

Index

- , (comma), format specifier, 329
- . (dot)
 - format specifier, 329
 - member access (dot) operator, 229–230
- ?:(question mark colon), conditional operator, 231
 - RTF
 - courier font, 0
- () (parentheses), casting operator, 230
- & (ampersand)
 - NOT operator, 218
 - pointer type operator, 236
- &= (ampersand equal), AND assignment operator, 227
- && (ampersands), logical AND, 215–217
- * (asterisk)
 - multiplication operator, 213–215
 - pointer type operator, 236
- *= (asterisk equal), multiplication assignment operator, 227
- [^] (carat), exclusive-OR operator, 218
- [^]= (carat equal), logical exclusive-OR assignment operator, 228
- = (equal sign), assignment operator, 226
- == (equal signs), equality operator
 - checking object equality, 89
 - comparing structs, 191
 - description, 223–224
- != (exclamation equal), inequality operator, 224
- ! (exclamation point), logical negation, 215–217
- >= (greater than equal), greater than or equal operator, 225–226
- >>= (greater than equal), shift right assignment operator, 228
- > (greater than sign), greater than operator, 225
- >> (greater than signs), right-shift operator, 221–222
- > (hyphen greater than), pointer type operator, 236
- < (left angle bracket), less than operator, 224
- <= (left angle bracket equal), less than or equal operator, 225
- << (left angle brackets), left-shift operator, 221–222
- <<= (left angle brackets equal), shift left assignment operator, 228
- <= (less than equal), less than or equal operator, 225
- <<= (less than equal), shift left assignment operator, 228
- < (less than sign), less than operator, 224
- << (less than signs), left-shift operator, 221–222
- = (minus equal), decrement operator, 226–227
- > (minus greater than), pointer type operator, 236
- (minus sign), subtraction operator, 213–215
- (minus signs), decrement operator, 220–221
- %= (percent equal), modulus assignment operator, 227
- % (percent sign)
 - format specifier, 329
 - modulus operator, 213–215
- + = (plus equal), increment operator, 226
- + (plus sign)
 - addition operator, 213–215
 - string concatenation operator, 219–220, 307–312
- ++ (plus signs), increment operator, 220–221

- # (pound sign), format specifier, 329
- > (right angle bracket), greater than operator, 225
- >= (right angle bracket equal), greater than or equal operator, 225
- >> (right angle brackets), right-shift operator, 221–222
- >>= (right angle brackets equal), shift right assignment operator, 228
- / (slash), division operator, 213–215
- /= (slash equal), division assignment operator, 227
- |= (vertical bar equal), OR assignment operator, 228
- | (vertical line), OR operator, 218
- || (vertical lines), logical OR, 215–217
- [] (square brackets)
 - indexing operator, 230
 - pointer type operator, 236
- ~ (tilde), complement operator, 218–219
- 0 (zero), format specifier, 329

- A
- Abort method, 417–418
- abstract keyword
 - class access modifier, 94–95
 - method access modifier, 101–102
 - properties, 124
- Abstract methods
 - C# vs. Java, 59
 - inheritance, 157–158
- Access level default, 58
- Access modifier rules, 58
- ActiveX Data Objects (ADO), 467–468
- ADO (ActiveX Data Objects), 467–468
- ADO.NET. *See also* databases; `DataSet` class.
 - `DataSet` objects, 483
 - `DELETE` queries, 493–495
 - `INSERT` queries, 493–495
 - .NET data provider, 482
 - OLE DB data provider, 483
 - overview, 481–483
- parameters, binding at runtime, 491–492
- Recordsets, getting, 495–496
- SELECT queries, 487–492
- SQL Server, 483
- stored procedures, 495–496
- UPDATE queries, 493–495
- Ampersand (&)
 - NOT operator, 218
 - pointer type operator, 236
- Ampersand equal (&=), AND assignment operator, 227
- Ampersands (&&), logical AND, 215–217
- Anchoring controls, 581–584
- Anonymous inner classes, 82
- APIs, C# vs. Java
 - `java.awt` package, 666–668
 - `java.io` package, 661
 - `java.lang` package, 657–659
 - `java.sql` package, 662
 - `java.swing` package, 664–666
 - `java.text` package, 663
 - `java.util` package, 659–660
 - `org.w3c.dom` package, 664
- AppendText method, 380–381
- Application domains, 6–7
- Applications
 - isolation. *See* application domains.
 - multiple, in single process. *See* application domains.
- Arithmetic operators, 213–215
- Array class, 279–282
- ArrayList interface, 344–347
- Arrays
 - of arrays, 285–287
 - C# vs. Java, 277–279
 - collections as, 290–291, 343–344, 344–347
 - copying, 287–290
 - jagged, 285–287
 - multidimensional, 283–285
 - one-dimensional, 279–282
 - rank, 283
- `System.Array` class, 279–282

- as operator, 183–185, 233–234
ASP.NET, 7, 56
Assemblies
 application configuration, 636–641
 C# *vs.* Java, 77–78
 config file, sample, 639–641
 creating, 5–6, 625–629
 dictionary section, 638–639
 dynamic, 6
 GAC (Global Assembly Cache), 632
 manifests, 624–625
 modules, 625
 name/value sections, 638
 overview, 5–6, 624
 programmatic access, 629–634
 resources, 625
 sharing, 632
 single-tag sections, 637–638
 static, 6
 strong names, 632
 versioning, 634–636
Assignment operators, 226–229
Asterisk (*)
 multiplication operator, 213–215
 pointer type operator, 236
Asterisk equal (*=), multiplication assignment operator, 227
Asynchronous I/O, 385–387
Atomic operations, 426
Attributes
 C# *vs.* Java, 81–82
 global, 453
 multiuse, 453
 overview, 452–456
 targets, 452
Authenticated identity, 19
Authentication, 19
Authorization, 19
- B
Background threads, 401
base keyword, 153–156
- Bean managed persistence (BMP), 53
Binary formatter, 387
Binding, 132–133
BitArray interface, 347–349
Bitwise operators, 217–219. *See also* Boolean values; logical operators.
BMP (bean managed persistence), 53
bool keyword, 203
Boolean data types, 203
Boolean values, 347–349. *See also* bitwise operators; logical operators.
Boxing, 61–62, 205–206
Bubbling up exceptions, 266
BufferedStream class, 375
Business logic layer, 52–53
byte keyword, 195–196
- C
C#, overview
 accessing classes in different namespaces, 35–36
 class names *vs.* file names, 31
 classpaths, 35, 36
 compiling programs, 29–30
 constructor, default, 47
 EXE file, entry point, 32
 EXE file, *vs.* Java class file, 36
 ILDASM (Intermediate Language Disassembler), 42–47
 importing packages, 32–33
 Main() method, 31
 multiple classes per source file, 33–35
 namespaces, 32
 .NET SDK, 29
 performance, 37–38
 reference data type, 204–205
 running programs, 32
 tools, 38–47
 using keyword, 32–33
 value types, 189–193
Visual Studio .NET, 38–42
writing your first program, 30–36

- C# *vs.* Java
abstract methods, 59
access level, default, 58
access modifier rules, 58
access modifiers, equivalent, 58
anonymous inner classes, 82
arrays, 277–279
ASP.NET, 56
assemblies, 77–78
attributes, 81–82
BMP (bean managed persistence), 53
boxing, 61–62
business logic layer, 52–53
calling remote objects, 53
checked exceptions, 82
class members, 70
classes, 61–62, 67–69
CLR (Common Language Runtime), 55–56
CMP (container managed persistence), 54
collections library, 75
common language elements, 67–75
common objects, 65–67
constants, 79
constructors, 59–60, 69
containers, 52
control flow statements, 63–64
data layer, 53–54
data types, 61–62
delegates, 81
destructors, 59–60, 69
dynamic class loading, 82
enum construct, 61–62
enums, 80
exception support, 64–65
explicit base interfaces, 74–75
IIS (Internet Information Server), 52
indexers, 80
indexing parameters, 80
inheritance object hierarchy, 58
input/output, 76–77
instance constructors, 69
interfaces, 59, 74–75, 170–174
interfaces containing fields, 82
Internet Explorer, 56
Java byte code, 54
Java overview, 50–51
`java.lang.Object` class, 65–67
`java.lang.String` class, 65–67
JDBC driver, 53
JVM (Java Virtual Machine), 54–55
listeners, 81
managed code, 54
method modifiers, 73
methods, 70, 73–74
missing elements, C#, 82
missing elements, Java, 78–82
namespaces, 77–78
.NET suite, 51–52
OOP model, 57–60
operators, 62–63
packages, 77–78
polymorphism, 58
presentation layer, 52
properties, 79–80
RMI (Remote Method Invocation), 54
runtime architecture, 54–57
safe code, 54
serialization, 54
shell executables, 56–57
singleton classes, 69
static constructors, 69
static methods, 58
structs, 61–62, 78–79
threading, 75–76
transactions, 52
typed collections, 82
unboxing, 61–62
underlying types, 80
variables, 70–72
virtual methods, 57
XML Web services, 53
- C# *vs.* Java, APIs
 `java.awt` package, 666–668
 `java.io` package, 661

- `java.lang` package, 657–659
- `java.sql` package, 662
- `java.swing` package, 664–666
- `java.text` package, 663
- `java.util` package, 659–660
- `org.w3c.dom` package, 664
- CachedRowSet class. *See* `DataSet` class.
- Calling
 - remote objects, 53
 - superclasses, 153–156
- Carat (^), exclusive-OR operator, 218
- Carat equal (^=), logical exclusive-OR assignment operator, 228
- Casting, 158–163. *See also* virtual dispatching.
- Casting operators, 230
- `char` keyword, 199–200
- Checked exceptions, 25, 82, 261
- `checked` operator, 235–236
- Class access modifiers
 - `abstract`, 94–95
 - inheritance, 95–96
 - `internal`, 90
 - nested types, 90–93
 - `new`, 95
 - `private`, 93–94
 - `protected`, 93–94
 - `public`, 90
 - `sealed`, 95
- `class` keyword, 204–205
- Class library, 25–26
- Class members, 70
- Class names *vs.* file names, 31
- Classes. *See also entries for specific classes.*
 - access modifiers, 67–68
 - accessing in different namespaces, 35–36
 - anonymous inner, C# *vs.* Java, 82
 - C# *vs.* Java, 61–62, 67–69
 - dynamic loading, 82
 - instance members, 90
 - `java.lang.Object`, 65–67
 - `java.lang.String`, 65–67
 - members, 90
 - metadata, 90
 - multiple per source file, 33–35
 - nesting, 67
 - overriding superclasses, 57, 58, 146–153
 - singleton, 69
 - static members, 90
- Classpaths, 35, 36
- Clients, 3
- CLR (Common Language Runtime)
 - C# *vs.* Java, 55–56
 - exception management, 22–25
 - exceptions, 272–273
 - garbage collection, 14–17
 - overview, 4–5
 - vs.* JVM, 10–14
- CLS (Common Language Specification), 9–10
- CLS-compliant components, 9–10
- CMP (container managed persistence), 54
- COFF (Common Object File Format), 11–12
- `CollectionBase` class, 361–362
- Collections.
 - as arrays, 290–291, 343–344, 344–347
 - of Boolean values, 347–349
 - `CollectionBase` class, 361–362
 - comparing objects, 355–358
 - custom, 362–367
 - `DictionaryBase` class, 361–362
 - enumerating through elements, 359–361
 - FIFO (first-in, first out) mechanism, 352
 - generating hash codes, 358–359
 - `IEnumerable` interface, 339–341
 - `IDictionary` interface, 340–341
 - `IDictionaryEnumerator` interface, 340–341
 - `IEnumerable` interface, 339–341
 - `IHashCodeProvider` interface, 340–341
 - `IList` interface, 339–341
 - `IMap` interface, 339–341
 - key-value pairs, 349–351, 354–355
 - LIFO (last-in, first out) mechanism, 353–354

- Collections. (*cont.*)
 System.Collections interfaces, 339–341
 type-safe, 361–362
- Collections classes
 accessibility, 343–344
 ArrayList, 344–347
 BitArray, 347–349
 Hashtable, 349–351
 IComparer, 355–358
 IEnumerator, 359–361
 IHashCodeProvider, 358–358
 immutability, 343
 list of, 342
 Queue, 352
 SortedList, 354–355
 Stack, 353–354
 thread safety, 341
 type safety, 341
- Collections interfaces, 339–341
- Collections library, 75
- Comma (,), format specifier, 329
- Commas, formatting numbers, 326–328
- Common Language Runtime (CLR). *See* CLR (Common Language Runtime).
- Common Language Specification (CLS), 9–10
- Common Object File Format (COFF), 11–12
- Common Type System (CTS), 8–9
- Compact garbage collectors, 17
- Comparing strings, 305–307
- Compilers, 12–13
- Compiling C# programs, 29–30
- Component classes, 561–564
- Component controls, 561–564
- Composition *vs.* inheritance, 163–164
- Concatenating strings, 307–312
- Conditional operators, 231
- config file, sample, 639–641
- Connecting to databases, 485–487
- const keyword, 121–122
- Constants, 79
- Constructors
- C# *vs.* Java, 59–60, 69
- default, 47
- definition, 8–9
- instance, 69
- overview, 96–98
- static, 69
- Container managed persistence (CMP), 54
- Container nodes, 543
- Containers, 52
- Contract of cooperation, 5
- Control flow
 C# *vs.* Java, 63–64
 do while loop, 248–249
 fall through, 250–253
 foreach statement, 253–255
 if statement, 243–247
 jump statements, 256
 for loop, 249–250
 short-circuiting, 244
 switch statement, 250–253
 while loop, 247–248
- Controls (GUI), 561–564
- Copy method, 381–382
- Copying arrays, 287–290
- Copying garbage collectors, 17
- Creating
 assemblies, 5–6, 625–629
 GUI projects, 552–554
 object metadata, 11
 processes, 651–652
 threads. *See* threads, creating.
 threads with delegates, 404–417
- Cryptography, 20
- csc command, 30
- CTS (Common Type System), 8–9
- CultureInfo class, 322
- Currency, formatting, 322–324
- Custom collections, 362–367
- Custom exception objects, 267–270
- Custom format specifiers, 328–331
- D
- Data layer, 53–54
- Data sources, 481

-
- Data types
 - boxing, 61–62, 205–206
 - C# reference, 204–205
 - C# value types, 189–193
 - C# vs. Java, 61–62
 - `class` keyword, 204–205
 - compatibility checking, 232
 - converting. *See also* operator; casting
 - `delegate` keyword, 204–205
 - `enum`, 192–193
 - enum construct, 61–62
 - enumerator lists, 192–193
 - events, 8
 - fields, 8
 - `interface` keyword, 204–205
 - Java reference, 188
 - Java value types (primitives), converting to namespace equivalent, 61–62
 - Java value types (primitives), overview, 187–188
 - methods, 8–9
 - nested types, 8
 - in .NET, 8–9
 - pointers, 209–211
 - properties, 9
 - referents, 209–211
 - size, getting, 234
 - `struct`, 189–192
 - structs, 61–62
 - `System.Type` object, getting, 208
 - `type`, getting, 234
 - `typeof` operator, 208
 - unboxing, 61–62, 206–207
 - uniformity. *See also* CTS (Common Type System).
 - unmanaged, 209–211
 - unsafe code, 209–211
 - Data types, built-in
 - `bool` keyword, 203
 - Boolean types, 203
 - `byte` keyword, 195–196
 - `char` keyword, 199–200
 - `decimal` keyword, 202–203
 - decimal value types, 202–203
 - `double` keyword, 202
 - explicit numeric conversions, 203–204
 - `float` keyword, 200–202
 - floating-point types, 200–202
 - implicit conversions, 203–204
 - `int` keyword, 197–198
 - integral types, 194–200
 - `long` keyword, 198–199
 - overview, 193
 - `sbyte` keyword, 195–196
 - `short` keyword, 196–197
 - `uint` keyword, 197–198
 - `ulong` keyword, 198–199
 - `ushort` keyword, 196–197
- Database drivers, 484
- Databases. *See also* ADO.NET; `DataSet` class.
 - C# namespaces, 482
 - connecting to, 485–487
 - data sources, 481
 - deleting data, 493–495, 502–513
 - getting data, 487–492
 - inserting data, 493–495, 502–513
 - JDBC (Java Database Connectivity), 483–484
 - metadata, 498–499
 - rolling back transactions, 498
 - savepoints, 498
 - saving data from `DataSet` class, 502–513
 - transaction isolation levels, 499
 - transaction processing, 496–498
 - updating data, 493–495, 502–513
- `DataSet` class. *See also* ADO.NET; databases.
 - definition, 499
 - loading data into, 500–502
 - saving data from, 502–513
- `DataSet` objects, 483
- Date and time
 - format specifiers, 334
 - formatting, 332–335
 - parsing, 337
- `DateFormat` class, 332
- `DateTimeFormatInfo` class, 322, 332–335
- Decimal data types, 202–203
- `decimal` keyword, 202–203

- Decimal point. *See* dot.
- Decimal points, formatting, 326–328
- Declarative security, 17–18
- Decrement operators, 220–221
- Deep copies, 395
- `delegate` keyword, 204–205
- Delegates
- C# *vs.* Java, 81
 - creating threads, 404–417
 - exception handling, 463
 - multicasting, 465–467
 - as static members, 463–464
 - `ThreadStart`, 404–417
 - using, 461–464
- `Delete` method, 381–382
- `DELETE` queries, 493–495
- Deleting database data, 493–495, 502–513
- Destroying objects, 127–129
- Destructors, 59–60, 69
- Detecting process completion, 653–654
- Dictionary section, assemblies, 638–639
- `DictionaryBase` class, 361–362
- Directories, 380–387
- `Directory` class, 380
- `Dispose()` method, 128
- `do while` loop, 248–249
- Docking controls, 584
- Document Object Model (DOM), 515
- DOM (Document Object Model), 515
- DOM API. *See also* XML (Extensible Markup Language).
- container nodes, 543
 - definition, 533
 - querying for specific nodes, 537
 - searching XML documents, 543–547
 - traversing the DOM tree, 534–537
 - writing XML, 539–543
- XPath, 543–547
- XSLT transformation, 547–549
- Dot (.)
- format specifier, 329
 - member access (dot) operator, 229–230
- Double buffering, 589
- `double` keyword, 202
- Drawing shapes, 584–589
- Dynamic assemblies, 6
- Dynamic binding, 132–133
- Dynamic class loading, 82
- E**
- .ear (enterprise application archive) files, 623
- Early binding, 132–133
- Enterprise applications, 49–50
- Enum construct, 61–62
- `enum` data type, 192–193
- Enumerating through collections, 359–361
- Enumerator lists, 192–193
- Enums, 80
- Equal sign (=), assignment operator, 226
- Equal signs (==), equality operator
 - checking object equality, 89
 - comparing structs, 191
 - description, 223–224
- Equality testing
 - objects, 89
 - strings, 299
- `Equals()` method, 89
- Errors. *See* exceptions.
- Escaping specifiers, 331–332
- Event arguments, 468
- Event programming
 - example, 468–477
 - listeners, 467–468
- Events
 - data types, 8
 - definition, 468
- Evidence, 18
- Evidence-based security, 18–19
- Exception management
 - in the CLR, 22–25
 - delegates, 463
 - in Java, 25
 - overview, 22–25
- Exception specification, 270–272

- Exceptions
 bubbling up, 266
 C# *vs.* Java, 64–65, 82
 catching all possible, 264–266
 checked, 25
 checked/unchecked, 261
 CLR (Common Language Runtime), 272–273
 custom exception objects, 267–270
 definition, 22–25
 design considerations, 273–275
 handling, 274–275
 inheritance, 270–272
 propagating, 24–25
 supported, viewing, 272–273
 System.Exception object, 261
 try-catch-finally construct, 260–267
 unchecked (runtime), 25
 unhandled, pop-up box, 262
 vs. errors, 261
- Exclamation equal (!=), inequality operator, 224
- Exclamation point (!), logical negation, 215–217
- EXE file, 32, 36
- Exists method, 381–382
- Exiting processes, 654
- Experiences, .NET Framework, 3
- Explicit base interfaces, 74–75
- Explicit interface declaration, 174–179
- Explicit numeric conversions, 203–204
- Extensible Markup Language (XML). *See XML*
 (Extensible Markup Language).
- extern keyword, 100
- F
- Fall through, 250–253
- Fields
 const keyword, 121–122
 data types, 8
 internal keyword, 120
 private keyword, 120
 protected keyword, 120
 public keyword, 120
- readonly keyword, 122–123
- static keyword, 123
- usage guidelines, 117–120
- volatile keyword, 123–124
- FIFO (first-in, first out) mechanism, 352
- File class, 380
- Files, 380–387
- FileStream class, 373–374
- finalize() method, 16
- First-in, first out (FIFO) mechanism, 352
- Fixed-point numbers, formatting, 326–328
- float keyword, 200–202
- Floating-point data types, 200–202
- Flow control. *See* control flow.
- for loop, 249–250
- foreach statement, 253–255
- Format specifiers
 custom, 328–331
 date and time, 334
 definition, 321–322
 numerical formats, 323
- Formatting
 CultureInfo class, 322
 custom objects, 335–337
 date and time, 332–335
 DateTimeFormatInfo class, 322, 332–335
 IFormatProvider interface, 322
 NumberFormatInfo class, 322
 parsing strings, 337
 pattern strings, 321
- Formatting numbers
 with commas, 326–328
 currency, 322–324
 decimal points, 326–328
 escaping specifiers, 331–332
 fixed-point, 326–328
 hexadecimal specifiers, 331–332
 negative numbers, 331–332
 percentages, 328
 positive numbers, 331–332
 scientific notation, 324–326
 section specifiers, 331–332

- Formatting numbers (*cont.*)
 by value, 331–332
 zero values, 331–332
- Forms (Windows GUI)
 designing, 554–555
 modifying, 555–561
 Windows, 561–564
- Forms Designer, 553
- Function pointers. *See* pointers.
- G
- GAC (Global Assembly Cache), 632
- Garbage, 15
- Garbage collection
 in the CLR, 14–17
 definition, 14
 destroying objects, 127–129
 functional description, 15
 in the JVM, 17
 managed data, 14
 managed heaps, 14
 performance, 16
 roots, 15
 weak references, 16
- Garbage collectors
 compact, 17
 copying, 17
 definition, 17
 generational, 16–17
 incremental, 17
- Generational garbage collectors, 16–17
- Getters/setters, 125
- Getting
 data type size, 234
 data types, 234
 database data, 487–492
 object metadata. *See* reflection.
 Recordsets, 495–496
 System.Type object, 208
- Global Assembly Cache (GAC), 632
- Global attributes, 453
- goto statement, 64
- Graphical User Interfaces (GUIs). *See* GUIs
 (Graphical User Interfaces).
- Graphics class, 585–589
- Greater than equal ($>=$), greater than or equal operator, 225–226
- Greater than equal ($>>=$), shift right assignment operator, 228
- Greater than sign ($>$), greater than operator, 225
- Greater than signs ($>>$), right-shift operator, 221–222
- GUIs (Graphical User Interfaces)
 anchoring controls, 581–584
 choosing a library, 551–552
 component classes, 561–564
 component controls, 561–564
 controls, 561–564
 creating a new project, 552–554
 designing a form, 554–555
 developing an application, 551–552
 docking controls, 584
 double buffering, 589
 drawing, sample program, 590–587
 drawing shapes, 584–589
 example, 565–580
 Forms Designer, 553
 model view controller, 563
 modifying a form, 555–561
 modular, 551–552
 namespaces, 562
 rubber-band effect, 590
 System.Drawing.Graphics class, 585–589
 Windows forms, 561–564
- H
- Hash codes, generating, 358–359
- HashMap class. *See* Hashtable class.
- Hashtable class, 341
- Hashtable interface, 349–351
- Hexadecimal specifiers, 331–332
- HotSpot compiler, 13
- Hotspots, 13

- Hyphen greater than (->), pointer type operator, 236
- I**
- `ICollection` interface, 339–341
 - `IComparer` interface, 355–358
 - Identities, 19
 - `IDictionary` interface, 340–341
 - `IDictionaryEnumerator` interface, 340–341
 - `IDisposable` interface, 128
 - `IEnumerable` interface, 339–341
 - `IEnumerator` interface, 359–361
 - `if` statement, 243–247
 - `IFormatProvider` interface, 322
 - `IHashCodeProvider` interface, 340–341, 358–358
 - IIS (Internet Information Server), 52
 - ILDASM (Intermediate Language Disassembler), 42–47
 - `IList` interface, 339–341
 - `IMap` interface, 339–341
 - Implicit conversions, 203–204
 - Importing Java packages, 32–33
 - Increment operators, 220–221
 - Incremental garbage collectors, 17
 - Indexers
 - `C# vs. Java`, 80
 - definition, 445
 - integer indexes, 445–446
 - multiple indexes, 448–451
 - string indexes, 446–448
 - Indexing operators, 230
 - Indexing parameters, 80
 - Inheritance
 - abstract methods, 157–158
 - `base` keyword, 153–156
 - class access modifiers, 95–96
 - exceptions, 270–272
 - interfaces, 179–183
 - is-a relationships, 131
 - LRU (least recently used) cache, 163
 - method access modifiers, 146–153
 - methods, 106
 - overview, 131
 - properties, 125–126
 - single class, 131
 - static methods, 156–157
 - superclasses, calling, 153–156
 - superclasses, overriding, 57, 58, 146–153
 - vs.* composition, 163–164
 - Inheritance object hierarchy, 58
 - Initializing strings, 305–307
 - Input/output. *See I/O*.
 - `INSERT` queries, 493–495
 - Inserting database data, 493–495, 502–513
 - Instance constructors, 69
 - Instance members, 90
 - `int` keyword, 197–198
 - Integer indexes, 445–446
 - Integral data types, 194–200
 - `interface` keyword, 204–205
 - Interfaces
 - `C# vs. Java`, 59, 74–75, 170–174
 - containing fields, 82
 - explicit declaration, 174–179
 - inheritance, 74–75, 179–183
 - multiple, 174–179
 - `as` operator, 183–185
 - overview, 167–170
 - Interlocked operations, 437
 - Intermediate Language Disassembler (ILDASM), 42–47
 - `internal` keyword
 - class access modifier, 90
 - fields, 120
 - method access modifier, 99
 - properties, 124
 - Internet Explorer, 7, 56
 - Internet Information Server (IIS), 52
 - I/O
 - `AppendText` method, 380–381
 - asynchronous, 385–387
 - binary formatter, 387

- I/O (*cont.*)
 C# *vs.* Java, 76–77
 Copy method, 381–382
 deep copies, 395
 Delete method, 381–382
 directories, 380–387
 Directory class, 380
 Exists method, 381–382
 File class, 380
 files, 380–387
 Move method, 381–382
 Open method, 382–385
 OpenRead method, 382–385
 OpenText method, 382–385
 OpenWrite method, 382–385
 readers, 377–379
 serialization, basic, 388–392
 serialization, custom, 393–398
 serialization, overview, 387–388
 SOAP formatter, 387
 System.IO.BufferedStream class,
 375
 System.IO.FileStream class, 373–
 374
 System.IO.MemoryStream class, 376
 System.IO.NetworkStream class,
 376–377
 System.IO.Stream class, 369–371
 System.IO.TextReader class, 377–379
 System.IO.TextWriter class, 377–379
- is operator, 232
is-a relationships, 131
Isolated storage, 20
- J
Jagged arrays, 285–287
.jar files. *See also* .ear (enterprise application archive) files; .war (Web application archive) files.
- Java
 byte code, 54. *See also* MSIL (Microsoft Intermediate Language).
- C# comparison. *See C# vs. Java.*
exception management, 25
reference, 188
security management, 20–21
Java Database Connectivity (JDBC), 483–484
Java packages. *See also* namespaces.
 C# *vs.* Java, 77–78
 importing, 32–33
 java.awt, 666–668
 java.io, 661
 java.lang, 657–659
 java.sql, 662
 java.swing, 664–666
 java.text, 663
 java.util, 659–660
Java value types (primitives)
 converting to namespace equivalent, 61–62
 overview, 187–188
Java Virtual Machine (JVM). *See* JVM (Java Virtual Machine).
 java.awt package, 666–668
 Javac compiler, 13
 java.io package, 661
 java.lang package, 657–659
 Java.lang.Exception class. *See* System.Exception object.
 java.lang.Object class, 65–67, 86–89
 java.lang.String class, 65–67
 java.lang.StringBuffer class. *See* StringBuilder class.
 java.sql package, 662
 java.swing package, 664–666
 java.text package, 663
 java.text.DateFormat class, 332
 java.util package, 659–660
 java.util.HashMap class. *See* Hashtable class.
 JDBC (Java Database Connectivity), 483–484
 JDBC driver, 53
 JIT (just in time) compiler, 12–13
 Jump statements, 256
 Just in time (JIT) compiler, 12–13

JVM (Java Virtual Machine)

- C# *vs.* Java, 54–55
- garbage collection, 17
- vs.* CLR, 10–14

K

- Key-value pairs, collections, 349–351, 354–355
- Killing processes, 651–652

L

- Last-in, first out (LIFO) mechanism, 353–354
- Late binding, 132–133
- Least recently used (LRU) cache, 163
- Left angle bracket (<), less than operator, 224
- Left angle bracket equal (<=), less than or equal operator, 225
- Left angle brackets (<<), left-shift operator, 221–222
- Left angle brackets equal (<<=), shift left assignment operator, 228
- Left-associative operators, 238
- Less than equal (<=), less than or equal operator, 225
- Less than equal (<<=), shift left assignment operator, 228
- Less than sign (<), less than operator, 224
- Less than signs (<<), left-shift operator, 221–222
- LIFO (last-in, first out) mechanism, 353–354
- Listeners, 81, 467–468
- Loader, 12
- Logical operators, 215–217. *See also* bitwise operators; Boolean values.
- `long` keyword, 198–199
- LRU (least recently used) cache, 163

M

- `Main()` method, 31
- Main threads, 401
- Managed code, 5, 54
- Managed data, 14
- Managed execution environment, 10–14
- Managed heaps, 14

Manifests, assemblies, 624–625**Member access (dot) operators**, 229–230**Members**, 90**MemoryStream class**, 376**Metadata**

- classes, 90
 - databases, 498–499
 - object, creating, 11
 - object, getting. *See* reflection.
- Method access modifiers, 146–153
- Method invocation, 137–144
- Method modifiers, 73
- Method parameters
- and conversion rules, polymorphism, 144–146
 - `out` keyword, 115–116
 - overview, 109–113
 - `params` keyword, 116–117
 - `ref` keyword, 113–115

Methods

- `abstract`, C# *vs.* Java, 59
- C# *vs.* Java, 70, 73–74
- data types, 8–9
- `Equals()`, 89
- `finalize()`, 16
- inheritance, 106
- invoking remotely, 644–649
- `Main()`, 31
- modifiers, C# *vs.* Java, 73
- overloading, 106–109
- overridden base method, 101
- overview, 98
- `ReferenceEquals()`, 89

RMI (Remote Method Invocation), C# *vs.*

Java, 54

`static`, C# *vs.* Java, 58

threading, 417

`virtual`, C# *vs.* Java, 57, 136–137**Methods, access modifiers**`abstract`, 101–102`extern`, 100`internal`, 99`new`, 102–105

- Methods, access modifiers (*cont.*)
 override, 101
 private, 99
 protected, 99
 public, 99
 sealed, 99–100
 static, 99
 unsafe, 100
 virtual, 101
- Microdiscovery, 144
- Microsoft Intermediate Language (MSIL), 5, 11–12
- Microsoft Internet Explorer, 7, 56
- Minus equal (-=), decrement operator, 226–227
- Minus greater than (->), pointer type operator, 236
- Minus sign (-), subtraction operator, 213–215
- Minus signs (—), decrement operator, 220–221
- Model view controller, 563
- Modular GUIs, 551–552
- Modules, assemblies, 625
- Move method, 381–382
- MSIL (Microsoft Intermediate Language), 5, 11–12
- Multicasting, 465–467
- Multidimensional arrays, 283–285
- Multiuse attributes, 453
- N**
- Namespaces
 C# *vs.* Java, 77–78
 databases, 482
 GUIs, 562
 used in this book, 26
 XML, 518
- Name/value section, assemblies, 638
- Negative numbers, formatting, 331–332
- Nesting
 data types, 8, 90–93
 try-catch-finally blocks, 266
- .NET data provider, 482
- .NET Framework, overview. *See also* CLR
 (Common Language Runtime); JVM (Java Virtual Machine); MSIL (Microsoft Intermediate Language).
application domains, 6–7
assemblies, 5–6
class library, 25–26
clients, 3
CLS (Common Language Specification), 9–10
CTS (Common Type System), 8–9
definition, 3
exception management, 22–25
experiences, 3
garbage collection, 14–17
JIT (just in time) compiler, 12–13
language interoperability. *See* CLS (Common Language Specification).
managed execution environment, 10–14
memory management. *See* garbage collection.
namespaces in this book, 26
PE (portable executable) files, 5
runtime hosts, 7–8
security management, 17–21
servers, 4
tools, 4
user-centric services, 4
XML Web services, 3–4
- .NET SDK, 29
- .NET suite, 51–52
- NetworkStream class, 376–377
- new keyword
 class access modifier, 95
 method access modifier, 102–105
 properties, 124
- new operator, 231
- Nodes
 container, 543
 displaying attributes, 523–525
 querying for specific, 537
- Null objects, 86
- NumberFormatInfo class, 322
- Numerical format specifiers, 323

- O
Object class, 65–67, 86–89
Object creation operators, 231
Objects
 checking for equality, 89
 comparing in collections, 355–358
 custom, formatting, 335–337
 definition, 85
 destroying, 127–129
 `java.lang.Object` class, C# equivalent, 86–89
 null, 86
 pointers, 85
 references, 85
OLE DB data provider, 483
One-dimensional arrays, 279–282
OOP model, 57–60
Open method, 382–385
OpenRead method, 382–385
OpenText method, 382–385
OpenWrite method, 382–385
Operators. *See also* entries for specific operators.
 arithmetic, 213–215
 assignment, 226–229
 bitwise, 217–219
 C# vs. Java, 62–63
 casting, 230
 conditional, 231
 decrement, 220–221
 increment, 220–221
 indexing, 230
 left-associative, 238
 logical, 215–217
 member access (dot), 229–230
 object creation, 231
 overflow exception control, 235–237
 overloading, 62–63, 238–241
 pointer type, 237
 precedence, 237
 relational, 223–226
 right-associative, 238
 shift, 221–222
 string concatenation, 219–220
 type information, 231–234
 `org.w3c.dom` package, 664
 out keyword, 115–116
 Overflow exception control operators, 235–237
Overloading
 methods, 106–109
 operators, 238–241
Overridden base method, 101
override keyword, 137–144
Override keyword, 156
Overriding superclasses, 57, 58, 146–153
- P
Packages. *See* Java packages; namespaces.
params keyword, 116–117
Parentheses (()), casting operator, 230
Parsers (XML), 515
Parsing strings, 337
Pattern matching, 313–318
Pattern strings, 321
PE (portable executable) files, 5
Percent equal (%=), modulus assignment operator, 227
Percent sign (%)
 format specifier, 329
 modulus operator, 213–215
Percentages, formatting, 328
Performance, 37–38
Period. *See* dot.
Permission requests, 18–19
Plus equal (+=), increment operator, 226
Plus sign (+)
 addition operator, 213–215
 string concatenation operator, 219–220, 307–312
Plus signs (++) , increment operator, 220–221
Pointer type operators, 237
Pointers
 data type, 209–211
 mixing with references, 210

- Pointers (*cont.*)
 references, 85
 type-safe. *See* delegates.
- Policy-driven security, 18–19
- Polymorphism
 binding, 132–133
 C# *vs.* Java, 58
 dynamic binding, 132–133
 early binding, 132–133
 example, 133–136
 late binding, 132–133
 method invocation, 137–144
 method parameters and conversion rules, 144–146
 microdiscovery, 144
 override keyword, 137–144
 overview, 132
 static binding, 132–133
 virtual dispatching, 143–144, 156. *See also* casting.
 virtual keyword, 137–144
 virtual method calls, 137–144
 and virtual methods, 132–133
 VPTR (vpointer), 133
 VTABLEs, 133
- Portable executable (PE) files, 5
- Positive numbers, formatting, 331–332
- Pound sign (#), format specifier, 329
- Precedence, operators, 237
- Presentation layer, 52
- Primitive data types. *See* Java value types.
- Principals (security), 19
- `private` keyword
 class access modifier, 93–94
 fields, 120
 method access modifier, 99
 properties, 124
- Processes
 creating, 651–652
 detecting completion, 653–654
 exiting, 654
 invoking methods remotely, 644–649
- killing, 651–652
overview, 642
querying, 649
querying current application domain, 642–643
querying remote application domain, 643–644
redirecting output, 652–653
`System.Diagnostics` namespace, 649
- Programmatic security, 18
- Programming platforms. *See also* C# *vs.* Java.
 APIs (application programming interfaces), 50
 definition, 49
 enterprise applications, 49–50
- Propagating exceptions, 24–25
- Properties
 C# *vs.* Java, 79–80
 data types, 9
 getters/setters, 125
 inheritance, 125–126
 keywords, 124
 overview, 124, 439–444
 sealed, 125
 virtual, 125
- `protected` keyword
 class access modifier, 93–94
 fields, 120
 method access modifier, 99
 properties, 124
- Protection domains, 21
- `public` keyword
 class access modifier, 90
 fields, 120
 method access modifier, 99
 properties, 124
- Pull model, 517
- Q
- Qualified names, 229–230
- Querying
 current application domain, 642–643

- processes, 649
remote application domain, 643–644
Question mark colon (?:), conditional operator, 231
Queue interface, 352
- R
- Rank (arrays), 283
Readers, 377–379
ReaderWriterLock class, 433–437
Reading XML, 516–517, 521–530
`readonly` keyword, 122–123
Read/write locking, 433–437
Recordsets, getting, 495–496
Redirecting process output, 652–653
`ref` keyword, 113–115
ReferenceEquals() method, 89
References, 85
Referents, 209–211
Reflection
 browsing members, 604–610
 core classes, 600–604
 definition, 599
 fields, setting, 610–618
 generating dynamic IL, 618–622
 invoking methods, 610–618
 properties, setting, 610–618
 querying members, 604–610
 `System.Type` class, 600–604
 uses for, 599–600
Reflection emit, 618
Regex class, 313–318
Relational operators, 223–226
Remote Method Invocation (RMI), 54
Resources, assemblies, 625
Right angle bracket (>), greater than operator, 225
Right angle bracket equal (>=), greater than or equal operator, 225
Right angle brackets (>>), right-shift operator, 221–222
Right angle brackets equal (>>=), shift right assignment operator, 228
Right-associative operators, 238
RMI (Remote Method Invocation), 54
Role-based security, 19
Rolling back transactions, 498
Roots, 15
RowSet class. *See DataSet class.*
Rubber-band effect, 590
Running C# programs, 32
Runtime architecture, 54–57
Runtime (unchecked) exceptions, 25
Runtime hosts, 7–8
- S
- Safe code, 54
Savepoints, 498
SAX (Simple API for XML), 517
`sbyte` keyword, 195–196
Scientific notation, formatting, 324–326
`sealed` keyword, 95, 99–100
Sealed properties, 125
Section specifiers, 331–332
Security management
 authenticated identity, 19
 authentication, 19
 authorization, 19
 cryptography, 20
 declarative security, 17–18
 evidence, 18
 evidence-based security, 18–19
 identities, 19
 isolated storage, 20
 in Java, 20–21
 overview, 17–21
 permission requests, 18–19
 policy, 18
 policy-driven security, 18–19
 principals, 19
 programmatic security, 18
 protection domains, 21
 role-based security, 19
Security policy, 18

- SELECT queries, 487–492
Serialization
 basic, 388–392
 C# *vs.* Java, 54
 custom, 393–398
 overview, 387–388
Servers, 4
Setters/getters, 125
Sharing assemblies, 632
Shell executables, 7, 56–57
Shift operators, 221–222
short keyword, 196–197
Short-circuiting, 244
Simple API for XML (SAX), 517
Simulating threads, 401–402
Single class inheritance, 131
Single-tag sections, assemblies, 637–638
Singleton classes, 69
`sizeof` operator, 234
Slash (/), division operator, 213–215
Slash equal (/=), division assignment operator, 227
SOAP formatter, 387
SortedList interface, 354–355
Splitting strings, 311–312
SQL Server, 483
Square brackets ([])
 indexing operator, 230
 pointer type operator, 236
Stack interface, 353–354
Starting threads, 405–406
States, threads, 407
Static assemblies, 6
Static binding, 132–133
Static constructors, 69
`static` keyword
 abstract methods, 157–158
 fields, 123
 method access modifier, 99, 101
 properties, 124
Static members, 90, 463–464
Static methods
 C# *vs.* Java, 58
 inheritance, 156–157
 virtual dispatching, 156
Stopping threads, 417–418
Stored procedures, 495–496
Stream class, 369–371
String class.
 CLS-compliant public methods, 296, 297
 compare methods, 296, 298
 concatenation methods, 296, 298
 `Equals` method, 307
 `IComparable` interface, 293
 `IEnumerable` interface, 293
 immutability, 293
 interfaces, 293–295
 Java, 65–67
 methods, summary table, 300–304
String concatenation operators, 219–220
String indexes, 446–448
StringBuffer class. *See* `StringBuilder` class.
StringBuilder class, 307
Strings. *See also* formatting.
 comparing, 305–307
 concatenating, 307–312
 initializing, 305–307
 parsing, 337
 pattern matching, 313–318
 splitting, 311–312
 testing equality, 299
 tokenizing, 311–312
Strong names, 632
struct data type, 189–192
Structs, 61–62, 78–79
super keyword. *See* base keyword.
Superclasses
 calling, 153–156
 overriding, 57, 58, 146–153
switch statement, 64, 250–253
System.Array class, 279–282

`System.Collections` classes. *See* `Collections` classes.

`System.Collections` interfaces, 339–341

`System.Diagnostics` namespace, 649

`System.Drawing.Graphics` class, 585–589

`System.Exception` object, 261

`System.IO.BufferedStream` class, 375

`System.IO.Directory` class, 380

`System.IO.File` class, 380

`System.IO.FileStream` class, 373–374

`System.IO.MemoryStream` class, 376

`System.IO.NetworkStream` class, 376–377

`System.IO.Stream` class, 369–371

`System.IO.TextReader` class, 377–379

`System.IO.TextWriter` class, 377–379

`System.String` class. *See* `String` class; strings.

`System.Text.RegularExpressions.Regex` class, 313–318

`System.Threading` classes, 403–404

`System.Threading` delegates, 404

`System.Threading` namespace, 402–404

`System.Threading.Thread` class, 404–417

`System.Type` class, 600–604

`System.Type` object, getting, 208

`System.Xml.XmlReader` class, 517–519

`System.Xml.XmlWriter` class, 517–519

T

Targets, 452

`TextReader` class, 377–379

`TextWriter` class, 377–379

`Thread` class, 404–417

Thread pooling, 418–422

Thread safety, 341

Thread synchronization

- atomic operations, 426
- interlocked operations, 437
- overview, 426–433

`ReaderWriterLock` class, 433–437

read/write locking, 433–437

wait queues, 432

Threads

- `Abort` method, 417–418
- background, 401
- C# vs. Java, 75–76
- delegates, 404–405
- life cycle, 406–417
- locking. *See* thread synchronization.
- main, 401
- methods, 417
- simulating, 401–402
- starting, 405–406
- states, 407
- stopping, 417–418

`System.Threading` classes, 403–404

`System.Threading` delegates, 404

`System.Threading` namespace, 402–404

Threads, creating

- asynchronous callback, 424–426
- in the background, 422–424
- manually, 418

`System.Threading.Thread` class, 404–417

`System.Threading.Threadpool` class, 418–422

`System.Threading.Timer` class, 422–424

`ThreadStart` delegate, 404–417

Thread-safe data, 402

`ThreadStart` delegate, 404–417

Tilde (~), complement operator, 218–219

Time. *See* date and time.

Tokenizing strings, 311–312

Tools, 4, 38–47

Transaction isolation levels, 499

Transaction processing, 496–498

Transactions, 52

`try-catch-finally` construct, 64–65, 260–267

Type class, 600–604
Type information operators, 231–234
Type safety, 341
Typed collections, 82
`typeof` operator, 208, 234
Type-safe code, verifying, 12
Type-safe collections, 361–362

U

`uint` keyword, 197–198

`ulong` keyword, 198–199

Unboxing, 61–62, 206–207

Unchecked exceptions, 25, 261

`unchecked` operator, 236–237

Underlying types, 80

Unmanaged data types, 209–211

Unsafe code, 209–211

Unsafe context, 209

`unsafe` keyword, 100

Unsafe mode, 209

`UPDATE` queries, 493–495

Updating database data, 493–495, 502–513

User-centric services, 4

`ushort` keyword, 196–197

`using` keyword, 32–33

V

Validating parsers, 515

Variable scoping, 126–127

Variables, 70–72

Versioning, 634–636

Vertical bar equal (`|=`), OR assignment operator, 228

Vertical line (`()`), OR operator, 218

Vertical lines (`||`), logical OR, 215–217

Virtual dispatching, 143–144, 156

`virtual` keyword

abstract methods, 157–158

method access modifier, 101

polymorphism, 137–144

static methods, 156

Virtual methods

C# *vs.* Java, 57

calling, 137–144

polymorphism, 132–133

Virtual properties, 125

Visual Studio .NET

description, 38–42

developing an application, 552–561

`volatile` keyword, 123–124

Vpointer (VPTR), 133

VPTR (vpointer), 133

VTABLEs, 133

W

Wait queues, 432

.war (Web application archive) files, 623

Weak references, 16

Well-formedness, 515

`while` loop, 247–248

Windows forms, 561–564

designing, 554–555

modifying, 555–561

Windows, 561–564

Writing XML, 516–517, 530–533, 539–543

X

XML (Extensible Markup Language). *See also* DOM API.

displaying node attributes, 523–525

documents, searching, 543–547

DOM (Document Object Model), 515

namespaces, 518

and .NET, 516–517

parsers, 515

parsing to create objects, 525–528

printing object state, 530–533

pull model, 517

reading, 516–517, 521–530

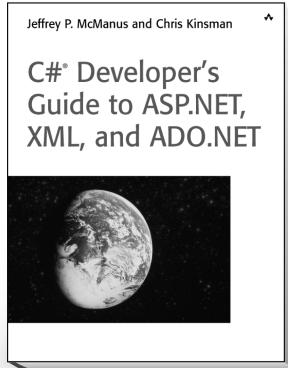
sample document, 519–520

SAX (Simple API for XML), 517

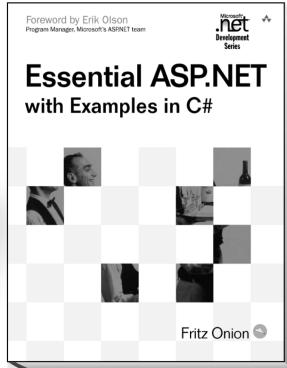
support in Java, 515–516

- validating against a DTD, 528–530
 - validating parsers, 515
 - well-formedness, 515
 - writing, 516–517, 530–533, 539–543
 - XmlNodeReader class, 519
 - XmlReader class, 517–519
 - XmlTextReader class, 518
 - XmlTextWriter class, 519
 - XmlValidatingReader class, 519
 - XML Web services, 3–4, 53
 - XmlNodeReader class, 519
 - XmlReader class, 517–519
 - XmlTextReader class, 518
 - XmlTextWriter class, 519
 - XmlValidatingReader class, 519
- Z
- Zero (0), format specifier, 329
 - Zero values, formatting, 331–332

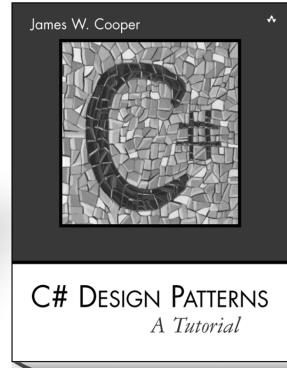
Also from Addison-Wesley



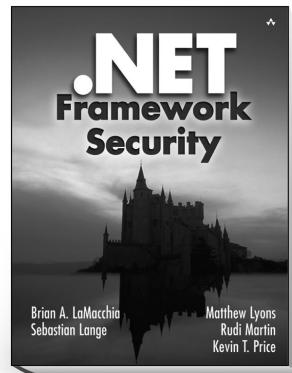
0-672-32155-6



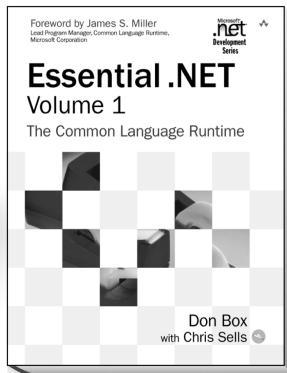
0-201-76040-1



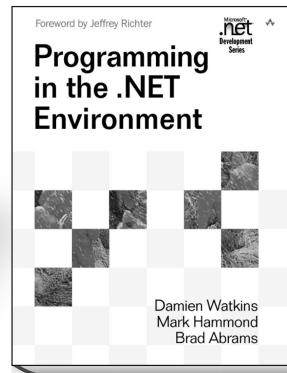
0-201-4453-2



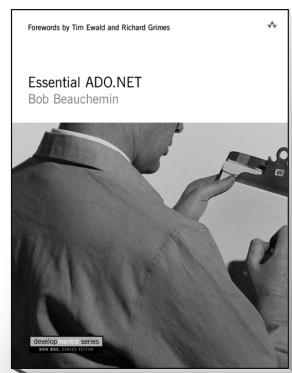
0-672-32184-X



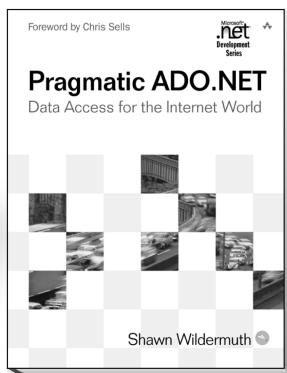
0-201-73411-7



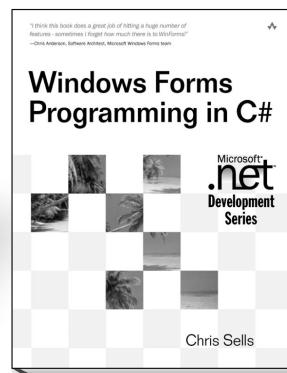
0-201-77018-0



0-201-75866-0



0-201-74568-2



0-321-11620-8

For more information about these and other titles on C#, .NET, and Java, please visit
<http://www.awprofessional.com>