

Index

SYMBOLS

- & (ampersand), at beginning of command, 65, 68**
- ' (apostrophe), 119, 416, 431–432**
- = (assignment operator), 72, 220–222**
- * (asterisk)**
 - as multiplication operator, 218, 219
 - in .* pattern, 54, 80
 - as wildcard character, 12, 48, 49, 59, 72, 73, 125, 171–173, 200, 202
- @ (at sign), in associative arrays, 249**
- \ (backslash), at beginning of commands, 23, 472**
- ^ (caret)**
 - as metacharacter, 81, 85, 290, 357, 508
 - in \$^ variable, 61
- : (colon), 196, 346, 375, 469**
- , (comma), 121**
- { } (curly braces), for designating script blocks, 71–72**
- / (division operator), 218, 219**
- \$ (dollar sign)**
 - at beginning of expression, 64, 65
 - in \$\$ variable, 61
- (minus sign)**
 - in assignment operator, 220
 - as subtraction operator, 218, 219
 - in unary operators, 226–227
- ! (-not operator), 225**
- () (parentheses)**
 - role in parsing, 64
 - using with methods, 264
- % (percent sign), in assignment operator, 220, 221**
- . (period)**
 - for running PowerShell scripts, 23, 472
 - when using PowerShell command mode, 64–66, 68
- | (pipeline)**
 - default formatter, 151–154
 - defined, 78
 - examples, 13, 14, 19–20, 41–42
 - grouping objects, 83–85
 - layout, 19
 - .NET objects in, 35–38, 57
 - overview, 19–20, 78–85
 - past limitations, 56–57
 - role in sorting objects, 81–83
 - separators in, 13, 78
 - sequence of commands, 78–81
 - symbol, 19, 78
 - syntax, 19
- + (plus sign)**
 - as addition operator, 218, 219, 248
 - in assignment operators, 220
 - in unary operators, 226–227
- ? (question mark)**
 - in \$? variable, 61
 - as wildcard, 59, 72, 125, 171
- “ (quotation mark, double)**
 - at beginning of expression, 64, 65
 - in commands, 71, 431–432
 - parameters and, 119–120
- ' (quotation mark, single), 119, 416, 431–432**
- > (redirection operator), 18, 41, 407, 445–446**
- >> (redirection operator), 199, 445–446**
- ; (semicolon), 20, 198**
- / (slash)**
 - in assignment operator, 220, 221
 - in division operator, 218, 219

[] (square brackets)

[] (square brackets)

in PowerShell syntax for indicating .NET elements, 34, 268, 324

in syntax for designating array elements, 236

in wildcard searches, 59, 290, 429

- (subtraction operator), 218, 219

- (unary operator), 226–227

- (unary operator), 226–227

\$_ (underscore) variable, 61, 71

A

abbreviated commands, 76–78

Access, as COM application, 303–305

Active Directory, 50

\$AD variable, 60

add-content cmdlet, 93

add-history cmdlet, 92

addition operator (+), 218, 219, 248

add-member cmdlet, 95

add-psSnapin cmdlet, 92

administrators, setting execution policies for, 372, 373

Alias provider, 48, 184, 185, 346

aliases

as abbreviations for cmdlets, 77–78

background, 51–52

creating, 108–111

default, 101

displaying lists, 52–53, 107–108, 346

as drives, 48, 346–347

finding, 101–102, 347–348

list of PowerShell cmdlets for working with, 348

overview, 48–49, 77–78, 100–101

in scripts, 87

in WMI command line utility, 51–52

AllSigned execution policy

defined, 207

as security issue, 372, 373, 374

signed scripts and, 209

unsigned scripts and, 208

ampersand (&), at beginning of command, 65, 68

-and operator, 225

apostrophe ('), 119, 416, 431–432

-append parameter, 449

application domain, current, 353–357

Application log, 478

\$Args variable, 61

-argumentList parameter

for new-object cmdlet, 311, 314, 315, 316–317

for trace-command cmdlet, 419

-arguments parameter, 294

arithmetic operators, 217, 218–219

arrays

adding elements, 242

associative, 249–250

concatenating, 248

creating, 235–239

defined, 235

modifying structure, 241–245

naming, 236

shortening elements, 242–243

typed, 239–240

working from end, 245–247

-asSecureString parameter, 212, 213

assemblies, finding, 354–355

assignment operator

equals sign (=), 72, 220–222

minus sign (-), 220

percent sign (%), 220, 221

plus sign (+), 220

slash (/), 220, 221

associative arrays, 249–250

-asString parameter, 481

asterisk (*)

as multiplication operator, 218, 219

in .* pattern, 54, 80

as wildcard character, 12, 48, 49, 59, 72, 73, 125,

171–173, 200, 202

at sign (@), in associative arrays, 249

automatic variables, 60–62

-autoSize parameter, 155, 157

-average parameter, 201

B

-backgroundColor parameter, 215, 216

backslash (\), at beginning of commands, 23, 472

-band operator, 79

batch files (.bat)

limitations, 28

maintenance, 28

versus scripting languages, 28

-begin parameter, 261

-bor operator, 79

break statement, 394, 395

C

C#, upgrade path to, 58

caret (^)

as metacharacter, 81, 85, 290, 357, 508

in \$^ variable, 61

case

- .NET Framework and, 18, 326–327
- in PowerShell cmdlets, 18, 92, 327
- category **parameter, 400**
- categoryActivity **parameter, 400**
- categoryReason **parameter, 400**
- categoryTargetName **parameter, 400**
- categoryTargetType **parameter, 400**
- ccontains **operator, 80**

cd alias, 49

- ceq **operator, 80, 223**

Certificate namespace, 374–376

- certificate **parameter, 377**

Certificate provider, 184, 185, 346**certificates**

- code-signing, 376–379
- creating, 376–377
 - as drives, 51, 374–375, 376
 - finding, 377–378
- cge **operator, 79, 223**
- cgt **operator, 79, 223**
- character **parameter, 201**

child items, obtaining, 55. See also get-childItem cmdlet

- childPath **parameter, 452**

CIM repository, 499**CIM Studio tool, 499–500****CIMOM (Common Information Model Object Manager), 497, 498****classes, .NET Framework**

- calling, 34
- displaying by using `get-member`, 35, 36
- finding information about, 324–333
- library, 36, 37–38

- cle **operator, 79, 223**

clear command, 17**clear-content cmdlet, 93****clear-host cmdlet, 17****clear-item cmdlet, 93****clear-itemProperty cmdlet, 93****clear-variable cmdlet, 95, 231–232**

- clike **operator, 80, 223**

Clone() method, 257, 265**closing Windows PowerShell, 10****cls cmdlet, 18, 52**

- clt **operator, 79, 223**

- cmatch **operator, 80, 224**

CMD.exe

- background, 27
- exiting PowerShell to prompt, 10
- limitations, 27–29

- versus PowerShell, 428, 455
- similarity of PowerShell to prompt, 17

CmdletInfo object, 152, 153**cmdlets**

- aliases for abbreviating, 77–78
- availability, 11–12, 75–76
- case-insensitivity, 18, 92, 327
- combining in pipelines, 19–20, 78–85
- completing by using Tab key, 76–77, 112–113, 443–444
- cycling through recently used, 18
- defined, 11
- exploring, 11–14
- for file actions, 449–450
- finding, 11–14
- finding parameters, 121–124
- focusing search, 12
- getting help with, 14–17
- getting recent-use history, 18
- grouping, 83–85
- naming scheme, 57–58, 85
- overview, 20–21
- parameters and, 117–131
- separator between, 20
- syntax, 20–21
- testing combinations, 21–23
- verbosity issue, 85–87
- verbs as, 13
- viewing list, 11–12
- ways to abbreviate, 76–78
- for working with paths, 450–453
- cne **operator, 80, 223**
- cnotcontains **operator, 80**
- cnotlike **operator, 80, 224**
- cnotmatch **operator, 80, 224**

code debugging

- handling syntax errors, 403–408
 - as PowerShell feature, 59
 - `set-psDebug` cmdlet, 408–413
 - tracing, 418–423
 - `write-psDebug` cmdlet, 413–418

colon (:), 196, 346, 375, 469**COM (Component Object Model)**

- accessing, 56
- creating objects by using `new-object` cmdlet, 293–294
- Internet Explorer and, 294–299
- Microsoft Access and, 303–305
- Microsoft Excel and, 302–303
- Microsoft Word and, 301–302
- network shares and, 305–306

COM (Component Object Model) (continued)

COM (Component Object Model) (continued)

- object scripting, 43–44
- PowerShell support, 293
- role of synthetic types, 306–308
- specific applications and, 294–308
- Windows Script Host and, 30, 299–301

comma (,), 121

command line, PowerShell

- applying in exploratory way, 39–41
- approaches to parsing, 63–69
- batch file tools versus scripting languages, 28
- command mode versus expression mode, 63–69
- existing utilities, 53–55
- tool history, 26–27
- tool inconsistency, 28
- using utilities from, 54
- Windows utilities, 53–55

command mode

- examples, 66–69
- versus expression mode, 63–65
- mixing expressions with, 69

-command parameter, 419

-Command PowerShell startup option, 10

command prompt

- >> as, 19
- creating, 113–115
- customizing, 113–115

command shell providers. See providers

commands, PowerShell, using Tab key to complete, 76–77, 112–113, 443–444. See also cmdlets; scripts, PowerShell

Common Information Model Object Manager (CIMOM), 497, 498

- comObject parameter, 43, 56, 294, 311

Compare() method, 265, 268–270

compare-object cmdlet, 95

CompareTo() method, 265, 271

comparing strings, 268–271

comparison operators, 217, 222–224

completing commands by using Tab key

- cmdlet examples, 443–444
- as PowerShell abbreviated command form, 76–77
- registry example, 112–113
- when not to use, 113

Component Object Model (COM)

- accessing, 56
- creating objects by using new-object cmdlet, 293–294
- Internet Explorer and, 294–299
- Microsoft Access and, 303–305

- Microsoft Excel and, 302–303
- Microsoft Word and, 301–302
- network shares and, 305–306
- object scripting, 43–44
- PowerShell support, 293
- role of synthetic types, 306–308
- specific applications and, 294–308
- Windows Script Host and, 30, 299–301

concatenated strings, 65

concatenating arrays, 248

conditional expressions, 250–256

- confirm parameter
 - for clear-variable cmdlet, 231
 - as common parameter, 132, 165, 166
 - for new-item cmdlet, 203
 - for new-psDrive cmdlet, 204
 - for new-variable cmdlet, 229
 - for remove-psDrive cmdlet, 187
 - for remove-variable cmdlet, 233
 - for set-authenticodeSignature cmdlet, 377
 - for set-executionPolicy cmdlet, 371
 - for set-variable cmdlet, 228
 - for stop-service cmdlet, 180–181
- \$ConfirmPreference variable, 61

console files, loading, 90

\$ConsoleFileName variable, 61

Constant option, 348–349, 388

consumers, WMI, 499

Contains() method, 265, 271–272

- contains operator, 79

continue statement, 394–395

ConvertFrom-SecureString cmdlet, 94

convert-path cmdlet, 93, 450

ConvertTo-html cmdlet, 95

ConvertTo-SecureString cmdlet, 94

copying strings, 267

- copy-item cmdlet, 93
- copy-itemProperty cmdlet, 93

CopyTo() method, 265, 272–273

core library, .NET, 355–358

Core snapin, 90–92

count property, 21, 22

CPUs. See Win32_Processor WMI class

- credential parameter
 - for get-content cmdlet, 197
 - for join-path cmdlet, 452
 - for new-item cmdlet, 203, 204
 - for remove-item cmdlet, 167
 - for resolve-path cmdlet, 453
 - for test-path cmdlet, 451

curly braces ({}), for designating script blocks, 71–72
current application domain, 353–357
current working location, system state information, 338, 340–341
 CurrentDomain **property, 354**
custom drives, 447–448

D

data stores, as drives, 45
databases, accessing data in Access databases from PowerShell command line, 303–305
date and time examples, 22, 34–38
 DateTime **object**
 creating by casting strings, 318–319
 creating by using `new-cmdlet cmdlet`, 312–317
 role in casting strings, 288–289
 ToString() **method, 22**
 DayOfWeek **property, 288**
 -debug **parameter, 132**
 -debugger **parameter, 419**
debugging code
 handling syntax errors, 403–408
 as PowerShell feature, 59
 set-psDebug **cmdlet, 408–413**
 tracing, 418–423
 write-psDebug **cmdlet, 413–418**
 \$DebugPreference **variable, 61, 414, 415**
decimal point (.)
 for running PowerShell scripts, 23, 72
 when using PowerShell command mode, 64, 65, 66, 68
default formatter, 151–154
 default **option, 255–256**
 Definition **property, 323, 349**
 -delimiter **parameter, 197, 198**
 -description **parameter**
 for new-psDrive **cmdlet, 204**
 for new-variable **cmdlet, 229**
 for set-variable **cmdlet, 228**
developers, setting execution policies for, 372, 373
 dir **alias, 47, 48, 52, 55, 78, 433**
 DirectoryInfo **object, 434–435**
 -displayErrors **parameter, 156**
 -displayName **parameter, 178**
division operator (/), 218, 219
dollar sign (\$)
 at beginning of expression, 64, 65
 in \$\$ **variable, 61**
DOS machines, 26

dot (.) notation
 for running PowerShell scripts, 23, 72
 when using PowerShell command mode, 64, 65, 66, 68
 do/while **statement, 259–260**
drives. See also get-psDrive cmdlet
 aliases as, 48, 346–347
 certificates as, 51, 346, 374–375, 376
 creating, 187, 204–205
 custom, creating, 447–448
 defined, 45
 finding, 434
 finding information about, 425–426
 in fully qualified path names, 427
 functions as, 349–350
 mapping, 305–306
 namespaces as, 45–51
 PowerShell definition, 45
 relationship to providers, 45, 186
 removing, 94, 187–188
 variables as, 49, 346, 350

E

elseif clause, 252–253
 -encoding **parameter**
 for get-content **cmdlet, 197**
 for out-file **cmdlet, 449**
 -end **parameter, 261**
end users, setting execution policies for, 372, 373
 EndsWith() **method, 265, 273–274**
 Environment **provider, 184, 185, 346, 351, 468–469**
environment variables
 defined, 465
 editing information about, 467–468
 exploring by using `get-childItem cmdlet`, 470–471
 finding information about, 466–467
 modifying, 466, 471–473
 overview, 465–468
 as PowerShell tool, 351–353
 -eq **operator**
 as comparison operator, 223
 versus = operator, 72
 using with where-object **cmdlet, 79, 144**
equal sign (=), as assignment operator, 72, 220–222.
 See also -eq operator
 Equals() **method, 265, 274–275**
error handling, system state information, 338, 345–346
 \$Error **variable, 61, 383–387, 388**

-errorAction parameter

`-errorAction` **parameter, as common parameter, 132, 397–398**

`$ErrorActionPreference` **variable, 61, 390–392, 397, 398**

`-errorId` **parameter, 400**

`-errorRecord` **parameter, 400**

errors

- nonterminating, 381, 382, 392
- related variables, 345–346, 383–392
- syntax, 403–423
- system, 381–401
- terminating, 381, 382
- trapping, 392–397

`-errorVariable` **parameter, as common parameter, 132, 399**

`$ErrorView` **variable, 61, 389–390**

event logs

- displaying entries in Windows Event Viewer, 478
- displaying entries onscreen by using PowerShell, 480–492
- overview, 477–480
- PowerShell versions, 477
- writing entries to file, 492–494

Event Viewer

- Filter tab, 478, 479
- opening, 477
- viewing logs, 478

Excel, as COM application, 302–303

`-exception` **parameter, 400**

Exchange Management Shell, 50

Exchange Server 2007, 38, 39, 50

exclamation point, in `-!` operator, 225

`-exclude` **parameter**

- for `clear-variable` cmdlet, 231
- for `get-childitem` cmdlet, 191
- for `get-content` cmdlet, 197
- for `get-variable` cmdlet, 230
- for `remove-item` cmdlet, 167
- for `remove-variable` cmdlet, 232
- for `set-variable` cmdlet, 228
- for `stop-service` cmdlet, 178
- for `test-path` cmdlet, 451

`-excludeProperty` **parameter, 144**

execution policies

- checking by using `get-executionPolicy` cmdlet, 209–212
- defined, 207
- enabling scripts by using `set-executionPolicy` cmdlet, 9, 23
- overview, 207–209
- as security issue, 370–374

`$ExecutionContext` **variable, 61**

`-executionPolicy` **parameter, 371, 373**

exiting Windows PowerShell, 10

`-expand` **parameter, 156**

`-expandProperty` **parameter, 144, 146**

`export-alias` **cmdlet, 95, 105–106, 348**

`export-clixml` **cmdlet, 95**

`export-console` **cmdlet, 92**

`export-csv` **cmdlet, 95**

expression mode

- versus command mode, 63–65
- examples, 65–66
- mixing commands with, 69

`-expression` **parameter, 419**

expressions

- conditional, 250–256
- regular, 289–291

extended wildcards, 59–60

F

`$False` **variable, 61**

file system

- cmdlets for file actions, 449–450
- finding file characteristics, 436–439
- finding files and folders, 425, 427–431
- finding hidden files, 442–443
- removing items, 166–175

`FileInfo` **object**

- retrieving, 434, 435
- `select-object` cmdlet and, 440, 441
- working with methods, 436–437
- working with properties, 438–439

`-filePath` **parameter**

- for `out-file` cmdlet, 449
- for `set-authenticodesignature` cmdlet, 377, 378
- for `trace-command` cmdlet, 419

files and folders

- finding, 434–436
- finding file characteristics, 436–439

`FileSystem` **provider**

- defined, 184, 185, 346, 425
- multiple drives and, 195–196
- overview, 45, 46, 47, 426

`-filter` **parameter**

- for `get-childitem` cmdlet, 191
- for `get-content` cmdlet, 197
- for `remove-item` cmdlet, 167
- for `test-path` cmdlet, 451

filtering processes

- using `where-object` cmdlet, 41, 71–75, 79–81, 137–144
- using wildcards, 72–73

filters, 349–350

`-filterScript` **parameter, 138, 142–143**

finding

- aliases, 101–102, 138, 347–348
- assemblies, 354–355
- certificates, 377–378
- cmdlets, 11–14
- drives, 434
- file characteristics, 436–439
- files and folders, 434–436
- hidden files, 442–443
- information about drives, 425–426
- information about environment variables, 466–467
- information about .NET classes, 324–333
- members of .NET Framework objects by using `get-member` cmdlet, 35, 36, 50, 320–323
- parameters, 121–124
- PowerShell commands, 11–14, 75–76
- providers, 183–184, 425
- services that are running by using `get-service` cmdlet, 74–75, 138–140
- Windows processes that are running by using `get-process` cmdlet, 69–71

`findstr` **command, 53, 54**

`-first` **parameter, 145, 148–150**

`for` **statement, 256–258, 382**

`-force` **parameter**

- for `clear-variable` cmdlet, 231
- for `format-table` cmdlet, 156
- for `get-childitem` cmdlet, 191, 442–443
- for `get-content` cmdlet, 197
- for `new-item` cmdlet, 203
- for `new-variable` cmdlet, 229
- for `remove-item` cmdlet, 166
- for `remove-psDrive` cmdlet, 187
- for `remove-variable` cmdlet, 232
- for `set-variable` cmdlet, 228
- for `stop-service` cmdlet, 178
- for `trace-command` cmdlet, 419

`foreach` **statement, 238, 260–261**

`foreach-object` **cmdlet, 92**

`-foregroundColor` **parameter, 215, 216**

`format-custom` **cmdlet, 95**

`$FormatEnumerationLimit` **variable, 61**

`format-list` **cmdlet**

- adding to final pipeline step, 323
- defined, 95

displaying information, 57, 186

example, 58

`-force` parameter in, 386

overview, 161–162

`format-table` **cmdlet**

`-autoSize` parameter in, 157

defined, 95, 155

displaying information, 57

`-groupBy` parameter in, 158–159

`-hideTableHeaders` parameter in, 158

list of parameters, 155–156

`-property` parameter in, 156–157

specifying column labels, 159–161

specifying column widths, 159–161

`format-wide` **cmdlet, 95**

`-full` **parameter, 60**

`FullName` **property, 355**

fully qualified path names, 427–430

`Function` **drive, 349–350, 431**

`Function` **provider, 184, 185, 346**

functions, 349–350

G

`gci` **alias, 52. See also** `get-childitem` **cmdlet**

`-ge` **operator, 79, 144, 223**

`get` **verb, 57**

`get-acl` **cmdlet, 94**

`get-alias` **cmdlet**

versus `Alias` provider, 48

defined, 95, 348

overview, 107–108

using to find available aliases, 52

verbosity example, 85–86

`GetAssemblies()` **method, 354**

`get-authenticodeSignature` **cmdlet, 94, 376, 378**

`get_Chars()` **method, 265, 275**

`get-childitem` **cmdlet**

aliases, 48–49, 55, 78

defined, 93

versus `dir` alias, 47

examples, 41–42, 68–69

`-force` parameter in, 442–443

overview, 191–194

`Registry` provider and, 47–48

retrieving aliases, 103–104

retrieving environmental variable information, 351–353

`get_ChildNodes()` **method, 289**

`get-command` **cmdlet**

defined, 92

versus `findstr` command, 54

using to find commands, 11, 12, 13–14, 20–21, 75–76

`get-content` **cmdlet**, **21–22, 54, 93, 196–201**
`get-credential` **cmdlet**, **94**
`get-culture` **cmdlet**, **95**
`get_CurrentDomain()` **method**, **354**
`get-date` **cmdlet**, **21, 22, 36–37, 56, 95, 490–491**
`GetEnumerator()` **method**, **265**
`get-eventlog` **cmdlet**, **93, 480–494**
`get-executionPolicy` **cmdlet**, **94, 371**
`GetHashCode()` **method**, **265**
`get-help` **cmdlet**
 defined, 92
 displaying detail, 60
 finding parameters, 121–124
 overview, 14–17
`get-history` **cmdlet**, **18, 92**
`get-host` **cmdlet**, **95**
`get-item` **cmdlet**, **93**
`get-itemProperty` **cmdlet**, **93**
`get_Length()` **method**, **265, 275**
`get-location` **cmdlet**, **93, 194–196, 338–345**
`get-member` **cmdlet**, **35, 36, 50, 95, 320–323**
`GetMember()` **method**, **326–328**
`GetMembers()` **method**, **324–325**
`GetMethod()` **method**, **329**
`GetMethods()` **method**, **328–329**
`get-pfxCertificate` **cmdlet**, **94**
`get-process` **cmdlet**
 defined, 93
 filtering results by using `where-object` cmdlet, 73, 81, 138–140
 finding running Windows processes, 69–71
 .NET Framework System.Diagnostics.Process objects and, 34
 parameters for, 118–121
 in scripting example, 21, 22
 sorting processes by using `sort-object` cmdlet, 81–83
`GetProperties()` **method**, **330–331**
`GetProperty()` **method**, **331–333**
`get-psDrive` **cmdlet**
 defined, 45, 93
 versus `get-psProvider` cmdlet, 102–103
 using to find available drives, 184–188
`get-psProvider` **cmdlet**
 defined, 45, 93
 versus `get-psDrive` cmdlet, 102–103
 using to find available providers, 183–184
`get-psSnapin` **cmdlet**, **90–91, 92**
`get-service` **cmdlet**
 compared with Services MMC snap-in, 73–74

 defined, 57, 93, 357
 examples, 125–127
 finding running services, 74–75, 138–140
 overview, 358–359
`get-traceSource` **cmdlet**, **95, 422–423**
`GetType()` **method**, **265, 276**
`GetTypeCode()` **method**, **265, 276**
`get-UICulture` **cmdlet**, **95**
`get-unique` **cmdlet**, **95**
`get-variable` **cmdlet**, **95, 230–231**
`get-wmiObject` **cmdlet**
 defined, 93, 502
 examples, 503–509
 exploring Windows system characteristics, 509–514
 finding WMI classes and members, 506–509
 list of parameters, 503
graphical user interface (GUI), **26**
greater-than operator, **79, 144, 223, 407**
greater-than-or-equal-to operator, **79, 144, 223**
`-groupBy` **parameter**
 format-table cmdlet and, 155, 158–159
 group-object cmdlet and, 158–159
`group-object` **cmdlet**
 defined, 95
 examples, 152
 overview, 83–85
 as verb-noun naming scheme example, 57–58
`-gt` **operator**, **79, 144, 223, 407**
GUI (graphical user interface), **26**

H

`help` **alias**, **111–112**
`help` **function**, **15**
`-Help` **PowerShell startup option**, **10–11**. *See also*
 `get-help` **cmdlet**
hidden files, finding, **442–443**
`-hideTableHeaders` **parameter**, **155, 158**
hiding table headers, **158**
`HKEY_CLASSES_ROOT` **hive**, **456**
`HKEY_CURRENT_CONFIG` **hive**, **456**
`HKEY_CURRENT_USER (HKCU)` **hive**, **456, 459, 461–464**
`HKEY_LOCAL_MACHINE (HKLM)` **hive**, **456, 458, 459–461**
`HKEY_USERS` **hive**, **456**
`$Home` **variable**, **61**
Host snapin, **90, 93**
`$Host` **variable**, **61**

hyphen (-)

- in assignment operator, 220
- as subtraction operator, 218, 219
- in unary operators, 226–227

I

- ieq operator, 224**
- if statement, 251–254**
- ige operator, 224**
- ignoreWhitespace parameter, 201**
- igt operator, 224**
- ile operator, 224**
- ilike operator, 224**
- ilt operator, 224**
- imatch operator, 224**
- import-alias cmdlet, 95, 108, 348**
- import-clixml cmdlet, 95**
- import-csv cmdlet, 95**
- include parameter**
 - for clear-variable cmdlet, 231
 - for get-childitem cmdlet, 191–194
 - for get-content cmdlet, 197, 200–201
 - for get-variable cmdlet, 230
 - for remove-item cmdlet, 167, 173–174
 - for remove-variable cmdlet, 232
 - for set-variable cmdlet, 228
 - for stop-service cmdlet, 178
 - for test-path cmdlet, 450
- includeChain parameter, 377**
- IndexOf() method, 265, 277–278**
- IndexOfAny() method, 265, 277–278**
- ine operator, 224**
- inotlike operator, 224**
- inotmatch operator, 224**
- \$Input variable, 61**
- InputFormat PowerShell startup option, 10**
- inputObject parameter**
 - for format-table cmdlet, 156
 - for measure-object cmdlet, 201
 - for out-file cmdlet, 449
 - for select-object cmdlet, 145
 - for where-object cmdlet, 142–143
- inputOption parameter, for trace-command cmdlet, 419**
- Insert() method, 265, 278**
- installing**
 - .NET Framework 2.0, 4–7
 - Windows PowerShell, 7–8, 368

Internet Explorer

- closing, 299
- as COM application, 294–299
- launching, 294–295
- manipulating in PowerShell, 294–299
- invoke-expression **cmdlet, 95**
- invoke-history **cmdlet, 92**
- invoke-item **cmdlet, 93**
- is operator, 79**
- IsNormalized() method, 265**
- isnot operator, 79**
- isValid parameter, 451**
- itemType parameter, 203**

J

- join-path cmdlet, 93, 450, 451–452**

L

- last parameter, 145, 148–150**
- LastIndexOf() method, 265, 278–279**
- LastIndexOfAny() method, 265, 278–279**
- le operator, 79, 144, 223**
- Length property, 241, 245, 266, 440**
- less-than operator, 79, 144, 223**
- less-than-or-equal-to operator, 79, 144, 223**
- like operator, 79, 144, 223**
- line parameter, 201**
- list parameter, 481**
- listenerOption parameter, for trace-command cmdlet, 419, 420**
- literalName parameter, 185**
- literalPath parameter**
 - for get-childitem cmdlet, 191
 - for push-location cmdlet, 341
 - for resolve-path cmdlet, 453
- logical operators, 218, 225–226**
- logName parameter, 480**
- logs. See event logs**
- looping, 256–261**
- ls alias, 52, 78**
- lt operator, 79, 144, 223**

M

- managed resources, 496–497**
- Management snapin, 90, 93–94**
- mapping drives, 305–306**
- match operator, 54, 79, 80, 108, 144, 223, 289**

-maximum parameter

`-maximum` **parameter, 201**
`$MaximumAliasCount` **variable, 61**
`$MaximumDriveCount` **variable, 61**
`$MaximumErrorCount` **variable, 61, 387**
`$MaximumFunctionCount` **variable, 61**
`$MaximumHistoryCount` **variable, 18, 61**
`$MaximumVariableCount` **variable, 61**
`measure-command` **cmdlet, 95**
`measure-object` **cmdlet, 95, 201–203**
`MemberInfo` **object, 324, 326**
`-memberType` **parameter, 320, 322, 328, 330**
memory
 Win32_CacheMemory WMI class, 511–512
 Win32_PhysicalMemory WMI class, 510–511
`-message` **parameter**
 for `write-debug` cmdlet, 414
 for `write-error` cmdlet, 400
metacharacter (^), 81, 85, 290, 357, 508
`MethodInfo` **object, 328, 329**
methods
 for `FileInfo` object, 436–437
 for finding information about .NET classes, 324–333
 for `String` class, 264–266
Microsoft Access, as COM application, 303–305
Microsoft Excel, as COM application, 302–303
Microsoft Management Console (MMC), 39
Microsoft Word, as COM application, 301–302
`Microsoft.Management.Automation.Core`
 namespace, 425
`Microsoft.PowerShell.Core` **snapin, 90, 91, 92**
`Microsoft.PowerShell.Host` **snapin, 90, 93**
`Microsoft.PowerShell.Management` **snapin, 90, 93–94**
`Microsoft.PowerShell.Security` **snapin, 90, 94**
`Microsoft.PowerShell.Utility` **snapin, 90, 95–96**
`-minimum` **parameter, 201**
minus sign (-)
 in assignment operator, 220
 as subtraction operator, 218, 219
 in unary operators, 226–227
`move-item` **cmdlet, 93**
`move-itemProperty` **cmdlet, 93**
multiplication operator (*), 218, 219
`$MyInvocation` **variable, 61**

N

`-name` **parameter**
 for `clear-variable` cmdlet, 231
 for `get-childItem` cmdlet, 191

 for `get-member` cmdlet, 60, 320, 322
 for `get-process` cmdlet, 177
 for `get-psDrive` cmdlet, 185
 for `get-variable` cmdlet, 230
 for `new-item` cmdlet, 203
 for `new-psDrive` cmdlet, 204
 for `new-variable` cmdlet, 229
 as positional parameter, 133
 for `remove-psDrive` cmdlet, 187
 for `remove-variable` cmdlet, 232
 for `set-variable` cmdlet, 228
 for `stop-service` cmdlet, 178
 for `trace-command` cmdlet, 419, 420

named parameters, 124–125

namespaces. See also providers

 defined, 338
 as drives, 45–51
 as providers, 346

naming cmdlets, 57–58

`-ne` **operator, 79, 223**

`-neg` **operator, 144**

`$NestedPromptLevel` **variable, 61**

.NET Framework 2.0

 as basis for PowerShell architecture, 34–35
 calling classes, 34
 case sensitivity issue, 18, 326–327
 class library, 36, 37–38
 COM object types, 306–308
 core library, 355–358
 creating .NET objects, 311–319
 downloading Software Developer's Kit, 310
 finding information about .NET classes, 324–333
 future emphasis on, 31
 information resources, 310–311
 installing, 4–7
 matching strings using regular expressions, 54
 PowerShell and, 309–311
 reflection, 324–333
 succinct PowerShell ways to manipulate objects, 56
 type system, 306–308

network shares, mapping drives to local machines, 305–306

`new` **verb, 57**

`new-alias` **cmdlet, 95, 108–109, 348**

`-newest` **parameter, 480**

`new-item` **cmdlet, 93, 203–204**

`new-itemProperty` **cmdlet, 93**

`new-object` **cmdlet**

`-comObject` parameter in, 43, 56, 294, 311
 creating new COM objects, 293–294

creating new .NET objects, 311–317
 defined, 95
 -strict parameter in, 294, 311
 -typeName parameter in, 294, 311, 313
 new-psDrive **cmdlet**, **93**, **204–205**
 new-service **cmdlet**, **93**, **357**, **360**
 new-timespan **cmdlet**, **95**
 new-variable **cmdlet**, **95**, **229–230**
 -noClobber **parameter**, **449**
 -NoExit **PowerShell startup option**, **10**
 -NoLogo **PowerShell startup option**, **10**
 -noNewLine **parameter**, **214**
 -NonInteractiver **PowerShell startup option**, **10**
nonterminating errors, **381**, **382**, **392**
 -NoProfile **PowerShell startup option**, **10**
 Normalize() **method**, **265**
 -not **operator**, **225**
 -notcontains **operator**, **79**
Notepad
 opening files in, 106
 saving .ps1 scripts in, 22
not-equal-to operator (-ne), **79**, **223**
 -notlike **operator**, **79**, **144**, **223**
 -notmatch **operator**, **79**, **144**, **223**
 \$now **variable**, **34–37**
 \$null **variable**, **62**
numbers, at beginning of expression, **63–64**

O

-object **parameter**, **214**
objects, measuring properties, **201–203**
 -off **parameter**, **408**
operators
 arithmetic, 217, 218–219
 assignment, 72, 217, 220–222
 comparison, 217, 222–224
 for filtering objects, 79
 logical, 218, 225–226
 overview, 217–218
 precedence, 219–220
 redirection, 18, 41, 407, 445–446
 special, 218
 unary, 218, 226–227
 for where-object cmdlet, 144
 -option **parameter**
 for new-alias cmdlet, 109, 348
 for new-variable cmdlet, 229
 for set-alias cmdlet, 348
 for set-variable cmdlet, 228
 for trace-command cmdlet, 419–420
 -or **operator**, **225**

out-default **cmdlet**, **96**
 out-file **cmdlet**, **96**, **449–450**, **492**
 out-host **cmdlet**, **96**
 out-null **cmdlet**, **96**
 out-printer **cmdlet**, **96**
 -outputBuffer **parameter**, **132**
 -OutputFormat **PowerShell startup option**, **10**
 -outputVariable **parameter**, **132**
 out-string **cmdlet**, **96**
 OverloadDefinitions **property**, **325**

P

PadLeft() **method**, **265**, **279–280**
 PadRight() **method**, **265**, **279–280**
parameters
 absence of, 117, 118
 for clear-variable cmdlet, 231
 common, 132, 397–399
 example, 117
 finding for cmdlets, 121–124
 for format-table cmdlet, 155–156
 for get-authenticodeSignature cmdlet, 378
 for get-childItem cmdlet, 191
 for get-eventlog cmdlet, 480–481
 for get-location cmdlet, 195, 196–197, 338–341
 for get-member cmdlet, 320–323
 for get-process cmdlet, 118–121
 for get-psDrive cmdlet, 185
 for get-service cmdlet, 358–360
 for get-traceSource cmdlet, 422
 for get-variable cmdlet, 230
 for get-wmiObject cmdlet, 503
 for join-path cmdlet, 451–452
 list, 132
 for measure-object cmdlet, 201
 named, 124–125
 for new-item cmdlet, 203
 for new-object cmdlet, 293–294, 311–317
 for new-psDrive cmdlet, 204
 for new-service cmdlet, 360–361
 for new-variable cmdlet, 229
 for out-file cmdlet, 449–450
 overview, 117
 for pop-location cmdlet, 344–345
 positional, 127–130
 for push-location cmdlet, 341–344
 for remove-item cmdlet, 166–167
 for remove-psDrive cmdlet, 187
 for remove-variable cmdlet, 232–233
 for resolve-path cmdlet, 453
 for restart-service cmdlet, 361

parameters (continued)

parameters (continued)

- for `select-object` cmdlet, 144
- for `set-authenticodeSignature` cmdlet, 377–378
- for `set-psDebug` cmdlet, 408–413
- for `set-service` cmdlet, 362
- for `set-traceSource` cmdlet, 422
- for `set-variable` cmdlet, 228
- for `start-service` cmdlet, 362–363
- for `stop-service` cmdlet, 363–364
- for `suspend-service` cmdlet, 364
- for `test-path` cmdlet, 450–451
- for `trace-command` cmdlet, 419–420
- variables as, 133–135
- wildcards in values, 125–127, 200
- for `write-error` cmdlet, 400–401
- for `write-psDebug` cmdlet, 414

parentheses ()

- role in parsing, 64
- using with methods, 264

parser, PowerShell, 63–69

`-passThru` parameter

- for `new-variable` cmdlet, 229
- for `pop-location` cmdlet, 344, 345
- for `push-location` cmdlet, 341
- for `set-location` cmdlet, 188, 190
- for `set-variable` cmdlet, 228
- for `stop-service` cmdlet, 178

Path environment variable, 54, 67, 353

path names

- fully qualified, 427–430
- in PowerShell, 426–433
- relative, 430–431
- running commands and, 431–433

`-path` parameter

- for `copy-item` cmdlet, 170
- defined, 166
- for `get-childItem` cmdlet, 191
- for `get-content` cmdlet, 196
- for `join-path` cmdlet, 452
- for `new-item` cmdlet, 203
- for `push-location` cmdlet, 341, 344
- for `remove-item` cmdlet, 166, 167
- for `resolve-path` cmdlet, 453
- for `test-path` cmdlet, 450

PathInfo object, 344, 345

paths, cmdlets for, 450–453

`-pathType` parameter, for `test-path` cmdlet, 450

percent sign (%), in assignment operator, 220, 221

period (.)

- for running PowerShell scripts, 23, 472
- when using PowerShell command mode, 64, 65, 66, 68

\$PID variable, 62

pipelines (|)

- default formatter, 151–154
- defined, 78
- examples, 13, 14, 19–20, 41–42
- grouping objects, 83–85
- layout, 19
- .NET objects in, 35–38, 57
- overview, 19–20, 78–85
- past limitations, 56–57
- role in sorting objects, 81–83
- separators in, 13, 78
- sequence of commands, 78–81
- symbol, 19, 78
- syntax, 19

plus sign (+)

- as addition operator, 218, 219, 248
- in assignment operators, 220
- in unary operators, 226–227

pop-location cmdlet, 93, 342, 344–345

positional parameters

- overview, 127–131
- `-property` parameter as, 156–157

PowerShell (Windows)

- approaches to parsing, 63–69
- architecture, 33–34
- backward compatibility, 51–56
- clearing screen in, 17–18
- closing, 10
- complete coverage forecast, 55
- current working folder, 23
- cycling through recently used commands, 18
- defined, 3
- enabling scripts, 9–10
- error handling in, 381–382
- exiting, 10
- exploring Windows systems, 69–76
- extended wildcards, 59–60
- extensibility, 51–56
- finding available commands, 11–14, 75–76
- installing, 7–8, 368
- loading console files, 90
- loading snapins, 90–91
- long term roadmap, 55
- minimizing default risk, 368–374
- need for, 25–31
- .NET Framework basis, 34–35, 309–311
- as object-based, 35–58
- path names in, 426–433
- repeating last-used command, 18
- response to errors, 58, 66–67

- starting, 8–11, 89–90
 - support for code debugging, 59
 - synthetic types, 306–308
 - system state information, 338–350
 - unsigned, 23
 - upgrade path to C#, 58
 - using to explore Windows registry, 458–461
 - working with file system, 425–453
 - PowerShell Analyzer, 41**
 - PowerShell **command, 89–90**
 - precedence, operator, 219–220**
 - preference variables, 115–116**
 - process parameter, 261**
 - profile files, 97–100**
 - \$Profile variable, 62**
 - profile.ps1 file, 98**
 - \$ProgressPreference variable, 62**
 - prompt function, 113**
 - prompt parameter, 212, 213, 216**
 - prompts. See command prompt**
 - properties**
 - expanding, 146–147
 - for FileInfo object, 438–439, 440, 441
 - measuring, 201–203
 - selecting, 145–146
 - property parameter**
 - for format-table cmdlet, 155, 156–157
 - for measure-object cmdlet, 201
 - for select-object cmdlet, 145–146
 - PropertyInfo **object, 330, 331**
 - PropertyItem **class, 152–153**
 - providers. See also drives; get-psProvider cmdlet**
 - defined, 45, 183
 - finding, 183–184, 425
 - in fully qualified path names, 427
 - namespaces as, 346
 - overview, 45–51
 - relationship to drives, 45, 186
 - removing items, 166–175
 - WMI, 499
 - .pscl extension, 90**
 - PSConsoleFile PowerShell startup option, 10**
 - psDrive parameter, 195, 338, 339**
 - \$PSHome variable, 62**
 - psHost parameter, for trace-command cmdlet, 419**
 - .psl extension, 22**
 - psProvider parameter**
 - for get-location cmdlet, 195, 338, 339
 - for get-psDrive cmdlet, 185
 - for new-psDrive cmdlet, 204
 - for remove-psDrive cmdlet, 187
 - push-location cmdlet, 94, 341–344**
 - \$PWD variable, 62**
- ## Q
- question mark (?)**
 - in \$? variable, 61
 - as wildcard, 59, 72, 125, 171
 - quotation mark, double (“)**
 - at beginning of expression, 64, 65
 - in commands, 71, 431–432
 - parameters and, 119–120
 - quotation mark, single (’), 119, 416, 431–432**
- ## R
- readCount parameter, 196**
 - read-host cmdlet, 96, 134–135, 211, 212–214**
 - recommendedAction parameter, 400**
 - recurse parameter**
 - for copy-item cmdlet, 170
 - for get-childitem cmdlet, 191
 - for new-item cmdlet, 167
 - for remove-item cmdlet, 166, 173–174
 - redirection operator (>>), 445–446**
 - redirection operator (>), 18, 41, 407, 445–446**
 - reflection, .NET, 324–333**
 - RegEdit utility. See Registry Editor**
 - registry**
 - adding new keys, 457–458
 - allowed types, 457
 - backing up, 456, 457
 - list of hives, 456
 - making changes to, 461–464
 - navigating by using PowerShell, 458–461
 - overview, 455–458
 - removing items, 166–175
 - Registry Editor**
 - accessing registry keys, 308
 - modifying ExecutionPolicy key value, 209–211
 - running, 456
 - Registry provider, 47–48, 184, 185, 346**
 - regular expressions, 289–291**
 - relative path names, 430–431**
 - remote machines, 513–514**
 - RemoteSigned execution policy, 208, 209, 211, 370, 372, 374**
 - Remove() method, 266, 280–281**
 - remove-item cmdlet, 94, 166–175**
 - remove-itemProperty cmdlet, 94**
 - remove-psDrive cmdlet, 94, 187**
 - remove-psSnapin cmdlet, 92**

remove-variable **cmdlet, 96, 232–233**
rename-item **cmdlet, 94**
rename-itemProperty **cmdlet, 94**
Replace() **method, 266, 281–282**
\$ReportErrorShowExceptionClass **variable, 62**
\$ReportErrorShowInnerException **variable, 62**
\$ReportErrorShowSource **variable, 62**
\$ReportErrorShowStackTrace **variable, 62**
repository, CIM, 499
-resolve **parameter, for join-path cmdlet, 452**
resolve-path **cmdlet, 94, 450, 453**
restart-service **cmdlet, 94, 357, 361, 364**
Restricted **execution policy, 207, 208, 209, 370, 372, 373**
resume-service **cmdlet, 94, 357**
-root **parameter, 204**

S

-scope **parameter**

for clear-variable cmdlet, 231
for get-psDrive cmdlet, 185
for get-variable cmdlet, 230
for new-psDrive cmdlet, 204
for new-variable cmdlet, 229
for remove-psDrive cmdlet, 187
for remove-variable cmdlet, 233
for set-variable cmdlet, 228

Script Library, WMI, 497–498

scripts, PowerShell

avoiding aliases in, 87
for COM objects, 43–44
enabling, 9–10, 68, 207–217
execution policies, 207–212, 370
minimizing default risk, 368–374
overview, 21–23, 41–43
path issues, 22, 23
path names and, 431–432
running, 22–23, 368, 431–432
saving .ps1 scripts in Notepad, 22
signing, 376–379
testing cmdlet combinations, 21–23

scroll bars, 18

security

execution policies and, 370–374
minimizing default risk, 368–374
running scripts, 368–374

Security log, 478

Security snapin, 90, 94

select-object **cmdlet**

defined, 96, 144
-expandProperty parameter in, 146–147

finding positional parameters, 127–128
-first parameter in, 148–150
-last parameter in, 148–150
overview, 144–145
properties, 144–145
-property parameter in, 145–146
for selecting most recent event log entries, 487–488
specifying properties to pass along, 104
-unique parameter in, 147–148
using to explore files, 439–442
select-string **cmdlet, 96**
semicolon (;), **20, 198**
services, list of cmdlets, **357**
Services MMC snap-in. See get-service **cmdlet**
set **verb, 57**
set-acl **cmdlet, 94**
set-alias **cmdlet, 55, 96, 109–111, 348**
set-authenticodeSignature **cmdlet, 94, 376, 377**
set-content **cmdlet, 22, 94**
set-date **cmdlet, 96**
set-executionPolicy **cmdlet, 9, 94, 371**
set-item **cmdlet, 94**
set-itemProperty **cmdlet, 94**
set-location **cmdlet, 46, 47–48, 49, 94, 188–190**
set-psDebug **cmdlet, 92, 408–413, 415, 417**
set-service **cmdlet, 94, 357, 362**
set-traceSource **cmdlet, 96, 422**
set-variable **cmdlet, 96, 228–229**
shell. See Windows PowerShell
shell aliases, system state information, **338, 346–349**
shell functions, system state information, **338, 349–350**
shell variables, system state information, **338, 350**
\$ShellId **variable, 62**
-showErrors **parameter, 156**
single quotation mark ('), **119, 416, 431–432**
signed scripts, **376–379**
slash (/)
in assignment operator, 220, 221
in division operator, 218, 219
snapins
loading, 90–91
Microsoft.PowerShell.Core snapin, 90, 91, 92
Microsoft.PowerShell.Host snapin, 90, 93
Microsoft.PowerShell.Management snapin, 90, 93–94
Microsoft.PowerShell.Security snapin, 90, 94
Microsoft.PowerShell.Utility snapin, 90, 95–96
viewing cmdlets available, 91–92
sorting pipeline objects, **81–83**

sort-object cmdlet, 81–83, 96, 104, 484
spelling mistakes, 349
Split() method, 266, 282–285
split-path cmdlet, 94, 450
square brackets ([])
 in PowerShell syntax for indicating .NET elements, 34, 268, 324
 in syntax for designating array elements, 236
 in wildcard searches, 59, 290, 429
-stack parameter
 for get-location cmdlet, 195, 338
 for push-location cmdlet, 341, 343
-stackName parameter
 for get-location cmdlet, 195, 338
 for push-location cmdlet, 341, 343, 344
starting Windows PowerShell, 8–10, 89–90
start-service cmdlet, 94, 357, 362–363
start-sleep cmdlet, 96
StartsWith() method, 266, 285
start-transcript cmdlet, 93
startup options, PowerShell, 10–11
-static parameter, 320, 322
-step parameter, 408, 411–413
stop-process cmdlet, 94, 175–178
stop-service cmdlet, 94, 178–181, 357, 363–364
stop-transcript cmdlet, 41, 93
StoreCountAndDate.ps1 script, 22
-strict parameter, 294, 311
strings. See also System.String class
 casting to other classes, 287–291
 comparing, 268–270
 copying, 267
Substring() method, 266
subtraction operator (-), 218, 219
-sum parameter, 201
Suspend option, 412, 413
suspend-service cmdlet, 94, 357, 364
switch statement, 254–256
syntax errors, 403–408
synthetic types, 306–308
system errors, 381–401
System log, 478
system state, 338–350
System.AppDomain class, 353, 354
System.Boolean object, 315
System.Collections.HashTable object, 249
System.____ComObject type, 306–307
System.DateTime class, 34, 35. See also DateTime object
System.Diagnostics.Process object, 34
System.Management.Automation.Cmdlet class, 381

System.Management.Automation.Core namespace, 185
System.Management.Automation.PSCmdlet class, 381
System.Reflection namespace, 324
System.Resources namespace, 356–357
System.Security.SecureString object, 213–214
System.String class
 Length property, 266
 list of methods, 265–266
 overview, 263–264
 working with methods, 267–287
System.Type class, 333

T

Tab key, using to complete commands, 76–77, 112–113, 443–444
-targetObject parameter, 400
tee-object cmdlet, 96
terminating errors, 381, 382
test-path cmdlet, 94, 450–451
ThrowTerminatingError() method, 381
time and date examples, 34–38
-timeStampServer parameter, 377
ToCharArray() method, 266, 285–286
ToLower() method, 266, 286–287
ToLowerInvariant() method, 266, 286–287
ToString() method, 266
-totalCount parameter, 197, 198
ToUpper() method, 266, 287
ToUpperInvariant() method, 266, 287
-trace parameter, 408–410
trace-command cmdlet
 defined, 96, 419
 examples, 420–421
 multiple positional parameters, 129–131
 parameters for, 419–420
tracing, 418–423
trap statement, 392–397, 410–411
Trim() method, 266, 287
TrimEnd() method, 266, 287
TrimStart() method, 266, 287
\$True variable, 62
typed arrays, 239–240
-typeName parameter, 294, 311, 313

U

unary operators (- and -), 226–227
underscore (\$) variable, 61, 71
-unique parameter, 145, 147–148

Unrestricted **execution policy**, **208**, **209**, **370**, **372**, **373**, **374**

update-formatData **cmdlet**, **96**, **162**

update-typeData **cmdlet**, **96**, **162**

URI **object**, **287–288**

Utility **snapi**n, **90**, **95–96**

V

-value **parameter**

for new-item cmdlet, 203

for new-variable cmdlet, 229

for set-variable cmdlet, 228

-valueOnly **parameter**, **230**

Variable **provider**, **184**, **185**, **346**

variables

assigning values to, 228–229, 352

automatic, 60–62

clearing value, 231–232

creating, 229–230

as drives, 49, 346, 350

environment, 351–353

error-related, 345–346, 383–392

manipulating, 227–233

overview, 49–51

as parameters, 133–135

preference, 115–116

removing, 232–233

retrieving, 230–231

-verbose **parameter**

as common parameter, 132, 165, 166

for stop-service cmdlet, 181–182

\$VerbosePreference **variable**, **62**

verbosity, **85–87**

verbs, in cmdlets, **13**

-view **parameter**, **156**

visible **property**, **295–296**

Visual Studio **2005**, **310**

vertical bar (|)

default formatter, 151–154

defined, 78

examples, 13, 14, 19–20, 41–42

grouping objects, 83–85

layout, 19

.NET objects in, 35–38, 57

overview, 19–20, 78–85

past limitations, 56–57

role in sorting objects, 81–83

separators in, 13, 78

sequence of commands, 78–81

symbol, 19, 78

syntax, 19

W

-wait **parameter**, **197**, **199**, **200**

\$WarningPreference **variable**, **62**

WBEM (Web-based enterprise management), **30**

-whatIf **parameter**

for clear-variable cmdlet, 231

as common parameter, 166

defined, 132, 165

for new-item cmdlet, 203

for new-psDrive cmdlet, 204

for new-variable cmdlet, 229

overview, 39

for remove-item cmdlet, 170–171

for remove-psDrive cmdlet, 187

for remove-variable cmdlet, 233

set-authenticodeSignature cmdlet and, 377

for set-executionPolicy cmdlet, 371

for set-variable cmdlet, 228

stop-process cmdlet and, 175–178

stop-service cmdlet and, 178–180

\$WhatIfPreference **variable**, **62**

where-object **cmdlet**

defined, 92, 137

for filtering event log entries, 484–487, 489, 490

filtering overview, 138–140

-filterScript parameter in, 142–143

finding aliases for, 138, 347–348

-inputObject parameter in, 142–143

list of operators, 144

-match operator and, 290–291, 347

multiple filter tests, 138–140

operators for use in filtering, 79–81, 144

overview, 41, 137–138

parameters for, 142–143

simple filtering, 138, 139–140

together with GetProperties() method, 330

using to filter processes, 71–72, 73

using to filter services, 74–75

while **statement**, **258–259**

-width **parameter**, **449**

wildcards

asterisk as, 12, 48, 49, 59, 72, 73, 125, 171–173, 200, 202

extended, 59–60

for filtering processes, 72–73

in parameter values, 125–127, 200

question mark as, 59, 72, 125, 171

for removing items, 171–173

Win32_CacheMemory **WMI class**, **511–512**

Win32_Date **WMI class**, **497**

Win32_PhysicalMemory **WMI class, 510–511**

Win32_Process **WMI class, 497, 508–509**

Win32_Processor **WMI class, 497, 509–510**

Win32_Service **WMI class, 497, 512–513**

windir **environment variable, 465–466**

Windows Explorer, 368

Windows Management Instrumentation (WMI)

accessing, 56

CIM Studio tool, 499–500

defined, 30–31

managed resources layer, 496–497

overview, 496–502

relationship to PowerShell, 38–39, 495

role of aliases, 51–52

tools, 499–502

WMI consumers layer, 499

WMI infrastructure layer, 497–499

WMI Object Browser, 500–502

Windows PowerShell

approaches to parsing, 63–69

architecture, 33–34

backward compatibility, 51–56

clearing screen in, 17–18

closing, 10

complete coverage forecast, 55

current working folder, 23

cycling through recently used commands, 18

defined, 3

enabling scripts, 9–10

error handling in, 381–382

exiting, 10

exploring Windows systems, 69–76

extended wildcards, 59–60

extensibility, 51–56

finding available commands, 11–14, 75–76

installing, 7–8, 368

loading console files, 90

loading snapins, 90–91

long term roadmap, 55

minimizing default risk, 368–374

need for, 25–31

.NET Framework basis, 34–35, 309–311

as object-based, 35–58

path names in, 426–433

repeating last-used command, 18

response to errors, 58, 66–67

starting, 8–11, 89–90

support for code debugging, 59

synthetic types, 306–308

system state information, 338–350

unsigned, 23

upgrade path to C#, 58

using to explore Windows registry, 458–461

working with file system, 425–453

Windows Script Host (WSH), 30, 299–301

Windows systems

exploring processes with `get-process` cmdlet, 69–71

exploring services with `get-service` cmdlet, 73–75

exploring with `get-wmiobject` cmdlet, 509–514

filtering processes by using wildcards, 72–73

filtering processes with `where-object` cmdlet, 69–71

finding running processes by using `get-process` cmdlet, 69–71

WMI consumers, 499

WMI Object Browser, 500–502

WMI providers, 499

WMI Script Library, 497–498

WMI (Windows Management Instrumentation)

accessing, 56

CIM Studio tool, 499–500

defined, 30–31

managed resources layer, 496–497

overview, 496–502

relationship to PowerShell, 38–39, 495

role of aliases, 51–52

tools, 499–502

WMI consumers layer, 499

WMI infrastructure layer, 497–499

WMI Object Browser, 500–502

Word, as COM application, 301–302

`-word` **parameter, 201**

`-wrap` **parameter, 155**

`write-debug` **cmdlet, 96, 413–418**

`write-error` **cmdlet, 96, 396, 400–401**

`WriteError()` **method, 381**

`write-host` **cmdlet**

defined, 96

examples, 211, 394–395, 418

overview, 214–217

using to display expression results, 66

versus `write-debug` cmdlet, 413

`write-output` **cmdlet, 96**

`write-progress` **cmdlet, 96**

`write-verbose` **cmdlet, 96**

`write-warning` **cmdlet, 96**

WSH (Windows Script Host), 30, 299–301

X

XML files, console files as, 90

XML object, 289