

Index

Symbols

' (apostrophe), 52*
 := (colon equals), 285*
 - (hyphen), 110
 ^ (exponentiation operator), 104
 & (ampersand), 66–67, 109–110
 ! (exclamation point), 70
 # (pound sign)
 date literals, 73
 floating-point literals, 70
 Like operator, 110
 % (percent), 67
 () (parentheses)
 arrays, 86
 operator precedence, 100
 parameter declaration, 188
 * (asterisk), 110
 , (comma), 126
 . (period), 55
 / (forward slash), 73
 : (colon), 51–52, 72
 ? (question mark), 110
 @ (at character), 71
 [] (brackets), 110–111
 [(left bracket), 110
 \ (backward slash), 103
] (right bracket), 110
 _ (underscore), 51–54
 + (plus operator), 109, 126
 = (equals), 117–118
 (<>) angle brackets, 284

A

Abstract types, 265–267
 Accessibility
 declarations and, 57–58
 namespaces and, 159

overview, 21
 protected, 248–250
 shadowing and, 300
 Accessor methods, 28, 217–218
 add_Click method, 241
 AddHandler statement
 dynamic event handling, 31–32, 232–233
 implementing events using delegates, 242
 AddressOf keyword, 232, 235–236
 Alias
 Declare statements using, 206–207
 imported namespaces or types using,
 163–164
 Ampersand (&), 66–67, 109–110
 And operator, 106, 107–108
 AndAlso operator, 106
 ANSI standard, 207–208
 Apostrophe ('), 52
 Arguments
 late binding and, 196–198
 method invocation and, 193–194
 named, 196
 reference parameters and, 194–196
 Arithmetic operators, 103–104
 Array creation expression, 9–10
 Arrays, 85–92
 .NET Framework type hierarchy and, 254
 arrays of, 90–92
 creating and sizing, 87–88
 erasing, 88–89
 inheritance conversion, 252–254
 initializing, 89–90
 overview, 8–11
 parameter, 25–26, 191–193
 using, 85–87
 Assemblies
 overview, 2–3

400 ■ INDEX

Assemblies (*continued*)
 versioning and, 295

Assignment statements
 initializers as, 118
 overview, 11
 using, 122–123

Asterisk (*), 110

Asynchronous invocation, 237–239

At character (@), 71

Attributes, 283–294
 applying, 284–287
 defined, 284
 defining, 287–291
 indicating obsolete types, 305–306
 overview, 43
 storing and reading, 291–293
 using, 283–284

AttributeTargets values, 288–289

Auto character set, 208

B

Bang operator, 224–225

Banker's rounding method, 81–82

Base class
 conversions, 250–254
 inheritance, 246–247
 overriding from, 262

BeginInvoke method, 238

Binary comparisons
 with Like operator, 111
 overview, 105–106

Bitwise operators, 107–108

Boolean data type
 advanced conversions, 81–82
 overview, 63–64
 supported conversions, 80

Boxing
 defined, 254–256
 DirectCast operator and, 256–259

Brackets
 angle, 284
 square, 110–111

Branching statements
 Exit statements as, 134–135
 GoTo statements and labels as, 135–137
 overview, 13–14
 Return statements as, 135

ByRef keyword, 189

Byte data type
 conversion progression and, 80–81
 enumeration conversion and, 95–97
 enumeration of, 93–94
 overview, 65–66

ByVal keyword, 189

C

Case insensitivity, 48–49

Case statements, 125–127

Catch statements, 16, 146–148

Char data type, 72–73

Character translation, 207–208

Chr function, 113

Classes. *see also* Inheritance
 abstract, 265–267
 accessibility, 21
 attributes, 287
 constructors, 21–22, 178–181
 declarations and, 54–55
 finalization and resource disposal, 182–184
 memory management and, 169–171
 nested types, 22–23, 181–182
 overview, 20–21
 reference types, 173–176
 shared vs. instance, 176–177

CLS (Common Language Specification), 291

Collection
 For Each statement and, 129–130
 types, 132–134

Colon (:), 51–52, 72

Colon equals (:=), 285

COM
 consuming interfaces in, 278
 memory management in, 171

Comma (,), 126

Comments, 52–53

Common Language Specification (CLS), 291

Comparison operators, 4–6, 104–106

Compatibility
 decimal data types, 71
 default properties, 224
 floating-point data types, 69
 integer data types, 65

Compound assignment operators, 11–12,
 122–123

Conditional compilation statements, 164–166

Conditional methods, 198–199

Conditional statements
 If statement, 124–125
 overview, 13
 Select statement, 125–127

#Const statements, 165–166

Constant expressions, 113

Constants
 declaring local, 118–119
 overview, 24

Constructors
 abstract classes and, 266
 overview, 21–22
 shared, 181

- structure, 180
- using, 178–179
- Conversion, 76–82
 - advanced, 81–82
 - For Each statement and, 130
 - enumeration, 95–97
 - inheritance, 250–254
 - interface, 277–278
 - operators, 7–8, 77–79
 - overview, 76
 - supported, 80–81
 - widening and narrowing, 79–80
- copy-in/copy-out, 195, 209
- Covariance, 252–254
- CType operator, 78, 258–259
- Currency data type, 71
- Current property, 133–134
- CurrentCulture property, 106

D

- Data types. *see* Fundamental types
- Date data type
 - advanced conversions, 81–82
 - date literals and, 73–74
 - overview, 73
- Date literals, 73–74
- Decimal data type
 - conversion progression and, 80–81
 - Decimal literals, 71–72
 - overview, 70–71
- Decimal integer literals, 66–67
- Decimal literals, 71–72
- Declarations
 - determining accessibility of, 57–58
 - forward references and, 56–57
 - overview, 53–56
- Declarative event handling
 - implementing events using delegates, 242–243
 - overview, 31
 - working with, 229–231
- Declare statements
 - character translation and, 207–208
 - overview, 26–27
 - String parameters and, 208–211
 - working with, 205–207
- Default properties
 - compatibility and, 224
 - dictionary lookup operator and, 224–225
 - overview, 29–30
 - shadowing and, 301–302
 - using, 223
- Delegates, 233–243
 - defined, 233
 - implementing events with, 239–243

- invoking asynchronously, 237–239
- overview, 34–35
- using, 233–237
- Design, 51–52
- Dictionary lookup operator, 224–225
- DirectCast operator
 - overview, 78
 - using, 256–259
- Dispose, 184
- Division operators
 - floating-point vs. integer, 103
 - resolution and, 102
- Do statement, 12, 131–132
- Double data type
 - conversion progression and, 80–81
 - floating-point data types, 68–69
- Dynamic event handling, 31–32, 231–233

E

- Element type
 - arrays of arrays, 90–91
 - overview, 85–86
- Elements, array, 85–86
- Else block, 124–125
- #End Region statements, 166
- End statements, 137
- EndInvoke method, 238
- Entry points, 61
- Enumerations
 - collection types, 132–134
 - conversions, 95–97
 - .NET Framework type hierarchy and, 254
 - overview, 6
 - underlying types, 93–94
 - using, 92–93
- equals sign (=), 117–118
- Erase statement, 88–89
- Err property, 151
- Error statement, 15–16, 146
- Essential.NET Volume 1: The Common Language Runtime (Box)*, 295
- Event-driven programming, 227
- Events, 227–244. *see also* Delegates
 - declarative handling, 31, 229–231
 - defining and raising, 227–229
 - dynamic handling, 31–32, 231–233
 - implementing using delegates, 239–243
 - interface, 270, 275–276
 - overview, 30–31
- Exception handling, 143–154
 - defined, 143
 - floating-point division operator, 103
 - GoTo statements and, 136
 - overview, 15–16

402 ■ INDEX

Exception handling (*continued*)
 structured, 16–17, 146–149
 throwing exceptions, 144–146
 unstructured, 17, 149–153
 using, 143–144

Exclamation point (!), 70

Exit statement, 13–14, 134–135

Exponent, 68

Exponentiation operator (^), 104

Expressions, constant, 113

F

Fields

defined, 213

overview, 23–24

read-only, 214–216

using, 213–214

Finalization, 182–184

Finalize method, 183

Finally statement

disposable type used with, 184

structured exception handling with, 16,
 147–148

Flag enumerations, 94

Floating-point data types

converting, 81–82

division operator, 103

floating-point literals, 69–70

overview, 68–69

For Each statement

collection types enumerated by, 132–134

as loop statement, 12

overview, 129–131

For statement, 12, 127–129

Forward references, 56–67

Forward slash (/), 73

Free-format languages, 50

Friend access level, 21, 58

Fully qualified names, 160–161

Function return variables, 188

Functions, 187–188

Fundamental types, 63–84

Boolean data type, 63–64

Char data type, 72–73

comparison operators, 5

conversion operators, 7–8

conversions. *see* Conversion

Date data type, 73

Decimal data type, 70–72

of enumerations, 93–94

floating-point data types, 68–70

integer data types, 65–67

logical operators, 5

numeric operators, 4–7

Object data type, 74–76

overview, 3–4

String data type, 73

string operators, 5–6

table of, 64

used in attributes, 290–291

G

Garbage collection

defined, 171

finalization and, 182–184

GDI, 189

Get accessor, 217–218, 262

GetCustomAttributes method, 292–293

GetEnumerator method, 132–134

GetType operator, 112–113

GoSub statement, 137

GoTo statement

as branching statement, 14, 135–137

unstructured exception handling and, 17

H

Heap, 17–18, 170–171

Hello, World!, 1–3

Hexadecimal integer literals, 66–67

Hierarchy

.NET Framework, 254–259

inheritance, 247–248

Hyphen (-), 110

I

IDisposable interface, 183–184

IEnumerable interface, 132–133

If statement, 13

IL (intermediate language), 3

Immutable types, 216

Implements statement

event implementations, 275–276

overview, 271–273

private interface implementations, 273–275

Implicit locals, 120–122

Imports statement

applying attributes with, 286–287

fully qualified names used with, 162

importing namespaces, 33–34, 161–164

Imports System statement, 162

Index, of array, 8, 85

Indexed properties

declaring as default properties, 223–224

overview, 28–29

working with, 220–222

Inheritance, 245–268

abstract classes and methods in, 265–267

conversions, 250–254

defined, 246

interfaces supporting, 278–280

- interfaces vs., 269–270
- .NET Framework type hierarchy and, 254–259
- overriding, 39–41, 259–265
- overview, 36–37
- protected accessibility and, 37–38, 248–250
- single, 247
- using, 245–248
- Initializers
 - array, 9–10, 89–90
 - local variable, 117–118
 - static local, 120–122
- Instance members
 - of classes and structures, 20–21
 - shared members vs., 176–177
- Integer data types
 - advanced conversions of, 81–82
 - conversion progression and, 80–81
 - enumeration based on, 93–94
 - enumeration conversion and, 95–97
 - floating-point values vs., 68
 - integer literals, 66–67
 - overview, 65–66
- Integer division operator (\), 103
- Integer remainder operator, 103
- Interface keyword, 270
- Interfaces
 - consuming, 277–278
 - defining, 270–271
 - inheritance and, 269–270, 278–280
 - overview, 41–43
- Interfaces, implementing, 271–277
 - events, 275–277
 - overview, 271–277
 - private, 273–276
- Intermediate language (IL), 3
- Invocation, asynchronous, 237–239
- Is keyword, 126
- Is operator, 174
- IsDefined method, 97
- J**
- Jagged arrays
 - defined, 11, 90
 - performance and, 92
- JIT (Just-In-Time compiling), 59
- L**
- Labels, 135–137
- Late binding
 - of indexed properties, 222
 - of methods, 196–198
- Left-associative, operators as, 99
- Let assignments, 122
- Library clause, 205
- Like operator, 110–111
- Line continuations, 51, 54
- Line orientation, 49–51
- Literals
 - Char and String, 72–73
 - date, 73
 - Decimal, 71–72
 - defined, 63
 - floating-point, 69–70
 - integer, 66–67
 - Nothing, 74
- Local declaration statements, 115–122
 - constants, 118–119
 - implicit locals, 120–122
 - initializers, 117–118
 - overview, 115–117
 - static locals, 119–120
 - type characters, 117
- Logical operators, 5, 106–107
- Long data type
 - conversion progression and, 80–81
 - enumeration based on, 93–94
 - enumeration conversion and, 95–97
 - overview, 65–66
- Loop control variable, 127
- Loop statement, 12
- Looping statements, 127–134
 - collection types, 132–134
 - For Each statement, 129–131
 - overview, 12
 - For statement, 127–129
 - While and Do statements, 131–132
- Loose typing, 75–76
- M**
- Main method, 2, 60
- Mantissa, 68
- Memory management, 169–171
 - COM's method of, 171
 - heap and stack, 170–171
 - overviews, 17–19
 - working with, 169
- Metadata, 3
- Method invocation, 193–199
 - arguments and reference parameters, 194–196
 - conditional methods, 198–199
 - late binding, 196–198
 - named arguments, 196
 - overview, 193–194
- Methods, 187–212
 - abstract, 265–267
 - constructor, 178–181
 - Declare statements and, 26–27, 205–211
 - defined, 24, 187

404 ■ INDEX

- Methods (*continued*)
 - as functions, 187–188
 - interfaces containing, 270
 - overloading, 199–205
 - overriding, 262–267
 - overview, 24–27
 - parameters and, 24–26, 188–193
 - as subroutines, 187–188
 - using Overloads keyword, 304–305
 - Modifiers, 283–284
 - Modules
 - accessibility, 21
 - defined, 155
 - fully qualified names and, 160
 - overview, 19–20
 - working with, 155–157
 - MoveNext method, 132–134
 - Multicast delegates, 35, 236
 - Multicast events, 228–229
 - Multidimensional arrays
 - initializing, 90, 92
 - sizing, 87
 - MustInherit class, 40
 - MustOverride methods, 40
 - MyBase keyword, 40, 263
 - MyClass keyword, 263–265
- N**
- Named arguments, 196
 - Namespaces
 - declarations and, 54–55
 - defined, 155
 - fully qualified names, 160–161
 - importing, 161–164
 - overview, 32–34
 - working with, 157–159
 - Naming conventions
 - assigning alias to Declare statement, 206
 - assigning alias to imports, 163–164
 - attributes, 284–285, 288
 - declarations, 53–57
 - interface, 272–273, 281
 - properties, 225
 - versioning and. *see* Shadowing
 - NaN (Not a Number), 68, 103
 - Narrowing conversion, 79–80
 - Nested arrays of arrays, 91
 - Nested namespaces, 158
 - Nested types, 22–23, 181–182
 - .NET Framework
 - defining attributes, 286–287
 - modules, 155
 - overview, 59–60
 - program startup and termination, 60–61
 - type hierarchy, 254–259
 - .NET language, overview
 - arrays, 8–11
 - attributes, 43
 - case insensitivity, 48–49
 - classes, 20–24
 - comments, 52–53
 - delegates, 34–35
 - events, 30–32
 - exception handling, 15–17
 - fields, 23–24
 - fundamental types, 3–8
 - Hello, World!, 1–3
 - inheritance, 36–41
 - interfaces, 41–43
 - line orientation, 49–52
 - memory management, 17–19
 - methods, 24–27
 - modules, 19–20
 - namespaces, 32–34
 - properties, 28–30
 - readability, 47–48
 - statements, 11–15
 - structures, 20–24
 - versioning, 44–45
 - New operator, 173–174
 - Next statement, 128
 - Not a Number (NaN), 68, 103
 - Not operator, 106, 107–108
 - Nothing keyword
 - clearing arrays, 89
 - Object data type and, 74
 - setting reference variables to, 173–175
 - NotOverridable modifier, 41
 - Null reference, 173–174
 - Numeric data types
 - advanced conversions, 81–82
 - converting, 80–81
 - defined, 63
 - Numeric operators, 3–7
- O**
- Object data type
 - inheritance and, 247–248
 - .NET Framework type hierarchy and, 254–259
 - overloading vs., 202
 - overview, 74–76
 - resolution and, 102
 - Obsolete code, 305–306
 - Octal integer literals, 66–67
 - On Error statements
 - unstructured exception handling with, 17, 150–151
 - using Resume Next statement, 153
 - Operators, 99–113

- arithmetic, 103–104
 - bitwise, 107–108
 - comparison, 104–106
 - constant expressions, 113–114
 - logical, 106–107
 - precedence of, 99–101
 - resolution of, 101–102
 - shift, 108–109
 - string, 109–111
 - type, 111–113
 - Option Explicit statement, 121
 - Option statement, 286–287
 - Option Strict statement
 - disallowing late binding, 222
 - good programming and, 76
 - variable declarations and, 115, 187
 - Optional parameters
 - method invocation and, 193–194
 - .NET Framework languages supporting, 193
 - overloading vs., 202
 - overview, 190–191
 - parameter arrays and, 191–193
 - Or operator
 - as bitwise operator, 107–108
 - defining attributes, 288
 - as logical operator, 106
 - OrElse operator, 106
 - Overloading
 - overview, 26
 - properties, 221–222
 - resolution of, 202–205
 - versioning and, 302–305
 - working with, 199–202
 - Overloads keyword
 - shadowing and, 303–305
 - versioning and, 44–45
 - Overriding, 259–267
 - abstract methods, 266–267
 - methods, 262–267, 287
 - MyBase and, 263–265
 - MyClass and, 263–265
 - overviews of, 39–41, 259–261
 - properties, 261–262
 - shadowing vs., 298–300
- P**
- Parameter arrays
 - method invocation and, 193–194
 - overview, 25–26
 - working with, 191–193
 - Parameters
 - attribute, 288–289
 - defined, 188
 - optional, 190–191
 - overview, 24–26
 - property, 28
 - reference, 189–190
 - value, 188–189
 - Parentheses ()
 - arrays, 86
 - operator precedence, 100
 - parameter declaration, 188
 - Percent sign (%), 67
 - Period (.), 55
 - Person class
 - conversions, 250–252
 - designating as abstract type, 265–267
 - inheritance and, 246
 - Plus operator (+), 109, 126
 - Positional arguments, 196
 - Pound sign (#)
 - date literals, 73
 - floating-point literals, 70
 - Like operator, 110
 - Precedence, operator, 99–101
 - Preprocessing statements, 164–166
 - Preserve keyword, 87
 - Private access level
 - defined, 21
 - inheritance and, 248
 - namespaces and, 159
 - overview, 57–58
 - Private interface implementations, 273–275
 - Program flow statements, 137–138
 - Properties, 216–225
 - attribute, 285, 290
 - default, 223–224, 301–302
 - defined, 216
 - defining accessors, 217–218
 - dictionary lookup operator and, 224–225
 - indexed, 220–222
 - interface, 270
 - overriding, 261–262
 - overview, 28–30
 - ReadOnly/WriteOnly, 218–219
 - using Overloads keyword, 304–305
 - Protected access level, 21, 37–38
 - Protected Friend access level
 - defined, 21
 - overriding, 262
 - overview, 250
 - Protected members access level, 248–250
 - Public access level
 - defined, 21
 - interfaces and, 270
 - overview, 57–58
- Q**
- QueryInterface, 278
 - Question mark (?), 110

406 ■ INDEX

R

RaiseEvent statement, 30–31, 228
 Rank, array, 86
 ReadLine method, 2
 ReadOnly fields, 23
 ReadOnly modifier, 218–219
 ReDim statement, 9, 87–88
 Reference counting, 171
 Reference parameters
 arguments and, 194–196
 indexed properties as, 221
 overview, 189–190
 Reference types
 classes as, 20, 173–176
 memory management and, 17–19
 .NET Framework type hierarchy and,
 254–256
 #Region statements, 166
 REM keyword, 53
 remove_Click method, 241
 RemoveHandler statement
 dynamic event handling, 31–32, 232–233
 implementing events using delegates, 242
 Resolution
 operator, 101–102
 overload, 202–205
 Resource disposal, 182–184
 Resume Next statement, 152–153
 Resume statement, 152–153
 Return statement
 as branching statement, 14
 overview, 135
 returning values from functions, 188

S

Select statement, 13, 125–127
 Set accessor, 217–218, 262
 Set assignments, 122
 SetLastError method, 206
 Shadowing, 295–302
 accessibility and, 300
 default properties and, 301–302
 defined, 297
 overriding vs., 298–300
 overview, 295–298
 Shadows keyword, 44–45, 297–298
 Shared constructors, 181
 Shared events, 231–233
 Shared members, 20–21, 176–177
 Shift operators, 108–109
 Short-circuiting, 106–107
 Short data type
 conversion progression and, 80–81
 enumeration based on, 93–94
 enumeration conversion and, 95–97

overview, 65–66
 Single data type
 conversion progression and, 80–81
 floating-point data types, 68–69
 Single inheritance, 247
 Stack, 170–171
 Startup, program, 60–61
 Statement separators, 51–52
 Statements, 115–142
 assignment, 11, 118, 122–123
 branching, 13–14, 134–137
 Case, 125–127
 Catch, 16, 146–148
 conditional, 13, 124–127
 Conditional compilation, 164–166
 #Const, 165–166
 Declare. *see* Declare statements
 Do, 12, 131–132
 For Each, 12, 129–131, 132–134
 End, 137–138
 #End Region, 166
 On Error, 17, 150–151, 153
 Exit, 13–14, 134–135
 For, 12, 127–129
 GoTo, 14, 17, 135–137
 If, 124–125
 Local declaration. *see* Local declaration
 statements
 Looping. *see* Looping statements
 Option, 286–287
 Option Explicit, 121
 Option Strict, 76, 115, 187, 222
 overview, 11–15
 preprocessing, 164–166
 program flow, 137–138
 #Region, 166
 Return, 14, 135, 188
 Select, 13, 125–127
 Stop, 137–138
 structured exception handling, 146–148
 SyncLock, 14–15, 138–140
 While, 12, 131–132
 With, 12–13, 123–124
 Static locals, 119–120
 Step keyword, 129
 Stop statements, 137–138
 String concatenation operator (&), 109–110
 String data type
 advanced conversions, 81–82
 comparisons, 105–106
 overview, 72
 resolution and, 102
 string literals, 72–73
 String operators, 4–6, 109–111
 String parameters, 208–211

StringBuilder class, 210
Strong typing, 76
Structured exception handling, 146–149
Structures
 accessibility, 21
 constructors, 21–22, 180–181
 finalization and resource disposal, 182–184
 memory management and, 169–171
 nested types, 22–23, 181–182
 .NET Framework type hierarchy and, 254
 overview, 20–21
 shared vs. instance, 176–177
 value types and, 171–173
Subroutines, 187–188
Synchronous invocation, 237
SyncLock statement, 14–15, 138–140
Syntax, 49
System.ApplicationException, 145
System.Attribute class, 287–288
System.AttributeUsageAttribute attribute, 288
System.Delegate.Combine method, 236–237
System.Delegate.Remove method, 237
System.Diagnostics.ConditionalAttribute attribute, 199
System.Enum.IsDefined method, 97
System.EventHandler delegate type, 241
System.Exception, 144–145
System.IDisposable interface, 131, 183–184
System.ObsoleteAttribute, 305–306

T
Termination, program, 60–61
Text comparisons
 Like operator, 111
 overview of, 105–106
Thread.CurrentThread property, 106
Throw statement
 exception handling, 15
 rethrowing exceptions, 148–149
 throwing exceptions, 144–146
To keyword, 126
true byref, 195
Try statement, 16–17, 146–147
Type characters, 117
Type operators, 111–113
Type-safe programming, 63
TypeOf operator, 111–112
Types. *see also* Fundamental types
 defined, 63

U
Unboxing
 defined, 255
 DirectCast operator and, 256–259
underscore (_), 51–54
Unicode standard, 207–208
Universal type, 254
Unstructured exception handling
 overview, 149–152
 Resume and Resume Next statement,
 152–153
Until keyword, 131–132

V
ValidOn parameter, 288
Value parameters, 188–189
Value types
 boxing and unboxing, 254–256
 memory management and, 18
 overview, 171–172
 structures as, 20, 172–173
Variables, array, 8–9, 85
Variant parameters, 191
Variant type, 254
Versioning, 295–306
 obsolete code and, 305–306
 overloading and, 302–305
 overview, 44–45
 shadowing. *see* Shadowing

W
When clause, 148
While keyword, 131–132
While statement, 12, 131–132
Widening conversion, 79
With statement
 overview, 12–13
 using dictionary lookup operation in,
 225
 working with, 123–124
WithEvents modifier, 31, 229–231
WriteLine method, 2
WriteOnly modifier, 218–219

X
XEventHandler delegate type, 240
Xor operator, 106–108

