we will weaken the security a little bit (just a little). We will allow scripts that reside on your local system to run. However, scripts that come from network resources (including the Internet) will need a digital signature from a trusted script author. Later in this book you learn how to digitally sign WPS scripts. You also learn to restrict your system to scripts that you or your colleagues have signed.

To allow the script to run, enter the following:

```
Set-ExecutionPolicy remotesigned
```

Then, start the script again (see Figure 1.9). Now you should see a message that the user account has been created (see Figure 1.10).



```
Windows PowerShell
PS C:\Documents\hs>
PS C:\Documents\hs>
PS C:\Documents\hs> Set-ExecutionPolicy remotesigned
PS C:\Documents\hs> C:\temp\CreateUser.ps1
Creating User on Computer localhost
User created: Dr. Holger Schwichtenberg
PS C:\Documents\hs> _
```

Figure 1.9 Running your first script to create a user account

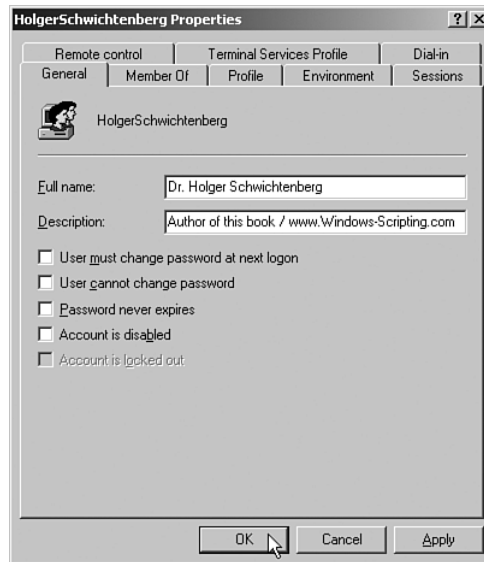


Figure 1.10 The newly created user account

Downloading and Installing PowerShell Community Extensions

WPS 1.0 includes only 129 commandlets. You might ask why I wrote *only*. You will notice soon that the most important commandlets are those with the verbs `get` and `set`. And the number of those commandlets is quite small compared to the large number of objects that Windows operating systems provide. All the other commandlets are, more or less, related to WPS infrastructure (for example, filtering, formatting, and exporting).

PowerShell Community Extensions (PSCX) is an open source project (see Figure 1.11) that provides additional functionality with commandlets such as `Get-DhcpServer`, `Get-DomainController`, `Get-MountPoint`, `Get-TerminalSession`, `Ping-Host`, `Write-GZip`, and many more. Microsoft leads this project, but any .NET software developer is invited to contribute. New versions are published on a regular basis. At the time of this writing, version 1.1.1 is the current stable release.

DOWNLOAD POWERSHELL COMMUNITY EXTENSIONS

www.codeplex.com/PowerShellCX

PSCX is provided as a setup routine that should be installed after WPS has been installed successfully.



Figure 1.11 PowerShell Community Extension website

You can incorporate additional functionality of PSCX into WPS by using a profile script (see Figure 1.12). Just copy this profile script to your *My Documents/Windows PowerShell* directory, if you want, during PSCX setup. As a beginner, you should use this option.

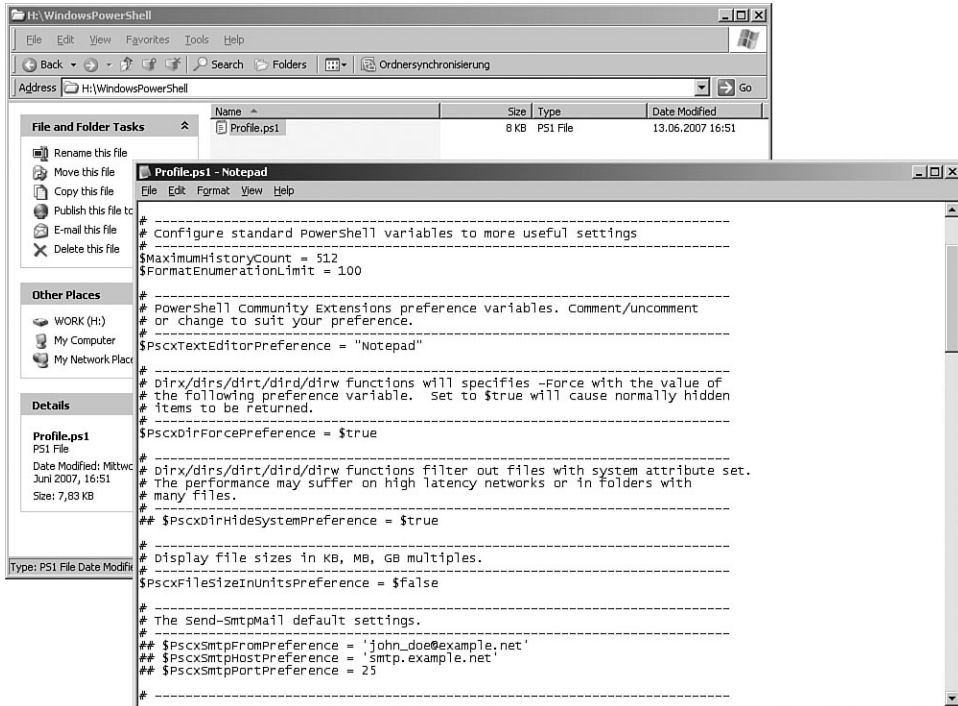



Figure 1.12 The PSCX profile script that was created during PSCX setup

Testing the PowerShell Extensions

The installation of PSCX changes the WPS console just a bit. Instead of the current path, the prompt now contains a counter. However, the path does display in the window's title.

Start WPS and type **Get-DomainController** (if your computer is a member of an Active Directory) or test PSCX by using **Ping-Host** with any computer on your network (see Figure 1.13).



```
ex PoSh C:\WINDOWS
Windows PowerShell
Copyright (C) 2006 Microsoft Corporation. All rights reserved.

1# Get-DomainController

ServerName      : E02
DnsHostName     : E02.IT-Visions.local
Site            : Default-First-Site
Domain          : IT-Visions.local
DN              : CN=E02.OU=Domain Controllers,DC=IT-Visions,DC=local
GlobalCatalog  : True

2# ping-host www.IT-Visions.de
Pinging www.it-visions.de with 32 bytes of data:
Reply from 195.234.228.60 bytes=32 time=16ms TTL=117
Reply from 195.234.228.60 bytes=32 time=15ms TTL=117
Reply from 195.234.228.60 bytes=32 time=16ms TTL=117
Reply from 195.234.228.60 bytes=32 time=15ms TTL=117

Ping statistics for www.it-visions.de:
    Packets: Sent = 4, Received = 4 (100% loss)
    Approximate round trip times: min = 15ms, max = 16ms, avg = 15ms

3#
```

Figure 1.13 Testing Get-DomainController and Ping-Host

Downloading and Installing the PowerShellPlus

Unfortunately, Microsoft does not provide a script editor for WPS yet. However, a few third-party editors support WPS (see Chapter 9, “PowerShell Tools”). Throughout this book, we use PowerShellPlus Editor, which is free for noncommercial use.

A previous editor called PowerShell IDE from the same author was free even for commercial use. However, PowerShell IDE never made it to a final release and was discontinued.

The PowerShellPlus Editor is part of PowerShellPlus. PowerShellPlus consists of the editor and a console that provides IntelliSense while using the PowerShell interactively.

POWERSHELLPLUS WEBSITE www.powershell.com

PowerShellPlus does not need any setup. It is a true .NET application with XCopy deployment. You just unpack the ZIP file to the directory of your choice and start the PowerShellPlus.exe that is part of the package.

Testing the PowerShell Editor

The PowerShellPlus has, according to the WPS console, two modes: an interactive mode and a script mode (see Figure 1.14). After starting the PowerShellPlus, you will see the interactive mode. You can use any commandlet (or pipeline). When you press Return, the commandlet is executed, and the result displays in the same window. The handy feature is the IntelliSense. If you enter **Get-P**, you will see a drop-down list of the available commandlets that start with these letters.

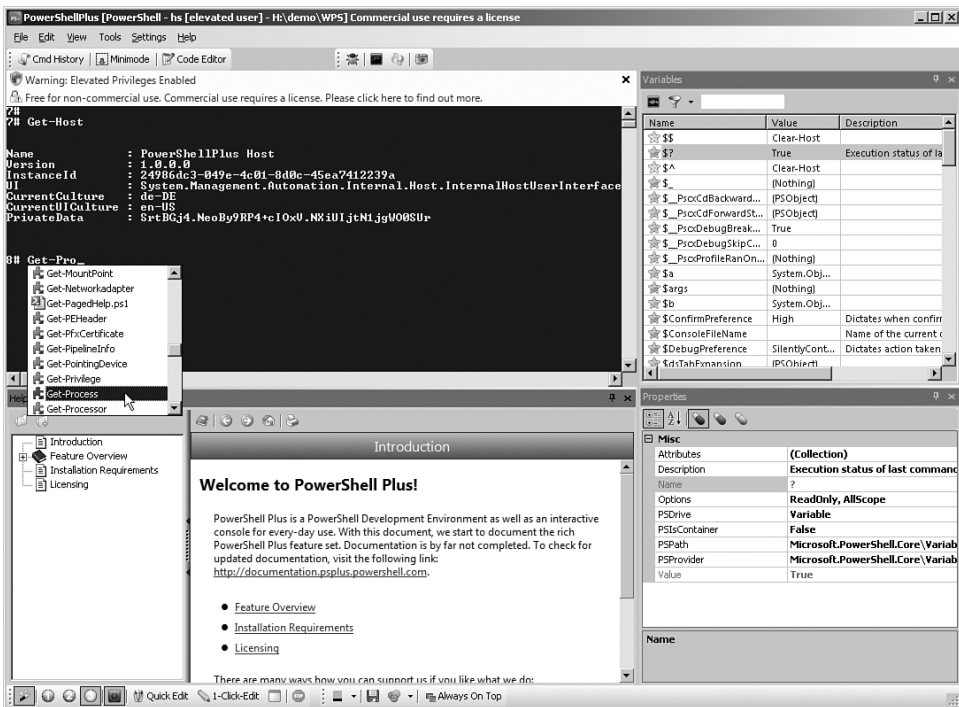


Figure 1.14 WPS IDE in interactive mode

To use the PowerShellPlus in script mode, click Code Editor and create a new script file (New/PowerShell Script) or open an existing script PS1 file (Open). Now open the script file `CreateUser.ps1` that you created earlier. You will see line numbers, and you will encounter the same IntelliSense features that you have in interactive mode. To run the script,

click the Run symbol in the toolbar (see Figure 1.15). The result will display in the interactive Windows in the background.

WARNING Make sure the user account does not exist before running the script. Otherwise the script will fail with the error “The account already exists.”

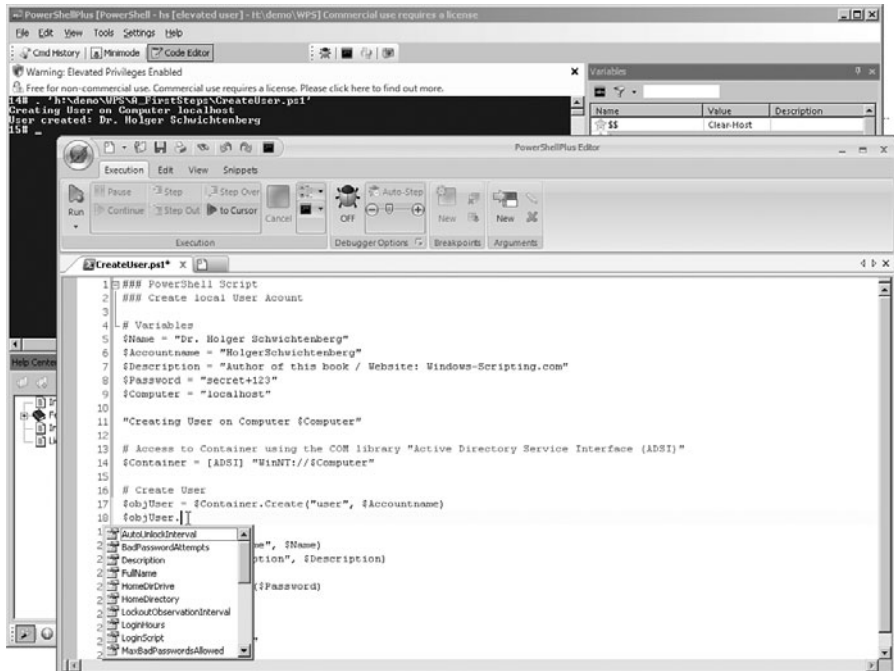


Figure 1.15 WPS IDE in script mode

Another great feature is debugging. Place the cursor on any line in your script and click the Debugging icon. Next, go to any line and press F9. This creates a red circle next to that line, called a *breakpoint*. Now run the script. You will see the PowerShellPlus Editor executing the script in slow motion, marking the current line yellow and stopping at the line with the breakpoint (see Figure 1.16). In the Variables Inspector window, you can inspect the current value of all variables. In the interactive window, you can type any WPS command that will be executed within the current context. That is, you can interactively access all script variables. To continue the script, press F8 or click the Continue icon in the toolbar.

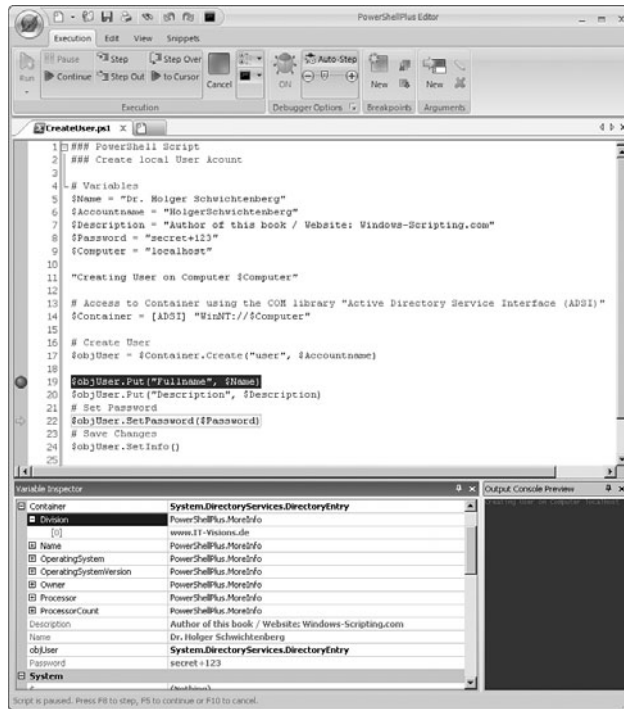


Figure 1.16 Script debugging with the WPS IDE

Code snippets are also a nice feature of the PowerShellPlus. In a script file, click Snippet/Insert on the toolbar or select Insert Snippet in the context menu in the main Editor window. You will be able to select a snippet. You can create your own snippets with the PowerShellPlus (via Snippets/New on the toolbar).

Summary

Windows PowerShell is a new .NET-based environment for scripting and is an interactive command-line shell. WPS is an optional feature on Windows Server 2008 and an add-on for Windows XP, Vista, and Server 2008. Commands in WPS are called commandlets. The PSCX extends WPS with additional commandlets.

The PowerShellPlus is an alternative shell for WPS commands and an editor for WPS scripts.

In the next chapter, you learn much more about commandlets and pipelines. You also learn how to get help if you are seeking a command or the available options for a commandlet.

INDEX

Symbols

& (ampersand) operator, 109

@ (at symbol) in hash tables, 107

= (equals sign), 109

() (parentheses) in methods, 64

+ (plus sign operator), 54, 108

"" (quotation marks) in parameters, 26

;(semicolons) in commands, 90

* (star operator), 108, 356

| (vertical line) for pipelines, 43

A

access control entries.

See ACEs

access control lists.

See ACLs

access rights, 403-406

accessing

databases

commands, 383-385

connections, 380-382

data readers, 386-388

DataSets. *See*

DataSets

provider-independent, 382-383

www.IT-Visions.de

extensions, 396-399

directory services, 313

file shares, 221

hash tables, 107

WMI

collections, 146

members, 142-144

objects, 137-138

ACEs (access control entries), 225, 402

adding to ACLs, 418-419

contents, 402

deleting from ACLs, 421-423

reading, 410-411

ACLs (access control lists), 225, 401-402

ACEs

adding, 418-419

contents, 402

deleting, 421-423

reading, 410-411

classes, 406

control holders, 408

inheritance

hierarchy, 406

ObjectSecurity, 406

reading ACLs, 408-409

resources, 407

commandlets, 401

configuring, 425-426

reading, 408-409

transferring, 424

Active Directory

extensions

PSCX, 361

Quest, 365

www.IT-Visions.de, 362-364

group members

assignments, 345

creating/filling, 345

deleting, 346

listing, 343-344

organizational units, 346-347

schema

documentation, 338

website, 450

searching, 314

indexed attributes, 354

multivalued attributes, 355-356

result restrictions, 357

star operator, 356

structure, 365-367

user accounts

authentication, 341

creating, 339-340

deleting, 342

moving, 343

- passwords, 340
 - renaming, 342
 - user class, attributes, 335-338
 - Active Directory Service Interfaces. *See* ADSI
 - Active Directory Management Objects (ADMO), 365
 - AD Access Change/Break in RC2 website, 449
 - ADAM (Active Directory Application Mode), 365
 - add-content commandlet, 429
 - binary files, 238
 - text files, writing, 236
 - add-directoryentry commandlet, 362, 429
 - add-history commandlet, 429
 - add-member commandlet, 429
 - Add-PSSnapin commandlet, 175-176, 429
 - add-user commandlet, 362, 429
 - adding
 - ACEs to ACLs, 418-419
 - snap-ins, 175
 - users to groups, 345
 - virtual web servers, 308-311
 - AddPrinterConnection() method (Win32_Printer class), 287
 - ADMO (Active Directory Management Objects), 365
 - ADO.NET, 373
 - architecture, 374
 - data providers, 375
 - data source control elements, 377
 - DataReader object, 376-378
 - DataSet object, 376-378
 - SQL Servers, listing available, 376
 - ADSI (Active Directory Service Interfaces), 314
 - architecture, 316
 - deficiencies, 321-323
 - directory services, compared, 320
 - DirectoryEntry class, 318-319
 - integration, 316
 - object model, 318
 - property cache, 329
 - search queries, 319
 - aliases, 29
 - creating, 30-31
 - enumerating, 29
 - properties, 68
 - ambiguous commandlets, 180
 - ampersand (&) operator, 109
 - Analyzer, 164
 - analyzing pipeline content, 59
 - alias properties, 68
 - code properties, 68
 - ETS, 69-70
 - get-member commandlet, 62, 66-69
 - get-pipelineinfo commandlet, 60
 - methods, 64
 - note properties, 67
 - properties, 65
 - property sets, 66
 - script properties, 67
 - AppendChild() method, 246
 - AppendData right, 403
 - architecture
 - Active Directory, 365-367
 - ADO.NET, 374
 - ADSI, 316
 - arrays, 105-106
 - associative, 106-108
 - declaring, 105
 - defining, 105
 - joining, 105
 - listing, 105
 - multidimensional, 106
 - at symbol (@) in hash tables, 107
 - attributes, 213
 - directory entries
 - reading, 328
 - writing, 329
 - FileSystemAccessRule objects, 410
 - indexed, 354
 - mailboxes, 304
 - multivalued, 355-356
 - Property, 318
 - services, 278
 - user class (Active Directory), 335-338
 - authentication, 58, 341
 - autostart applications, 263
- B–C**
- binary files, 238
 - BIOS settings, 282
 - boot configuration settings, 282

- calculated parameters, 27-28
- calculations (pipelines), 76
- calling methods, 64
- castrating objects, 73-74
- Change() method
 - (Win32_Service class), 278
- ChangePermission
 - right, 403
- checking XML files, 242-243
- classes
 - attributes, 213
 - CmdletInfo, 179
 - COM
 - COM objects, 135
 - creating instances, 133
 - existing instances, 134
 - DateTime, 102
 - DbProviderFactories, 382
 - DirectoryEntries, 319
 - DirectoryEntry, 318-319
 - DriveInfo, 208
 - FileInfo, 214
 - group policies, 367
 - Hashtable, 107
 - IIsApplicationPool, 305
 - IIsComputer, 305
 - IIsWebServer, 305
 - IIsWebService, 305
 - IIsWebVirtualDir, 305
 - MailMessage, 302
 - ManagementDateTime-
 - Converter, 145
 - .NET, 129
 - assemblies, loading, 131
 - constructor
 - parameters, 130
 - enumerations, 132
 - help, 38-40
 - instances, creating, 130
 - object analysis, 132
 - static members, 130
 - ObjectSecurity, 406
 - security, 406
 - control holders, 408
 - inheritance
 - hierarchy, 406
 - ObjectSecurity, 406
 - reading ACLs, 408-409
 - resources, 407
 - SmtpClient, 302
 - String, 99
 - TimeSpan, 103
 - user (Active Directory), 335-338
 - WebClient, 300
 - Win32_Computersystem, 281
 - Win32_Desktop, 315
 - Win32_LogicalDisk, 207-210
 - Win32_NetworkAdapter
 - Configuration, 296
 - Win32_NTLogEvent, 291
 - Win32_Operating
 - System, 281
 - Win32_PerfRawData, 292
 - Win32_Product, 259
 - Win32_Service, 277
 - Win32_Share, 221
 - Win32_StartupCommand, 263
 - Win32_Trustee, 226
 - WMI, 135
 - available, listing, 148
 - collections,
 - accessing, 146
 - instances, creating, 149
 - object access, 137-138
 - object adapter, 139
 - object analysis, 140
 - object filtering/
 - selecting, 146-147
 - properties/methods, 142-144
 - queries, 147
 - static class
 - members, 144
 - System.Management
 - object model, 135
 - type indicators, 139
 - WPS support, 136
 - XMLDocument, 229, 244
- clear-content commandlet, 429
- clear-item commandlet, 206, 429
- clear-itemproperty
 - commandlet, 429
- clear-variable commandlet, 429
- clipboard, 200
- close-dbconnection
 - commandlet, 430
- cmdlet development
 - guidelines
 - website, 450
- Cmdlet help website, 450
- CmdletInfo class, 179
- code properties, 68
- COM classes
 - COM objects, 135
 - instances
 - creating, 133
 - existing, 134
- command mode, 33, 154
- command-processing
 - modes, 33
- commandlets
 - add-content, 429
 - binary files, 238
 - text files, writing, 236
 - add-directoryentry, 362, 429

- add-history, 429
- add-member, 429
- add-pssnapin, 175, 429
- add-user, 362, 429
- ambiguous, 180
- case sensitivity, 29
- clear-content, 429
- clear-item, 206, 429
- clear-itemproperty, 429
- clear-variable, 429
- close-dbconnection, 430
- compare-object, 78, 430
- convert-html, 251
- convert-path, 430
- convert-xml, 249, 430
- convertfrom-base64, 430
- convertfrom-secure-string, 430
- convertto-base64, 430
- convertto-html, 430
- convertto-macos9line-ending, 430
- convertto-securestring, 430
- convertto-unixline-ending, 430
- convertto-windowsline-ending, 430
- copy-item, 212, 254, 430
- copy-itemproperty, 430
- data access, 396
- debugging parameters, 171
- definition, 25
- disable-mailbox, 304
- disconnect-terminal-session, 430
- Exchange Server 2007, 184-185
- export-alias, 431
- export-bitmap, 431
- export-clixml, 248, 431
- export-console, 431
- export-csv, 239, 431
- expression integration, 33
- extensions, 174-175, 181
- external, 33-34
- file system administration, 205-206
- foreach-object, 105, 235, 431
- format-byte, 431
- format-custom, 431
- format-hex, 431
- format-list, 431
- format-table, 431
- format-wide, 431
- format-xml, 244, 431
- get-, 35
- get-acl, 401, 431
- get-adobject, 314, 358, 431
- get-alias, 30, 432
- get-authenticodesignature, 432
- get-bios, 432
- get-cdromdrive, 432
- get-childitem, 432
 - directory content, 210
 - Filter parameter, 211
 - include parameter, 211
 - registry keys, 253
- get-clipboard, 200, 432
- get-command, 432
- get-computerinfo, 432
- get-computername, 432
- get-content, 206, 432
 - binary files, 238
 - files, reading, 229, 235
- get-credential, 58, 432
- get-culture, 188, 432
- get-currentuser, 432
- get-datarow, 396
- get-datatable, 396
- get-date, 102, 432
- get-dbconnection, 432
- get-dbrow, 433
- get-dbtable, 433
- get-dhcpserver, 433
- get-directory, 362
- get-directorychildren, 433
- get-directoryentry, 362, 433
- get-directoryvalue, 362, 433
- get-disk, 206, 433
- get-domaincontroller, 324, 433
- get-eventlog, 290, 433
- get-executionpolicy, 433
- get-exportedtype, 433
- get-fileversioninfo, 433
- get-foregroundwindow, 434
- get-hash, 434
- get-help, 35, 434
- get-history, 186, 434
- get-host, 187, 434
- get-item, 434
 - file properties, 213
 - registry keys, 254
- get-itemproperty, 255, 434
- get-ITVisions, 434
- get-keyboard, 434
- get-location, 206, 434
- get-mailbox, 303
- get-mailboxdatabase, 303
- get-member, 62, 66-68, 434
 - alias properties, 68
 - code properties, 68
 - methods, 64
 - note properties, 67
 - output, reducing, 69
 - properties, 65
 - property sets, 66
 - script properties, 67
- get-memorydevice, 434

- get-metadata, 435
- get-mountpoint, 435
- get-networkadapter, 435
- get-PEheader, 435
- get-pfxcertificate, 435
- get-pipelineinfo, 60, 435
- get-pointingdevice, 435
- get-privilege, 435
- get-process, 11, 435
 - processes, enumerating, 267-268
 - processes, filtering, 268
- get-process | out file, 55
- get-process | out-printer, 55
- get-processor, 435
- get-psdrive, 83, 206, 435
- get-psprovider, 84, 435
- get-pssnapin, 435
- get-pssnapinhelp, 435
- get-random, 435
- get-reparsepoint, 435
- get-service, 272, 435
- get-service i, 13
- get-shortpath, 436
- get-sounddevice, 436
- get-storagegroup, 303
- get-tabexpansion, 436
- get-tapedrive, 436
- get-terminalsession, 436
- get-tracesource, 173, 436
- get-uiculture, 188, 436
- get-unique, 436
- get-usbcontroller, 436
- get-variable, 436
- get-videocontroller, 436
- get-wmiobject, 135,
 - 144, 436
 - hardware information, 284
 - list parameter, 148
- group-object, 74, 436
- help, 35, 38
- import-alias, 436
- import-bitmap, 436
- import-clixml, 436
- import-csv, 240, 436
- import-dbcommand, 437
- invoke-dbcommand, 396
- invoke-expression,
 - 109, 437
- invoke-history, 437
- invoke-item, 437
- invoke-scalardb-command, 437
- join-path, 437
- join-string, 102, 437
- listing of, 35
- measure-command,
 - 173, 437
- measure-object, 76, 437
- move-item, 206,
 - 212, 437
- move-itemproperty, 437
- navigation, 84
- new-alias, 30, 437
- new-hardlink, 218, 437
- new-item, 206, 437
 - registry keys, 254
 - text files, creating, 236
- new-itemproperty, 256, 437
- new-junction, 218, 437
- new-mailboxdatabase, 303
- new-object, 437
- new-psdrive, 438
- new-service, 278, 438
- new-shortcut, 217, 438
- new-storagegroup, 303
- new-symlink, 220, 438
- new-timespan, 103, 438
- new-variable, 438
- nouns, 29
- out-clipboard, 438
- out-default, 51, 438
- out-file, 55, 236, 438
- out-host, 51, 438
- out-null, 438
- out-printer, 55, 287, 438
- out-string, 438
- output, 49
 - printing, 55
 - single values, 53-54
 - standard, 51-53
 - suppressing, 55
 - text files, 55
- parameters, 26-27
 - calculated, 27-28
 - case sensitivity, 29
 - filtering output, 28
 - placeholders, 29
 - quotation marks, 26
 - sequence, 27
- ping-host, 296, 438
- pipelines
 - calculations, 76
 - castrating objects, 73-74
 - classic commands, 46
 - comparing objects, 78
 - content, analyzing. *See* pipelines, content analyzing
 - creating, 43
 - filtering objects, 70-72
 - grouping objects, 74-75
 - intermediate steps, viewing, 76
 - objects, 44-46
 - output, 49-55
 - Pipeline Processor, 47-49
 - ramifications, 78
 - sorting objects, 74
 - user input, 56-58
- placeholders, 29
- pop-location, 438

- PSCX, 181-182, 214
- push-location, 438
- Quest, 183-184
- read-host, 56, 438
- remove-directoryentry, 362, 438
- remove-item, 206, 212, 254, 438
- remove-itemproperty, 257, 438
- remove-mountpoint, 439
- remove-psdrive, 439
- remove-pssnapin, 439
- remove-reparsepoint, 439
- remove-variable, 439
- rename-item, 206, 212, 439
- rename-itemproperty, 439
- resize-bitmap, 439
- resolve-assembly, 215, 439
- resolve-host, 299, 439
- resolve-path, 439
- restart-service, 277, 439
- resume-service, 439
- SCVMM, 185
- select-object, 70, 73, 439
- select-string, 237, 439
- select-xml, 244-246, 439
- send-smtpmail, 302, 439
- set-acl, 401, 440
- set-alias, 30, 440
- set-authenticodesignature, 120, 440
- set-clipboard, 200, 440
- set-content, 206, 440
 - binary files, 238
 - text files, writing, 236
- set-datarow, 396
- set-datatable, 396
- set-date, 104, 440
- set-dbtable, 440
- set-directoryvalue, 362, 440
- set-distributiongroup, 304
- set-executionpolicy, 119, 440
- set-filetime, 214, 440
- set-foregroundwindow, 440
- set-item, 206, 440
- set-itemproperty, 214, 440
- set-location, 206, 254, 441
- set-privilege, 441
- set-psdebug, 173, 441
- set-service, 278, 441
- set-tracesource, 173, 441
- set-variable, 441
- set-volumelabel, 210, 441
- snap-ins, 179
- sort-object, 74, 441
- split-path, 441
- split-string, 101, 441
- start-process, 269-270, 441
- start-service, 277, 441
- start-sleep, 122, 441
- start-tabexpansion, 441
- start-transcript, 441
- stop-process, 270, 441
- stop-service, 277, 441
- stop-terminalsession, 441
- stop-transcript, 441
- suspend-service, 442
- syntax, 26
- test-assembly, 442
- test-dbconnection, 396, 442
- test-path, 442
- test-xml, 243, 442
- trace-command, 442
- tree-object, 78, 442
- update-formatdata, 442
- update-typedata, 442
- verbose parameter, 172
- where-object, 70, 442
- write-bzip2, 442
- write-clipboard, 200, 442
- write-debug, 442
- write-error, 53, 442
- write-gzip, 442
- write-host, 53, 442
- write-output, 443
- write-progress, 443
- write-tar, 443
- write-verbose, 443
- write-warn, 53
- write-warning, 443
- write-zip, 220, 443
- commands
 - database access, 383-385
 - history, 186-187
 - separating, 90
- comments, 90
- CommitChanges()
 - method, 329
- compare-object
 - commandlet, 78, 430
- comparing objects, 78
- complex pipelines, 48-49
- compression (files), 220-221
- computers
 - BIOS, 282
 - boot configurations, 282
 - event logs, 290
 - entries, 290-291
 - names, 290
 - remote access, 291

- hardware
 - information, viewing, 284-285
 - printers, 286-289
- performance counters, 292-293
- pinging, 295
- product aviation
 - settings, 282
- recovery settings, 283
- serial numbers, 282
- settings, viewing, 281-283
- software versions, 282
- configuring
 - ACLs, 425-426
 - date and time, 104
 - files
 - date and time, 214
 - share permissions, 225-228
 - networking, 296-298
- confirm parameter, 171
- connections
 - databases, 380-382
 - printers, 287
- consoles
 - interactive mode, 11
- WPS, 151
 - command history, 186-187
 - command mode, 154
 - functions, 152
 - interpreter mode, 154
 - PowerTab, 156
 - snap-ins, loading, 175-176
 - tab completion, 153
 - Vista user account control, 155
- constant values
 - (variables), 95
- constructors (.NET classes), 130
- control structures, 110-112
- convert-html commandlet, 251
- convert-path commandlet, 430
- convert-xml commandlet, 249, 430
- convertfrom-base64 commandlet, 430
- convertfrom-securestring commandlet, 430
- convertto-base64 commandlet, 430
- convertto-html commandlet, 430
- convertto-macos9lineending commandlet, 430
- convertto-securestring commandlet, 430
- convertto-unixlineending commandlet, 430
- convertto-windowslineending commandlet, 430
- ConvertToDateTime() method, 145
- copy-item commandlet, 212, 254, 430
- copy-itemproperty commandlet, 430
- copying
 - files/folders, 212
 - registry keys, 254
- CreateDirectories right, 403
- CreateElement() method, 246
- CreateFiles right, 404
- creating
 - CSV files, 239
 - directory entries, 332
 - Explorer links, 217
 - file shares, 223-224, 229-232
 - groups
 - Active Directory, 345
 - policy links, 369-370
 - hardlinks, 218
 - junction points, 218
 - mailboxes, 303
 - organizational units, 346-347
 - public folders, 305
 - registry keys, 254-257
 - symbolic links, 220
 - user accounts, 339-340
 - websites from CSV files, 309-311
- CSV files, 239
 - creating, 239
 - exporting, 239
 - importing, 240
 - websites, creating, 309-311
- customizing
 - file properties, 214
 - service configuration, 278-279
 - strings, 100
 - XML documents, 246
- D**
- data
 - adapters, 391
 - providers, 375
 - readers, 386-388
 - types, 92
 - listing of, 92
 - registry, 257
 - variables, 91-93

- databases
 - access
 - commands, 383-385
 - connections, 380-382
 - data readers, 386-388
 - DataSets. *See* DataSets
 - provider-independent, 382-383
 - www.IT-Visions.de extensions, 396-399
 - ADO.NET, 373
 - architecture, 374
 - data providers, 375
 - data source control elements, 377
 - DataReader object, 376-378
 - DataSet object, 376-378
 - enumerating data providers, 375
 - SQL Servers, listing available, 376
 - example, 379
 - mailboxes, 303
 - DataReader object, 376-378
 - DataSets, 389
 - data adapter, 391
 - object model, 376-378, 390
 - provider-independent example, 394-395
 - provider-specific example, 391-393
 - XML exports/imports, 395
 - DataTable objects, 390
 - date and time, 102-103, 145
 - files, 214
 - periods of time, 103
 - remote computers, 104
 - setting, 104
 - WMI date format conversions, 145
 - DateTime class, 102
 - DbProviderFactories class, 382
 - deactivating mailboxes, 304
 - debug parameter, 171
 - debugging
 - commandlet parameters for, 171
 - PowerShellPlus, 21
 - PowerShellPlus Editor, 163
 - step-by-step, 173
 - verbose parameter, 172
 - declaring
 - arrays, 105
 - variables, 91
 - default naming
 - context, 324
 - Delete right, 404
 - DeleteSubdirectoriesAndFiles right, 404
 - DeleteTree() method, 342
 - deleting
 - ACEs to ACLs, 421-423
 - directory entries, 332
 - files/folders, 212
 - group policy links, 370-372
 - junction points, 219
 - print jobs, 288
 - registry keys, 254, 257
 - text file content, 236
 - users
 - Active Directory, 342
 - groups, 346
 - virtual web servers, 311
 - dependent services, 274-276
 - dialog boxes
 - authentication, 58
 - user input, 57
 - digital signatures, 120-121
 - directory content
 - files/folders operations, 212-213
 - viewing, 210-212
 - directory services
 - access, 313
 - ADSI
 - compared, 320
 - deficiencies, 321-323
 - paths, 323-325
 - programming, 325
 - ADSI property cache, 329
 - binding meta objects to directory entries, 325-326
 - container objects, 331
 - directory entries, 332
 - directory entry attributes, 328-329
 - directory entry existence, checking, 327
 - impersonation, 327
 - object properties, 330
 - www.IT-Visions.de commandlets, 362-364
 - DirectoryEntries class, 319
 - DirectoryEntry class, 318-319
 - disable-mailbox
 - commandlet, 304
 - disconnect-terminalsession
 - commandlet, 430
 - DLL registration, 175
 - DNs (distinguished names), 323

- documents
 - binary files, 238
 - CSV files, 239
 - creating, 239
 - exporting, 239
 - importing, 240
 - HTML, 251
 - text files
 - content, deleting, 236
 - reading, 235-236
 - searching, 237
 - writing to, 236-237
- XML, 241
 - checking, 242-243
 - converting to XHTML files, 249
 - customizing, 246
 - formatting, 244
 - object pipeline, 248
 - reading, 241
 - searching with XPath, 244
- domain controllers (Active Directory), 366-367
- domains (Active Directory), 366
- dot sourcing, 118
- downloading
 - PSCX, 17
 - RSS feeds, 301
 - WPS, 8
- DownloadString()
 - method, 300
- DriveInfo class, 208
- drives
 - defining, 87-88, 255
 - free space, viewing, 208-210
 - listing all, 206-207
 - names, 210
 - network, 210
 - providers, 83-84
- E**
 - e-mail, sending, 302
 - ending processes, 270
 - enumerating
 - aliases, 29
 - data providers, 375
 - file shares, 223
 - group policies, 367-369
 - .NET classes, 132
 - processes, 267-268
 - services, 272-273
 - environment variables,
 - viewing, 283
 - equals sign (=), 109
 - ErrorAction parameter, 125-127
 - errors (scripts), 122
 - creating, 128
 - handling, 125-127
 - history, 128
 - standard reactions, 127
 - trap blocks, 128
 - trapping example, 123-125
 - ETS (Extended Type System), 44
 - pipeline content,
 - analyzing, 69-70
 - website, 450
 - event logs, 290
 - entries, 290-291
 - filtering, 14
 - names, 290
 - remote access, 291
 - Exchange Management Shell website, 451
 - Exchange Server 2007, 302
 - basic operations, 302
 - databases, listing, 303
 - functionality, testing, 303
 - mailboxes, 303-304
 - management shell,
 - 184-185
 - public folder
 - management, 305
 - scripts website, 451
 - storage groups, 303
 - executable files
 - PE header information, 215
 - PSCX commandlets, 214
 - viewing, 215
 - ExecuteFile right, 404
 - execution policies, 119
 - execution time,
 - measuring, 173
 - Exists() method, 327
 - Explorer links, 216-217
 - export-alias commandlet, 431
 - export-bitmap
 - commandlet, 431
 - export-clicxml commandlet, 248, 431
 - export-console
 - commandlet, 431
 - export-csv commandlet, 239, 431
 - exporting
 - CSV files, 239
 - DataSets, 395
 - expressions, 32-33
 - Extended Reflection, 44
 - Extended Type System.
 - See* ETS
 - extensions
 - commandlets,
 - 174-175, 181
 - PSCX
 - Active Directory, 361
 - commandlets, 181-182
 - LDAP filters, 358

Quest, 365
www.IT-Visions.de, 183
 Active Directory,
 362-364
 database access,
 396-399
external commandlets,
 33-34

F

file system administration
 access rights, 403-406
 commandlets, 205-206
 directory content,
 viewing, 210-212
 drives
 free space, displaying,
 208-210
 listing all, 206-207
 names, 210
 network, 210
 executable files
 PE header
 information, 215
 PSCX commandlets,
 214
 viewing, 215
file compression,
 220-221
file properties
 customizing, 214
 date/time information,
 214
 viewing, 213
file shares
 accessing, 221
 creating, 223-224
 enumerating, 223
 mass creation, 229-232
 permissions, 225-228
files/folders operations,
 212-213

links, 216
 Explorer, 216-217
 hardlinks, 217-218
 junction points,
 218-219
 symbolic, 220
FileInfo class, 214
files
 binary, 238
 compression, 220-221
 copying, 212
 CSV, 239
 creating, 239
 exporting, 239
 importing, 240
 websites, creating,
 309-311
 deleting, 212
 executable
 PE header
 information, 215
 PSCX commandlets,
 214
 viewing, 215
HTML, 251
moving, 212
names, 34
properties
 customizing, 214
 date/time information,
 214
 viewing, 213
renaming, 212
retrieving from HTTP
 servers, 300-301
shares
 accessing, 221
 creating, 223-224
 enumerating, 223
 mass creation, 229-232
 permissions, 225-228

text
 content, deleting, 236
 reading, 235-236
 searching, 237
 writing to, 236-237
XML, 241
 checking, 242-243
 converting to XHTML
 files, 249
 customizing, 246
 DataSet
 exports/imports, 395
 formatting, 244
 object pipeline, 248
 reading, 241
 searching with
 XPath, 244
FileSystemAccessRule
 objects, 410
filling groups, 345
Filter parameter
 (get-childitem
 commandlet), 211
filtering
 event logs, 14
 LDAP queries, 358
 objects, 70-72
 conditions, 70
 heterogeneous
 pipeline content, 72
 parameter output, 28
 processes, 268
 RSS feeds, 301
 WMI objects, 146-147
flags (parameters), 420
folders
 copying, 212
 deleting, 212
 moving, 212
 public, 305
 renaming, 212

- foreach-object commandlet, 105, 235, 431
- forests (Active Directory), 366
- format-byte commandlet, 431
- format-custom
 - commandlet, 431
- format-hex commandlet, 431
- format-list commandlet, 431
- format-table commandlet, 431
- format-wide commandlet, 431
- format-xml commandlet, 244, 431
- formatting XML files, 244
- free space (drives), 208-210
- FullControl right, 404
- G**
- get-acl commandlet, 401, 431
- get-adobject commandlet, 314, 358, 431
- get-alias commandlet, 30, 432
- get-authenticodesignature
 - commandlet, 432
- get-bios commandlet, 432
- get-cdromdrive
 - commandlet, 432
- get-childitem commandlet, 432
 - directory content, 210
 - Filter parameter, 211
 - include parameter, 211
 - registry keys, 253
- get-clipboard commandlet, 200, 432
- get-command
 - commandlet, 432
- get-commandlet, 35
- get-computerinfo
 - commandlet, 432
- get-computername
 - commandlet, 432
- get-content commandlet, 206, 432
 - binary files, 238
 - files, reading, 229, 235
- get-credential
 - commandlet, 58, 432
- get-culture commandlet, 188, 432
- get-currentuser
 - commandlet, 432
- get-datarow commandlet, 396
- get-datatable commandlet, 396
- get-date commandlet, 102, 432
- get-dbconnection
 - commandlet, 432
- get-dbrow commandlet, 433
- get-dbtable commandlet, 433
- get-dhcpserver
 - commandlet, 433
- get-directorychildren
 - commandlet, 362, 433
- get-directoryentry
 - commandlet, 362, 433
- get-directoryvalue
 - commandlet, 362, 433
- get-disk commandlet, 206, 433
- get-domaincontroller
 - commandlet, 324, 433
- get-eventlog commandlet, 290, 433
- get-executionpolicy
 - commandlet, 433
- get-exportedtype
 - commandlet, 215, 433
- get-fileversioninfo
 - commandlet, 215, 433
- get-foregroundwindow
 - commandlet, 434
- get-hash commandlet, 434
- get-help commandlet, 35, 434
- get-history commandlet, 186, 434
- get-host commandlet, 187, 434
- get-item commandlet, 434
 - file properties, 213
 - registry keys, 254
- get-itemproperty
 - commandlet, 255, 434
- get-ITVisions commandlet, 434
- get-keyboard commandlet, 434
- get-location commandlet, 206, 434
- get-mailbox commandlet, 303
- get-mailboxdatabase
 - commandlet, 303

- get-member commandlet,
 - 62, 66-68, 434
 - alias properties, 68
 - code properties, 68
 - methods, 64
 - note properties, 67
 - output, reducing, 69
 - properties, 65
 - property sets, 66
 - script properties, 67
- get-memorydevice
 - commandlet, 434
- get-metadata commandlet, 435
- get-mountpoint
 - commandlet, 435
- get-networkadapter
 - commandlet, 435
- get-peheader commandlet, 215, 435
- get-pfxcertificate
 - commandlet, 435
- get-pipelineinfo
 - commandlet, 60, 435
- get-pointingdevice
 - commandlet, 435
- get-privilege commandlet, 435
- get-process commandlet, 11, 267-268, 435
- get-process | out file
 - commandlet, 55
- get-process | out-printer
 - commandlet, 55
- get-processor commandlet, 435
- get-psdrive commandlet, 83, 206, 435
- get-psprovider
 - commandlet, 84, 435
- get-pssnapin commandlet, 435
- get-pssnapinhelp
 - commandlet, 435
- get-random commandlet, 435
- get-reparsepoint
 - commandlet, 435
- get-service commandlet, 272, 435
- get-service i commandlet, 13
- get-shortpath commandlet, 436
- get-sounddevice
 - commandlet, 436
- get-storagegroup
 - commandlet, 303
- get-tabexpansion
 - commandlet, 436
- get-tapedrive commandlet, 436
- get-terminalsession
 - commandlet, 436
- get-tracesource
 - commandlet, 173, 436
- get-uiculture commandlet, 188, 436
- get-unique commandlet, 436
- get-usbcontroller
 - commandlet, 436
- get-variable commandlet, 436
- get-videocontroller
 - commandlet, 436
- get-wmiobject
 - commandlet, 135, 144, 436
 - hardware information, 284
 - list parameter, 148
- GetAccessRules()
 - method, 411
- GetDrives() method, 206
- GetFactoryClasses()
 - method, 375
- GetOwner() method, 417
- GetType() method, 93
- GPMC (Group Policy Management Console), 367, 450
- GPMGMT component, 367
- graphical user interfaces, 196
 - clipboard, 200
 - input window, 196-198
 - objects, displaying, 198-200
- group-object commandlet, 74, 436
- Group Policy Management Console (GPMC), 367, 450
- grouping objects, 74-75
- groups
 - Active Directory
 - creating/filling, 345
 - deleting users, 346
 - members, 343-345
 - policies, 367
 - classes, 367
 - enumerating, 367, 369
 - links, 369-372
 - WMI management, 314-315

H

- handling script errors, 125-127
- hardlinks, 217-218

hardware
 information, viewing,
 284-285
 printers
 connections, 287
 listing all, 286
 print jobs, 287-289
 status, 286
 hash tables, 106-108
 Hashtable class, 107
 help
 commandlets, 35, 38
 get-commandlet, 35
 .NET classes, 38-40
 PSL, 90
 tool, 169
 Help Editor website, 449
 heterogeneous pipeline
 content, 72
 hexadecimal numbers, 96
 history
 commands, 186-187
 WPS, 4-5
 host information, 187-188
 HTML files, 251

I

IADs interface, 317
 IDE, 156-157
 IEnumerable interface, 319
 IIS (Internet Information
 Services), 305
 classes, 305
 virtual web servers
 adding, 308-311
 deleting, 311
 listing, 307
 IISApplicationPool
 class, 305
 IISComputer class, 305
 IISWebServer class, 305
 IISWebService class, 305
 IISWebVirtualDir
 class, 305
 import-alias commandlet,
 436
 import-bitmap
 commandlet, 436
 import-clicxml
 commandlet, 436
 import-csv commandlet,
 240, 436
 import-dbcommand
 commandlet, 437
 importing
 CSV files, 240
 DataSets, 395
 include parameter
 (get-childitem
 commandlet), 211
 indexed attributes
 (Active Directory
 searches), 354
 input boxes, 56
 input windows, 196-198
 InputBox() method, 56
 Install() method
 (Win32_Product()
 class), 263
 installed services,
 viewing, 13
 installing
 PowerShellPlus, 19
 printers, 287
 PSCX, 17
 services, 278
 software, 263
 WPS, 8-10
 installutil.exe, 175
 IntelliSense
 PowerShellPlus
 commandlet
 names, 159
 commandlet
 parameters, 160
 .NET classes, 161
 path names, 160
 variables, 162
 PrimalScript
 class names, 169
 commandlets, 168
 parameters, 168
 interactive mode, 11, 14
 console window, 11
 event logs, filtering, 14
 IDE, 156
 installed services,
 viewing, 13
 pipeline features, 13
 running processes,
 viewing, 11
 tab completion, 13
 interfaces
 ADSI
 architecture, 316
 deficiencies, 321-323
 directory services,
 compared, 320
 DirectoryEntry class,
 318-319
 integration, 316
 object model, 318
 property cache, 329
 search queries, 319
 graphical user
 interfaces, 196
 clipboard, 200
 input window, 196-198
 objects, displaying,
 198-200
 IADs, 317
 IEnumerable, 319
 intermediate steps
 (pipelines),
 viewing, 76
 Internet Information
 Services. *See* IIS

- interpreter mode (WPS console), 154
- inventory (software)
 - script, 260-261
 - searching, 260
 - viewing, 259
- invoke-dbcommand
 - commandlet, 396
- invoke-expression
 - commandlet, 109, 437
- invoke-history
 - commandlet, 437
- invoke-item commandlet, 437
- invoke-scalardbcommand
 - commandlet, 437
- J-K**
- Join() method, 102
- join-path commandlet, 437
- join-string commandlet, 102, 437
- joining
 - arrays, 105
 - hash tables, 108
 - strings, 102
- junction points, 218-219
- keys (registry)
 - copying, 254
 - creating, 254
 - deleting, 254
 - entries, 255-257
 - hierarchy script, 115-117
 - reading, 253-254
- Kill() method (Process class), 270
- L**
- LDAP queries
 - example, 350
 - executing, 351
 - filters, 358
 - search example, 352
 - search filters
 - website, 450
 - syntax, 349-350
 - user login name
 - searches, 353-354
- links
 - file system, 216
 - Explorer, 216-217
 - hardlinks, 217-218
 - junction points, 218-219
 - symbolic, 220
 - group policies
 - creating, 369-370
 - deleting, 370-372
 - parameter flags, 420
 - list parameter
 - get-eventlog
 - commandlet, 290
 - get-wmiobject
 - commandlet, 148
 - ListDirectory right, 404
 - listings
 - ACEs
 - adding, 419
 - deleting, 422-423
 - details, 411
 - ACL transfers, 424-426
 - Active Directory
 - domain controllers, 366
 - domains/forests, 366
 - search result
 - restrictions, 357
 - user accounts,
 - passwords, 341
 - Active Directory groups
 - creating, 345
 - deleting members, 346
 - listing members, 344
 - member assignments, 346
- Active Directory user
 - accounts
 - authentication, 341
 - creating, 340
 - deleting, 342
 - moving, 343
 - renaming, 342
 - binary files, 238
 - COM classes
 - existing instances, 134
 - instantiating, 133
 - database access
 - data readers, 387-388
 - provider-independent
 - command objects, 384
 - www.IT-Visions.de
 - extensions, 398-399
 - database connections
 - Microsoft Access, 381
 - Microsoft SQL Server, 381-382
 - Microsoft SQL Server Express, 382
 - provider-independent, 383
 - DataSets
 - provider-independent
 - example, 394-395
 - provider-specific
 - example, 392-393
 - dialog box user input
 - example, 57
 - directory container
 - objects, 331
 - directory entries, 332
 - directory objects
 - customizing, 330
 - fetching, 328
 - properties, 331

- directory service
 - operations via `www.IT-Visions.de` commandlets, 363-364
 - downloading files via HTTP, 300
 - downloading/filtering RSS feeds, 301
 - drive free space, viewing
 - `DriveInfo` class, 208
 - `Win32_LogicalDisk` class, 209-210
 - drive names, 210
 - e-mail, sending, 302
 - executable files, viewing, 215
 - files
 - date and time, configuring, 214
 - share permissions, creating, 226-228
 - shares, creating, 224, 230-232
 - formatted output, 55
 - `get-wmiobject` commandlet, 144
 - group policies
 - enumerating, 368-369
 - links, 370-372
 - input windows, 196-198
 - LDAP
 - searches, executing, 352
 - user login name search, 354
 - networks, configuring, 297-298
 - objects, displaying, 198-200
 - organizational units, creating, 347
 - print jobs, canceling, 288
 - protocol entries, fetching, 291
 - registry example, 258
 - scripts
 - dot sourcing, 118
 - error testing example, 123-125
 - registry key hierarchy, 115-117
 - services
 - configuration, customizing, 278
 - enumerating, 272
 - SIDs
 - displaying, 414
 - SDDL names, 416
 - well-known, 415
 - software
 - installations, testing, 265-266
 - installing, 264
 - inventory script, 260-261
 - inventory solution with WPS, 8
 - inventory solution with WSH, 5-7
 - searching inventory, 260
 - uninstalling, 264
 - strings
 - customizing, 100
 - joining, 102
 - splitting, 101
 - subroutines, 112
 - system owners, reading, 418
 - text files
 - reading, 235
 - writing to, 236
 - user accounts, creating, 14
 - user input, 56
 - user profiles
 - PSCX, 190-195
 - script, 188
 - variable resolution
 - within a string, 99
 - virtual web servers
 - information, viewing, 308
 - waiting for process ending, 271
 - websites, creating from CSV files, 309-311
 - WMI
 - classes, instantiating, 149
 - date format conversions, 145
 - XML files
 - customizing, 247
 - fetching, 242
 - loading
 - assemblies, .NET classes, 131
 - snap-ins, WPS console, 175-176
 - locking variables, 95
 - logical operators, 72
- ## M
- mailboxes (Exchange Server)
 - attributes, 304
 - creating, 303
 - deactivating, 304
 - listing, 303
 - managing, 303-304
 - moving, 304
 - `MailMessage` class, 302
 - `ManagementBaseObject` class, 135

- ManagementClass
 - class, 135
- ManagementDateTimeConverter class, 145
- ManagementObject
 - class, 135
- mass creation, file shares, 229-232
- measure-command
 - commandlet, 173, 437
- measure-object
 - commandlet, 76, 437
- measuring execution
 - time, 173
- methods, 64
 - AddPrinterConnection(), 287
 - AppendChild(), 246
 - calling, 64
 - Change(), 278
 - CommitChanges(), 329
 - ConvertToDateTime(), 145
 - CreateElement(), 246
 - DeleteTree(), 342
 - DownloadString(), 300
 - Exists(), 327
 - GetAccessRules(), 411
 - GetDrives(), 206
 - GetFactoryClasses(), 375
 - GetOwner(), 417
 - GetType(), 93
 - InputBox(), 56
 - Install(), 263
 - Join(), 102
 - Kill(), 270
 - object pipelines, 45-46
 - PurgeAccessRules(), 421
 - RefreshCache(), 329
 - RemoveAccessRule(), 421
 - SelectNodes(), 229, 244
 - SelectSingleNode(), 244
 - SetInfo(), 317
 - Slit(), 101
 - String class, 99
 - Subtract(), 103
 - ToDateTime(), 145
 - ToTimeString(), 60
 - Uninstall(), 264
 - WM classes, 142, 144
- Modify right, 404
- move-item commandlet, 206, 212, 437
- move-itemproperty
 - commandlet, 437
- moving
 - files/folders, 212
 - mailboxes, 304
 - user accounts, 343
- multidimensional
 - arrays, 106
- multivalued attributes
 - (Active Directory searches), 355-356
- N**
- name resolution, 299
- names
 - drives, 210
 - event logs, 290
 - files/folders, 212
 - SDDL, 416-417
- navigating
 - Active Directory, 361
 - commandlets, 84
 - drives, defining, 87-88
 - paths, 85-86
 - registry, 83-84
- .NET
 - 3.0 Redistributable
 - package website, 10
 - classes, 129
 - assemblies, loading, 131
 - constructor
 - parameters, 130
 - enumerations, 132
 - help, 38-40
 - instances, creating, 130
 - library documentation
 - for FileSystemRights enumeration website, 450
 - object analysis, 132
 - static members, 130
 - Community website, 449
 - regular expressions
 - website, 450
 - tools and software
 - components reference website, 449
- NetCmdlets from
 - nsoftware website, 450
- networking
 - configuring, 296-298
 - drives, 210
 - e-mail, sending, 302
 - Exchange Server 2007, 302
 - basic operations, 302
 - databases, listing, 303
 - functionality, testing, 303
 - mailboxes, 303-304
 - public folder
 - management, 305
 - storage groups, 303
 - file retrieval from HTTP servers, 300-301

IIS, 305
 classes, 305
 virtual web servers,
 307-311
 name resolution, 299
 pinging computers, 295
 new-alias commandlet,
 30, 437
 new-hardlink commandlet,
 218, 437
 new-item commandlet,
 206, 437
 registry keys, 254
 text files, creating, 236
 new-itemproperty
 commandlet,
 256, 437
 new-join commandlet,
 218, 437
 new-mailboxdatabase
 commandlet, 303
 new-object commandlet,
 437
 new-psdrive commandlet,
 438
 new-service commandlet,
 278, 438
 new-shortcut commandlet,
 217, 438
 new-storagegroup
 commandlet, 303
 new-symlink commandlet,
 220, 438
 new-timespan
 commandlet,
 103, 438
 new-variable commandlet,
 438
 nonterminating errors, 122
 note properties, 67
 nouns (commandlets), 29

numbers, 96-98
 assigning to untyped
 variables, 96
 hexadecimal, 96
 random, 98

O

object model (DataSets),
 390
 objects
 castrating, 73-74
 comparing, 78
 displaying, 198-200
 filtering, 70-72
 conditions, 70
 heterogeneous
 pipeline content, 72
 grouping, 74-75
 .NET classes, 132
 orientation, pipelining,
 44
 pipelines, 44
 HTML files, 251
 methods, 45-46
 parameters, 46
 XML documents, 248
 sorting, 74
 WMI
 accessing, 137-138
 adapter, 139
 analysis, 140
 filtering/selecting,
 146-147
 ObjectSecurity class, 406
 operators, 72, 108-109
 organizational units,
 creating, 346-347
 out-clipboard
 commandlet, 438
 out-default commandlet,
 51, 438

out-file commandlet, 55,
 236, 438
 out-host commandlet,
 51, 438
 out-null commandlet, 438
 out-printer commandlet,
 55, 287, 438
 out-string commandlet,
 438
 output, 49
 get-member
 commandlet
 alias properties, 68
 code properties, 68
 methods, 64
 note properties, 67
 properties, 65
 property sets, 66
 reducing, 69
 script properties, 67
 mixing literals and
 variables, 54
 printing, 55
 single values, 53-54
 standard, 51-53
 pagewise, 51
 restricting, 52
 suppressing, 55
 text files, 55

P

p parameter (out-host
 commandlet), 51
 pagewise output, 51
 parameters, 26-27
 calculated, 27-28
 case sensitivity, 29
 debugging, 171
 ErrorAction, 125-127
 Filter, 211
 filtering output, 28

- flags, linking, 420
- include, 211
- LDAP queries, 349
- list
 - get-eventlog
 - commandlet, 290
 - get-wmiobject
 - commandlet, 148
- .NET class constructors, 130
- object pipelines, 46
- p, 51
- placeholders, 29
- quotation marks, 26
- sequence, 27
- start-process
 - commandlet, 270
- parentheses () in
 - methods, 64
- passwords (user
 - accounts), 340
- paths, 85-86, 323-325
- pausing
 - print jobs, 288
 - scripts, 122
- performance counters, 292-293
- periods of time, 103
- permissions (file shares), 225-228
- ping-host commandlet, 296, 438
- pinging computers, 295
- Pipeline Processor, 47-49
- pipelines
 - | (vertical line), 43
 - calculations, 76
 - classic commands, 46
 - complex, 48-49
 - content, analyzing, 59
 - alias properties, 68
 - code properties, 68
 - ETS, 69-70
 - get-member
 - commandlet, 62, 66-69
 - get-pipelineinfo
 - commandlet, 60
 - methods, 64
 - note properties, 67
 - properties, 65
 - property sets, 66
 - script properties, 67
 - creating, 43
 - features, 13
 - heterogeneous
 - content, 72
 - intermediate steps, view-
 - ing, 76
- objects, 44
 - castrating, 73-74
 - comparing, 78
 - filtering objects, 70-72
 - grouping, 74-75
 - HTML files, 251
 - methods, 45-46
 - orientation, 44
 - parameters, 46
 - sorting, 74
 - XML, 248
- output, 49
 - printing, 55
 - single values, 53-54
 - standard, 51-53
 - suppressing, 55
 - text files, 55
- Pipeline Processor, 47-49
- ramifications, 78
- user input, 56
 - authentication dialog
 - boxes, 58
 - dialog boxes, 57
 - input box, 56
- placeholders, 29
- plus sign (+) operator, 54, 108
- policies
 - execution, 119
 - group, 367
 - classes, 367
 - creating links, 369-370
 - deleting links, 370-372
 - enumerating, 367-369
- pop-location commandlet, 438
- PowerShell
 - Analyzer, 164
 - Community Extensions.
 - See PSCX
 - documentation
 - website, 449
 - download website, 449
 - Help, 169
 - IDE, 156-157
 - Pipeline Processor, 47-49
 - remoting website, 449
 - Script Language.
 - See PSL
- PowerShell
 - 2.0, 445-447
- PowerShellPlus, 19, 158
 - debugging, 21, 163
 - functions, 158
 - installing, 19
 - IntelliSense
 - commandlets, 159-160
 - .NET classes, 161
 - path names, 160
 - variables, 162
- PrimalScript,
 - compared, 166
- testing, 20
- variables, viewing all, 164
- website, 19

- PowerShellPlus Editor, 22
- PowerTab, 156
- predefined variables, 93
- PrimalScript, 165
 - IntelliSense
 - class names, 169
 - commandlets, 168
 - parameters, 168
 - PowerShellPlus,
 - compared, 166
 - website, 166
 - WPS script output, 167
- printers
 - connections, 287
 - jobs, 287-289
 - listing all, 286
 - output, 55
 - print jobs, 287-289
 - status, 286
- priority parameter
 - (start-process commandlet), 270
- processes, 267
 - ending, 270
 - enumerating, 267-268
 - filtering, 268
 - running, 11
 - starting, 269-270
 - waiting for ending, 271
- product activation
 - settings, 282
- profiles, 189-195
- programming directory
 - services, 325
 - ADSI property cache, 329
 - binding meta objects to
 - directory entries, 325-326
 - container objects, 331
 - directory entries
 - attributes, 328-329
 - creating, 332
 - deleting, 332
 - existence, checking, 327
 - impersonation, 327
 - object properties, 330
 - properties, 65
 - alias, 68
 - code, 68
 - directory objects, 330
 - files
 - customizing, 214
 - date/time information, 214
 - viewing, 213
 - note, 67
 - PSSnapIn, 179
 - script, 67
 - WMI classes, 142-144
 - Property attribute
 - (DirectoryEntry class), 318
 - property sets, 66
 - Prosser, Karl, 164
 - provider factories, 384
 - providers, 83-84
 - independent data access, 382-383
 - listing of, 84
 - viewing, 84
 - PSCX (PowerShell Community Extensions), 17, 181
 - Active Directory
 - navigation, 361
 - commandlets, 181-182
 - downloading, 17
 - executable files
 - commandlets, 214
 - installing, 17
 - LDAP filters, 358
 - testing, 18
 - website, 181, 449
 - PSL (PowerShell Script Language), 89
 - arrays, 105-106
 - associative, 106-108
 - declaring, 105
 - defining, 105
 - joining, 105
 - listing, 105
 - multidimensional, 106
 - command separation, 90
 - comments, 90
 - control structures, 110-112
 - data types, 92
 - date and time, 102-103
 - periods of time, 103
 - remote computers, 104
 - setting, 104
 - hash tables, 106-108
 - accessing, 107
 - defining, 107
 - joining, 108
 - help, 90
 - numbers, 96-98
 - assigning to untyped variables, 96
 - hexadecimal, 96
 - random, 98
 - operators, 108-109
 - strings, 99
 - customizing, 100
 - joining, 102
 - splitting, 101
 - variables, 91
 - constant values, 95
 - data types, 91-93
 - declaring, 91
 - example, 94
 - predefined, 93
 - resolution, 95

- PSSnapIn property (CmdletInfo class), 179
- public folders, managing, 305
- PurgeAccessRules() method, 421
- push-location commandlet, 438
- Q**
- queries
 - LDAP
 - example, 350
 - executing, 351
 - filters, 358
 - search example, 352
 - syntax, 349-350
 - user login name searches, 353-354
 - WQL, 147
- Quest
 - extensions (Active Directory), 365
 - Management Shell for Active Directory, 183-184
- quotation marks (“ ”) in parameters, 26
- R**
- ramifications (pipelines), 78
- random numbers, 98
- Read right, 404
- read-host commandlet, 56, 438
- ReadAndExecute right, 404
- ReadAttributes right, 404
- ReadData right, 405
- ReadExtendedAttributes right, 405
- reading
 - ACEs, 410-411
 - ACLs, 408-409
 - binary files, 238
 - directory entry
 - attributes, 328
 - registry keys, 253-255
 - system owners, 417
 - text files, 235-236
 - XML files, 241
- ReadPermissions right, 405
- recovery settings, 283
- reflection mechanism, 44
- RefreshCache() method, 329
- registry, 253
 - data types, 257
 - drives, defining, 255
 - example, 257-258
 - keys
 - copying, 254
 - creating, 254
 - deleting, 254
 - entries, 255-257
 - hierarchy script, 115-117
 - reading, 253-254
 - navigating, 81
 - commandlets, 84
 - drives, 83-84
 - providers, 83-84
- regular expressions, 71
- relational operators, 72
- remote computers, date and time, 104
- remove-directoryentry commandlet, 362, 438
- remove-item commandlet, 206, 212, 254, 438
- remove-itemproperty commandlet, 257, 438
- remove-mountpoint commandlet, 439
- remove-psdrive commandlet, 439
- remove-pssnapin commandlet, 439
- remove-reparsepoint commandlet, 439
- remove-variable commandlet, 439
- RemoveAccessRule() method, 421
- rename-item commandlet, 206, 212, 439
- rename-itemproperty commandlet, 439
- renaming
 - files/folders, 212
 - users, 342
- resize-bitmap commandlet, 439
- resolution (variables), 95
- resolve-assembly commandlet, 215, 439
- resolve-host commandlet, 299, 439
- resolve-path commandlet, 439
- resources (security classes), 407
- restart-service commandlet, 277, 439
- restricting output, 52
- resume-service commandlet, 439
- retrieving files from HTTP servers, 300-301
- rights (access), 403-406
- RSS feeds, 301
- running processes, viewing, 11

S

- schemas (Active Directory), 338
- script mode, 14-15, 156
- scripts
 - DataSet providers
 - independent example, 394-395
 - specific example, 391-393
 - debugging, 21
 - digital signatures, 120-121
 - dot sourcing, 118
 - errors, 122
 - creating, 128
 - handling, 125-127
 - history, 128
 - standard reactions, 127
 - trap blocks, 128
 - trapping example, 123-125
 - Exchange Server scripts
 - website, 451
 - pausing, 122
 - properties, 67
 - registry key hierarchy, 115-117
 - security, 118-119
 - software inventory, 260-261
 - starting, 117
 - user accounts,
 - creating, 14
- SCVMM (System Center Virtual Machine Manager), 185
- SDDL (Security Descriptor Definition Language), 416
 - ACLs, configuring, 425-426
 - names, 416-417
- SDK website, 450
- SDs (security descriptors), 225, 402
- search queries (ADSI), 319
- searching
 - Active Directory, 314
 - indexed attributes, 354
 - multivalued attributes, 355-356
 - result restrictions, 357
 - star operator, 356
 - software inventory, 260
 - text files, 237
 - XML files, 244
- security, 402
 - access rights, 403-406
 - ACLs, 402
 - ACEs, 402
 - adding ACEs, 418-419
 - configuring, 425-426
 - deleting ACEs, 421-423
 - reading ACEs, 410-411
 - transferring, 424
 - classes, 406
 - control holders, 408
 - inheritance
 - hierarchy, 406
 - ObjectSecurity, 406
 - reading ACLs, 408-409
 - resources, 407
 - descriptors (SDs), 225, 402
 - owners, reading, 417
 - scripts, 118-119
 - SIDs
 - displaying, 414
 - SDDL names, 416-417
 - well-known, 414-416
 - user accounts, 402
- Security Descriptor Definition Language. *See* SDDL
- security descriptor (SDs), 225, 402
- security identifiers. *See* SIDs
- select-object commandlet, 70, 73, 439
- select-string commandlet, 237, 439
- select-xml commandlet, 244-246, 439
- selecting WMI objects, 146
- SelectNodes() method (XMLDocument class), 229, 244
- SelectSingleNode() method (XmlDocument class), 244
- semicolons (;) in commands, 90
- send-smtpmail commandlet, 302, 439
- sending e-mail, 302
- sequence (parameters), 27
- serial numbers (computers), 282
- servers
 - HTTP, 300-301
 - SQL, 376
 - virtual web servers
 - adding, 308-311
 - deleting, 311
 - listing, 307
- services
 - attributes, 278
 - configuration,
 - customizing, 278-279
 - dependent, 274-276
 - directory, 325
 - access, 313
 - ADSI. *See* ADSI

- binding meta objects
 - to directory entries, 325-326
- container objects, 331
- directory entries, 332
- directory entry
 - attributes, 328-329
- directory entry existence, checking, 327
- impersonation, 327
- object properties, 330
- paths, 323-325
- www.IT-Visions.de
 - commandlets, 362-364
 - enumerating, 272-273
 - installed, viewing, 13
 - installing, 278
 - starting, 276-277
 - stopping, 277
- set-acl commandlet, 401, 440
- set-alias commandlet, 30, 440
- set-authenticodesignature commandlet, 120, 440
- set-clipboard commandlet, 200, 440
- set-content commandlet, 206, 440
 - binary files, 238
 - text files, writing, 236
- set-datarow commandlet, 396
- set-datatable commandlet, 396
- set-date commandlet, 104, 440
- set-dbtable commandlet, 440
- set-directoryvalue commandlet, 362, 440
- set-distributiongroup commandlet, 304
- set-executionpolicy commandlet, 119, 440
- set-filetime commandlet, 214, 440
- set-foregroundwindow commandlet, 440
- set-item commandlet, 206, 440
- set-itemproperty commandlet, 214, 440
- set-location commandlet, 206, 254, 441
- set-privilege commandlet, 441
- set-psdebug commandlet, 173, 441
- set-service commandlet, 278, 441
- set-tracesource commandlet, 173, 441
- set-variable commandlet, 441
- set-volumelabel commandlet, 210, 441
- SetInfo() method, 317
- settings (computers), 281-283
- SharePoint Provider website, 449
- SIDs (security identifiers), 402
 - displaying, 414
 - SDDL names, 416-417
 - well-known, 414-416
- signing scripts, 120-121
- single value output, 53-54
- SMTP (Simple Mail Transfer Protocol), 302
- SmtplibClient class, 302
- snap-ins
 - adding, 175
 - commandlets, 179
 - listing, 178
 - loading in WPS console, 175-176
- Snover, Jeffrey, 117
- software, 259
 - autostart, 263
 - installed list of, 262
 - installing, 263
 - inventory
 - script, 260-261
 - searching, 260
 - solution with WPS, 8
 - solution with WSH, 5-7
 - viewing, 259
 - not installed with Windows Installer, 262
 - uninstalling, 264
 - versions, viewing, 282
- sort-object commandlet, 74, 441
- sorting objects, 74
- Split() method, 101
- split-path commandlet, 441
- split-string commandlet, 101, 441
- splitting strings, 101
- SQL Servers, listing available, 376
- standard output, 51-53
 - pagewise, 51
 - restricting, 52
- star operator (*), 108, 356
- start-process commandlet, 269-270, 441
- start-service commandlet, 277, 441

- start-sleep commandlet, 122, 441
- start-tabexpansion commandlet, 441
- start-transcript commandlet, 441
- starting
 - processes, 269-270
 - scripts, 117
 - services, 276-277
- static members
 - .NET classes, 130
 - WMI classes, 144
- step-by-step debugging, 173
- stop-process commandlet, 270, 441
- stop-service commandlet, 277, 441
- stop-terminalsession commandlet, 441
- stop-transcript commandlet, 441
- stopping services, 277
- storage limitations (public folders), 305
- String class, 99
- strings, 99
 - customizing, 100
 - joining, 102
 - representation, 60
 - splitting, 101
- subroutines, 112
- Subtract() method, 103
- suspend-service commandlet, 442
- symbolic links, 220
- Synchronize right, 405
- syntax
 - commandlets, 26
 - LDAP queries, 349-350

- logical operators, 72
- regular expressions, 71
- relational operators, 72
- System Center Virtual Machine Manager (SCVMM), 185
- system information, 187-188
- system owners, reading, 417
- System.Management
 - object model, 135
- System.Management namespace
 - documentation
 - website, 450

T

- tab completion, 13, 153
- TakeOwnership right, 405
- targets (junction points), 219
- terminating errors, 122
- test-assembly commandlet, 214, 442
- test-dbconnection commandlet, 396, 442
- test-path commandlet, 442
- test-xml commandlet, 243, 442
- testing
 - Exchange Server 2007 functionality, 303
 - PowerShellPlus, 20-22
 - PSCX, 18
- text files
 - content, deleting, 236
 - reading, 235-236
 - searching, 237
 - writing to, 236-237

- time and date, 102-103
 - periods of time, 103
 - remote computers, 104
 - setting, 104
- TimeSpan class, 103
- ToDateTime()
 - method, 145
- ToString() method, 60
- trace-command commandlet, 442
- tracing, 173
- transferring ACLs, 424
- Traverse right, 405
- tree-object commandlet, 78, 442
- trial and error website, 451
- type indicators (WMI classes), 139

U

- Uninstall() method (Win32_Product class), 264
- uninstalling software, 264
- WPS, 10
- update-formatdata commandlet, 442
- update-typedata commandlet, 442
- user accounts
 - Active Directory, 335-338
 - authentication, 341
 - creating, 339-340
 - deleting, 342
 - moving, 343
 - passwords, 340
 - renaming, 342
- creating, 14
- security, 402

user administration
 Active Directory
 authentication, 341
 deleting users, 342
 moving users, 343
 renaming users, 342
 user accounts, 339-340
 user class attributes,
 335-338
 WMI, 314-315
 users
 adding to groups, 345
 deleting from
 groups, 346
 input, 56
 authentication dialog
 boxes, 58
 dialog boxes, 57
 input box, 56
 object user interface
 mapping website, 450
 profile script, 188

V

variables, 91
 constant values, 95
 data types, 91-93
 declaring, 91
 example, 94
 predefined, 93
 resolution, 95
 VBScript command con-
 versions website, 451
 verbose parameter,
 171-172
 vertical line (|) for
 pipelines, 43
 viewing
 commandlets list, 35
 computer settings,
 281-283
 directory content,
 210-212

drive free space (file
 system), 208, 210
 drives list, 206-207
 environment
 variables, 283
 executable files, 215
 file properties, 213
 hardware information,
 284-285
 installed services, 13
 objects, 198-200
 pipeline intermediate
 steps, 76
 providers, 84
 running processes, 11
 SIDs, 414
 software inventory,
 259, 262
 virtual web servers
 adding, 308-311
 deleting, 311
 listing, 307
 Vista user account
 control, 155

W

waiting for process
 ending, 271
 WebClient class, 300
 websites
 Active Directory
 schema, 450
 AD Access
 Change/Break in
 RC2, 449
 Cmdlet development
 guidelines, 450
 Cmdlet help, 450
 data providers, 375
 ETS, 450
 Exchange Management
 Shell, 451

Exchange Server
 scripts, 451
 Group Policy
 Management
 Console with Service
 Pack 1, 450
 Help Editor, 449
 LDAP search filters, 450
 .NET Framework
 3.0 Redistributable
 package, 10
 class library
 documentation for
 FileSystemRights
 enumeration, 450
 Community, 449
 Framework regular
 expressions, 450
 tools and software
 components
 reference, 449
 NetCmdlets from
 nsoftware, 450
 PowerShell
 Analyzer, 165
 documentation, 449
 download, 449
 Help, 169
 remoting, 449
 PowerShellPlus, 19
 PrimalScript, 166
 PSCX, 17, 181, 449
 SDK, 450
 SharePoint Provider, 449
 System.Management
 documentation, 450
 trial and error, 451
 user object user
 mapping, 450
 VBScript command
 conversions, 451
 Windows PowerShell
 graphical help
 file, 449

- WMI schema class
 - reference, 450
- WPS download, 9
- www.IT-Vision.de WPS
 - extensions, 183
- well-known security
 - principals, 414-416
- WhatIf parameter, 171
- where-object commandlet,
 - 70, 442
- wildcards, 29
- Win32_Computersystem
 - class, 281
- Win32_Desktop class, 315
- Win32_LogicalDisk class
 - drive free space,
 - viewing, 209-210
 - drives, viewing, 207
- Win32_NetworkAdapter-Configuration
 - class, 296
- Win32_NTLogEvent
 - class, 291
- Win32_Operating System
 - class, 281
- Win32_PerfRawData
 - class, 292
- Win32_Product class, 259
- Win32_Service class, 277
- Win32_Share class, 221
- Win32_StartupCommand
 - class, 263
- Win32_Trustee class, 226
- windows
 - console, 11
 - input, 196, 198
- Windows Forms
 - PropertyGrid
 - control, 198
- Windows PowerShell.
 - See* WPS
- WMI (Windows Management Instrumentation), 135
 - classes, 135
 - available, listing, 148
 - collections,
 - accessing, 146
 - IIsApplicationPool, 305
 - IIsComputer, 305
 - IIsWebServer, 305
 - IIsWebService, 305
 - IIsWebVirtualDir, 305
 - instances, creating, 149
 - object access, 137-138
 - object adapter, 139
 - object analysis, 140
 - object filtering/
 - selecting, 146-147
 - properties/methods, 142-144
 - queries, 147
 - static class
 - members, 144
 - System.Management
 - object model, 135
 - type indicators, 139
- Win32_Computer-system, 281
- Win32_Desktop, 315
- Win32_LogicalDisk, 207-210
- Win32_NetworkAdapter Configuration, 296
- Win32_NTLogEvent, 291
- Win32_Operating-System, 281
- Win32_PerfRawData, 292
- Win32_Product, 259
- Win32_Service, 277
- Win32_Share, 221
- Win32_Startup-Command, 263
- Win32_Trustee, 226
- WPS support, 136
- date format
 - conversions, 145
- groups, managing, 314-315
- objects
 - accessing, 137-138
 - adapter, 139
 - analysis, 140
 - schema class reference
 - website, 450
 - users, managing, 314-315
- WMI Query Language (WQL), 147
- WorkingDirectory parameter (start-process commandlet), 270
- WPS (Windows PowerShell).
 - See also* scripts
 - definition, 3
 - benefits, 5
 - console, 151
 - command history, 186-187
 - command mode, 154
 - functions, 152
 - interpreter mode, 154
 - PowerTab, 156
 - snap-ins, loading, 175-176
 - tab completion, 153
 - Vista user account
 - control, 155
 - downloading, 8
 - graphical help file
 - website, 449
 - history, 4-5
 - installing, 8-10

- interactive mode, 11-14
 - console window, 11
 - event logs, filtering, 14
 - installed services,
 - viewing, 13
 - pipeline features, 13
 - running processes,
 - viewing, 11
 - tab completion, 13
 - script mode, 14-15
 - software inventory
 - solution, 8
 - uninstalling, 10
 - WMI support, 136
- WQL (WMI Query Language), 147
- Write right, 405
- write-bzip2 commandlet, 442
- write-clipboard
 - commandlet,
 - 200, 442
- write-debug commandlet, 442
- write-error commandlet, 53, 442
- write-gzip commandlet, 442
- write-host commandlet, 53, 442
- write-output commandlet, 443
- write-progress
 - commandlet, 443
- write-tar commandlet, 443
- write-verbose
 - commandlet, 443
- write-warn commandlet, 53
- write-warning
 - commandlet, 443
- write-zip commandlet, 220, 443
- WriteAttributes right, 405
- WriteData right, 406
- WriteExtendedAttributes right, 406
- writing
 - binary files, 238
 - directory entry
 - attributes, 329
 - text files, 236-237
- WSH software inventory
 - solution, 5, 7
- www.IT-Visions.de
 - extensions, 183
 - Active Directory, 362-364
 - database access, 396-399

X-Z

- XML files, 241
 - checking, 242-243
 - converting to XHTML files, 249
 - customizing, 246
 - DataSet exports/
 - imports, 395
 - formatting, 244
 - object pipeline, 248
 - reading, 241
 - searching with XPath, 244
- XMLDocument class, 229, 244
- XPath, 244