

# *index*

---

## **Symbols**

- symbol 41
- ./ prefix 45
- .\ prefix 45
- (see!) argument 228
- @ symbol 37, 95
- @( ... ). *See* array subexpressions
- & operator 415
- & symbol 37
- # character 58
- % alias 684
- + symbol 59
- += operator 685
- > operator 181
- >> operator 181
- \$\_ variable 37, 217–218, 224, 265, 572, 677
- \$? variable 564–565, 621, 794
- \$( ... ). *See* subexpressions
- \$count variable 379
- \$error variable 560–561

## **Numerics**

- 2> operator 181
- 2>&1 operator 181
- 2>> operator 181
- 32-bit applications, 64-bit applications vs. 793
- 32-bit operating systems 499
- 64-bit applications, vs. 32-bit applications 793
- 64-bit operating systems 499

## **A**

- abstraction 8
- access controls, and endpoints 533–535
- access restriction 543
- access to current tab 623
- accessing COM objects 767
- AccessMode module 388
- accidental code injections 439
- accidental execution 276
- accumulated results, in variables 208
- acronyms 37
- Action parameter 855
- Action parameter 842
- Action property, on PSBreakpoint object 654
- action script block, in breakpoints 658
- actions
  - asynchronous events 854
  - running upon module removal 389–391
- Actions.Create() method 789
- Active Directory Services Interface (ADSI) 75
- active scope 652
- ActiveScript engine 783
- ActiveX Data Objects (ADO) 75
- adaptation 761
  - extending objects 401
  - of existing member 427
  - synthetic members 402
- adaptation layer, COM 762
- adapter mechanism 77
- add members 401
- add property 405

- Add() method 261, 746
- add/remove software, appwiz.cpl command 597
- add\_Click() method 850, 855
- addition operator 113–116
- addition, with hash tables 115
- Add-Member cmdlet 423, 425
  - addressing, remoting target 518–520
    - connecting to nondefault ports 518–519
    - DNS names and IP addresses 518
    - proxy servers 520
    - using URI 519
  - using to extend objects 402–408
    - adding AliasProperty members 404–405
    - adding NoteProperty members 405–406
    - adding ScriptMethod members 406–407
    - adding ScriptProperty members 407–408
- Add-Type cmdlet 710, 723
  - compiling code with 440–446
    - defining new .NET class with C# language 440–442
    - defining new enum types at runtime 442–443
    - dynamic binary modules 443
  - defining new types with 729–739
    - creating Singleton member definitions 730–734
    - interoperation with P/Invoke signatures 734–735
    - Path parameter set 737–739
    - TypeDefinition parameter set 736–737
  - example 731
    - IgnoreWarnings parameter 731
    - Language parameter 730
    - Path parameter set 737
    - singleton member definitions 730
    - UsingNamespace parameter 730
- Admin Script PowerShell IDE 752
- Administrator privileges 517
- Administrators group 517
- administrators, in other domains
  - enabling remoting for 517–518
- ADO (ActiveX Data Objects) 75
- ADSI (Active Directory Services Interface) 75
  - After parameter 601
- agentless monitoring using remoting 462
- aggregating events with GROUP
  - keyword 874–875
- algorithms
  - comparison 128
  - module search 329–330
- Alias attribute, creating parameter aliases
  - with 303–305
- Alias parameter 334
- alias property 404
- alias: drive 667
- aliased members 405
- aliases
  - and elastic syntax 47–50
  - listing definitions 667
  - predefined 48
  - why use parameter alias 49
- AliasesToExport element 370
- AliasProperty members, adding to
  - objects 404–405
- All flag 351
- AllMatches switch 692
- Allow Automatic Configuration of Listeners
  - policy 453
- AllowClobber parameter 480
- AllowEmptyCollection attribute 306
- AllowEmptyString attribute 306
- AllowNull attribute 306
- AllowNullExample function 306
- AllSigned execution policy 899
- alternate layouts, in ISE 609
- Alt-F4 shortcut, ISE 611
- AMD64 processor 500
- Anchor property 749
- and operator 149
- anonymous code 658
- anonymous filter 266
- anonymous functions 398
- APIs (application programming interfaces) 7
  - debugging with host 580–593
    - catching errors with strict mode 582–584
    - Set-StrictMode cmdlet 584–589
    - static analysis of scripts 589–593
  - interoperation with 69
- AppActivate() method 778
- AppDomain class 725

- appending error records 183
- Application Data directory 668
- Application event log 879
- Application log 602
- application output redirection, in ISE 617
- Applications list, controlling access to 540
- Applications property, From
  - \$ExecutionContext.SessionState 539
- applications. *See also* native commands
  - issues in ISE 616
  - managing with COM 779–783
    - looking up word definitions using Internet Explorer 779–781
    - spell checking using Microsoft Word 781–783
  - native executables and PowerShell paths 666
- appwiz.cpl command 597
- \$args scalar 238
- \$args variable 241, 278, 289
  - passing arguments using 237–239
  - simplifying processing with multiple assignment 240
- \$args.length 246
- argument processing, by ForEach-Object cmdlet 227–228
- ArgumentList parameter 358
- arguments 60, 174
  - handling variable numbers of 245–246
  - passing to scripts 278–280
  - passing using \$args variable 237–239
  - printing 222
  - specifying to switch parameters 250–252
  - vs. parameters 39
- Arguments parameter 813
- arithmetic operation 24
- arithmetic operators 112–119
  - addition 113–116
  - multiplication 116–117
  - subtraction, division, and modulus 117–119
- [array] class 170
- array assignment 94
- Array class 434
- array concatenation 113
- array literals 91
- array operations
  - convert array to string 677
  - determining unique members 684
  - discarding empty elements 683
  - grouping by key property 686
  - indexing with negative values 685
  - number of occurrences 684
- array operators 112, 162–173
  - comma 162–165
  - indexing and slicing 167–170
  - multidimensional 171–173
  - range 165–167
- array slicing 197, 603
- array subexpression operator 158
- ArrayList class 261
- arrays 91–96
  - 0 origin 24
  - as reference types 93–94
  - collecting pipeline output as 91–92
  - concatenation of 25
  - empty arrays 94–96
  - indexing of 92
  - multiplication of 116
  - of characters 435
    - converting to string 406
    - string as 406
  - of indexes 172
  - polymorphism in 92–93
  - presentation in tracing 641
  - resizing 259
  - subexpressions 160–162
- as operator 152, 154
- As parameter 712
- ASCII encoding 679
- AsCustomObject parameter 417
- AsCustomObject() method 382
- AsJob parameter 481, 489–490, 492, 802–803, 882
- AsSecureString parameter 917
- assemblies 721–725
  - default 722
  - loading
    - dynamically 722–723
    - using Load() method 724
    - with Add-Type 723
    - with System.Reflection.Assembly class 723–725

- pros and cons of dynamic linking 721
- versioning and 721–722
- assembly manifest 721
- Assembly property 444
- AssemblyName parameter 759
- assets
  - definition 893
  - threats, mitigation, and 893–897
    - authentication, authorization, and roles 894–895
    - avoiding lawn gnome mitigation 893–894
    - blacklisting/whitelisting 894
    - code injection 896–897
    - input validation 895–896
- assignable elements 122
- assignable expressions 155
- assignment expressions
  - as value expressions 123
  - syntax 120
- assignment operators 119–124, 187
  - as value expressions 123–124
  - multiple 120–123
- asynchronous events 853–855
  - .NET 855–860
    - managing subscriptions 859–860
    - writing timer event handler 856–859
  - eventing cmdlets 854–855
  - handling with scriptblocks 860–863
    - automatic variables 860
    - dynamic modules and event handler state 862–863
  - subscriptions, registrations, and actions 854
- attacks, defined 892
- authentication
  - authorization, roles, and 894–895
  - connecting user 514–518
    - enabling remoting for administrators in other domains 517–518
    - forwarding credentials in multihop environments 515–517
  - target computer 511–514
  - to server 511
- Authentication parameter 803
- Author element 367
- authorization, authentication, roles, and 894–895

- automatic type conversion 154
- automatic variables 224, 437, 860
- automatically generated help fields 315
- automating applications 765
- automation interfaces 761, 763
- Automation model 765
- AutoReset property 857
- AutoReset timer property 728
- AutoSize switch 65–66
- AWK 224

## B

- background color, setting in ISE 625
- background jobs 481–493
  - cmdlets 483–487
    - removing jobs 487
    - waiting for jobs to complete 486
  - commands 483
  - multiple 487–489
    - running in existing sessions 492–493
    - starting on remote computers 489–492
- background processing, using ISE tabs 628
- backquote character 52
- backslash character 54
- backtick character 52, 669
- base class 423
- bash shell 6, 38
- bash, Windows 4
- Basic type 514
- bastion server 515
- Before parameter 601
- begin blocks, functions with 266–267
- begin clause 61, 397, 419
- begin keyword 266
- Begin() function 420
- BeginInit() method 726
- BeginInvoke() method 726
- begin-processing clause 62
- binary data operations 674
- binary modules 353
  - creating 355–357
    - dynamic 443
    - nesting in script modules 357
    - vs. snap-in modules 354–355
- binary operator 179, 225

- binary tree 163
- binding
  - event action 857–858
  - objects, data and methods 400
  - parameters pipelines and 62–63
- bitmap files
  - dumping 676
  - working with 677
- BitsTransfer module 363
- bitwise operators 148–150
- blacklisting/whitelisting 894
- block of code, trap statement scope 570
- .BMP files. *See* bitmap files
- boilerplate preamble 551
- BOM (Byte order mark) 613
- [bool] parameter 252–253
- [bool] type 107
- [bool] type accelerator 252
- Boolean parameters 252–257
- bootstrap remoting 453
- bottom-tested variant, of while loop 204
- bound variables 414
- boundaries 543
- Bourne shell 9
- bp function 647
- braces, mandatory in statement lists 201
- branching 215
- break keyword 571, 655
- break statement 212–216, 220
- breakpoint command 646–647
- breakpoint ID 654
- breakpoint line, highlight in ISE 649
- breakpoints
  - conditional breakpoints 654
  - objects 653–656
  - setting
    - on commands 656–657
    - on variable assignment 657–658
- browser cache path issues 668
- browser windows
  - management module 770–777
    - adding graphical front end 773–774
    - defining control actions 775–776
    - XAML 774–777
  - managing using COM 767–770

- building objects, in PowerShell 400
- built-in \$PSHOME variable 333
- built-in commands 42
- built-in type conversion 104
- Button control 755
- Button object 850, 855
- by reference 94
- bypass security boundary, using scriptblocks 541
- bypass type adapter 824
- byte order mark (BOM) 613

## C

- C# language 201, 261, 440–442, 570, 575
- \$c2 variable 417
- caching security 905
- calc process 822, 842, 869
- calculated field 411
- calculated module exports 346–347
- calculated property 805
- call (& ) operator 286
- call operator 537, 569
  - executing script block with 438
  - script blocks 396
- CallExit function 280
- calling functions 242
- calling modules
  - defining modules vs. 384–387
    - accessing defining module 385–386
- Call-JScript function 785
- Call-VBScript function 784
- CancelTimeout parameter 526
- candidate conversion 106
- CanInvoke property 626–627
- canonical aliases 673
- captures 563
- capturing error objects 555
- capturing error records 556, 560
- capturing errors 566
- capturing script output 593
- case flag 218
- case option 218
- case-insensitive 116
- case-insensitive keywords 199
- casesensitive option 217
- CaseSensitive parameter 689

- case-sensitivity, comparison operators
  - and 127–128
- cast notation 96, 185
- casting 153
  - strings to arrays of characters 406
  - to void 156
- catch keyword 575
- categories, of COM objects 794
- CategoryInfo property 557
- cd alias, For Get-Location 665
- ceq operator 125
- Certificate snap-in 908
- certificates
  - exporting 913
  - self-signed 905–909
  - using to sign scripts 909–912
    - setting up test 909–910
    - signing test 910–912
  - testing integrity 912
- Certmgr.exe (Certificate Manager tool) 913
- chained cast 102
- change-tolerant scripts 137
- [char] class 677
- char array 729
- character classes 682
- character encodings 674, 679
- checksum function 680
- child jobs
  - and nesting 489–490
  - with Invoke-Command cmdlet 490–492
- ChildJob property 491
- ChildJobs property 490
- Church, Alonzo 394
- CIM (Common Information Model)
  - namespaces 8, 807–810
- CIM\_Process class 799
- CIM\_Process.Terminate() method 799
- class definition, removing 432
- class keyword, implementing 431
- Class parameter 807, 818
- class-based event registrations, Microsoft
  - WMI 867–870
  - using WIN32\_ProcessTrace events 868–870
  - verifying that events fired 870
- classes
  - defined 12, 429
  - Microsoft WMI, using Get-WmiObject cmdlet
    - to find 806–807
- cleaning up breakpoints 656
- Clear() method 636
- Clear-EventLog cmdlet 597
- Clear-Item cmdlet 673
- Click event 746
- Click() method 746, 757
- clipboard, Windows 17
- clippy function 268
- CliXML format 632
- Clixml format 717
- clobbering output 184
- Clone() method 90
- Close() method 746
- closures 414–417, 638
- CLRVersion element 367–368
- Cmd.exe 281
- cmd.exe 4, 187
  - /c option 565
  - and PSPaths 666
  - and transcript files 596
  - command completion 20
  - command equivalents 673
  - convenience aliases 673
  - security 889
- Cmdlet parameter 334
- cmdlet Verb-Noun syntax 47
- CmdletBinding attribute 289, 296
  - \$PSCmdlet variable 293
  - ConfirmImpact property 293
  - DefaultParameterSetName property 293
  - SupportsShouldProcess property 290–293
- cmdlets 13, 42–43, 483
  - background jobs 483–487
    - removing jobs 487
    - waiting for jobs to complete 486
  - commands and 38–42
  - flow control using 223–231
    - ForEach-Object cmdlet 223–228
    - Where-Object 228–231
  - formatting and output 64–70
  - Microsoft WMI 801–824
    - common parameters 802–803
    - Get-WmiObject cmdlet 804–813

- cmdlets (*continued*)
  - invoke-WmiMethod cmdlet 819–822
  - remove-WmiObject cmdlet 822–824
  - Set-WmiInstance cmdlet 813–819
  - variable 188–193
    - getting and setting options 189–191
    - indirectly setting 188–189
    - names vs. values 192–193
    - using PSVariable objects as references 191–192
  - verb-noun pairs 13
  - WS-Man 831–832
    - invoking methods with Invoke-WSManAction cmdlet 841–846
    - retrieving management data with Get-WSManInstance cmdlet 832–839
    - updating resources using Set-WSManInstance cmdlet 840–841
  - WSMan implementation 509–511
    - establishing remote connection 510–511
    - testing connections 510
- CmdletsToExport element 370
- code
  - compiling with Add-Type cmdlet 440–446
  - defining new .NET class with C# language 440–446
  - defining new enum types at runtime 442–443
  - dynamic binary modules 443–446
  - example
    - basic expressions and variables 23–25
    - navigation and basic operations 22–23
- code execution 893
- code injection 896–897, 924
- code injection attacks 439
- Code reuse role 324
- CodeMethod type 403
- CodeProperty type 403
- code-signing 907
- coding exercise 890
- collection comparisons 129
- collection type 108
- collections 24
  - of numbers 263
  - of objects 215
  - using comparison operators with 129–131
- colon character, in variable names 186
- color names 625
- COM (Component Object Model) 760–796
  - adapter issues and limitations 793
  - automating Microsoft Windows with 764–777
    - browser window management module 770–777
    - managing browser windows using COM 767–770
    - Shell.Application class 765–766
  - classes, identifying and locating 762–764
  - in ISE 618
  - Interop assembly 794
  - issues with 793–796
    - 64-bit vs. 32-bit applications 793
    - interop assemblies, wrappers, and typelibs 793
    - threading model problems 793
  - managing applications with 779–783
    - looking up word definitions using Internet Explorer 779–781
    - spell checking using Microsoft Word 781–783
  - Microsoft Windows Task Scheduler 786–793
    - Schedule.Service class 786–787
    - tasks 787–793
  - objects 761–762
  - WScript.Shell class 777–779
  - WSH ScriptControl class 783–786
    - embedding JScript code 785–786
    - embedding VBScript code 784
- comma operator 162–166, 197
- command aliases, for DOS and UNIX 22
- command completion 13, 20–21
- command discovery 394
- command editing, in console host 16
- command history 6
- command information 399
- command input editor, ISE 608
- command interpreter, vs. shell 6
- command line debugger 638
- command lines 6–7, 267
- command mode 54, 59
- command not found error, and private commands 540

- command not found exception 422
- command output, parsing using regular
  - expressions 136–137
- command pane 608–609
- Command parameter 285
- command path, managing 898
- command resolution, in constrained sessions 548
- command switches, using switch parameters to
  - define 248–252
- command type 396
- command visibility
  - controlling 536–539
  - in remoting 535
- CommandInfo object 382, 395–396, 476, 536
- CommandInfo, FunctionInfo subclass 399
- command-line debugging 652–660
  - breakpoints
    - objects 653–656
    - setting 656–658
  - debugger limitations and issues 658–660
- command-line editing 16
- command-mode parsing 54–56
- CommandPaneUp, ISE menu item 625
- commands
  - anatomy of 38
  - and cmdlets 38–42
  - background jobs 483
  - break-down of 39
  - breakpoint 646–647
  - built-in 42
  - categories of 42–46
    - cmdlets 42–43
    - functions 43
    - native commands 44–46
    - scripts 44
  - considerations when running
    - remotely 493–501
  - executables 495–496
  - processor architecture 498–501
  - profiles and remoting 494–495
  - reading and writing to console 496–497
  - remote output vs. local output 497–498
  - remote session startup directory 494
- converted 40
- determining if errors in 564–566
- executing in ISE 614–616
  - executing other in debug mode 651
  - first element of 39
  - invoking 394–396
  - native, issues with 616–617
  - no concurrent in session 468
  - offset in pipeline 559
  - prefixing 45
  - proxy, creating with steppable
    - pipelines 420–423
  - running in traditional shells 209
  - setting breakpoints on 656–657
  - with built-in remoting 448–449
- comma-separated values 143
- comment block 904
- comment syntax 58–60
- comments
  - comment-based help 316–318
  - tags used in 318–321
    - .COMPONENT help 320
    - .EXTERNALHELP help 320–321
    - .FORWARDHELPCATEGORY help 320
    - .FORWARDHELPTARGETNAME help 320
    - .LINK help 320
    - .PARAMETER help 319
    - .REMOTEHELPRUNSPACE help 320
- Common Information Model (CIM) 8, 709
- ComObject parameter 761
- CompanyName element 367
- comparison operators 124–131
  - and case-sensitivity 127–128
  - case sensitivity factor 124
  - design rational 125
  - left-hand rule 126
  - scalar 125–127
    - basic comparison rules 126
    - type conversions and comparisons 126–127
  - using with collections 129–131
- compile time 436
- compiled script 438
- compile-time error 185
- compiling code, with Add-Type cmdlet 440–446
  - defining new .NET class with C#
    - language 440–442
  - defining new enum types at runtime 442–443
  - dynamic binary modules 443–446

- complete statement 56
- compiled programs 722
- .COMPONENT help tag 320
- Composing solutions role 324
- composite management applications, mash-ups 324–325
- compound assignment operators 120
- compression, of properties in serialization 509
- ComputerName parameter 602, 805
- COMtools.psm1 module 770–771
- Concatenate parameter 513
- concatenated statements 80
- concatenation, of arguments 240
- concrete system resources 799
- concurrency, adding to remoting
  - examples 455–457
- concurrent connections
  - fan-in remoting 528
  - fan-out remoting 528
  - limiting 522
- concurrent operation
  - using remoting 455
  - with jobs 487
- concurrent sessions, in ISE 607
- condition part, of if statement 202
- condition test 206
- conditional breakpoints 654
- conditional matches 219
- conditional statement 199–203
- configuration script
  - boilerplate 551
  - updating 549
- configuration updates, scope of change 549
- ConfigurationName parameter 530
- configurations 530–535
  - creating custom 531–533
    - registering endpoint configuration 532–533
    - session configuration 531–532
  - setting security descriptors on 534–535
- Configuring the environment role 324
- Confirm flag 292
- ConfirmImpact property 293
- \$ConfirmPreference preference variable 293
- connection patterns, remote services 527–530
  - fan-in 528–530
  - fan-out 527–528
- connections
  - establishing remote 510–511
  - persistent, remoting sessions and 462–473
  - testing 510
- Connect-WSMan cmdlet 509, 514
- Connect-WSMan command 525
- console
  - reading and writing to 496–497
  - threading differences between ISE and 618
- console APIs 735
- console applications, in ISE 616
- console editing features 16
- console host 14–16
- console objects 616
- Console.ReadLine API 496
- Console.WriteLine API 496
- [ConsoleColor] parameter 310
- constant expression folding, in PowerShell 562
- constant expressions 562
- constant variables 184
- constrained application environment, in remoting 530
- constrained endpoint 543
- constraining
  - execution environments 543
  - sessions 535–543
    - controlling command visibility 536–539
    - setting language mode 539–543
- construction elements, module manifests 370–375
  - loader elements 371–373
  - module component load order 374–375
- Constructors type 106
- containment operators 130–131
- contains operator 130–131
- content elements, module manifests 375–376
- context properties, searching with 692
- context-sensitive keywords 199
- continue keyword 571
- continue statement 212–215, 220, 571
- contract parameter 305
- control actions, defining 775–776
- control flow, in trap statement 572
- control structures 393
- control transfer, in trap statements 570
- Controls member 751
- convenience aliases 48

- conventions, used in examples 15
- conversion and precision 73, 102
- conversion error 185
- conversion rule 113
- conversions, of types 101–109
  - .NET-based custom 104–107
  - built-in 104
  - in parameter binding 107–109
  - overview 101–104
- ConvertFrom-SecureString cmdlet 918
- ConvertTo-SecureString cmdlet 918
- ConvertTo-Xml cmdlet 711–714
- copying elements 114
- copying, into Windows clipboard 17
- Copy-Item cmdlet 239, 673
  - LiteralPath parameter 670
- Copyright element 367
- Core cmdlet noun 664
- core cmdlets 664–665, 673
- Count property 131, 238
- \$count variable 342, 379, 416
  - using with Get-Count and Reset-Count 338–339
  - variables and aliases exported 345
- counter module 343, 382
- counter2 module 350, 353
- counting loop 205
- countUp function 350
- Create() method 627, 776, 819, 828, 842
- CreateElement() 695
- CreateProcess() API 898, 922
- credential dialog 621
- credential information 452
- Credential parameter 511
- credentials
  - and scheduled tasks 789–792
  - forwarding in multihop environments 515–517
  - passing securely 516
- CredSSP (Credential Security Service Provider) 509, 515–517
- CredSSP type 515
- critical operations 566
- Critical type 599
- cryptography 889
- CSV file 188
- Ctrl-Break 616
- Ctrl-C 616
- Ctrl-F5 637
- Ctrl-N 611, 625
- Ctrl-O 612
- Ctrl-R 611
- CUA (Common User Access) 17, 610
- currency symbol 181
- current directory 208
- current execution line, displayed in debugger 649
- Current property 264
- current scope 280
- current state 769
- current working directory 111
- CurrentDomain property 725
- CurrentFile property, ISE object model 623
- CurrentPowerShellTab property, ISE Object model 623
- custom drives 665
- custom hosts 546
- custom menu items
  - hotkey collisions 634
  - removing 636
  - updating 635
- custom objects 393, 417–418
- custom remoting endpoint 543
- custom services 527–552
  - access controls and endpoints 533–535
  - configurations 530–531
  - constrained execution environments 543
  - constraining sessions 535–543
    - controlling command visibility 536–539
    - setting language mode 539–543
  - remote service connection patterns 527–530
    - fan-in 528–530
    - fan-out 527–528
- CustomClass keywords 428–433
- customizing ISE
  - setting font and font size 624
  - using object model 621

**D**

- Danom virus 890–891
- data abstraction 428
- Data General 9

- data structure, example of 122
- data, processing 25–30
  - problem-solving pattern 29–30
  - selecting properties from objects 27–28
  - sorting objects 25–27
  - with ForEach-Object cmdlet 28–29
- DateTime objects 11, 128, 585, 805
- DayOfWeek property 406
- DCE/RPC (Distributed Computing Environment/Remote Procedure Call) 802
- DCOM protocol 802
- (DDL) dynamic link library 42, 721
- dead objects 803
- Debug menu
  - Disable All Breakpoints item 649
  - Display Call Stack 649
  - Enable All Breakpoints item 649
  - List-Breakpoints item 649
  - Remove All Breakpoints item 649
  - Run/Continue item 648
  - Step Into item 648
  - Step Out item 648
  - Step Over item 648
  - Stop Debugger item 648
  - Toggle Breakpoint item 649
- Debug mode 651
- debug mode prompt 649
- debug statements 259
- debugger shortcut commands 651
- debugging 63
  - command-line 652–660
    - breakpoints 653–658
    - debugger limitations and issues 658–660
  - problems in function output 259–262
  - scripts 638–647
    - nested prompts and Suspend operation 643–647
  - Set-PSDebug cmdlet 638–643
  - v2 debugger 647–652
  - with host APIs 580–593
    - catching errors with strict mode 582–584
    - Set-StrictMode cmdlet 584–589
    - static analysis of scripts 589–593
- [decimal] value 75
- declaring parameters 241
- declaring types 73
- decrement operator 155
- default assemblies 722
- default clause 216
- default file encoding 676
- default presentation, overriding 558
- default prompts, in remote sessions 470
- default remoting port (HTTP) 518
- default remoting port (HTTPS) 518
- default session configuration, creating 534
- Default type 514
- default values, initializing function parameters
  - with 246–247
- default, security by 897–898
  - disabling remoting 897
  - managing command path 898
  - no execution of scripts 897–898
  - notepad 897
- DefaultParameterSetName property 293
- Definition property 477, 540, 547
- definitions
  - managing in session 267–269
  - of words, looking up using Internet Explorer 779–781
- delegation
  - and delegates 850–853
  - non-GUI synchronous event
    - example 851–853
- Delete() method 825
- deleting
  - functions 399
  - variables 583
- Delimiter parameter 678–679
- denial-of-service (DoS) attacks 892
- Depth parameter 713
- depth, default serialization 498
- Descendants() method 710
- Description element 367
- Description property 388, 805
- descriptors, security 534–535
- Deserialized Property 508
- design decision, contentious issues 125
- desk.cpl command 598
- Desktop Management Task Force 8
- destructive conversion 102

- DeviceID property 840
- DHCPEnabled property 805
- diagnosing problems, using Eventlog 602
- diagnostics error 562
- diagnostics, tracing and logging 553
- dialog boxes, WinForms 747–750
- DialogResult property 749
- Dictionaries type, serialization in 507
- Digest type 515
- digital signature 905
- dir alias, Get-ChildItem 664
- dir command
  - comparison between two files 11
  - new PowerShell console 14–15
  - Path parameter 738
  - positional parameters 249
  - using pipelines 158
- DirectoryInfo object 423, 427
- Disable All Breakpoints debug menu item 649
- Disable-PSBreakPoint cmdlet 653
- Disable-PSSessionConfiguration cmdlet 531
- Disable-WsManCredSSP cmdlet 509
- discarding error messages 568
- discarding output 183
- Disconnect-WsMan cmdlet 509
- Display Call Stack debug menu item 649
- display settings, desk.cpl command 598
- display, width of 65
- DisplayName event 875
- DisplayName property 626, 628
- Distributed Computing Environment/Remote Procedure Call (DCE/RPC) 802
- distributed object model 802
- division 74
- division by zero error 561
- division operator 117–119
- DLL (dynamic link library) 42, 721
- DLR (Dynamic Language Runtime) 729
- DMTF (Distributed Management Task Force) 799
- DNS (Domain Name Service) names, and IP addresses 518
- Do It button 744
- doc comments 315
- Dock property, on winforms objects 746
- DockPanel control 775
- documentation comments 315
- documentation package, PowerShell 14
- documenting 314–321
  - help
    - automatically generated fields 315
    - comment-based 316–318
    - creating manual content 315–316
    - tags used in comments 318–321
    - .COMPONENT help 320
    - .EXTERNALHELP help 320–321
    - .FORWARDHELPCATEGORY help 320
    - .FORWARDHELPTARGETNAME help 320
    - .LINK help 320
    - .PARAMETER help 319
    - .REMOTEHELPRUNSPACE help tag 320
- documents
  - analyzing word use in 683–684
  - test 703–704
- dollar sign 186
- domain controller 511
- domain, extracting 137
- domain-specific languages 428, 709
- DoS (denial-of-service) attack 892
- dot operator 173–177, 197
- dot script 283
- DotNetFrameworkVersion element 367–368
- dot-sourcing 615
  - scripts and functions 283–284
- [double] type 74
- double assignment, Fibonacci example 121
- double quotes 52, 437
- double-clicking on script 897
- double-colon operator 177–178, 197
- double-quoted strings 78–79
- do-while loop 204–205
- DPAPI (Windows Data Protection API) 918
- drives
  - and providers 665
  - creating custom 665
  - function: drive 672
  - PowerShell drive abstraction 665
  - variable: drive 672
- DriveType property 840

- DSL (Domain-Specific Language) 686, 709
- dynamic binary modules 445
- dynamic code generation 658
- Dynamic Language Runtime (DLR) 729
- dynamic languages 73, 400, 428
  - debugging 643
  - security 895
- Dynamic Link Libraries (DDL) 42, 721
- dynamic linking, pros and cons of 721
- dynamic modules 412–418
  - and event handler state 862–863
  - binary 443–446
  - closures 414–417
  - creating custom objects from 417–418
  - script 412–414
- dynamic parameters, and dynamicParam
  - keyword 311–314
  - steps for adding 312–314
  - when to use 314
- dynamic scoping 272–273, 430
  - defined 269
  - implementation of 415
  - passing arguments from enclosing scope 926
  - same name of variables 652
- dynamic typing 72, 586
- dynamically generating scriptblocks 642
- dynamicParam block 312

## E

- ea parameter 567
- Eclipse minicomputer 9
- edit.com program 45, 617
- editor
  - ISE 610–614
    - files 612–613
    - syntax highlighting in ISE panes 614
    - tab expansion in editor pane 613–614
    - running current pane contents 614–615
- editor buffer, ISE editor pane 631
- editor keystrokes 16
- editor pane
  - hiding in ISE 610
  - ISE (Integrated Scripting Environment) 18–20, 607
  - making changes in 631–632

- EjectPC() method 766
- elastic syntax 46, 239
  - aliases and 47–50
  - definition 48
- Element() method 710
- elements, adding to XML objects 695–697
- elevated privileges, and remoting 468
- elevation of privilege, defined 892
- elseif clauses 201
- elseif keyword 201–202
- emits objects 257
- empty arrays 94–96, 211
- Enable All Breakpoints debug menu item 649
- Enable Strong Protection box, Certificate Export Wizard 914
- Enabled timer property 728
- Enable-PSBreakPoint cmdlet 653
- Enable-PSRemoting cmdlet 450, 452, 531
- Enable-PSSessionConfiguration cmdlet 531
- Enable-PSTrace 334
- Enable-WSManCredSSP cmdlet 509, 516
- enabling V1 strict mode 583
- encapsulating data and code 400
- encoded pipeline 505
- encoding
  - command arguments 505
  - serialization 505
  - used in strings 77–78
- Encoding parameter 311, 679
- encoding parameter 184
- encryption
  - in remoting 514
  - public key, and one-way hashing 904–905
- encryption key, security 918
- end blocks, functions with 266–267
- end clause 267, 397
- End() function 420
- end-of-parameters parameter 41
- endpoints
  - access controls and 533–535
  - registering configurations 532–533
  - remoting configuration 532
  - unregistering 533
  - verifying existence 533
- end-processing clause 62

- engine events
  - generating in functions and scripts 876–877
  - predefined 875–876
  - registrations 875–877
- EnterNestedPrompt() method 646
- enterprise, enabling remoting in 452–454
- Enter-PSSession cmdlet 450, 469, 518, 619, 621
- Enter-PSSession command 34
- EntryType filter 599
- EntryWritten event 879
- enum types
  - defining new at runtime 442–443
  - serialization in 506
- enumerable collection array 114
- Enumerate parameter 837
- enumerating collection, update issues 635
- enumerating, hash tables 87–88
- enumeration types 683
- enumerations 814
  - filtering results 837–838
  - singleton resources vs. 836–837
- EnumerationTimeoutms setting 525
- enumerators 636
- en-US subdirectory 363
- \$ENV: environment provider 376
- \$ENV: MYAPPDIR variable 376
- \$ENV: PATH 329
- \$ENV: PROCESSOR\_ARCHITECTURE 376
- \$ENV: PSMODULEPATH directory 362, 368
- \$ENV: PSMODULEPATH variable 329
- \$ENV drive 619
- env namespace 186
- \$ENV:HOMEPATH environment 494
- \$ENV:PATH environment variable 898
- \$ENV:PATHEXT variable 898
- \$ENV:PROCESSOR\_ARCHITECTURE variable 499
- \$env:PSExecutionPolicyPreference environment 903
- environment variables 186, 197, 619
- environmental forces, definition 6
- eq operator 124, 127
- \$error[0] 562
- error action policy 566, 578
- error action preference 566–567, 571
- error buffer
  - circular bounded buffer 560
  - controlling size 560
  - operations 561
- error codes
  - \$LASTEXITCODE variable 565
  - use in PowerShell 554
- error messages 244, 568
- error objects 182, 259, 564
- error processing subsystem 554
- error record exception property 559
- error records 554
  - as formatted text 556
  - as strings 556
  - displaying all properties 558
- error stream 259
- error subsystem, architecture of 554
- \$Error variable 559–564
- \$Error.Clear() method 560
- ErrorAction parameter 567–569
- ErrorActionPreference parameter 569
- \$ErrorActionPreference variable 567, 569, 571
- ErrorDetails Property 557
- ERRORLEVEL variable 281
- ErrorRecord 554, 572
- errors 553–605
  - capturing error objects 560
  - capturing session output 593–596
  - catching 431
  - debugging with host APIs 580–593
    - catching errors with strict mode 582–584
    - Set-StrictMode cmdlet 584–589
    - static analysis of scripts 589–593
  - event log 597–605
    - EventLog cmdlets 597–602
    - viewing 603–605
  - getting detailed information about 559
  - handling 554–569
    - \$Error variable and -ErrorVariable parameter 560–564
    - controlling actions taken on errors 566–569
    - determining if commands had errors 564–566
    - error records and error stream 555–560
  - object references 563
  - redirecting 181–182

- errors (*continued*)
  - runtime behavior 566
  - that terminate execution 569–580
    - throw statement 578–580
    - trap statement 570–575
    - try/catch/finally statement 575–578
  - types of 554, 599
- ErrorVariable parameter 560, 562–564
- escape character 53, 669
- Escape processing. *See* quoting
- escape sequence processing 54
- evaluates 228
- evaluation order, in foreach loop 209
- event log 597–605
  - accessing from PowerShell 603
  - EventLog cmdlets 597–602
  - viewing 603–605
- event log entries
  - event categories in PowerShell log 603
  - PowerShell state transitions 603
  - properties 604
  - types 603
- event log tasks
  - clearing and event log 597
  - creating new event log 597
  - listing available logs 598
  - setting log size limits 597
  - writing new event log entry 597
- \$Event variable 861
- Event viewer, and Show-Event cmdlet 597
- \$Event.Entry.Message 880
- \$Event.SourceArgs 861
- event-based script 849
- EventHandler variables 745
- EventIdentifier 865
- eventing cmdlets 854–855
- EventLog cmdlets 597–602
- EventLog events 879–882
- EventLog object 879
- events 847–887
  - asynchronous 853, 855
    - .NET events 855–860
    - event handling with scriptblocks 860–863
    - eventing cmdlets 854–855
    - subscriptions, registrations, and actions 854
- engine
  - event registrations 875–877
  - generating events in functions and scripts 876–877
  - predefined 875–876
  - forwarding, remoting and 877–882
  - handling 848–849
  - Microsoft WMI 866–875
  - Microsoft WMI intrinsic classes 871–874
  - queued, and Wait-Event cmdlet 863–866
  - synchronous 849–853
    - delegates and delegation 850–853
    - in GUIs 850
  - workings of 882–887
- Events member 751
- EventSubscriber cmdlet 861
- \$EventSubscriber variable 861
- EventWatcher1 879
- exact matches 219
- example code 22–35
  - basic expressions and variables 23–25
  - flow control statements 30–31
  - navigation and basic operations 22–23
  - processing data 25–30
    - problem-solving pattern 29–30
    - selecting properties from objects 27–28
    - sorting objects 25–27
    - with ForEach-Object cmdlet 28–29
  - remoting and Universal Execution Model 32
  - scripts and functions 31–32
- examplemodule.dll 356
- Exception property 558
- exceptions. *See also* errors
  - accessing in trap block 572
  - C# and VB.Net 554
  - catching all exceptions 570
  - related to error records 572
  - rethrowing 571
  - terminating error 570
  - throwing 574
- Exchange server 505
- executables 495–496
- executing code, at runtime 436
- execution
  - errors that terminate 569–580
  - throw statement 578–580

- execution (*continued*)
  - trap statement 570–575
  - try/catch/finally statement 575–578
  - of scripts, none by default 897–898
  - policy, enabling scripts with 898–903
- execution context, and remoting 466
- execution environments, constrained 543
- execution policy 910, 915
  - and implicit remoting 473
  - for scripts 276–278
- execution stopped error 568
- \$ExecutionContext variable 437, 539
- ExecutionPolicy parameter 901
- ExecutionTimeLimit property 789
- executive job 490
- exit code 565
- exit command 34
  - in remoting 470
  - issued in constrained sessions 550
- exit scripts 280
- exit statement, exiting scripts and 280–281, 645
- exit with code 0 565
- Exit() API 643
- exiting constrained sessions, with Exit-PSSession function 550
- Exit-PSSession cmdlet 450
- Exit-PSSession function 550
- expandable strings 78
- ExpandString() method 437–438
- Explicit Cast Operator type 107
- explore objects 423
- Explore() method 766
- Explorer, as a shell 6
- Export-Clixml cmdlet 711, 714–718
- Exported member term 326
- ExportedCommand property 353
- ExportedCommands 350, 381
- ExportedFunctions member 332, 381
- exporting certificates 913
- Export-ModuleMember 413
- Export-ModuleMember cmdlet 325–326, 343–347
  - controlling module member visibility with calculated module exports 346–347
  - controlling export 343–346
- exports 334
  - accessing using PSMODULEINFO object 381–382
  - calculated module 346–347
  - elements 337
  - of functions, controlling 343–344
  - of variables and aliases, controlling 344–346
- Expression Blend 610
- expression member, with Select-Object 411
- expression mode 54
- expression oriented syntax, with try/catch statements 578
- expression-mode parsing 54–56
- expression-oriented language 589
- expressions 231
  - basic 23–25
  - operators in. *See* operators in expressions
  - using try/catch/finally statement in 578
- extended type system 64
- extending
  - ISE 622–638
    - \$psISE variable 622–623
    - custom menus 633–638
    - Options property 624–625
    - tabs and files 625–629
    - text panes 629–633
  - objects 423
  - PowerShell language 428–436
    - adding CustomClass keywords 428–433
    - little languages 428
    - type extension 433–436
  - runtime 445
- extensibility points, in ISE 622
- EXTensible Application Markup Language. *See* XAML
- Extensible Stylesheet Language Transformations. *See* XSLT language 710
- External command lookups 548
- external commands
  - error handling 565
  - in sessions 538
- external executables 44
- .EXTERNALHELP <XML HELP FILE PATH>
  - help tag 320

## F

- f operator 103, 179–180, 197
- F5 execute current text buffer 615
- F8 execute selected text 615
- factorial function 262
- FailFast() method 853, 880
- Failure Audit type 599
- \$false variable 131, 184
- fan-in remoting 528–530
- fan-out remoting 527–528
- Fibonacci sequence 121
- fidelity 497
- fields 66, 682
- file association 897
- file encodings 675–676
- file length 188
- File menu, ISE 620
- file names, matching 132
- file not found error 562
- file operations 663
  - concatenating multiple files 675
  - display file contents 675
  - formatting and output subsystem 686
  - reading 674
  - renaming 673
  - searching file hierarchy 691
  - writing binary data 679
  - writing pre-formatted data 679
  - writing to files 679
- file option 222
- File parameter 285
- file paths 667, 669
- File property 352
- file search tool
  - defining appearance 754–756
  - specifying behavior 756–758
- file system
  - listing directories 664
  - working with 22
- file system provider 187, 667
- FileInfo object 175
- FileList manifest element 375
- FilePath option 461
- FilePath parameter 494
- files 625–629
  - adding file checker menu item 637
  - creating new 613
  - default association, notepad 897
  - loading and saving 697–701
  - opening 612–613
  - processing 672–681
  - processing with switch statement 221–222
  - saving list of open 632–633
  - searching with Select-String cmdlet 688–693
    - getting all matches in line 692–693
    - list and -quiet parameters 690–691
    - trees of files 691
    - with context properties 692
    - specifying format and password 914
  - FileSystemWatcher object 864
  - FileVersionInfo property 33
  - filter keyword 265, 267
  - Filter parameter 810–812
  - filtering
    - enumeration results 837–838
    - where cmdlet 33
  - Filtering output, using Get-Member cmdlet 557
  - filters 398
    - and functions 265–266
    - filtering EventLog entries 599
  - finally keyword 575
  - FindName() method 776
  - findstr command 688
  - firewall exception, for WinRM Service 453
  - fl command 48
  - flattened results 226
  - floating point 24
  - flow control 198, 234–235
    - adding new 428
    - conditional statement 200–203
    - labeled loops and break and continue statements 212–215
    - looping statements 203–212
      - do-while loop 204–205
      - for loop 205–207
      - foreach loop 207–212
      - while loop 203–204
    - performance 233–235
    - statements as values 231–233
    - switch statement 215

- flow control (*continued*)
  - processing files with 221–222
  - using \$switch loop enumerator in 222
  - using regular expressions with 217–221
  - using wildcard patterns with 216–217
  - using cmdlets 223–231
    - ForEach-Object 223–228
    - Where-Object 228–231
- flushing changes 828–829
- Folder object 789
- folder structure, of modules 362–363
- foo imports 326
- foo variable 817
- for loop 30, 205–207
- Force flag 359
- Force option 388
- Force parameter 341, 380, 513
  - in process of remoting access 451
  - overwriting existing definition 439
  - removing jobs 487
  - using with SecureString cmdlets 919
  - viewing hidden files with 668
- foreach block 877
- foreach cmdlet 526
- foreach keyword 207, 225
- foreach loop 207–212, 875
  - and \$null value 211–212
  - debugging 639
  - defined 30
  - displaying hash tables 88
  - evaluation order in 209
  - removing items from collection 635
  - using \$foreach loop enumerator in 209–211
  - using range operator 167
- \$foreach loop enumerator, using in foreach
  - loop 209–211
- foreach statement 30, 87, 205, 207, 209, 234, 635–636
- \$foreach variable 209, 263, 701
- \$foreach.MoveNext() method 210
- ForEach-Object cmdlet 223–228, 393–394, 398, 684, 849
  - argument processing by 227–228
  - as anonymous filter 265
  - comparing order of execution with foreach
    - loop 209
    - comparing with foreach statement 31
  - definition and example 30
  - processing with 28–29
  - recognized as command or statement 207
  - using alias 702
  - using return statement with 227
- ForEach-Object cmdlet 169
- foreground color
  - setting in ISE 625
- forensic tools 916
- formal arguments 246
- formal parameters, declaring for
  - functions 241–257
    - adding type constraints to 243–245
    - handling mandatory 248
    - handling variable numbers of arguments 246
    - initializing with default values 246–247
    - mixing named and positional 242–243
    - switch parameters 248–257
- format operator 179–181
- format specifier element 180
- format string 179
- Format-Custom formatter 66
- Format-List cmdlet 332, 341, 350
  - using to display Registry 671
  - using to see error record 558
  - using to see log 600
  - using with Get-WSManInstance 832
- Format-List command 64–65
- formats, specifying for files 914
- FormatsToProcess element 370, 373
- FormatsToProcess manifest element 373
- Format-Table cmdlet 411
- Format-Table command 34, 64
- formatting output 179, 686
  - in interactive remoting 471
  - in non-interactive remoting 472
- formatting strings 179
- Format-Wide cmdlet 66
- Format-XmlDocument function 837
- Forward parameter 877
- .FORWARDHELPCATEGORY <CATEGORY>
  - help tag 320
- .FORWARDHELPTARGETNAME <COM-  
MAND-NAME> help tag 320

- forwarding events, remoting and 877–882
- fragments, of script code 569
- frameworks, for WPF 758–759
- freespace 29
- FullName property 425, 724
- full-screen applications 616
- full-screen editor 617
- fully qualified path 666
- FullyQualifiedErrorId property 558
- function body 238
- function calls, tracing 639
- function definition, changing 399
- function definitions 268, 546
- function drive 268, 274, 398, 439–440
- function keyword 44, 316, 397, 399
- Function parameter 334
- function provider 399, 547
- function visibility, constrained sessions 537
- function: drive 399, 672
- FunctionInfo object 399
- functions 31–32, 43, 236–274
  - body of 32
  - called like methods 586–587
  - calling 242
  - declaring formal parameters for 241–257
    - adding type constraints to 243–245
    - handling mandatory 248
    - handling variable numbers of arguments 245–246
    - initializing with default values 246–247
    - mixing named and positional 242–243
    - switch parameters 248–257
  - defining at runtime 398–400
  - definition 397
  - dot-sourcing scripts and 283–284
  - fundamentals of 237–240
    - \$args variable 237–240
    - ql and qs functions 239–240
  - generating events in 876–877
  - initializing parameters 580
  - managing definitions in session 267–269
  - parameterizing 237
  - returning values from 257–263
    - debugging problems in function output 259–262

- return statement 262–263
- using in pipeline 263–267
  - filters and functions 265–266
  - functions with begin, process, and end blocks 266–267
- variable scoping in 269–274
  - declaring variables 270–272
  - modifiers 272–274
- VBScript 784
- function-scoped variable 282
- FunctionsToExport element 370

## G

- GAC. *See* global assembly cache
- gateway server 515
- General Impression, Size, and Shape (GISS) 37
- generating elements 209
- generating script 279
- generic types 98–99, 739–740
- Get command 841
- GET() method 825
- Get/Update/Set pattern 253–257
- GetAssemblies() method 725
- Get-AuthenticodeSignature cmdlet, security 911
- Get-Bios command 477
- Get-Bios function 476
- Get-BrowserWindow function 770, 772–773
- Get-Character function 253, 255
- Get-ChildItem cmdlet 395, 402, 664, 691
- Get-ChildItem command 47
- Get-Command 332, 339, 342, 353, 384, 545
- Get-CommandString function 758
- Get-ConsoleWindow() method 734
- Get-Content cmdlet 221, 673–678
  - performance caveats 680–681
  - ReadCount parameter, and Where-Object cmdlet 229–231
  - reading files 697
  - sending data to pipeline 122
  - sending data to variables 143
  - using with binary files 188
- Get-Content command 47
- Get-Count 338–339, 342–343, 345
- Get-Count function 379–380, 383–384
- Get-Credential cmdlet 475, 917, 919

- Get-Date cmdlet 81, 182, 247, 395, 593
- Get-Date command 11, 395
- GetEnumerator() method 87, 720
- Get-Event cmdlet 854
- Get-EventLog cmdlet 597–598, 601, 603
  - filtering entries 602
  - InstanceID parameter 602
  - Message parameter 602
  - Source parameter 602
- Get-EventSubscriber cmdlet 854, 856, 859
- GetExportedTypes() method 725
- Get-HealthModel command 544
- Get-Help cmdlet 701
- Get-Help command 22, 325
- Get-Help Online about\_execution\_policies 31
- Get-HexDump function example 676–677
- Get-Item cmdlet 512, 564, 567
- Get-Item command, retrieving RootSDDL 534
- Get-ItemProperty cmdlet 672
- Get-Job cmdlet 483–485, 882
- GetLength() function 784
- Get-Location cmdlet 665, 673
- Get-MagicNumber function example 677–679
- Get-MailboxStatistics command 29
- Get-Member cmdlet 175, 557, 623, 653, 765, 857
  - examining objects 400
  - getting information of object using 13
  - static members 401
- GetMembers() method, listing object
  - members 119
- Get-Module cmdlet 333, 359
  - description of 325
  - finding modules 327–329
  - getting information about loaded module 331, 350–351
- GetNetworkCredential() method 790
- GetNewClosure() method 415
- Get-OkCancel() function 747
- Get-PageFaultRate command 531
- Get-PfxCertificate cmdlet 915
- Get-Process command 29, 40
- Get-ProgID function 763
- Get-PSBreakPoint cmdlet 653
- Get-PSCallStack cmdlet 653
- Get-PSDrive command 457
- Get-PSProvider cmdlet 664
- Get-PSSession cmdlet 472
- Get-PSSession command 472
- Get-PSSessionConfiguration cmdlet 531
- Get-Service cmdlet 711
- Get-Spelling.ps1 script 781
- GetTempFileName() method 881
- getter method 407
- GetType() method 73, 123, 142
- GetTypes() method 726
- Get-Variable cmdlet 651
- Get-Variable function 379
- Get-WmiObject cmdlet 804–813
  - Filter parameter 810–812
  - Microsoft WMI objects
    - selecting instances using filters and queries 810
    - selecting using -Query parameter 812–813
  - navigating CIM namespaces 807–810
  - using to find Microsoft WMI classes 806–807
- Get-WmiObject command 29
- Get-WordDefinition function 779
- Get-WSManCredSSP cmdlet 509
- Get-WSManInstance cmdlet
  - retrieving management data with 832–839
    - filtering enumeration results 837–838
    - getting Win32\_OperatingSystem resource 834–836
    - selecting instances 838–839
    - singleton resources vs. enumerations 836–837
    - targeting WS-Man resources using URIs 834
- gigabytes 83
- GISS (General Impression, Size, and Shape) 37
- \$global 387
- global assembly cache 722
- global context 186
- global environment, importing nested modules into
  - with -Global flag 352–353
- Global flag, importing nested modules into global environment with 352–353
- global functions 430–431
- global level 335
- global modifier 273
- Global parameter 353

- global scope 436
- \$global:name scope modifier 281
- globalization support, in ISE 607
- goto statement 212
- grammar 38
- graphical debugger 648–652
  - executing other commands in debug mode 651
  - hovering over variables to see values 652
- graphical environment 622
- graphical programming 445
- Graphical User Interfaces. *See* GUIs 743
- green play button 18
- grep command 688
- GROUP clause 874
- GROUP keyword, aggregating events
  - with 874–875
- Group Policy, enable remoting using 452
- GUI builder tools 752
- GUI debugger 653
- Gui flag 774
- Gui option 773
- Gui parameter 773
- GUID (globally unique ID) 762
  - serialization in 506
  - used as Job Instance ID 484
- GUIs (graphical user interfaces) 4, 743–759
  - creating winforms modules 750–753
  - defining in XAML 774–775
  - synchronous events in 850
  - WinForms library 744–750
    - simple dialog boxes 747–750
  - WPF 753–759
    - advantages of 758
    - file search tool 754
    - frameworks for 758–759
    - preconditions 753

**H**

- Handle property 839
- handlers, event
  - state 862–863
  - timer 856–859
- handles 226, 395
- handling
  - events 848–849, 860–863
  - remote EventLog events, example 879–882
- hanging applications, in ISE 616
- has-a relationship 13
- hash algorithm, MD5 889
- hash table argument 244
- hashing
  - one-way, public key encryption and 904–905
- hashtable operators 112
- hashtables 85–91
  - as reference types 90–91
  - counting unique words with 684–686
  - empty 429
  - enumerating 87–88
  - extending 435
  - modifying 88–89
  - sorting 87–88
  - use with Select-Object 411
- health model function 544
- Hello world program 3
- help
  - automatically generated fields 315
  - comment-based 316–318
  - creating manual content 315–316
  - tags
    - .COMPONENT help 320
    - .EXTERNALHELP help 320
    - .FORWARDHELPCATEGORY help 320
    - .FORWARDHELPTARGETNAME help 320
    - .LINK help 320
    - .PARAMETER help 319
    - .REMOTEHELPRUNSPACE help 320
  - help files, operating on 683
  - help subsystem, PowerShell 23
  - help topics, about\_Assignment\_operators 685
  - help viewer 611
  - helper function 430
  - HelpMessage property 302–303
  - here-strings 77, 80–82, 279, 638
    - loading XAML from 776–777
  - hex digits 114
  - hexadecimals 84–85, 180
  - hi function 470
  - hi member 425
  - hiding editor pane

- Ctrl-R 611
  - in ISE 610
- highlighting syntax, in ISE panes 614
- History tab 792
- HitCount property, on breakpoints 654
- hkln: registry drive 671
- home directories, of providers 667
- Home directory 667
- \$HOME variable 494
- host APIs, debugging with 580–593, 595
  - catching errors with strict mode 582–584
  - Set-StrictMode cmdlet 584–589
  - static analysis of scripts 589–593
- host application interfaces 643
- host application, PowerShell 14
- \$host member 581
- host programs 466
- \$host variable 467, 580, 646
- host version 581
- hosting fan-in remoting, IIS (Internet Information Services) 529
- \$hostName parameter 308
- hotkey sequences
  - defining in ISE 634
  - ISE pane positions 610
- hovering over variables 652
- Howard, Michael 892
- HTML tags 741
- HTTP (Hypertext Transport Protocol) 504
- HTTPS (Hypertext Transfer Protocol Secure) 512
- HTTPS (Secure HTTP) 504
- hygienic dynamic scoping 269

## I

- I/O redirection 68, 183
- IComparable interface 127
- IdleTimeout parameter 526
- IE (Internet Explorer) 668
- IEEE Specification 1003.2, POSIX Shell 9
- IEnumerable interface 210, 720
- ieq operator 125
- IETF (Internet Engineering Taskforce) 504
- if statement 200, 358, 376, 774
- IgnoreCase option 146
- IgnoreWarnings parameter 731

- IIS (Internet Information Services) 529
- IList interface 507
- IList type 108
- Impersonation parameter 803
- implementation decision, concatenation
  - hashtables 115
- implicit behavior, overriding 127
- Implicit Cast Operator type 106
- implicit remoting 473–481, 535
  - and execution policy 473
  - connection sequence 474
  - generating temporary modules 480
  - in constrained sessions 545
  - in InitialSessionState 546
  - local proxy functions 473
  - message flow 476
- Import-Clixml cmdlet 633, 714–718
- Imported member term 326
- importing SecureString 918
- Import-Module cmdlet 331, 353–354, 358, 414
  - and nested modules 350
  - description of 325
  - loading modules 331, 356
  - module loading another 326
  - using -ArgumentList parameter 358
  - using -Global flag 352
- Import-Module function 368, 372, 380
- Import-PSSession cmdlet 474, 551
- Import-PSSession, with constrained sessions 546
- imports 334
- increment operator 80, 155, 204
- \$increment variable 338, 345, 379, 384, 416
- indenting text, in editor pane 612
- index operation 396
- indexing 167–170
  - of arrays 92
  - with variables 173
- indirect method invocation 178–179
- indirect property name retrievals 176
- information disclosure attack 896
- Information type 599
- infrastructure script 461
- inheritance hierarchy 424
- inheritances 433
- Initialize-ComplexConstrained-
  - HMConfiguration.ps1 548

- Initialize-ConstrainedHMConfiguration.ps1 543
- initializer expressions 247–248, 580
- initializing multiple variables 123
- injection code 896–897
- inline documentation 60
- in-memory buffering 187
- in-memory sessions, local 619
- inner loops 214
- InnerException property 559
- InnerText() method 695
- in-process execution 619
- input redirection 182
- input validation 895–896
- \$input variable 263, 265, 455
- input, processing 264
- \$input.current.\$p expression 264
- InputObject parameter 39, 51
- InputObject parameter 813
- installation directory path 434
- installutil.exe program 354
- instance members 177, 819
- \_InstanceCreationEvent class 871
- InstanceDeletionEvent class 871
- InstanceID filter 599
- InstanceID parameter, on Get-Eventlog 602
- InstanceID property 600
- InstanceModificationEvent class 872
- InstanceOperationEvent class 872
- instances
  - creating 429, 432
  - extending 401, 434
  - setting properties of 816–819
  - using Microsoft WMI paths to target 814–816
- instantiating objects 727
- integer 74, 127
- integrated debugger, ISE 648
- Integrated Scripting Environment 14
  - automating 20
  - command entry window 17
  - Ctrl-C behavior 17
  - Ctrl-C in 17
  - editor window 17
  - F8 key in 18
  - help within 23
  - is scriptable 20
  - multiple session tabs 18
  - opening multiple files 18
  - output window 17
  - syntax highlighting 18
  - three parts of 17
  - three parts of the 17
  - using F1 key 23
  - Visual Studio comparison 20
- interactive command interpreter, security 897
- interactive environment 226
- interactive errors 562
- interactive InitialSessionState 546
- interactive mode 643
- interactive operation, constrained sessions 545
- interactive remoting 619
  - with custom configuration 549
- interactive sessions 450, 469–472, 621
  - multiple concurrent sessions 471
- interactive token, creating from credentials 515
- intercepting expressions 124
- interfaces, defined 12
- interleave user commands 607
- internal command lookups 548
- internal commands 541
- Internet Explorer, looking up word definitions
  - using 779–781
- Internet Information Services 529
- Internet, .NET framework and 740–743
  - processing RSS feeds 742–743
  - retrieving web pages 740–742
- InternetExplorer.Application.1 ProgID. 763
- Interop assemblies, COM and 761–796
- Interop library 762
- interpreter 199, 439, 639
- interpreter reentrancy 643
- Interval property 857
- Interval timer property 728
- intervening characters 136
- invalid file name, security 896
- invocation intrinsics 445
- InvocationInfo property 558–559
- Invoke method 178
- Invoke() method 382–383, 627, 852
- Invoke-Command cmdlet 32, 449, 518, 881–882
  - child jobs with 490–492
  - description of 450
  - syntax for 449

- Invoke-Expression cmdlet 104, 142, 377, 436–437, 643, 924
  - avoiding 923–927
  - security 895
- Invoke-Gui piece 774
- InvokeScript() method 437–438
- invoke-WmiMethod cmdlet 819–822
- Invoke-WSManAction cmdlet, invoking methods with 841–846
- invoking commands, indirectly 395
- invoking script blocks 393
- IP (Internet Protocol) addresses, DNS names and 518
- IP address 518
- IPAddress property 805
- ipconfig command 898
- is operator 152–154, 404
- IsChecked property 757
- ISE (Integrated Scripting Environment) 17–20, 260, 606–660
  - controlling pane layout 607–610
    - command pane 608–609
    - ISE toolbar 609–610
  - editor 610–614
    - files 612–613
    - syntax highlighting in ISE panes 614
    - tab expansion in editor pane 613–614
  - editor pane 18–20
  - executing commands in 614–616
    - running current editor pane
      - contents 614–615
      - selected text 615–616
  - extending 622–638
    - \$psISE variable 622–623
    - custom menus 633–638
    - Options property 624–625
    - tabs and files 625–629
    - text panes 629–633
  - key bindings 610
  - running scripts in 616–618
    - issues with native commands 616–617
    - threading differences between console and ISE 618
  - using multiple tabs 618–622
    - local in-memory session 619
    - remote session 619–622
- View Menu 610
- ISE menu item 634
- ISE object model
  - adding new session tab 631
  - hierarchy of 623
  - reloading saved tabs 633
  - saving list of open 632
  - setting editor buffer contents 632
- ISE panes 608
- isnot operator 152, 154
- ISO/IEC recommendations 84
- isolated execution environment. *See* AppDomain
- isolation, between sessions 619
- IT industry 8
- Item() property 768
- ItemNotFoundException 558
- iterating value 211
- iteration 215

**J**

- jagged arrays 171, 197
- JavaScript 401, 783
- \$jib variable 486
- job objects 486, 859
- jobs
  - background 481–493
    - cmdlets 483–487
    - commands 483
    - multiple 487–489
    - running in existing sessions 492–493
    - starting on remote computers 489–492
  - child
    - and nesting 489–490
    - with Invoke-Command cmdlet 490–492
  - removing jobs 487
  - waiting to complete 486
- join 177
- join method 177
- join operator 139–143, 406
- Join() methods 683
- joining strings 177
- Join-Path cmdlet 376
- JScript code, embedding in script 785–786
- JScript language 785
- jumping 214

## K

- Kerberos 514
- Kerberos type 515
- key codes, reading from PowerShell 581
- key-binding, ISE 610
- keyboard shortcuts, in debugger 648
- keys
  - in Registry 671
  - private protection, enabling strong 913–915
  - public encryption, and one-way hashing 904–905
- keys property 86
- key-value pairs 85
- keywords 199, 430, 433, 589
- Kidder, Tracey 9
- kilobytes 83
- Kleene, Stephen Cole 394
- Korn shell 9, 38

## L

- labeled loops, and break and continue
  - statements 212–215
- lambda calculus 394
- lambda expressions 394
- language development 9–11
- language elements 428
- language mode
  - sequence of operations 544
  - setting 539–543
- language restrictions, in module manifests 376–377
- LanguageMode property 539, 541
- last mile problem 8
- \$LAST variable 421–422
- \$last variable 421
- \$LASTEXITCODE variable 564–565
- LastWriteTime property 11
- lawn gnome mitigation, avoiding 893–894
- layout settings, ISE panes 610
- leading zeros, numeric comparison 126
- LeBlanc, David 892
- left aligned 180
- left operand 118
- left-hand rule operators 113
- legacy commands 44
- legitimately signed scripts 913
- Leibniz, Gottfried Wilhelm 5
- length of file object 221
- length property 164, 170, 173
- level of abstraction 8
- levels of indirection 175
- lexical analyzer 50
- lexical element 81
- lexical scoping 269
- lexical, ambiguity with type literals 178
- lfunc function 281
- libraries
  - of functions 283
  - simple 283, 747–750
  - WinForms 744–750
- lightweight data record 86
- like operator 132–133, 215, 217
- Limit-EventLog cmdlet 598
- lines, getting all matches in 692–693
- .LINK help tag 320
- linking, pros and cons of dynamic 721
- list comprehensions feature 159
- list of functions, function drive 398
- List parameter 690–691, 804
- List type, serialization in 507
- \$list variable 194
- ListAvailable parameter 328
- List-Breakpoints debug menu item 649
- LiteralPath parameter 670–671
- literals 96–101
  - accessing static members with 99–101
  - generic types 98–99
  - script block 397–398
  - type aliases 96–98
- little languages 428
- live objects 508, 803
- live shell session 613
- load order, of module components 374–375
- Load() method 724
- loader manifest elements, module
  - manifests 371–373
  - ModuleToProcess manifest element 371–372
  - NestedModules manifest element 372
  - RequiredAssemblies manifest element 372
  - ScriptsToProcess manifest element 372–373
  - TypesToProcess and FormatsToProcess manifest elements 373

- loading
  - by module name 331–333
  - files 697–701
  - removing loaded modules 335–337
  - tracing with `-Verbose` flag 333–334
- `LoadWithPartialName()` method 723
- local certificate authority, role in remoting 512
- local certificate store, defined 905
- local in-memory sessions 619
- local output, remote output vs. 497–498
- local proxy functions 473
- Local Security Policy MMC snap in 521
- local session 619
- Local User Administration dialog 920
- `LocalAccountTokenFilterPolicy` 517, 521
- Location property 354
- `LocationName` property 772
- logical complement 229
- logical disk object 830
- logical operators 148–150
- logical type containment 720
- lookup word definition 779
- loop 80, 639
- loop counter 206–207
- loop keyword 428
- loop processing 220
- loop statement 233
- looping construct, adding new 428
- looping statements 203–212
  - do-while loop 204–205
  - for loop 205–207
  - foreach loop 207–212
    - and `$null` value 211–212
    - evaluation order in 209
    - using `$foreach` loop enumerator in 209–211
  - while loop 203–204

**M**

- machines, monitoring
  - multiple machines 458
  - single machine 457–458
- magic number, determining binary file types 677
- `MakeCert.exe` program 906
- malware, defined 890
- MAML (Microsoft Assistance Markup Language)
  - format 316
- management model, Windows 35
- management objects 7, 797–846
  - Microsoft WMI 798–801
    - cmdlets 801–824
    - infrastructure 799–801
    - object adapter 824–830
    - putting modified objects back 828–830
  - WS-Man 830–846
- management, of types 72–77
- managing error records 560
- managing resource consumption, with quotas 523
- mandatory arguments 223, 248
- mandatory parameters
  - handling 248
  - in functions using `throw` statement 580
- Mandatory property 248, 297
- manifests 721
- manipulate script blocks 400
- manipulating code 400
- manual documentation, in help files 315–316
- `.map()` operator 209
- Margin property 756
- mash-ups, composite management
  - applications 324–325
- Match class, `System.Text.Success` property 688
- match group 135
- Match object, Value property 688
- `-match` operator 134–137, 215, 218, 687, 772
  - matching using named captures 135–136
  - parsing command output using regular expressions 136–137
- `Match()` method 687
- matched value 216
- `$matches` variable 134, 218
- matches, getting all in line 692–693
- MatchEvaluator class 851–852
- MatchEvaluator method 852
- MatchInfo class 692
- MatchInfo object 691
- matching process 216
- matching quote 52
- math operations, advanced 727
- `MaxEnvelopeSizeKB` setting 523
- maximum integer value 578

- \$MaximumErrorCount variable 560
- MaximumReceivedDataSizePerCommandMB parameter 523
- MaximumReceivedObjectSizeMB parameter 523
- \$MaximumVariableCount variable 523
- MaxPacketRetrievalTimeSeconds setting 525
- MaxProviderRequests setting 523
- MaxShellsPerUser controls 511
- MaxTimeoutms setting 524
- MD5 hash algorithm 889
- Measure-Command cmdlet 465
- Measure-Object cmdlet 710
- megabytes 83
- member collection 425
- member types 400
- member types, Add-Member cmdlet 403
- MemberDefinition property, On Add-Type cmdlet 730
- members, creating Singleton definitions 730–734
- memory consumption 560
- menus, custom 633–638
  - adding file checker item 637
  - submenu for Snippets 637–638
- merging streams 183
- Message filter 599
- Message parameter 602
- MessageData parameter 855, 863
- MessageData property 863
- metadata. *See also* module manifests 288, 391
- metaprogramming 392–446
  - building script code at runtime 436–440
    - \$ExecutionContext variable 437
    - creating elements in function drive 439–440
    - ExpandString() method 437–438
    - Invoke-Expression cmdlet 436–437
    - InvokeScript() method 438
    - script blocks 438–439
  - compiling code with Add-Type cmdlet 440–446
    - defining new .NET class with C# language 440–442
    - defining new enum types at runtime 442–443
    - dynamic binary modules 443–446
    - dynamic modules 412–418
    - closures 414–417
    - creating custom objects from 417–418
    - script 412–414
  - extending PowerShell language 428–436
    - adding CustomClass keywords 428–433
    - little languages 428
    - type extension 433–436
  - objects 400–410
    - adding note properties with New-Object cmdlet 409–410
    - public members 400–402
    - using Add-Member cmdlet to extend 402–408
  - script blocks 393–400
    - defining functions at runtime 398–400
    - invoking commands 394–396
    - literals 397–398
  - Select-Object cmdlet 410–412
  - steppable pipelines 418–423
    - creating proxy command with 420–423
  - type system 423–428
    - adding properties to 425–427
    - shadowing existing properties 427–428
- method call arguments 176
- method invocations 176, 178–179
- method operators 173–179
  - dot 174–177
  - indirect method invocation 178–179
  - static methods and double-colon operator 177–178
- methods
  - defined 11
  - functions called like 586–587
  - invoking with Invoke-WSManAction cmdlet 841–846
  - static, calling 819–822
- Methods type 403
- Microsoft Assistance Markup Language (MAML) format 316
- Microsoft Baseline Configuration Analyzer 462
- Microsoft Communications Protocol Program 505
- Microsoft Developers Network. *See* MSDN 424
- Microsoft Exchange 29
- Microsoft Management Console. *See* MMC
- Microsoft security response 890

- Microsoft study on improving offerings 7
- Microsoft Update 14
- Microsoft Windows
  - automating with COM (Component Object Model) 764–777
  - browser windows 767–777
  - Shell.Application class 765–766
  - connection issues 520–522
  - Task Scheduler 786–793
    - Schedule.Service class 786–787
    - tasks 787–793
  - Vista, connection issues 521–522
  - XP with SP3, connection issues 520–521
- Microsoft WMI (Windows Management Instrumentation) 8, 75, 522, 797, 866–875
  - cmdlets 801–824
    - common parameters 802–803
    - Get-WmiObject 804–813
    - invoke-WmiMethod 819–822
    - remove-WmiObject 822–824
    - Set-WmiInstance 813–819
  - documentation 807
  - event registrations
    - class-based 867–870
    - query-based 871–875
  - example of 29–30
  - infrastructure 799–801
  - methods 824
  - namespace hierarchy 814
  - object adapter 824–830
  - putting modified objects back 828–830
  - samples and resources, adapting 830
  - type accelerator 828
- Microsoft Word, spell checking using 781–783
- Microsoft.PowerShell endpoint 533
- Microsoft.PowerShell, session configuration 524
- \$mInfo variable 388
- minicomputer, Eclipse 9
- minute property 412
- mitigation
  - defined 893
  - serialization 505
  - threats, assets, and 893–897
    - authentication, authorization, and roles 894–895
    - avoiding lawn gnome mitigation 893–894
    - blacklisting/whitelisting 894
    - code injection 896–897
    - input validation 895–896
- mkdir command 67
- MkDir function 673
- mkdir function 268
- MMC (Microsoft Management Console) 32, 521, 597, 908, 922
- modeling
  - security 891–897
    - threat modeling 891–892
  - threats, assets, and mitigations 893–897
- models, defined 891
- Model-View-Controller (MVC) pattern 187
- modernized languages 418
- modifiers 272–274, 428
- modifying hashtables 88–89
- module boundary 541
- module exports 535
- module identity 368
- Module manifest term 326
- module manifests 361–391
  - construction elements 370–375
    - loader manifest elements 371–373
    - module component load order 374–375
  - content elements 375–376
  - controlling when modules can be unloaded 388–389
  - defining module vs. calling module 384–387
    - accessing calling module 385–386
    - accessing defining module 385–386
  - language restrictions in 376–377
  - module folder structure 362–363
  - production elements 366–370
    - module identity 368–370
    - runtime dependencies 368–370
- PSModuleInfo object 378–382
  - accessing module exports using 381–382
  - invocation in module context 378–381
  - methods 382–384
- running an action when module is removed 389–391
- setting module properties from inside script module 388
- structure of 363–366

- Module member term 326
- Module property 351, 383
- module scope 349
- Module type term 326
- ModuleList element 375
- ModuleName property 351–352
- module-qualified command name 548
- modules 7, 322–360
  - accessing exports using PSMModule Info object 381–382
  - basics of 325–327
    - single-instance objects 326–327
    - terminology 326
  - binary 353
    - creating 355–357
    - nesting in script modules 357–360
    - vs. snap-in modules 354–355
  - browser window management 770–777
    - adding graphical front end 773–774
    - defining control actions 775–776
    - XAML 774–777
  - component load order 374–375
  - controlling unloading of 388–389
  - dynamic
    - and event handler state 862–863
    - closures 414–417
    - creating custom objects from 417–418
    - script 412–414
  - dynamic binary 443–446
  - finding on system 327–330
    - imports and exports 334
    - loading module 331–334
    - module search algorithm 329–330
  - identity 368–370
  - invocation of PSMModuleInfo object in context of 378–381
  - removing loaded 335–337
  - role of 323–325
  - running actions when removed 389–391
  - setting properties from inside script module 388
  - winforms 750–753
  - writing script 337–353
    - controlling member visibility with Export-ModuleMember cmdlet 343–347
    - installing module 347
      - nested modules 350–353
      - review of scripts 338–340
      - scopes in script modules 348–350
      - turning into module 340–343
- ModuleToProcess element 370–372, 374
- ModuleVersion element 367–368
- modulus operator 117–119
- Monad project 5
- Monadology, The* (Leibniz) 5
- monitor size, and ISE 609
- monitoring, using remoting 457
- Move-Item cmdlet 673
- MoveNext() method 210, 223, 264
- MoveToNextAttribute() method 701
- MSDN (Microsoft Developers Network) 68, 424
  - blogs home page 741
  - documentation 762
- MSH/Cibyz worm 891
- [MS-PSRP] remoting protocol 504
- MSRPC (Microsoft Remote Procedure Call) 802
- MS-WSMV (Web Services Management Protocol Extensions for Windows Vista) 504
- MTA (multithreaded apartment) 618, 793
- multicore processors 618
- multidimensional arrays 171–173, 197
- multi-hop environments, forwarding credentials in 515–517
- multiline comments 59–60
- multiline option 147–148
- multimachine monitoring 457–462
  - multiple machines 458
  - parameterizing solution 459–462
  - resource management using throttling 458–459
  - single machine 457–458
- multiple assignment 187, 240
- multiple jobs
  - application of 488
  - performance considerations 489
- multiple machines 622
- multiplication operator 116–117
- multiplier suffixes, for numeric types 83–84
- multiplying numbers 113, 116
- multiscope catch 579
- multiscope trap 579
- multivalued arguments 243

MVC (Model-View-Controller) pattern 187  
\$MyInvocation.MyCommand.Module 385

## N

nadd function 244  
name member, with Select-Object 411  
name of host, obtaining 581  
Name property 175, 425, 491, 756, 815  
named captures, matching using 135–136  
named parameters 242–243  
names, of variables  
    syntax for 186–188  
    vs. variable values 192–193  
NAMESPACE class 808  
namespace collisions 535  
-Namespace parameter 730, 807  
namespace providers 800  
namespace qualifiers 272  
namespace, notation variables 186  
namespaces 665, 720  
name-value pair 193  
native commands 44–46, 565  
    issues with 616–617  
    Windows 39  
navigation 22–23  
-ne operator 124  
negative indexing 170  
Negotiate type 515  
nest prompt characters 56  
nested data structures 197  
nested function 387  
nested interactive session 644  
nested loops 214  
Nested module term 326, 350  
nested modules 350–353  
    binary in script 357–360  
    importing into global environment with -Global  
    flag 352–353  
nested pipelines 643  
nested prompts, and Suspend operation 643–647  
    breakpoint command 646–647  
    suspending script while in step mode 644–645  
nested session 646  
nested shell operation sequence 643  
nested statement 235

NestedModules element 370, 372  
nested-prompt level 646  
\$NestedPromptLevel variable 645  
nested-shell level 647  
nesting, child jobs and 489–490  
.NET assembly, loading 781  
.NET class  
    defining new with C# language 440–442  
.NET events 855–860  
    managing subscriptions 859–860  
    writing timer event handler 856–859  
        creating timer object 856  
        enabling 858–859  
        setting parameters 857  
.NET exceptions 579  
.NET framework 719–759  
    and Internet 740–743  
        processing RSS feeds 742–743  
        retrieving web pages 740–742  
    assemblies 721–725  
        default 722  
        loading 722–725  
        pros and cons of dynamic linking 721  
        versioning and 721–722  
    basics 720–721  
    GUIs 743–759  
        creating winforms modules 750–753  
        WinForms library 744–750  
        WPF 753–759  
    types  
        creating instances of 727–729  
        defining new with Add-Type  
        cmdlet 729–739  
        finding 725–727  
        generic 739–740  
.NET interop wrapper 794  
.NET libraries. *See* Assemblies  
.NET object model  
    leveraging 10–11  
    self-describing 10  
.NET/COM Interop library 761  
.NET-based custom type conversion 104–107  
netbooks 610  
network programming 740  
network tokens 515

- new interactive session 646
- new language features 433
- New Remote PowerShell Tab 620
- new tasks, XML representation of 791
- NewBoundScriptBlock() method 382–384, 416
- NewBoundScriptBlockScriptblock() method 862
- New-Control function 751
- New-Event cmdlet 854
- New-EventLog cmdlet 597
- new\_instance function 430–431
- New-Item cmdlet 439, 672–673
- newline character 56–57, 678
- New-Module cmdlet 325, 413–414
- New-ModuleManifest cmdlet 325, 363, 366
- NewName parameter 108
- New-Object cmdlet 172, 425, 761
  - adding note properties to objects with 409–410
  - examples of 729
  - limitations of 728
  - Property parameter 727–728
- New-PSSession cmdlet 450, 465, 468–469, 518, 879
- New-PSSessionOption cmdlet 519, 523, 526
- NewScriptBlock() method 437–438
- NewTask() method 788
- New-WSManSessionOption cmdlet 509
- NextMatch() method 688
- No to All 642
- noclobber parameter 184
- NoLanguage mode 550
- non-filesystem providers 672
- noninteractive remoting 450
- non-numeric string 114
- nonprintable characters 676
- nonstructured exit 212
- non-terminating errors 554, 566, 569
- non-zero value 566
- non-zero-length strings 253
- NoProfile option 494
- normal flow of control 849
- notcontains operator 130
- note member 433
- note properties
  - adding to objects with New-Object cmdlet 409–410
  - setting 408
- Notepad function 666
- notepad, default file association 897
- Notepad.exe command 666
- NoteProperty members, adding to
  - objects 405–406
- NoteProperty PSPath property 425
- NoTypeInfoInformation parameter 713
- nouns 428
- NTLM authentication 518
- \$null variable 583, 684–685
  - casting string to int 153
  - reference to uninitialized variable 184
  - using in integer expressions 570
- numbers, conversion to string 25
- numeric calculations 73
- numeric comparison 126
- numeric context 129
- numeric conversion rules 127
- numeric expressions, and \$null 583
- numeric types 82–85
  - hexadecimals 84–85
  - multiplier suffixes for 83–84
  - specifying 83
- \$numProcesses parameter 460

**O**

- [object] type 354
- object adapter
  - Microsoft WMI 824–830
  - type accelerators 825–828
- object model
  - .Net 10
  - customizing ISE 621
- object normalization 402
- object streaming model 554
- Object tab 713
- ObjectNotFound category 559
- object-oriented languages 245, 401
- objects 128, 400
  - adding note properties with New-Object cmdlet 409–410
  - creating custom from modules 417–418
  - defined 11
  - managing windows through 7–8

- objects (*continued*)
  - public members 400–402
  - rendering as XML 711–718
    - ConvertTo-Xml cmdlet 711–714
    - Import-Clixml and Export-Clixml cmdlets 714–718
  - representing in protocol stack 505–509
  - reviewing OOP 11–13
  - selecting properties from 27–28
  - sorting 25–27
  - type of 12
  - using Add-Member cmdlet to extend 402–408
    - adding AliasProperty members 404–405
    - adding NoteProperty members 405–406
    - adding ScriptMethod members 406–407
    - adding ScriptProperty members 407–408
  - using XML as 693–694
  - XML, adding elements to 695–697
- Off parameter, on Set-PSDebug 582
- \$OFS variable 103, 238, 387, 677–678
- one-dimensional arrays 171
- one-way hashing, public key encryption and 904–905
- OnRemove handler 480
- OnRemove property 390
- OOP (object-oriented programming) 11–13
- op\_class 112
- op\_Addition () method 112
- op\_Division() method 119
- Open file command, psEdit 612
- open tabs, listing In ISE 626
- opening multiple files, in ISE 612
- OpenTimeout parameter 526
- operand 114
- operating environment, object-based 8
- operating on binary data 149
- operations
  - basic 22–23
  - setting timeouts on 524–527
- Operations Manager, examining OpsMgr event log entries 600
- OperationTimeout parameter 526
- OperationTimeout value 526
- operator semantics 150
- operators 151–197, 589
  - array 162–173
    - comma operator 162–165
    - indexing and slicing 167–170
    - multidimensional 171–173
    - range operator 165–167, 170–171
  - for working with types 152–154
  - format 179–181
  - grouping 157–162
  - property and method 173–179
    - dot operator 174–177
    - indirect method invocation 178–179
    - static methods and double-colon operator 177–178
  - redirection 181–184
  - unary 154–157
- operators in expressions 110–150
  - arithmetic 112–119
    - addition operator 113–116
    - multiplication operator 116–117
    - subtraction, division, and modulus operators 117–119
  - assignment 119–124
    - as value expressions 123–124
    - multiple 120–121
  - comparison 124–131
    - and case-sensitivity 127–128
    - scalar 125–127
    - using with collections 129–131
  - logical and bitwise 148–150
  - pattern matching and text manipulation 131–148
    - join operator 139–143
    - match operator 134–137
    - regular expressions 133–134
    - replace operator 137–139
    - split operator 143–148
    - wildcard patterns and -like operator 132–133
- Option Explicit, in Visual Basic 582
- Options property 624–625
- orchestrating information 461
- original tables 115
- origin-zero arrays 92
- OS information 595
- out-default 67
- Out-Default cmdlet 421, 555

- Out-Default function 421
- outer loops 214
- OuterXML property 835
- Out-File cmdlet 68, 183, 679
- Out-GridView command 69–70
- Out-Host cmdlet 69
- Outlook Express 899
- Out-Null outputter 67
- Out-Printer cmdlet 68
- output 64–70
- Output Field Separator (\$OFS) variable 103, 238, 387, 677–678
- output messages 568
- output objects 555, 568
- output pane
  - accessing contents 629
  - ISE 609
  - saving contents 629–631
- output redirection 22, 111, 181–182
- output stream 675
- OutputAssembly parameter 738
- outputter cmdlets 67–70
- OutputType attribute 293–295
- Out-String cmdlet 69
- OverloadDefinitions property 682
- overriding method 423
- overwriting output 184

## P

- P/Invoke 733
- P/Invoke signatures, interoperation with Add-Type cmdlet 734–735
- PadLeft() method 677
- panes
  - controlling layout 607–610
    - command pane 608–609
    - ISE toolbar 609–610
  - editor, expansion in 613–614
  - ISE, syntax highlighting in 614
  - text 629–633
    - making changes in editor pane 631–632
    - saving list of open files 632–633
    - saving output pane contents 629–631
- param block 416, 773
- param keyword 31, 273

- param statement 242, 279–280, 397
- .PARAMETER <PARAMETER-NAME> help tag 319
- [Parameter()] attributes 355
- [Parameter] attribute 42
- parameter 41
- parameter aliases 49
- Parameter attributes, specifying 296–303
  - HelpMessage property 302–303
  - Mandatory property 297
  - ParameterSetName property 298–299
  - Position property 297–298
  - ValueFromPipeline property 300
  - ValueFromPipelineByPropertyName property 300–301
  - ValueFromRemainingArguments property 301–302
- parameter binding 39
  - pipelines and 62–63
  - remoting 509
  - type conversion in 107–109
- parameter initializers, terminating errors in 580
- parameter sets
  - Path 737–739
  - TypeDefinition 736–737
- parameterized properties 768
- ParameterizedProperty type 403
- parameterizing functions 237
- parameters 60
  - adding 459–462
  - creating aliases with Alias attribute 303–305
  - declaring 42, 241
  - for scriptblock 108–109
  - formal, declaring for functions 241–257
  - Microsoft WMI common 802–803
    - AsJob and ThrottleLimit 803
  - processing 248
  - setting, for timer event handler 857
  - validation attributes of 305–311
    - AllowEmptyCollection 306
    - AllowEmptyString 306
    - AllowNull 306
    - ValidateCount 307–308
    - ValidateLength 308
    - ValidateNotNull 307

- parameters (*continued*)
  - ValidateNotNullOrEmpty 307
  - ValidatePattern 308–309
  - ValidateRange 309
  - ValidateScript 310–311
  - ValidateSet 310
  - vs. arguments 39
- ParameterSetName property 298–299
- parent job 490
- parentheses 157, 176
- Parse() method 106
- parsing 36, 562
  - command output using regular expressions 136–137
  - comment syntax 58–60
    - multiline 59–60
  - expression-mode and command-mode 54–56
  - quoting 51–54, 669
  - statement termination 56–58
- parsing modes 56, 243
- partial type name 724
- PassThru parameter 365, 379, 405
- Password field 790
- password parameter 917
- Password property 922
- passwords
  - in remoting 511
  - specifying for files 914
- path components 667
- path issues, relative paths 666
- Path parameter 296, 675, 737–739, 814
- path parameter 249
- \_\_PATH property 814, 821, 826
- Path property 480
- path resolution 666
- path translation 666
- path-based pattern language 703
- PATHEXT environment variable 898
- paths
  - Microsoft WMI, using to target instances 814–816
  - processing 664–672
    - containing wildcards 667–668
    - LiteralPath parameter 670–671
    - providers and core cmdlets 664–665
    - PSDrives 665–667
    - Registry provider 671–672
      - suppressing wildcard processing 668–669
    - provider-specific path 666
    - special characters 667
- pattern matching 215, 219
  - and text manipulation 131–148
    - join operator 139–143
    - match operator 134–137
    - regular expressions 133–134
    - replace operator 137–139
    - split operator 143–148
    - wildcard patterns and -like operator 132–133
  - operations 150
  - suppressing 670
- \$Pattern parameter 772
- peer-to-peer networks 891
- performance
  - caveats for Get-Content cmdlet 680–681
  - flow control 233–235
  - in remoting 465
- Perl 37, 889
- PERL scripting language 133, 582
- persistent connections, remoting sessions
  - and 462–473
    - additional session attributes 466–468
    - interactive sessions 469–472
    - managing sessions 472–473
    - New-PSSession cmdlet 468–469
- Personal Information Exchange, Certificate Export Wizard 914
- petabytes 83
- .pfx files
  - signing scripts using 915–916
  - specifying name for 915
  - verifying creation of 915
- PHP 38
- physical type containment 720
- Pi constant 101
- \$PID variable 468
- pipe operator 60
- pipeline object flows 555–556
- pipeline output, as array 91–92

- pipelines 60–63, 199, 202, 263
  - and parameter binding 62–63
  - and streaming behavior 61–62
  - index of commands in 559
  - number of commands in 559
  - processing documents in 701–702
  - steppable 418–423
  - using functions in 263–267
    - filters and functions 265–266
    - functions with begin, process, and end blocks 266–267
- PKI (Public Key Infrastructure) 905
- Platform Invoke. *See* P/Invoke
- plus operator 93
- plus-equals operator 93
- point class 429
- Point type 748
- polymorphic behavior 196, 244
- polymorphic methods 112
- polymorphism, in arrays 92–93
- Popup() method 777
- Port parameter 518
- ports, connecting to nondefault 518–519
- Position property 297–298
- positional parameters 242–243, 249
- PositionMessage property 559
- POSIX 38
- postincrement operator 80
- PowerPoint 614
- PowerShell
  - aligning with C# syntax 38
  - and overloading 245
  - and WSMAN shells 511
  - as management tool 554
  - based on objects 11
  - case-insensitivity 21
  - categories of commands 39
  - command structure 13
  - console host 467
  - core types 505
  - creation of 8
  - details on supported platforms 14
  - downloading and installing 13
  - exact vs. partial match 49
  - expressions in 23
  - extending language 428–436
    - adding CustomClass keywords 428–433
    - little languages 428
    - type extension 433–436
  - grammar 201
  - help subsystem 23
  - host application 14
  - installation directory 689
  - IntelliSense in 21
  - lookup algorithm 272
  - namespace capabilities in 267
  - parameter binding 509
    - Property parameter 28
  - provider infrastructure 187
  - registry keys 671
  - remoting host process, wsmprovhost.exe 528
  - remoting schema 505
  - script file extensions 614
  - scripts in 565
  - secondary prompt in 28
  - setting the exit code 281
  - string handling 25
  - support for .NET decimal type 24
  - syntax for script blocks 393
  - terminology similar to other shells 38
  - type system 423
  - use of full .NET Framework 10
  - using interactively 202
  - using wildcard characters with help 23
  - VBScript and Jscript 785
  - viewing constraint variables 523
- PowerShell API 546, 619
- PowerShell Development Environments 752
- PowerShell drive abstraction. *See* PSDrive
- PowerShell foundations 36–70
  - aliases and elastic syntax 47–50
  - core concepts 38–46
  - formatting and output 64–70
  - language 37–38
  - parsing 50–60
    - comment syntax 58–60
    - expression-mode and command-mode 54–56
    - quoting 51–54
    - statement termination 56–58
  - pipelines 60–63
    - and parameter binding 62–63
    - and streaming behavior 61–62

- PowerShell Heresy 60
- PowerShell Hosts 467
- PowerShell installation directory, \$PSHOME variable 702
- PowerShell interpreter, function of 39
- PowerShell Job type, infrastructure extension point 482
- PowerShell language 589
- PowerShell Local Certificate Root 912
- PowerShell provider model 664
- PowerShell quick start guide 13–21
  - command completion 20–21
  - console host 14–16
  - ISE 17–20
  - obtaining program 14
  - starting program 14
- PowerShell SDK (software development kit) 546
- PowerShell session, termination 604
- PowerShell sessions 618
- PowerShell tokenizer API 637
- PowerShell.exe 605
  - launching from ISE 611
  - running from ISE 609
  - sta parameter 753
  - WindowStyle parameter 734
- powershell.exe console host 463
- PowerShellHostName element 367, 467
- PowerShellHostVersion element 367, 467
- PowerShellTabs property 625
- PowerShellVersion element 367, 369
- precision and conversion 73, 102
- predefined engine events 875–876
- predicate expressions 707, 810
- preference setting 569
- prefix operators 94
- prescriptive error messages 587
- PresentationCore, WPF required assemblies 753
- PresentationFramework, WPF required assemblies 753
- PrimalForms PowerShell IDE 752
- primary key 814
- printf-debugging 260
- private aliases, in constrained sessions 547
- private certificate, creating 913
- private function, calling 537
- private key protection, enabling strong 913–915
  - .pfx file 915
  - exporting certificate 913
  - specifying file format and password 914
  - starting CERTMGR.EXE and selecting certificate to export 913–914
- PrivateData element 375, 385–386
- PrivateData field 385
- probing 722
- problem-solving pattern 29–30
- process blocks, functions with 266–267
- Process clause 228
- process clause 61, 267, 397–398
- Process Id, using \$PID variable 468
- process keyword 254
- Process object 508
- process streaming 62
- process use, in ISE 619
- ProcessID property 799
- ProcessId property 843
- processing 663–718
  - data 25–30
    - problem-solving pattern 29–30
    - selecting properties from objects 27–28
    - sorting objects 25–27
    - with ForEach-Object cmdlet 28–29
  - files 672–681
  - paths 664–672
    - containing wildcards 667–668
    - LiteralPath parameter 670–671
    - providers and core cmdlets 664–665
    - PSDrives 665–667
    - Registry provider 671–672
    - suppressing wildcard processing 668–669
  - unstructured text 681–693
    - counting unique words with hashtables 684–686
    - manipulating text with regular expressions 686–688
    - searching files with Select-String cmdlet 688–693
    - System.String class 681–684
  - XML structured text 693–718
    - adding elements to objects 695–697
    - loading and saving files 697–701

- processing (*continued*)
  - processing documents in pipelines 701–702
  - processing with XPath 702–709
  - rendering objects as 711–718
  - using document elements as objects 693–694
  - XLinQ library 709–710
- Process-Message cmdlet 49
- ProcessName property 305
- process-object clause 62
- processor architecture 498–501
- ProcessorArchitecture element 367
- ProcessStartInfo object 922
- production elements
  - module manifests 366–370
  - module identity 368–370
  - runtime dependencies 368–370
- ProductVersion property 33
- \$PROFILE variable 494
- profiles, and remoting 494–495
- ProgID 762
  - Apple iTunes 763
  - Microsoft Word 763
- Program Files 376
- programming constructs 198
- programming languages 393
- Progress Record, serialization in 506
- prompt function, ISE 609
- prompt line element, ISE 608
- prompting
  - for target computer 620
  - using Read-Host cmdlet 581
- prompts 15
  - nested, and Suspend operation 643–647
  - while stepping 642
- properties
  - adding to type system 425–427
  - attempts to read nonexistent 585–586
  - compression of in serialization 509
  - defined 11
  - in registry 672
  - parameterized 768
  - selecting from objects 27–28
  - shadowing existing 427–428
- Properties member 424
- property bags 497–498, 506, 508
- property checks 586
- property dereference operator 173
- property names, viewing 557
- property notation 86
- property operators 173–179
  - dot 174–177
  - indirect method invocation 178–179
  - static methods and double-colon operator 177–178
- Property parameter 727–728, 751, 761
- PropertySet type 403
- protocol layers 504
- protocol stack
  - remoting 503–509
  - representing objects and types in 505–509
- prototypes 401
- provider abstraction 672
- provider capabilities 665
- provider cmdlets, with WSMAN 510
- provider infrastructure 402, 667
- provider model 664
- Provider paths, PSPATH 665
- providers
  - and core cmdlets 664–665
  - home directories 667
  - WSMAN implementation 509–511
    - establishing remote connection 510–511
    - testing connections 510
- provider-specific path 665
- proxy commands
  - creating with steppable pipelines 420–423
  - restricting features with 547
- proxy functions
  - implicit remoting 546
  - setting up, in constrained sessions 547
- proxy servers, addressing remoting target using 520
- ProxyAccessType setting 520
- ProxyAuthentication setting 520
- ProxyCredential setting 520
- PS\* properties 402
  - .ps1 extension 44
  - .ps1xml extension 434
- PSBase member 427
- PSBase property 824
- \$PSBoundParameters variable 254, 312

- PSBreakpoint object 654
- \$PSCmdlet variable 290, 293, 386–387
- \$PSCmdlet.SessionState.Module 386
- \$PSCmdlet.ThrowTerminatingError()
  - method 293
- PSComputerName property 34
- PSCredential object 790
- PSCustomObject type 411
- PSCustomType object 433
- PSDiagnostics module 334, 363
- PSDrives (PowerShell drives) 665–667
- psEdit command 612
- PSEventSubscriber objects 859, 861
- \$PSHOME variable 64, 689, 702
- \$PSHome variable 434
- \$PSHome/modules directory 362
- \$PSHome/types.ps1xml 717
- PSIsContainer property 402
- \$psISE variable 621–623, 629
- PSModuleInfo objects 378, 381–382, 388–390, 535
  - accessing module exports using 381–382
  - invocation in module context 378–381
  - methods
    - Invoke() 382–383
    - NewboundScriptblock() 383–384
- PSModuleObject 378
- \$PSModuleRoot 358
- PSObject class 423
  - PSBase member 695
  - synthetic object root 404
- PSObject layer 76
- PSObject property 423
- PSParser class 589
- PSPath, provider paths 665
- PSScriptMethod object 430
- PSScriptProperty 427
- \$PSScriptRoot variable 358
- PSSession attributes 466
- PSSession object, and runspaces 619
- PSSessionConfiguration cmdlet 531
- \$PSSessionConfigurationName variable 530
- \$PSSessionOption variable 519–520, 523, 526
- PSSessions type 463
- PSTypeConverter type 106

- \$psUnsupportedConsoleApplications variable 617
- PSVariable objects, using as references 191–192
- public decryption key 904
- public fields 400
- public key cryptography 722
- public key encryption 904–905
- Public Key Infrastructure (PKI) 905
- public keys, use in type names 724
- public members 400–402
- public methods 400
- public properties 400
- pure function 262
- pure synthetic objects 411, 433
- Put() method 828–829
  - PutType parameter 814
- Python 180
  - comparison to Visual Basic 39
  - regular expressions 133
  - security 889
- Python interpreter 271
- Python lambdas 397

## Q

- ql function 239–240
- qs function 239–240
- qualifiers 815
- Qualifiers attribute 815
- Query parameter, selecting Microsoft WMI objects
  - using 812–813
- query-based event registrations, Microsoft WMI 871–875
  - aggregating events with GROUP
    - keyword 874–875
  - Microsoft WMI intrinsic event classes 871–874
  - WITHIN keyword 871
- Queue type, serialization in 507
- queued events
  - and Wait-Event cmdlet 863–866
- quiet parameter 690–691
- quotas, managing resource consumption
  - with 523–524
- quotation marks 41
- quote list (ql) 239
- quote removal 669
- quoting 51–54, 669

## R

- range operator 165, 167, 170–171
- rank 172
- RBAC (role-based access control) 894
- read method 581
- Read mode, variable breakpoints 657
- ReadCount parameter 229–231, 677, 697
- read-evaluate-print loop 6
- Read-Host cmdlet 581, 917
- reading
  - files 674–679
    - Get-Content cmdlet 674–676, 680–681
    - Get-HexDump function example 676–677
    - Get-MagicNumber function example 677–679
    - writing files 679
  - key strokes 581
- ReadLine() method 581
- ReadOnly option 190
- Really Simple Syndication. *See* RSS
- Receive-Job cmdlet 483–484, 492
- recording errors 560
- Recurse parameter 809
- recurse parameter 249
- Recurse switch 41
- recursive definition 262
- recursive directory listing 249
- red stop button 18
- redefine functions 645
- redirection 276, 278, 568
  - error stream 555
  - into variables 556
  - merging output and error streams 556
  - redirecting error stream 560
  - stream merge operator 556
- redirection operator 24, 125, 181–184, 187, 555
- reducing attack surface 893
- reference types
  - array as 93–94
  - hash tables as 90–91
- references, using PSVariable objects as 191–192
- [regex] alias 97
- [regex] class 687, 851
- [regex] type 687
- regex flag 218
- Regex.Replace(String, MatchEvaluator) 852
- Register-EngineEvent cmdlet 854–855, 878
- Register-ObjectEvent cmdlet 854–855, 858–860, 879
- Register-PSSessionConfiguration cmdlet 523, 532
- RegisterTaskDefinition() method 789–790, 792
- Register-WmiEvent cmdlet 854–855, 873
- RegistrationInfo property 788
- registrations
  - asynchronous events 854
  - engine events 875–877
  - Microsoft WMI events
    - class-based 867–870
    - query-based 871–875
- Registry entry, for
  - LocalAccountTokenFilterPolicy 517
- Registry provider 665, 667, 671–672
- regular expressions 132–134, 218
  - alternation operator 687
  - creating from strings 687
  - default match 135
  - extracting text with 136
  - manipulating text with 686–688
    - splitting strings 687
    - tokenizing 687–688
  - Match method 687
  - matching any character 137
  - matching the beginning of a string 137
  - parsing command output using 136–137
  - quantifier specifications 687
  - submatches 134
  - using with switch statement 217–221
- rehydrated, serialization 505
- relative path resolution 667
- remainder modulus 120
- remote computers, starting background jobs
  - on 489–492
- remote connection, starting in ISE 620
- remote debugging, tracing script execution 658
- remote output, vs. local output 497–498
- remote session startup directory 494
- remote sessions 619–622
- Remote tab architecture 622
- .REMOTEHELPRUNSPACE <PSSession-VARIABLE> help tag 320

- remoteserver application 546
- RemoteSigned policy 337, 899
- remoting 32, 46, 447–502
  - additional setup steps for workgroup environments 451
  - and event forwarding 877–882
  - applying 454–462
    - basic remoting examples 454–455
    - multimachine monitoring 457–462
  - commands with built-in 448–449
  - configuration elements 530
  - configuration startup script 532
  - considerations when running
    - commands 493–501
    - executables 495–496
    - processor architecture 498–501
    - profiles and remoting 494–495
    - reading and writing to console 496–497
    - remote output vs. local output 497–498
    - remote session startup directory 494
  - cross-domain issues 517
  - custom services 502, 527–552
    - access controls and endpoints 533–535
    - configurations 530–531
    - constrained execution environments 543
    - constraining sessions 535–543
    - remote service connection patterns 527–530
  - disabling 897
  - enabling 450–451
  - EventLog access 602
  - implicit 473–481
  - infrastructure 503–527
    - addressing remoting target 518–520
    - authenticating 511–518
    - managing resource consumption 522–527
    - Microsoft Windows connection issues 520–522
    - remoting protocol stack 503–509
    - WSMan implementation cmdlets and providers 509–511
  - interactively 34
  - performance issues 465
  - persistent connections 463, 465
  - requirement for elevated sessions 517
  - resource management 472
  - sessions and persistent connections 462–473
    - additional session attributes 466–468
    - interactive sessions 469–472
    - managing sessions 472–473
    - New-PSSession cmdlet 468–469
  - subsystem 449–450
  - target machine 33
  - timeouts 526
  - transient connections 463
- remoting infrastructure
  - buffering 457
  - connection heartbeat 473
  - support for throttling 522
- remoting protocols
  - [MS-PSRP] 504
  - complete protocol stack 504
  - DCOM 448
  - HTTPS 504
  - Web Services for Management (WSMan) 504
- Remove All Breakpoints debug menu item 649
- Remove()method 635
- Remove-Event cmdlet 854, 865
- Remove-Item cmdlet 670, 672–673
- Remove-Item command 268
- Remove-Job cmdlet 483, 487
- Remove-Module cmdlet 325, 335, 339, 388
- Remove-PSBreakPoint cmdlet 653
- Remove-PSSession cmdlet 472
- Remove-WmiObject cmdlet 822–824
- removing class definition 432
- removing items, hash tables 88
- Rename-Item cmdlet 673
- renaming functions 399
- rendering objects 69
- REPL. *See* read-evaluate-print loop
- replace operator 134, 137–139, 632, 687, 732, 881
- Replace() method 851–852
- replacement strings, in event log entries 604
- repudiation, defined 892
- RequiredAssemblies element 370, 372, 375, 722
- RequiredAssemblies field 372
- RequiredAssemblies module 373
- RequiredModules element 367, 370, 372
- RequiredServices property 713

- Reset() member 210
- Reset-Count command 338, 342
- resizing arrays 259
- resource consumption, managing 522–527
- resource leaks, handles and garbage collection 701
- resource management, using throttling 458–459
- resources
  - singleton, vs. enumerations 836–837
  - updating using Set-WSManInstance cmdlet 840–841
- restricted execution policy 899
- restricted language, in constrained sessions 539
- RestrictedLanguage Mode 542
- restrictions, in sessions 541
- \$result variable 259
- resuming execution, after exceptions 574
- return statement 227, 262–263, 280
- returning function objects, ScriptControl 785
- reusable configuration script, for custom remoting configurations 544
- reverse arrays 406
- Reverse member 170
- reverse method 140, 406
- reversed in place, arrays 406
- rich error objects 554
- right aligned 180
- right operand 126, 153
- role-based access control (RBAC) 894
- roles, authentication, authorization, and 894–895
- root directory 111
- Root module term 326
- rootcimv2 namespace 800
- RootSDDL security descriptor, remoting access control 534
- RPC server not available error 802
- RSS (Really Simple Syndication) 4
- RSS feeds, processing 742–743
- Ruby language 400
- Run() method 778
- Run/Continue debug menu item 648
- runas.exe command 920
- running elevated 277
- running scripts from ISE, F5 key 649
- RunspaceId 862
- runspaces, defined 619

- runtime checks, in scripts 582
- runtime dependencies, module manifests 368–370
- runtime type casts 153
- runtime type conversion error 185

## S

- sample taxonomy, of classes and subclasses 12
- sandboxing, defined 897
- saving, files 697–701
- scalability
  - in scripting 459
- scalar arguments 238
- scalar comparisons 125–127
  - basic rules for 126
  - type conversions and comparisons 126–127
- scalar object 96, 210
- scalar value 95, 125, 210–211
- scale() method 433
- Scaling fan-in remoting, Issues 528
- Schedule.Service class 786–787
- scientific notation 75
- scopes
  - and scripts 281–284
    - dot-sourcing scripts and functions 283–284
    - simple libraries 283
  - in script modules 348–350
  - managing with script blocks 572
- scoping
  - behavior, F5 vs. F8 615
  - execution policy 900–903
  - rules 269, 281
  - variable, in functions 269–274
- script authoring, control of errors 569
- script block construction 398
- script block execution, in debug actions 654
- script blocks
  - as event handlers 746
  - defines a function 399
- script calls, calling another script 562
- script checking 582
- script code
  - building at runtime 436–440
    - \$ExecutionContext variable 437
    - creating elements in function drive 439–440
    - ExpandString() method 437–438

- script code (*continued*)
  - Invoke-Expression cmdlet 436–437
  - InvokeScript() method 438
  - script blocks 438–439
    - fragments of 569
- script commands 39
- \$Script counter variable 863
- script editor 18
- script line number, getting 559
- script modules
  - dynamic 412–414
  - nesting binary modules in 357–360
  - setting module properties from inside 388
  - writing 337–353
    - controlling member visibility with Export-ModuleMember cmdlet 343–347
    - installing module 347
    - nested modules 350–353
    - review of scripts 338–340
    - scopes in script modules 348–350
    - turning into module 340–343
- script name, getting 559
- Script Property
  - getter method 407
  - setter method implementation 408
  - setter script block 407
- script scope 281
- script tracing 639, 641
- script versioning 49
- [ScriptBlock]::Create() method 642
- ScriptBlock 506
- [scriptblock] alias 97
- [scriptblock] type accelerator 439
- scriptblock parameter 685
- scriptblocks 223, 226, 228, 393–400, 438–439
  - creating new scopes 572
  - defining functions at runtime 398–400
  - invoking commands 394–396
  - literals 397–398
  - parameters for 108–109
  - security 923
  - targeted execution in ISE 627
  - using with remoting 454
  - using with -split operator 148
- scripting debugger 653
- scripting languages 553
  - features of 7
  - security 890
  - vs. shell, advantages 7
- ScriptMethod members, adding to
  - objects 406–407
- ScriptMethod type 403
- ScriptProperty members, adding to
  - objects 407–408
- ScriptProperty type 403
- scripts 31–32, 44, 275–321
  - advanced functions and 276–287
    - documenting 314–321
    - dynamic parameters and dynamicParam keyword 311–314
    - exiting scripts and exit statement 280–281
    - managing scripts 284–285
    - passing arguments to scripts 278–280
    - running scripts from other applications 285–287
    - scopes and scripts 281–284
    - writing 287–311
  - applying strict mode V2 to 588–589
  - blocks, handling asynchronous events with 860–863
  - calling from another script 562
  - controlling access to 541
  - debugging 638–647
    - nested prompts and Suspend operation 643–647
    - Set-PSDebug cmdlet 638–643
  - enabling with execution policy 898–903
    - controlling and scoping 900–903
    - settings 899–900
  - execution policy 276–278
  - exit code 566
  - flow control in. *See* flow control
  - generating events in 876–877
  - hello world file 31
  - review of 338–340
  - running from cmd.exe 286
  - running in ISE 616–618
    - issues with native commands 616–617
    - threading differences between console and ISE 618

- scripts (*continued*)
  - signing 904–916
    - authorities 905
    - certificates 905
    - enabling strong private key protection 913–915
    - public key encryption and one-way hashing 904–905
    - using .pfx files 915–916
  - static analysis of 589–593
  - suspending while in step mode 644–645
  - using certificates to sign 909–912
    - setting up test script 909–910
    - signing test script 910–912
    - testing integrity of script 912
  - wheres
    - original 923–925
    - safer and faster version 925–927
  - writing secure 916–927
    - avoiding Invoke-Expression cmdlet 923–927
    - credentials 919–923
    - SecureString 916–919
- Scripts property, From
  - \$ExecutionContext.SessionState 539
- scripts, no execution of
  - by default 897–898
- script-scoped 282
- ScriptsToProcess element 370, 372–373
- ScriptToProcess 372
- SDDL (Security Descriptor Definition Language) 534
- search algorithm, modules 329–330
- search tools, file 754
  - defining appearance 754–756
  - specifying behavior 756–758
- searcher object 825
- Search-Help function 702
- searching
  - files, with Select-String cmdlet 688–693
  - with context properties 692
- secpol.msc 521
- secure by default principle 450
- secure computer 889
- secure environment 891
- Secure Hash Algorithm version 1 (SHA-1) 907
- secure hashing algorithm 889, 904
- secure remoted service, creating 551
- secure scripts 888
  - credentials 919–923
  - SecureString
    - class 916–917
    - cmdlets 918–919
    - object 917–918
- Secure String, serialization in 506
- securing PowerShell installations 916
- security 888–927
  - by default 897–898
    - disabling remoting 897
    - managing command path 898
    - no execution of scripts 897–898
    - notepad 897
  - introduction to 889–891
    - Danom virus 890–891
    - MSH/Cibyz worm 891
  - modeling 891–897
    - threat 891–892
    - threats, assets, and mitigations 893–897
  - modeling concepts 888
  - scripts
    - enabling with execution policy 898–903
    - signing 904–916
    - writing secure 916–927
- Security Descriptor Definition Language (SDDL) 534
- security descriptors, setting on
  - configurations 534–535
- security implications
  - Enable-PSRemoting cmdlet 451
  - TrustedHosts list 452
- security model 893
- Security Options, secpol.msc 521
- security response, Microsoft 890
- select command 34
- select elements 410
- selected text, executing in ISE 615–616
- selecting, instances 838–839
- Select-Members filter 726
- Select-Object cmdlet 378, 423, 679
  - defined 27
  - getting first lines of file with 911
  - selecting range of objects 410–412
  - using -Property parameter 28

- SelectorSet parameter 838, 840
- Select-String cmdlet 741
  - Quiet switch 690
  - searching files 688
- Select-Xml cmdlet 703–709
- self-describing, .Net object model 10
- Self-Remoting Commands
  - Clear-EventLog 448
  - Get-Counter 448
  - Get-EventLog 448
  - Get-HotFix 448
  - Get-Process 448
  - Get-Service 448
  - Get-WinEvent 448
  - Limit-EventLog 448
  - New-EventLog 448
  - Remove-EventLog 449
  - Restart-Computer 449
  - Set-Service 449
  - Show-EventLog 449
  - Stop-Computer 449
  - Test-Connection 449
  - Write-EventLog 449
- self-signed certificate 909
- semicolon character 56, 201, 204, 257
- Sender field 877
- \$Sender variable 861
- \$Sender.Event 861
- sending keystrokes 765
- SendKeys() method 778
- sensitive data, security 916
- separator 280
- serialization
  - core types 506
  - default depth 498, 713
  - IList interface 507
  - in systems management 505
  - object fidelity 716
  - of collections 507
  - shredding objects 716
  - using property bags 506
- <SerializationDepth> element 717
- serialized objects 46
- servers, proxy 520
- service architecture 527
- Service subnode, of WSMAN drive 525
- ServiceName property 305
- Services, ServiceController objects 712
- servicing. *See* Application servicing
- session boundary 541
- session isolation, in remoting 530
- Session parameter 492–493
- SessionOption parameter 520
- sessions 463
  - and hosts 467
  - capturing output 593–596
  - configurations 531–532
  - constraining 535–543
    - controlling command visibility 536–539
    - setting language mode 539–543
  - interactive 469–472
  - isolation 467
  - local in-memory 619
  - managing 472–473
  - managing definitions in 267–269
  - remoting 619–622
    - additional attributes 466–468
    - and persistent connections 462, 468–473
    - running background jobs in existing 492–493
- Set-Alias command 48
- Set-AuthenticodeSignature cmdlet 910
- Set-Content cmdlet 188, 673
- Set-CountIncrement 384
- Set-ExecutionPolicy cmdlet 277–278
- setIncrement function 338, 345
- Set-Item cmdlet 673
  - in constrained sessions 542
  - setting TrustedHosts list 513
- Set-Location cmdlet 673
- Set-PSBreakPoint cmdlet 653
- Set-PSDebug cmdlet 582, 638–643
  - Off parameter 582
  - statement execution 642–643
- Set-PSDebug, -Trace parameter 582
- Set-PSSessionConfiguration command 533
- Set-Service cmdlet 453
- Set-StrictMode cmdlet 584–589
  - applying strict mode V2 to scripts 588–589
  - attempts to read nonexistent properties 585–586

- Set-StrictMode cmdlet (*continued*)
  - empty variable references in strings 587–588
  - functions called like methods 586–587
  - uninitialized variable use in string
    - expansions 584–585
- settable property 405
- setting breakpoints, on lines in script 654
- setting token colors, syntax highlighting 625
- setting window styles 734
- Set-Variable cmdlet 379, 387, 651
- Set-Variable command 379
- Set-WmiInstance cmdlet 813–819
  - setting instance properties 816–819
  - using Microsoft WMI paths to target
    - instances 814–816
- Set-WSManInstance cmdlet, updating resources
  - using 840–841
- SHA-1 (Secure Hash Algorithm version 1) 907
- shadowing existing properties 427–428
- shared libraries 721
- shell environments 62
- shell function commands 39
- shell language 9
- Shell.Application class 765–766
- Shell.Application object 772
- Shell.Automation class 795
- ShellExecute API 922
- shells
  - as command-line interpreter 6
  - reasons for new model 7–8
    - managing windows through objects 7–8
  - scripting languages vs. 6–7
  - text-based 10
- Shift-F10, displaying context menu 612
- ShouldProcess() method 290
- Show switch 773
- \$showCmdlet module 358
- ShowDialog() method 747, 749, 776
- Show-ErrorDetails function 559
- Show-EventLog cmdlet 597
- ShowSecurityDescriptorUI parameter 535
- ShowToolBar, ISE menu item 625
- ShowWindow() API 772
- ShowWindow() method 734
- shredding objects 506, 716
- shutdown command 495
- side-by-side mode, ISE 610
- side-effects, of for statement 206
- signature information, security 911
- signatures, decrypting 904
- signing
  - authorities 905
  - scripts 904–916
    - certificates 905
    - enabling strong private key
      - protection 913–915
    - public key encryption and one-way
      - hashing 904–905
    - using .pfx files 915–916
- simple assignment 89
- simple matching 145
- Simple Object Access Protocol (SOAP) 504
- simplematch option 145
- single precision 74
- single quotes 53, 437
- single-instance objects, modules 326–327
- singleline mode 147
- single-quoted, strings 78–79
- single-threaded apartment. *See* STA 618
- Singleton member, creating definitions 730–734
- singleton resources, vs. enumerations 836–837
- Skip3 functions 681
- slicing 167–171
  - arrays 169–170
  - multidimensional arrays 172
  - using range operator 170
- sliding window 803
- snap-in modules, binary modules vs. 354–355
- snap-in, MMC extension 32
- Snippets menu
  - extending ISE 637
  - submenu for 637–638
- Snover, Jeffrey, PowerShell architect 118
- SOAP (Simple Object Access Protocol) 504
- software development kit (SDK), PowerShell 546
- sorting
  - hash tables 87–88
  - in descending order 26
  - objects 25–27
  - UNIX sort command 26

- Sort-Object cmdlet 25, 27, 87, 684
- Soul of a New Machine (Kidder) 9
- Source filter 599
- Source parameter, on Get-Eventlog 602
- \$SourceArgs variable 861
- \$SourceEventArgs variable 861
- SourceIdentifier 856, 859–860, 865, 878
- special behaviors operators 112
- special characters, using backtick 54
- special variable 209
- special-purpose applications, using remoting 530
- special-purpose endpoint 537
- spell checking, using Microsoft Word 781–783
- Spelling dialog box 783
- spelling errors 782
- splatting 839
  - in proxy functions 478
  - specifying remote connection settings 519
  - variables 193–197
- split operator 143–148, 631, 638, 681
  - options for 145–148
  - using scriptblocks with 148
- Split() method 681–682, 687
- SplitStringOptions parameters 682–683
- splitting strings, with regular expressions 687
- spoofing, defined 892
- SQL injection attacks 896
- SQL query 896
- square brackets 174
- ssh. *See* Invoke-Command cmdlet
- STA (single-threaded apartment) 618, 793
- sta parameter 618, 753
- Stack type, serialization in 507
- StackPanel layout control 755
- standard classes, WMI 807
- Start() method 859
- Start-Job cmdlet 483, 803
- Start-Job method 882–883
- Start-LocalUserManager command 920
- Start-LocalUserManager function 921
- Start-Process cmdlet 734, 743, 869, 921
- Start-Program function 666
- Start-Sleep cmdlet 456, 526, 627, 875
- StartTime property 128
- Start-Transcript cmdlet 593
- startup directories, remote session 494
- startup script, remoting 532
- state, event handler 862–863
- statement termination 56–58
- <statementList> section 204
- statements
  - as values 231–233
  - flow-control 223, 231
- static analysis, of scripts 589–593
- static checks, in scripts 582
- Static flag 99
- static members 170
  - accessing 177
  - accessing with literal 99–101
- static methods 177–178
  - calling 819–822
  - reference operator 177
- static reverse method 406
- static script checks 591
- static typing 72
- status line, ISE 608
- status variables 564
- stderr 259
- Step Into debug menu item 648
- step mode, suspending scripts while in 644–645
- Step Out debug menu item 648
- Step Over debug menu item 648
- Step parameter 642
- Step-Over debug command 650
- steppable pipelines 418–423
  - creating proxy command with 420–423
  - in proxy functions 478
- stepping mode 645
- stepping script 644
- stepping through scripts 638
- Stop Debugger debug menu item 648
- Stop-Job cmdlet 483
- Stop-Job method 883
- Stop-Process 869
- Stop-Transcript cmdlet 593–594
- stream combiner 183
- Stream parameter 69
- streaming behavior 43, 61–62

- strict mode 638
  - applying V2 to scripts 588–589
  - catching errors with 582–584
    - undefined variables 583–584
    - V1 582
  - In PERL 582
    - version 2 584, 588–589
- Strict parameter 762, 794
- Strict switch 761
- STRIDE model 892
- [string] class 681, 686
- [string].Join method 177
- string constructor 729
- string context 129
- string expansion 238
  - empty delimited variables 587
  - suppressing 437
  - uninitialized variable error 585
- string multiplication 116
- string operations
  - convert array to string 677
  - extracting fields from string 682
  - formatting hexadecimal numbers 677
  - joining strings 683
  - padding strings 677
  - parsing arithmetic expressions 687
  - splitting and joining strings 681
  - splitting into words 684
  - splitting on Whitespace character class 682
  - splitting strings 678
  - tokenizing strings 687
- String operations, casting to regular
  - expressions 687
- strings 58, 77–82
  - adding together 25
  - concatenation of 25, 113, 240, 683
  - empty variable references in 587–588
  - encoding used in 77–78
  - executing 437
  - here-strings 80–82
  - joining 177
  - single and double-quoted 78–79
  - splitting, with regular expressions 687
  - subexpression expansion in
    - complex 79–80
    - considerations for 80
- strong naming 722
- strongly typed languages 185
- structured error handling 554
- structured programming 213
- structured text, processing 693–718
- subclassing 401
- subdirectories, and dir command 41
- subexpression expansion, in strings 79
  - complex 79–80
  - considerations for 80
- subexpression operator 196
- subexpressions 57, 202, 206, 247
  - array 160–162
  - function of 157
  - with throw statement 580
- subkeys, in registry 671
- submatches 134
- Submenus collection, ISE object model 635
- SubscriptionId property 860
- subscriptions
  - .NET events 859–860
    - listing 859
    - removing 859–860
  - asynchronous events 854
- subshell 645
- Substring method 176
- subsystem remoting 449–450
- subtraction operation 120
- subtraction operator 117–119
- Success Audit type 599
- Success property 688
- Sum() method 264, 434–435, 732
- SumMethod.ps1xml file 435
- superclasses 867
- SupportEvent 856
- SupportEvent switch 856
- SupportsShouldProcess property 290–293
- Suspend operation, nested prompts and 643–647
- Suspended shell feature 645
- suspending sessions 643
- swapping files 188
- swapping two variables 120
- [switch] alias 97
- [switch] type 249
- \$switch loop enumerator, using in switch
  - statement 222

- switch parameters 41
  - using to define command switches 248–252
  - vs. Boolean parameters 252–257
- switch statement 30, 148, 199, 250, 781, 925
  - processing files with 221–222
  - using \$switch loop enumerator in 222
  - using regular expressions with 217–221
  - using wildcard patterns with 216–217
- switch value 216
- \$switch variable 222, 263, 701
- \$switch.movenext() method 222
- SwitchParameter type 107
- synchronous events 849–853
  - delegates and delegation 850–853
  - in GUIs 850
- .SYNOPSIS tag 317
- syntactic analysis 50
- syntactically complete statement 57
- syntax
  - for programmer-style activities 176
  - of foreach statement 207
- syntax checking custom menu item 637
- syntax errors 201, 592
- syntax highlighting
  - in ISE panes 614
  - setting token colors 625
- synthetic member objects 402, 404, 411, 427, 430
- synthetic properties 76, 402
- system dialogs 621
- system drives 672
- system health monitoring, remoting example 462
- System. ComObject type 767
- System.Array, extending 435
- System.Collections namespace 720
- System.Collections.ArrayList class 261
- System.Collections.ArrayList type 507, 560
- System.Collections.ArrayList.Add() method 720
- System.Collections.Generic.List 739
- System.Collections.Hashtable type 86
- System.Collections.IDictionary interface 85, 507
- System.Collections.IEnumerator interface 210
- System.Console APIs 496
- System.DateTime class 119
- System.Datetime type 118
- System.Delegate class 746, 850–851
- System.Diagnostics.EntryWrittenEventArgs 880
- System.Diagnostics.Process class 128
- System.Drawing assembly 748
- System.EventHandler class 746, 850–851
- System.Guid class 762
- System.Int32 type 96
- [System.IO.DirectoryInfo] object 585
- [System.IO.FileInfo] object 585
- System.IO.FileInfo objects 208
- [System.IO.FileInfo] type 586
- System.IO.FileSystemWatcher class 864
- System.IO.StringReader instance 776
- System.Management.Automation namespace 96
- System.Management.Automation.CommandInfo type 394
- System.Management.Automation.PSCustomObject type 409, 411
- System.Management.Automation.PSEventArgs 861
- System.Management.Automation.PSObject 404
- [System.Management.ManagementPath] object 830
- [System.Math] class 100
- [System.Math] type 727
- [System.Net.WebClient] type 740
- System.Object, root of object hierarchy 404
- System.Reflection.Assembly class, loading assemblies with 723–725
- System.Security.SecureString class 916
- System.Security.SecureString type 920
- System.String class 100, 681–684
  - analyzing word use in documents 683–684
  - SplitStringOptions parameters 682–683
  - testing types 404
- System.Text.RegularExpressions.Match class 687
- System.Text.RegularExpressions.Regex class 687
- System.Timers namespace 726
- System.Timers.ElapsedEventHandler class 726
- System.Timers.Timer class 856
- System.Version 368
- System.Windows.Forms namespace 723
- System.Windows.Window namespace 756
- System.XML.XmlDocument class 694
- System.Xml.XmlReader class 698
- SystemRoot environment variable 186
- SysWoW64 directory 499

## T

- \$t variable 408
- tab behavior, ISE 612
- tab completion 910
  - auto-correcting of capitalization 21
  - in editor pane 613
  - in ISE 617
  - on partial cmdlet names 20
  - on properties 21
  - on properties in variables 20
  - on variables 20
  - on wildcards 20
  - user extendable 21
  - within file system 20
- Tab key
  - for tab completion 13
  - using 20
- TabExpansion function 21
- tablet computers 610
- tabs 625–629
  - Editortabs 18
  - expansion in editor pane 613–614
  - using multiple 618–622
    - local in-memory session 619
    - remote session 619–622
  - working with in ISE 629
- tags, used in comments 318–321
- tampering with data, defined 892
- target computer, prompting for 620
- target object 561
- TargetObject property 558, 564
- Task Scheduler window 792
- Task Scheduler, Microsoft Windows. *See* Microsoft Windows, Task Scheduler
- tasks
  - creating new 788–789
  - credentials and scheduled 789–792
  - listing running 787
  - viewing life cycle of 792–793
- taskschd.msc 792
- tb function 252
- TCL/TK (Tool Command Language/Tool Kit) 743
- Technet website 674
- telecommunications 8
- telnet. *See* Invoke-Command cmdlet
- \$tempFile 881
- temporary file 187
- terabytes 83
- terminate partial operation 569
- Terminate() method 290, 823, 827, 842
- terminating errors 554, 566, 569
  - exception 570
  - generating in script 578
  - rethrowing 571
- terminating, PowerShell session 604
- terminator characters 56
- terminology 38
- test document 703–704
- test(1) command 125
- Test-ModuleContext module 386
- Test-ModuleManifest cmdlet 325, 365, 377
- Test-Path cmdlet 184
- TestProperty variable 817
- tests, scripts
  - integrity of 912
  - setting up 909–910
  - signing 910–912
- Test-Script function 590
- \$testv module 387
- \$testv variable 386–387
- Test-WSMan cmdlet 509
- text
  - inserting in ISE editor buffer 630
  - processing 663, 693
  - processing unstructured 681–693
    - counting unique words with hashtables 684–686
    - manipulating text with regular expressions 686–688
    - searching files with Select-String cmdlet 688–693
    - System.String class 681–684
  - selected, executing in ISE 615–616
  - XML structured, processing 693–718
- text manipulation, pattern matching and 131–148
  - join operator 139–143
  - match operator 134–137
  - regular expressions 133–134
  - replace operator 137–139

- text manipulation, pattern matching and (*continued*)
  - split operator 143–148
  - wildcard patterns and -like operator 132–133
- text panes 629–633
  - making changes in editor pane 631–632
  - saving list of open files 632–633
  - saving output contents 629–631
- Text property 630, 757
- text-based shells 10
- TextBox controls 757
- \$this member 407
- \$this variable 407, 745
- this.MyInvocation.MyCommand.Module 385
- this.SessionState.Module 386
- Thompson, Ken 133
- threading model
  - defined 618
  - problems in COM 793
- threat modeling 891–892
- threat to systems, defined 892
- threats, assets, mitigation, and 893–897
  - authentication, authorization, and roles 894–895
  - avoiding lawn gnome mitigation 893–894
  - blacklisting/whitelisting 894
  - code injection 896–897
  - input validation 895–896
- ThrottleLimit parameter 522, 803
- throttling, resource management using 458–459
- throw keyword 702
- throw statement 248, 578–580
- timeout interval 526
- Timeout parameter 863
- timeouts, setting on operations 524–527
- <TIMER>message 877
- timer event handler, writing 856–859
  - binding event action 857–858
  - creating timer object 856
  - enabling 858–859
  - setting parameters 857
- \$timer.Stop() method 858
- TlntSvr process 870, 872
- Toggle Breakpoint debug menu item 649
- Tokenize() method 589
- tokenizer analyzer 50
- Tokenizer API 589, 591
- tokenizing text 687–688
- tokens 50, 589, 591
- Tool Command Language/Tool Kit (TCL/TK) 743
- toolbar object 625
- toolkit, for building custom solutions 8
- top-level execution thread 646
- top-level match 135
- top-level properties, in ISE object model 623
- ToString() method 418, 429, 433, 438, 508, 572, 641, 679
- ToUpper() method 427
- trace message format 641
- trace mode 639
  - Trace parameter, on Set-Debug cmdlet 582
- Trace-Command cmdlet 63
- tracing function calls 640
- traditional dynamic scoping 270
- tradnum function 258
- transcript facility, in ISE 629
- transcript file 595
- transcript implementation 593
- transcript of script traces 641
- transcripts, session output captured with 595–596
- transformation 408
- transitional aliases 48
- transport mechanism, remoting 503
- trap statement 570–575
  - control transfer 572
  - flow chart 573
  - flow of control 573
- trees of files 691
- triggered clause 219
- Trojan Horse attack, defined 898
- \$true variable 184
- trust relationship, in remoting 511
- trusted certificate authority, role in remoting 512
- trusted third party organizations 905
- TrustedHosts element 514
- TrustedHosts list
  - authenticating target computer using 512–514
- try keyword 575
- try to pop GUI 496
- try/catch statement 570, 575, 772

- try/catch/finally statement 578
- try/finally statement 881
- type accelerators
  - Microsoft WMI 825–828
    - [WMI] type accelerator 826–827
    - [WMICLASS] type accelerator 827–828
    - [WMISEARCHER] type accelerator 825
  - WMI 828
- type aliases 96–98
- type already exists error 738
- type command 47
- type configuration files 434
- type constrained variables 586
- type constraints
  - adding to parameters 243–245
  - multiplication and arrays 117
- type conversion error 114
- type conversion operators 112
- type conversions 245
  - and comparisons 126–127
  - in multiple assignment 123
  - tracing mechanism 127
  - with XML documents 694
- type files, loaded at startup 434
- type inferencing 73
- type library 795
- type literal 153, 177
- type metadata 505
- Type parameter 557
- type parameters 739
- Type property 372
- type qualifiers, multiple assignment operators
  - with 121–123
- type references, assembly manifest 721
- type resolution 97
  - in complied programs 722
  - static linking 721
- type system 423–428
  - adding properties to 425–427
  - shadowing existing properties 427–428
  - updating definitions 435
- type-constrained function 244
- type-constrained variable 96, 115
- TypeConverter type 106
- TypeDefinition parameter set 736–737
- typeless languages 72
- typeless parameters 101, 243
- typelibs, problems in COM 793–796
- TypeNames member 424
- TypeNames property 497
- typeof() operator 178
- types 72–109
  - arrays 91–96
    - as reference types 93–94
    - collecting pipeline output as 91–92
    - empty arrays 94–96
    - indexing of 92
    - polymorphism in 92–93
  - conversions of 101–109
    - .NET-based custom 104–107
    - built-in 104
    - in parameter binding 107–109
    - overview 101–104
  - creating instances of 727–729
  - defining 429
  - defining new with Add-Type cmdlet 729–739
    - creating Singleton member
      - definitions 730–734
      - interoperation with P/Invoke
        - signatures 734–735
    - Path parameter set 737–739
    - TypeDefinition parameter set 736–737
  - enum, defining new at runtime 442–443
  - explicit operations 152
  - extending 433–436
  - finding 725–727
  - generic 739–740
  - hash tables 85–91
    - as reference types 90–91
    - enumerating 87–88
    - modifying 88–89
    - sorting 87–88
  - implicit operations 152
  - literals 96–101
    - accessing static members with 99–101
    - generic types 98–99
    - type aliases 96–98
  - management of 72–77
  - numeric 82–85
    - hexadecimals 84–85

- multiplier suffixes for 83–84
- specifying 83
- operators for working with 152–154
- representing in protocol stack 505–509
- strings 77–82
  - complex subexpressions in 79–80
  - encoding used in 77–78
  - expansion considerations for 80
  - here-strings 80–82
  - single and double-quoted 78–79
  - subexpression expansion in 79
- types files, default installed 434
- \$typeSpec variable 881
- TypesToProcess element 370, 373

## U

- UAC (User Access Control) 903
- UAC (User Account Control) 517
- unary comma operator 728
- unary operators 154–157
- unauthorized scripts 893
- unconstrained function 244
- undefined command 592
- undefined variable 208
- underlying store, WMI 828
- Unicode 78
- unified namespaces 184
- Uniform Resource Identifiers. *See* URIs 519
- uninitialized variables 184, 583
- Universal Execution Model 32
- UNIX command equivalents 673
- UNIX environment 428
- UNIX shell, convenience aliases 673
- unqualified operators, case insensitive by
  - default 125
- unraveling collections 210
- Unregister-Event cmdlet 854, 860
- Unregister-PSSessionConfiguration cmdlet 531
- unrestricted execution policy 900
- unsigned scripts 909
- unstructured text, processing 681–693
  - counting unique words with
    - hashtables 684–686
  - manipulating text with regular
    - expressions 686–688

- searching files with Select-String
  - cmdlet 688–693
  - System.String class 681–684
- Unsubscribe-Event 859
- unsupported application list
  - in ISE 617
- untrusted directory 898
- Update-Character function 255
- Update-TypeData cmdlet 435, 880–881
- updating
  - TrustedHosts list 513
- updating, ISE menu items 635
- URIs (Uniform Resource Identifiers)
  - addressing remoting target using 519
  - targeting WS-Man resources using 834
- usability testing 619
- User Access Control (UAC) 903
- User Account Control (UAC) 517
- User filter 599
- user portability aliases 673
- user profile, in remote sessions 470
- UserName property 922
- users
  - authenticating 514–518
    - enabling remoting for administrators in other
      - domains 517–518
    - forwarding credentials in multihop
      - environments 515–517
- usesCount module 351–352
- usesCount.psm1 module 350
- usesCount2 353
- using debugger, example 649
- UsingNamespace parameter 730
- UTC time 788
- \$utils variable 738

## V

- v2 debugger, graphical 648–652
  - executing other commands in debug mode 651
  - hovering over variables to see values 652
- ValidateCount attribute 307–308
- ValidateLength attribute 308
- ValidateNotNull attribute 307
- ValidateNotNullOrEmpty attribute 307
- ValidatePattern attribute 308–309

- ValidateRange attribute 309
- ValidateScript attribute 310–311
- ValidateSet attribute 310
- validating security 925
- validation 408, 895–896
- value expressions 123–124, 232
- Value member 425
- Value parameter 190
- ValueFromPipeline attributes 312
- ValueFromPipeline property 300
- ValueFromPipeline=true notation 42
- ValueFromPipelineByPropertyName
  - property 300–301, 305
- ValueFromRemainingArguments
  - property 301–302
- values
  - in Registry 671
  - of variables, variable names vs. 192–193
  - returning from functions 257–263
    - debugging problems in function
      - output 259–262
      - return statement 262–263
    - statements as 231–233
- ValueSet parameter 841, 843
- \$var variable 272
- variable assignment, tracing 640
- variable breakpoints, breaking on read or write 657
- variable checks 586
- variable initializer expression 248
- variable interpolation 437
- variable name notation 186
- variable namespace 111, 197
- variable operations 672
- Variable parameter 334
- variable provider 583
- variable reference 52
- variable scoping, in functions 269–274
  - declaring variables 270–272
  - modifiers 272–274
- variable syntax 399
- variable type attribute 185
- variable: drive 672
- variables 184–197
  - automatic 860
  - basic 23–25
  - cmdlets 188–193
    - getting and setting variable options 189–191
    - indirectly setting variable 188–189
    - using PSVariable objects as
      - references 191–192
      - variable names vs. variable values 192–193
  - declaring 184, 270–272
  - empty references in strings 587–588
  - expanding 438
  - hovering over to see values 652
  - indexing with 173
  - initializing 29
  - name syntax 186–188
  - saving expressions in 24
  - setting breakpoints on assignment 657–658
  - splatting 193–197
  - swapping 120
  - undefined 583–584
  - uninitialized use in string expansions 584–585
  - viewing values of 652
  - visibility in remoting 535
  - visibility of 269
- VariablesToExport element 370
- VariableValue property 817
- variant arrays 92
- VBScript 570
  - embedding code in script 784
  - regular expressions 133
  - WMI 807
  - WScript.Shell class 777
- verb-noun pairs 13
- Verbose flag 333–334, 860
- version file 33
- version of host, obtaining 581
- Version parameter 584
- Version property 369
- versioning
  - and assemblies 721–722
  - managing software changes 721
- View menu, ISE 610
- virtual memory 916
- virtual method 423
- VirtualMemorySize property 508
- viruses, Danom 890–891
- visibility of variable 208, 269

- Visibility property 536, 539
- Visual Basic 39, 570, 582
- Visual Studio 610, 648, 652, 749
- Visual Studio debugger 648
- Visual Studio SDK directory 906
- VM property 508
- VMS system 103
- voidable statements 156
- VolumeName property 840
- vPro technologies 798
- vulnerability 894
- vulnerability, defined 892

## W

- W3C (World Wide Web Consortium) 504
- Wait-Event cmdlet 854, 863–866, 877
- Warning type 599
- web pages, retrieving 740–742
- Web Services-Management. *See* WS-Man 830
- well formed string 137
- WhatIf parameter 108
- where alias 128
- Where-Object cmdlet 223, 225, 228–231, 393–394, 849, 923
- wheres function
  - original version 924
  - safe version 925
- wheres script
  - original 923–925
  - safer and faster version 925–927
- while loop 30, 203–204, 258
- while statement 201, 203
- whipupitude quotient 38
- whitelisting, blacklisting and 894
- whitespace 65, 201
- Whitespace character class, splitting on 682
- widening
  - defined 74
  - rules 116, 126
- Width property 755
- Wiktionary website 779
- wildcard option 217
- wildcard patterns
  - and -like operator 132–133
  - character ranges 132

- matching a single character 132
- matching string of characters 132
- using with switch statement 216–217
- wildcards 487, 612
  - in TrustedHosts list 514
  - limitations constraining scripts and applications 540
  - processing paths containing 667–668
  - suppressing processing of 668–669
- Win32 applications, In ISE 616
- Win32\_AddRemovePrograms class 806
- Win32\_Environment class 815–816
- win32\_logicaldisk 29
- Win32\_LogicalDisk class 840
- Win32\_NetworkAdapterConfiguration class 805
- Win32\_OperatingSystem resource 834–836
- Win32\_Process class 819, 824, 836, 842
- Win32\_ProcessStartTrace 868, 872
- Win32\_ProcessStopTrace 872
- Win32\_ProcessTrace 868, 872
- WIN32\_ProcessTrace events 868–870
- Win32\_Service class 872
- WindowName property 772
- Windows 7, enabling remoting 453
- Windows calculator application 778
- Windows commands, native 39
- Windows Console APIs, and remoting 495
- Windows dialog box 747
- Windows Forms application 750
- Windows Forms library 4
- Windows GUI application 778
- Windows Management Instrumentation. *See* Microsoft WMI 8
- Windows management surface 7
- Windows Presentation Foundation. *See* WPF 753
- Windows server applications 798
- Windows Vista
  - enabling remoting 453
- Windows XP
  - enabling remoting 453
  - supporting IIS-hosted remoting 530
- windows, managing through objects 7–8
- Windows, Microsoft. *See* Microsoft Windows
- Windows.Forms 4
- Windows() method 767, 772

- VisualStyle parameter 734
- WindowsUpdate.log file 221
- WinForms 4
- winforms assembly 723, 745
- WinForms library 744–750
  - simple dialog boxes 747–750
- winforms modules 750–753
- WinRM (Windows Remote Management) 452
  - changing configurations 533
  - restarting service 533
- winrm help uris command 837
- WinRM listener 452
- WITHIN keyword 871
- [WMI] type accelerator 826–827
- WMI (Windows Management Instrumentation), Microsoft. *See* Microsoft WMI
- WMI Query Language (WQL) 810
- WMI/CIM namespace 808
- WMIC (WMI command-line) 799
- [WMICLASS] type accelerator 827–828
- WmiObject 870
- [WMISEARCHER] type accelerator 825
- wof function 358
- Word, Microsoft. *See* Microsoft Word
- Word.Application object 783, 794
- words
  - analyzing use in documents 683–684
  - counting unique with hashtables 684–686
- worker function 431
- workgroup environments, additional setup steps for
  - remoting in 451
- World Wide Web Consortium (W3C) 504
- worms, MSH/Cibyz 891
- WPF (Windows Presentation Foundation) 753–759
  - advantages of 758
  - file search tool
    - defining appearance 754–756
    - specifying behavior 756–758
  - frameworks for 758–759
  - preconditions 753
- WPF XAML GUI builders 758
- WPIA namespace 732
- WPIA.ConsoleUtils class 737
- WPIA.Utils class 737
- WPIAForms module 750, 752
- WQL (WMI Query Language) 810
- wrapper functions, role in constrained sessions 538
- wrappers, problems in COM 793–796
- wrapping objects, object adaptation 423
- Write 657
- write method 581
- Write-EventLog cmdlet 597
- Write-Host cmdlet 195, 580–581, 596, 641, 654
- Write-InputObject 356
- Write-Output cmdlet 39, 51, 212
- writing
  - error objects 564
  - files
    - Get-Content cmdlet 674–676, 680–681
    - Get-HexDump function example 676–677
    - Get-MagicNumber function
      - example 677–679
  - secure code 926
  - secure scripts 916–927
    - avoiding Invoke-Expression cmdlet 923–927
    - credentials 919–923
    - SecureString 916–919
  - timer event handler 856–859
    - binding event action 857–858
    - creating timer object 856
    - enabling 858–859
    - setting parameters 857
- Writing Secure Code (Howard and LeBlanc) 892
- WScript.Shell class 777–779
- WSH ScriptControl class 783–786
  - embedding JScript code 785–786
  - embedding VBScript code 784
- WS-Man (Web Services-Management) 797, 830–846
  - cmdlets 831–832
    - and providers 509–511
    - invoking methods with Invoke-WSManAction 841–846
    - retrieving management data with Get-WSManInstance 832–839
    - updating resources using Set-WSManInstance 840–841
  - Remoting protocol 504
- WSMan configuration, TrustedHosts 511

- WSMan drive 509, 524
- WSMan provider 451
- WSMan shell 511
- WS-Man Specification 831
- WSMan-based transport 493
- wsmprovhost process 467
- wsmprovhost.exe, PowerShell remoting host process 528

## X

- x parameter 195
- \$x variable 271, 416
- x86 processor 500
- XAML (Extensible Application Markup Language)
  - defining GUI in 774–775
  - loading from here-string 776–777
- XAML loader 758
- XamlReader class 776
- XDocument objects, loading from file 710
- XLinq
  - Language Integrated Queries for XML 709
  - loading XDocument objects from file 710
  - XDocument class 710
- [xml] alias 97
- XML (Extensible Markup Language)
  - .NET framework classes 702
  - adding attributes to node 696
  - adding child nodes 696
  - bookstore inventory example 703
  - format-XmlDocument example 700
  - item property on XML object 696
  - loading XML documents 698
  - objects, adding elements to 695–697
  - rendering objects as 711–718
    - ConvertTo-Xml cmdlet 711–714
    - Import-Clixml and Export-Clixml cmdlets 714–718
  - representation of new tasks 791
  - Root property on XML object 696
  - saving document to file 695

- Select-Xml cmdlet 703
- structured text, processing 693–718
- System.XML.XmlDocument class 694
- System.Xml.XmlReader class 698
- using as objects 693–694
- XML document structure 712
- XmlNode class 695

- XML attributes 697
- XML configuration files 434
- XML data 401
- XML Document class 695
- XML DOM (Document Object Model) 698
- XML object adapter 694
- XML Path Language. *See* XPat 702
- XML processing 663
- Xml property 790
- XML provider 664
- XmlDocument class 693–694
- XmlDocument object 697
- XmlDocument, properties and navigation 694
- XmlNode object 697
- XMLReader class, loading and saving files
  - using 698–701
- XPath (XML Path Language)
  - attribute syntax 708
  - example patterns 703
  - predicate expression syntax 707
  - processing XML structured text with 702–709
    - basics of 703
    - select-Xml cmdlet 704–709
    - test document 703–704
  - used in pipeline 706
- XPath operators 708
- XSLT (Extensible Stylesheet Language Transformations) language 710

## Z

- Zbikowski, Mark 678
- zone of influence 888
- zsh shell 6, 38