



Index

Symbols

- # character, in keywords, 53
- #define directives, processing, 63, 65–66
- #include directives, requirement for, 6

A

- abstract accessors, 325–326, 333
- abstract classes
 - described, 274–275
 - interfaces, 391
- abstract methods, 25–28, 311–312, 313–315
- access
 - array elements, 369
 - interface members, 378–380
 - members, 79–86
 - See also* pointer element access; pointer member access; *this* access
- access modifiers, kinds of, 280
- access paths, hidden members, 380
- access restrictions, namespaces, 78
- accessibility
 - of class members, 19
 - constructed types, 491
 - of methods, 74
 - See also* declared accessibility
- accessibility constraints, types, 85
- accessibility domains, 81–84
- accessible names, hiding, 92
- accessing, protected members, 478
- accessor declarations, attributes, 417
- accessors
 - defined, 31, 317
 - described, 319–324
 - interface property declarations, 377
 - See also* abstract accessors; event accessors; get accessor; set accessors
- addition operators, 202–204
- additive operators
 - described, 13
 - precedence of, 150
- addresses, fixed pointer initializers, 447
- address-of operator, 441–442
- alert escape sequence, Unicode encoding, 59
- alias directives, using, 267, 268, 492
- aliases. *See* explicit keywords; keywords; reserved words
- analysis, lexical, 47–50
 - See also* lexical grammar; lexical structure
- anonymous invocations, delegates, 399
- anonymous methods, 525–538
 - blocks, 528
 - conversions, 526–527, 534–535
 - definite assignment states, 533–534
 - delegate instance equality, 533
 - evaluation, 532–533
 - expressions, 525
 - generics, 463–466
 - implementation example, 535–538

applicable function members ■ backslash escape sequence

- outer variables, 528–532
 - signatures, 525
- applicable function members, 166
- application domains, 73
- application entry point methods, generic
 - class declarations, 484
- application startup, 73–74
- application termination, 74
- argument lists, 162–165
- arguments
 - passing, 162
 - See also* type arguments
- arithmetic, and pointers, 443–444
- arithmetic expressions, using classes to mold, 27
- arithmetic operators
 - described, 198–207
 - See also* addition operators; division operators; multiplicative operators; remainder operators; subtraction operators
- array access, 178–179
- array covariance, 164, 223, 369–370
- array creation expressions, 184–186
- array elements. *See* elements
- array initializers
 - defined, 371
 - See also* variable initializers
- array type variables, possible contents, 11
- array types
 - defined, 36, 112
 - described, 8
- arrays, 367–372
 - described, 35–37
 - members of, 79
 - syntactic grammar, 610
 - See also* elements
- as operator, 216, 510
- assemblies
 - described, 6
 - file extensions, 5
 - See also* Intermediate Language (IL) instructions; runtime libraries
- assignment
 - structs, 359
 - See also* definite assignment
- assignment operators
 - described, 13, 221–226
 - exceptions and, 108
 - precedence of, 150
- assignment rules, output parameters, 118
- assignment states
 - instance variables of structs, 116
 - variables, 115, 118
 - See also* definite assignment states
- associated constant values, enum members, 395–397
- associativity, operators, 149–151
- atomacy, of variable references, 133
- attribute classes, 411–414
- attribute instances
 - described, 420–421
 - runtime retrieval, 421
- attribute parameter types, defined, 414
- attribute sections, defined, 415
- attributes, 411–427
 - classes, 411–414
 - constructed types, 493
 - described, 42–43
 - instances, 420–421
 - for interoperation, 427
 - partial types, 554
 - reserved attributes, 422
 - specification, 414–420
 - syntactic grammar, 613
 - for target assemblies, 263
- AttributeUsage attribute, 411, 422–423
- automatic memory management. *See* garbage collection

B

- backslash character, characters following, 58
- backslash escape sequence, Unicode encoding, 58

- backspace escape sequence, Unicode
 - encoding, 59
 - base access, 181
 - base classes
 - described, 20, 275–277
 - interfaces of constructed
 - types, 489–490
 - members, 83
 - partial class declarations, 555
 - specification of and
 - generics, 475–476
 - base interfaces
 - described, 374–375, 476
 - implementation using classes or structs, 381
 - partial types, 555
 - reimplementation of an interface, 390
 - base members, in nested types, 283
 - base types, 157
 - better conversions, 167
 - better function members, 167
 - binary numeric promotions, 155–156
 - binary operator overload
 - resolution, 153
 - binary operators
 - delegate combination, 204
 - described, 341–343
 - ID string example, 576
 - notation, 152
 - overloading, 151
 - precision, 106
 - binary subtraction operators, delegate removal, 206
 - binding
 - type parameters, 475
 - See also* name binding
 - bitwise complement operator, 195
 - block statements, definite assignment rules, 122
 - blocks
 - anonymous methods, 528
 - defined, 14, 46
 - described, 232–233
 - invariant meaning in, 172–173
 - iterator blocks, 539–540
 - names within, 90
 - reachability of function
 - members, 231
 - body. *See* class body; interface body; method body; struct body
 - bool value type
 - default value, 103
 - defined, 8, 109
 - boolean conditional logical operators, 219
 - boolean equality operators, 212
 - boolean expressions
 - assignment states, 121
 - described, 227
 - boolean literals, defined, 55
 - boolean logical operators, 218
 - boolean types, database example using structs, 364–366
 - boxed instances, invocations, 169
 - boxing
 - example, 10
 - struct types, 360
 - type parameters, 504–506
 - boxing conversions
 - defined, 137
 - described, 112–114
 - break statement
 - definite assignment rules, 125
 - described, 250
 - example, 16
 - byte value type
 - default value, 103
 - defined, 105
- C**
- <c> tag, 564
 - C# programming language
 - anonymous methods, 525–538
 - arrays, 367–372

callable entities ■ classifications

- attributes, 411–427
- basic concepts, 73–99
- C# 2.0 overview, 457–472
- classes, 273–354
- conversions, 135–146
- delegates, 399–405
- enums, 393–398
- exceptions, 407–410
- expressions, 147–228
- generics, 473–523
- interfaces, 373–390
- iterators, 539–551
- lexical structure, 45–71
- namespaces, 263–272
- overview, 3–43
- partial types, 553–557
- statements, 229–262
- structs, 355–366
- types, 101–114
- unsafe code, 429–453
- variables, 115–133
- callable entities, defined, 399
- calling. *See* invocations
- candidate user-defined operators, 154
- captured outer variables, 529
- carriage escape sequence, Unicode
 - encoding, 59
- carriage-return character, 49
- cast expressions
 - described, 197
 - explicitly converting operands, 155
- cast operators, overloading, 151
- catch clause, 255
- char value type
 - default value, 103
 - defined, 106
- character literals
 - described, 58–59
 - processing comments, 50
- character processing, Unicode encoding, 8
- checked operators, 190–193
- checked statement
 - defined, 14
- described, 258
- example, 17
- circular dependencies, static fields with
 - variable initializers, 351
- class base specifications, 275–277
- class body, 277
- class constraints, types, 501
- class declarations, 18, 273–277
- class member declarations, 280
- class members
 - declared accessibility, 80
 - described, 79, 277–287
 - kinds of, 18–19
 - See also* constants; constructors;
 - destructors; events; fields; indexers;
 - methods; nested types; operations;
 - properties
- class type variable, possible contents, 11
- class types
 - defined, 9, 110
 - described, 8
- classes, 273–354
 - compared to structs, 34, 35, 357, 358, 359, 361
 - constants, 287–289
 - described, 18–34
 - destructors, 352–354
 - events, 327–329
 - indexers, 333–338
 - instance constructors, 343–349
 - instance variables in, 116
 - interface implementations, 375, 380–391
 - methods, 299–317
 - operators, 338–343
 - properties, 317–326
 - syntactic grammar, 602–609
 - See also* attribute classes; base classes;
 - constants; destructors; events;
 - exception classes; fields; indexers;
 - instance constructors; methods;
 - operators; properties; static constructors; types
- classifications, expressions, 147–149

- cleanup. *See* garbage collection
- closed types, 489
- <code> tag, 564
- collection expression, generics, 511
- COM. components, attributes for
 - interoperation, 427
- comments, 49–50, 584
- comparison operators, 209
- compilation
 - attributes, 421
 - partial types, 472
- compilation units, 263–264
- compile-time processing, object
 - creation expressions, 183
- compile-time types, defined, 308
- compiling
 - field-like events, 330
 - programs, 45
- component-oriented programming,
 - defined, 3
- compound assignment operators
 - defined, 222
 - described, 224–225
- conditional attributes, 423–426
- conditional compilation
 - directives, 66–69
- conditional compilation symbols
 - conditional methods, 424
 - described, 63
 - preprocessing expressions, 64
- conditional logical operators, 218–220
- conditional methods, restrictions, 423
- conditional operators
 - described, 13, 220–221
 - precedence of, 150
- Console class, properties, 324
- constant expressions
 - conversion of, 137
 - defined, 105
 - described, 226–227
- constant values, enum
 - members, 395–397
- constants
 - char value type, 106
 - declaring for simple types, 105
 - described, 287–289
 - example of, 29
 - members of classes, 19
 - using static read-only fields for, 292
 - See also* enum types; local constants
- constituent types, 280
- constraints
 - generics, 500–507
 - inheritance, 503
 - partial types, 554
 - See also* accessibility constraints;
 - class constraints
- constructed types, generics, 459, 487–493
- constructor initializers, 344–345
- constructors
 - described, 30
 - example of, 29
 - executing, 346–348
 - ID string example, 574
 - members of classes, 19
 - structs, 361–362
 - See also* default constructors;
 - instance constructors; static constructors
- continue statement
 - definite assignment rules, 125
 - described, 251
 - example, 16
- contracts, and interfaces, 373
- Control-Z character, 48
- conversion operators
 - described, 341–343
 - in generic classes, 482
 - ID string example, 577
 - rules for, 341
 - See also* is operator
- conversions, 135–146
 - anonymous methods, 526–527

covariance ■ declarations

- as operator, 216
- boxing, 112–114, 137
- constructed types, 491–492
- method groups, 466, 534–535
- pointers, 437–438
- of simple types, 105
- type parameters, 506–507
- unboxing, 112, 114, 142
- See also* explicit conversions; explicit enumeration conversions; explicit numeric conversions; explicit reference conversions; implicit constant expression conversions; implicit conversions; implicit enumeration conversions; implicit numeric conversions; implicit reference conversions; predefined conversions; standard explicit conversions; standard implicit conversions
- covariance. *See* array covariance
- creation, arrays, 369
- .cs file extension, 4
- Current property, enumerator objects, 543

D

- data independence, and execution order, 99
- data structures. *See* classes; structs
- databases, examples of using
 - structs, 362–366
- decimal addition, 203
- decimal comparison operators, 211
- decimal division, 201
- decimal integers
 - as integer literals, 55
 - int and long values, 56
- decimal multiplication, 199
- decimal negation, 195
- decimal numeric types, precision, 9
- decimal remainder, 202
- decimal subtraction, 206
- decimal value type
 - conversion to an integral type, 139
- conversions, 140
 - default value, 103
 - defined, 8
 - described, 108–109
 - mixing in operations with double or float value types, 155
- declaration directives, 64
- declaration space
 - described, 76
 - nested blocks in, 77
 - rules, 75
 - See also* global declaration space; local variable declaration space
- declaration statements
 - defined, 14
 - definite assignment rules, 123
 - described, 234–236
- declarations
 - attributes, 417–418
 - compared to scope, 90
 - compared to using directives, 265
 - conditional compilation symbols, 63
 - constructors, 30
 - delegates, 399–402
 - enums, 393–394
 - explained, 75–77
 - interfaces, 373–375
 - namespaces, 94
 - new members hiding inherited members, 92
 - pointers, 434
 - properties, 31
 - scope of namespaces, classes, structs, or enumeration members, 89
 - structs, 355–356
 - unsafe contexts, 429
 - See also* class declarations; generic class declarations; generic delegate declarations; generic interface declarations; generic struct declarations; member declarations; nested type declarations; partial type declarations; type declarations

- declared accessibility
 - described, 80–81
 - inheritance of members, 77
 - nested types, 282–283
- decrement operators. *See* postfix
- increment and decrement operators
- decrementing, pointers, 442
- default constructors, 103, 348
- default value initialization fields, 296
- default values
 - enum members, 40
 - field initialization, 295
 - structs, 359–360
 - variables, 119–132
- defaults
 - base classes, 20, 489
 - declared accessibility, 80–81
 - default value for value types, 103
 - enum types, 39
 - instance constructors, 31
 - line indicators, 70
 - value expressions and generics, 508
- `#define` directives, processing, 63, 65–66
- definite assignment
 - explained, 119–132
 - `yield` return statement, 547
- definite assignment states
 - described, 533–534
 - instance variables, 120
- delegate combination, 204
- delegate creation expressions, 187–189
 - described, 527
 - generics, 521–522
- delegate declarations
 - attributes, 417
 - See also* generic delegate declarations
- delegate equality operators, 215
- delegate instance equality, anonymous methods, 533
- delegate invocations, 177
- delegate type variables, possible contents, 11
- delegate types
 - conversion from method groups, 534
 - defined, 9, 112
 - described, 8, 40–41, 400
 - implicit conversions, 526
- delegates
 - anonymous methods, 531, 533
 - compatible with anonymous methods, 465
 - described, 399–405
 - generic methods, 499
 - members of, 79
 - removal, 206
 - syntactic grammar, 612
- delimited comments, 49–50
- derived classes
 - declaring properties, 321
 - interface mappings, 387
- destructors
 - described, 34, 352–354
 - execution of and exceptions, 409
 - function members as, 157
 - ID string example, 575
 - member names reserved for, 287
 - members of classes, 19
 - structs, 362
- diagnostic directives, 69
- dimensions. *See* rank
- directives. *See* alias directives; conditional compilation directives; declaration directives; diagnostic directives; `#include` directives; line directives; preprocessing directives; region directives; `using` directive
- `Dispose` method, enumerator objects, 543
- division operators, 199–201
- `.dll` file extension, 5
- `DllImport` attribute, external methods, 315

do statement ■ events

- do statement
 - definite assignment rules, 124
 - described, 244
 - example, 15
 - documentation comments, 561–582
 - example, 577–580
 - overview, 561–562
 - processing documentation files, 572–577
 - recommended tags, 563–572
 - XML example, 580–582
 - documentation files, processing, 572–577
 - double quote escape sequence, Unicode encoding, 58
 - double value type
 - conversion to an integral type, 139
 - default value, 103
 - precision of, 107–108
 - dynamic allocation of memory, *new* operator, 183
 - dynamic type casts, example of, 38
- E**
- efficiency, structs compared to classes, 35
 - element access
 - described, 178–180
 - overloading, 152
 - element types
 - defined, 35, 368
 - types permitted, 36
 - elements
 - access, 369
 - defined, 35
 - described, 116
 - initializing default values, 119
 - in unsafe contexts, 448
 - embedded statement nonterminal, 229
 - empty statements, 233
 - encoding, of files, 45
 - end points, 230–231
 - entry points
 - in applications, 73–74
 - return types, 74
 - enum types
 - default value, 103
 - defined, 9
 - described, 7, 39–40, 393–398
 - explicit enumeration conversions, 141
 - syntactic grammar, 612
 - enumerable interfaces, defined, 540
 - enumerable objects, 544
 - enumeration addition operators, 203
 - enumeration comparison operators, 212
 - enumeration conversions. *See* explicit enumeration conversions
 - enumeration logical operators, 217
 - enumeration members
 - declared accessibility, 81
 - defined, 79
 - enumeration subtraction operators, 206
 - enumeration types, defined, 109
 - enumerator interfaces, defined, 539
 - enumerator objects, described, 540–543
 - equality operators, 13, 150
 - error conditions, exception handling, 407
 - error string format, 572
 - escape sequences, Unicode encoding, 51–52, 58–59, 585
 - evaluation, anonymous methods, 532–533
 - event access constructs, 160
 - event access, expressions as, 148
 - event accessors, 331–332
 - event assignment operators, 226
 - event declarations, attributes, 417
 - event handlers, 32
 - event string format, 572
 - events
 - argument list, 162
 - described, 32–33, 327–329
 - example of, 30
 - function members as, 157
 - ID string example, 576
 - interfaces, 377
 - member names reserved for, 287
 - members of classes, 19
 - type parameters, 500

- <example> tag, 564
 - exception classes, list of, 409–410
 - exception propagation, defined, 231
 - exception statements, generics, 510
 - <exception> tag, 565
 - exceptions
 - described, 407–410
 - and floating point operators, 108
 - and predefined implicit conversions, 135
 - .exe file extension, 5
 - execution order, 99
 - expanded form, of function members, 166
 - explicit base interfaces
 - defined, 489
 - relationship with base interfaces, 374
 - explicit conversions
 - described, 138–142
 - See also* standard explicit conversions
 - explicit enumeration conversions, 140
 - explicit interface member implementations
 - described, 381–384, 485
 - example of, 38
 - explicit keywords, conversion operator declarations, 341
 - explicit numeric conversions, 138–140
 - explicit reference conversions, 141
 - expression statements
 - defined, 14
 - definite assignment rules, 122
 - described, 236–237
 - example, 15
 - expressions, 147–228
 - anonymous methods, 525
 - boolean expressions, 227
 - classifications, 147–149
 - conditional operators, 220–221
 - definite assignment rules, 128–132
 - described, 11–13
 - function members, 157–169
 - generics, 508–511
 - lock statements, 259
 - member lookups, 156–157
 - pointers in, 438–445
 - primary expressions, 170–193
 - relational and type-testing operators, 209
 - in return statements, 253
 - shift operators, 207–208
 - simple names and member access, 498
 - syntactic grammar, 593–597
 - unary operators, 193–197
 - See also* arithmetic expressions; constant expressions; delegate creation expressions; operands; operators; preprocessing expressions
 - external constructors, defined, 344
 - external destructors, defined, 352
 - external indexer, defined, 336
 - external methods, 315–316
 - external operators, defined, 339
- F**
- features of C#, 3–4
 - field initialization, 295
 - field initializers, structs, 360
 - field string format, 572
 - field-like events, 329–331
 - fields
 - compared to properties, 31
 - described, 20–21, 290–298
 - example of, 29
 - ID string example, 574
 - members of classes, 19
 - See also* instance fields; static fields
 - file extensions
 - assemblies, 5
 - source files, 4

files ■ generics

- files
 - encoding of, 45
 - See also* documentation files; header files; source files
 - fixed statements, 446–449
 - fixed variables, 436–437
 - float value type
 - conversion to an integral type, 139
 - default value, 103
 - precision of, 107–108
 - floating point addition, 203
 - floating point comparison
 - operators, 210–211
 - floating point division, 200
 - floating point multiplication, 198
 - floating point negation, 194
 - floating point numeric types, precision, 9
 - floating point operators
 - exceptions and, 108
 - precision of, 108
 - floating point remainder, 201
 - floating point subtraction, 205
 - floating point types
 - compared to decimal type, 108
 - defined, 8
 - precision of, 107–108
 - See also* double value type; float value type
 - for statement
 - definite assignment rules, 124
 - described, 244–246
 - example, 16
 - foreach statement
 - definite assignment rules, 127
 - described, 246–248
 - example, 16
 - generics, 511
 - form feed escape sequence, Unicode
 - encoding, 59
 - fragmentation, of the heap, 447
 - fully qualified names
 - described, 94
 - interface members, 380
 - method invocations, property access, or indexer access, 382
 - nested types, 282
 - function member invocations
 - described, 168–169
 - runtime processing, 163
 - function members
 - assignment of output parameters, 120
 - defined, 278
 - definite assignment state, 121–132
 - described, 29–34, 157–169
 - enumerable objects, 544
 - iterator blocks, 539
 - iterators, 467
 - See also* constructors; destructors; events; indexers; instance function members; properties; static function members
 - function pointers, compared to delegates, 40, 112, 399
 - functional notation. *See* notation
- G**
- garbage collection
 - at application termination, 74
 - destructors, 34
 - explained, 95–98
 - fixed and moveable variables, 436
 - initialization of default variables, 119
 - pointers, 433
 - generic class declarations, 473–484
 - generic delegate declarations, 486–487
 - generic interface declarations, 484–486
 - generic methods, 493–500
 - generic struct declarations, 484
 - generic type instantiation, 460
 - generics, 473–523
 - anonymous methods, 463–466
 - class declarations, 473–484
 - constraints, 500–507
 - constructed types, 487–493

- delegate declarations, 486–487
- expressions and
 - statements, 508–511
- interface declarations, 484–486
- lookup rules revised, 511–522
- methods, 493–500
- overview, 457–463
- struct declarations, 484
- get accessor
 - defined, 31
 - described, 319
 - observable side effects, 323
- GetEnumerator methods,
 - defined, 544
- global attributes, compilation
 - units, 263
- global declaration space, defined, 75
- global namespace, defined, 78
- global scope, attributes, 417
- goto statement
 - definite assignment rules, 125
 - described, 251–253
 - example, 16
- governing types, switch
 - statements, 239
- grammar, 583–617
 - extensions for unsafe code, 614–617
 - syntactic grammar, 591–614
- grammar productions, defined, 45
- grammars, types of, 45–47
- grouping, and operators, 64

H

- handlers, events, 328
- header files, requirement for, 6
- heap, fragmentation of, 447
- Hello, World program, 4–5
- hexadecimal escape sequences, string
 - literals, 61
- hexadecimal integer literals, 55
- hidden members, interface
 - mapping, 386
- hiding
 - inherited members, 75, 279
 - in multiple-inheritance
 - interfaces, 380
 - and nested types, 283, 483
 - See also* name hiding
- horizontal tab escape sequence,
 - Unicode encoding, 59

I

- ID strings, 572–577
- identical simple names and type
 - names, 175–176
- identifiers
 - described, 52–54
 - lexical grammar, 585
- identity conversions, defined, 136
- IEEE 754 arithmetic
 - floating point addition, 203
 - floating point comparison
 - operators, 211
 - floating point division, 200
 - floating point multiplication, 198
 - floating point remainder, 201
- if statement
 - definite assignment rules, 123
 - described, 237–238
 - example, 15
- implicit constant expression
 - conversions, defined, 137
- implicit conversions
 - anonymous method
 - expressions, 526
 - class types to base class types, 20
 - described, 135–137
 - examples, 465
 - method groups, 534
 - null types, 492
 - to char value type, 106

implicit enumeration conversions ■ instance types

- to interface types, 38
- user-defined, 342
- See also* standard implicit conversions
- implicit enumeration conversions, 136
- implicit keywords, conversion operator
- declarations, 341
- implicit numeric conversions, 136
- implicit reference conversions, 136
- importing, types, 269
- #include directives, requirement for, 6
- <include> tag, 565
- increment operators. *See* postfix increment and decrement operators
- incrementing, pointers, 442
- indexer access
 - described, 179–180
 - expressions as, 148
 - fully qualified names, 382
 - values stored in, 149
- indexer access constructs, 161
- indexer declarations, attributes, 418
- indexers
 - argument lists, 162
 - declaring parameter arrays, 163
 - described, 32, 333–338
 - example of, 29
 - function members as, 157
 - ID string example, 576
 - interfaces, 377
 - member names reserved for, 287
 - members of classes, 19
 - signatures and overloading, 86
 - type parameters, 500
- indirection. *See* pointer indirection
- inferences. *See* type inferences
- inheritance
 - base classes, 79
 - base interfaces, 374
 - constraints and generics, 503
 - defined, 20
 - described, 279–287
 - enumeration members, 79
 - hiding through, 91–93
 - instance constructors, 31
 - interface implementation, 387–389
 - interface members, 376
 - members of a type, 77
 - structs, 358
 - See also* multiple inheritance
- inheritance chains, conditional methods, 425
- inherited members, hiding, 75
- inherited names, hiding, 92–93
- initializers
 - arrays, 370–372
 - See also* variable initializers
- initializing
 - instances of classes, 343
 - new closed constructed class types, 478
- initially assigned variables, 120
- instance constructor invocation
 - construct, 161
- instance constructors
 - argument list, 162
 - declaring parameter arrays, 163
 - defined, 30
 - described, 30, 343–349
 - function members as, 157
 - signatures and overloading, 86
 - structs, 361
 - See also* default constructor
- instance events, 332
- instance fields
 - defined, 20
 - described, 291
 - initialization, 298
- instance function members, defined, 168
- instance members
 - described, 280–281
 - protected access, 84–85
- instance methods
 - delegates, 41
 - described, 24–25, 308
- instance properties
 - defined, 31
 - described, 318
- instance types, generics, 475

- instance variable initializers,
 - defined, 345
- instance variables
 - described, 116
 - initializing default values, 119
- instances
 - string literals, 61
 - See also* attribute instances
- instantiations
 - delegates, 402–403
 - generic types, 460
 - local variables, 529–532
- int entry points, termination status code, 74
- int value type
 - default value, 103
 - defined, 105
- integer addition, 202
- integer comparison operators, 210
- integer division, 199
- integer literals, defined, 55–56
- integer logical operators, 217
- integer multiplication, 198
- integer remainder, 201
- integer subtraction, 205
- integer types, database examples using structs, 362–364
- integral types
 - conversions, 139
 - described, 105–107
 - See also* byte value type; char value type; int value type; long value type; sbyte value type; short value type; uint value type; ulong value type; ushort value type
- interface body, 375
- interface declarations. *See* generic interface declarations
- interface indexers, 377
- interface mapping, 384–387
- interface members, declared accessibility, 81
- interface methods, declarations, 376
- interface modifiers, 373
- interface type variables, possible contents, 11
- interface types
 - defined, 9, 111
 - described, 8
 - user-defined conversions, 342
- interfaces, 373–391
 - base classes and constructed interfaces, 489–490
 - declarations, 373–375
 - described, 37–39
 - fully qualified interface member names, 380
 - implementations, 380–391
 - members of, 79
 - structs, 356
 - syntactic grammar, 611
 - uniqueness, 485
 - See also* base interfaces; events; indexers; methods; properties
- Intermediate Language (IL)
 - instructions, processing, 6
- internal accessibility, defined, 19
- internal modifiers, defined, 80
- interoperation, attributes for, 427
- invariant meaning, in blocks, 172–173
- invocation expressions, 176–178
- invocation lists
 - anonymous method expressions, 532
 - defined, 401
- invocations
 - delegates, 403–405
 - See also* method invocations
- invocations. *See* function member invocation
- is operator
 - described, 215
 - generics, 510
- iteration statements
 - defined, 14

iteration variables ■ logical types

described, 243–248
 iteration variables, `foreach`
 statements, 246
 iterators, 539–551
 enumerable objects, 544
 enumerator objects, 540–543
 implementation example, 547–551
 iterator blocks, 539–540
 overview, 467–470

J

jagged arrays, defined, 36
 jump statements
 defined, 14
 described, 248–255
 See also `break` statement; `continue`
 statement; `goto` statement; `return`
 statement; `throw` statement

K

keywords
 # character in, 53
 defined, 54–55
 lexical grammar, 586
 See also `explicit` keywords; reserved
 words

L

labeled statements, 233–234
 labels, scope of, 88
 left shift, operator for, 208
 length
 of array dimensions, 367
 of array instances, 36
 lexical grammar
 defined, 45
 described, 47
 generics and right shift operator, 522
 lexical structure

analysis of, 47–50
 grammars, 45–47
 preprocessing directives, 61–71
 programs, 45
 tokens, 51–61
 libraries. *See* runtime libraries
 lifetime
 captured outer variables, 529
 local variables, 118, 537
 line directives, 70–71
 line terminators, 48–49
 list accessors, example, 31
 list class, example, 32
 <list> tag, 566
 literals
 creating simple types, 105
 described, 55–61
 in primary expressions, 170
 lexical grammar, 586–589
 local constant declarations
 described, 236
 example, 15
 local constants, scope of, 89
 local variable declaration space, 76
 local variable declarations
 described, 234–235
 example, 14
 local variables
 described, 23, 118–119
 fixed statements, 446
 instantiation, 529–532
 scope of, 88, 466
 lock statement
 defined, 14
 definite assignment rules, 128
 described, 259–260
 example, 17
 generics, 511
 logical negation operators, 195
 logical operators
 described, 13, 216
 precedence of, 150
 logical types. *See* boolean types

long value type
 default value, 103
 defined, 105
 lookups
 rules, revised for generics, 511–522
See also member lookups

M

machine-generated codes, partial types, 471
 Main methods
 as entry points, 74
 in Hello, World program, 4
 signatures, 73
 static constructors, 350
 mapping. *See* interface mapping
 member access, 173–176
 expressions, 498
 generics, 517–519
 member declarations
 declared accessibility, 80
 using alias directives, 266
 member lookups
 described, 156–157
 generics, 514
 type parameters, 503
 members
 access, 79–86
 accessibility domains, 81–84
 arrays, 369
 of constructed types, 490–491
 described, 77–79
 enums, 394–397
 generic classes, 476–477
 kinds of, 18–19
 partial types, 556
 scope of, 88
 structs, 356
See also events; fields; function members; indexers; instance members; methods; operators; properties; protected members; static members
 memory management
 dynamic allocation of memory, 183
 instances of `BitArray` classes, 336
 instances of classes, 18
 life cycle, 95
 pointers created by `fixed` statements, 447
 structs and classes, 34
See also garbage collection
 method body
 described, 316
 local variables, 23–24
 method declaration, attributes, 417
 method groups
 conversions, 466, 534–535
 delegates created from, 189
 expressions as, 147
 method invocation constructs, 158
 method invocations
 described, 176–177
 fully qualified names, 382
 generics, 496, 520–521
 overloaded candidate methods, 379
 method overloading
 described, 28
 Main method, 74
 method string format, 572
 methods, 299–317
 accessibility of, 74
 body, 316
 declaring parameter arrays, 163
 example of, 30
 function members as, 157
 generic, 462–463
 ID string example, 575
 interfaces, 376

modifiers ■ new modifier

- members of classes, 19
 - overriding, 308, 476
 - parameters, 301–307
 - signatures and overloading, 86
 - See also* abstract methods; anonymous methods; application entry point methods; conditional methods; Dispose method; external methods; generic methods; GetEnumerator methods; instance methods; override methods; parameters; return types; sealed methods; signatures; static methods; virtual methods
 - modifiers
 - enums, 394
 - partial types, 554
 - structs, 356
 - See also* interface modifiers
 - modifiers, class, 273–275
 - most derived implementation, defined, 309
 - moveable variables, 436–437
 - MoveNext method, enumerator objects, 541–543
 - multi-dimensional arrays
 - array initializers, 371
 - defined, 367
 - example of, 36
 - traversing, 247
 - multiple inheritance, interfaces, 37, 378
 - multiple interfaces, interface mapping, 386
 - multiplicative operators
 - described, 12, 198–199
 - precedence of, 150
 - multi-use attribute classes, defined, 412
- N**
- name binding, partial types, 557
 - name hiding, 90–93
 - named constants. *See* enum types
 - named parameters, 413
 - names
 - interface members, 382
 - introducing declaration space, 75–76
 - namespace declarations, 264–265
 - namespace member declarations, compilation units, 263
 - namespace members
 - described, 271
 - scope of, 88
 - namespace string format, 572
 - namespaces
 - for conditional compilation symbols, 63
 - declared accessibility, 80
 - defined, 4
 - described, 263–272
 - expressions as, 147
 - fully qualified names, 95
 - members, 78
 - syntactic grammar, 601
 - and type names, 93–95, 512–514
 - nested blocks, in declaration space, 77
 - nested classes, new modifiers, 274
 - nested members, accessibility domains, 81
 - nested scopes
 - explained, 87
 - names introduced by using alias directives, 267
 - nested type declarations, in generic classes, 483–484
 - nested types
 - described, 281–285
 - and hiding, 283, 483
 - partial type declarations, 553
 - nesting
 - commands, 50
 - conditional compilation directives, 67–68
 - generic class declarations, 473
 - hiding through, 90–91
 - .NET Framework class libraries, 5
 - new line escape sequence, Unicode encoding, 59
 - new modifier
 - in declarations, 376
 - delegates, 400
 - described, 280

- hiding inherited names, 92
 - interfaces, 374
 - new operator
 - array instances, 36–37
 - described, 183–189
 - instance constructors, 30
 - purpose, 18
 - struct constructors, 35
 - nonabstract classes, defined, 274
 - non-nested types, defined, 281
 - nonterminal symbols, grammar
 - notation, 45
 - nonvirtual invocations, virtual function
 - member invocations, 275
 - nonvirtual methods
 - compared to virtual methods, 309
 - described, 25
 - implementation, 308
 - nonvolatile fields, optimization
 - techniques, 293
 - normal form, of function members, 166
 - notation
 - grammar, 45
 - unary and binary operators, 152
 - notifications. *See* events
 - null escape sequence, Unicode
 - encoding, 59
 - null types, implicit conversions, 492
 - numeric conversions. *See* explicit
 - numeric conversions
 - numeric promotions, 154
 - numeric types, 8–9
- O**
- object class type, defined, 111
 - object creation expressions
 - described, 183–184
 - and generics, 508
 - object types, predefined
 - conversions, 482
 - object variables, possible
 - contents, 11
 - objects
 - described, 18
 - types as, 10, 101
 - See also* enumerable objects; enumerator objects
 - Obsolete attribute, 426–427
 - open types, 489
 - operands, when all simple type
 - constants, 105
 - operational notation. *See* notation
 - operations
 - rules, 339
 - with simple types, 105
 - operator declarations
 - attributes, 417
 - rules, 339
 - operator invocation constructs, 161
 - operator overloading
 - conditional logical operators, 219
 - described, 151–152
 - use for, 11
 - See also* overloading
 - operators
 - described, 33, 149–156, 338–343
 - enums, 397
 - example of, 30
 - generic classes, 481–483
 - grouping and, 64
 - hiding, 92
 - kinds of, 61
 - lexical grammar, 589
 - members of classes, 19
 - pointer comparison, 444
 - pointers, 443
 - signatures and overloading, 86
 - type parameters, 500
 - See also* address-of operator; binary operators; conversion operators; size of operator; unary operators

optimization ■ precision

optimization

- foreach statements, 247
- volatile and nonvolatile fields, 293
- order of declarations, 76
- order of evaluation, argument lists, 164
- order of members within types, 556
- outer variables
 - defined, 466
 - definite assignment states, 533
 - described, 528–532
- output parameters, 22, 117, 303–304
- overflow checking context
 - checked and unchecked operators, 192
 - integral-type arithmetic operations and conversions, 190
- overflow checking, checked and unchecked operators, 107
- overload resolution
 - with argument constructs, 158, 161
 - described, 165–167
 - numeric promotions, 154–156
 - parameter arrays, 306–307
 - unary operators, 153, 182
- overloadable operators
 - See also* binary operators; conversion operators; unary operators
- overloading
 - in generic classes, 479–480
 - generic methods, 495, 496
 - generic types, 473
 - indexers, 32
 - instance constructors, 30
 - and user-defined operator declarations, 152
 - See also* method overloading; operator overloading
- overridden base method, defined, 311
- override accessors, 325–326, 333
- override methods, 25–28, 311–312
- overriding
 - and generic classes, 480
 - methods, 308, 476

P

- pair-wise declarations, binary operators, 341
- <para> tag, 567
- parameter array methods, type
 - parameters, 480
- parameter arrays
 - declaring, 163
 - described, 22, 304–307
- parameters
 - defined, 21
 - kinds of, 21–23
 - methods, 301–307
 - scope of, 88, 525
 - See also* output parameters; parameter arrays; reference parameters; type parameters; value parameters
- <paramref> tag, 568
- parenthesized expressions, 173
- partial type declarations, 553–556
- partial types
 - described, 553–557
 - overview, 471–472
- passing, arguments, 162
- <permission> tag, 569
- pointer element access, 440–441
- pointer indirection, 439
- pointer member access, 439–440
- pointers
 - C# language compatibility to C and C++, 429
 - conversions, 437–438
 - in expressions, 438–445
 - See also* function pointers
- polymorphic behavior, 279
- positional parameters, 413
- postfix increment and decrement operators, 181–183
- precedence, of operators, 149–151
- precision
 - decimal value type and, 108
 - floating point types, 107–108
 - numeric types, 8–9

- unary and binary operators, 106
- predefined class types, 111
- predefined conversions, operators, 482
- predefined types, accessibility
 - domains, 81
- prefix increment and decrement
 - operators, 195–197
- prefixes, # character in keywords, 53
- preprocessing directives, 61–71, 589–591
- preprocessing expressions, 64
- primary expressions, 170–193
- primary operators
 - described, 12
 - precedence of, 150
- private accessibility, defined, 19
- private constructors, 348
- private modifiers, defined, 80
- program structure
 - compilation units, 45
 - overview, 5–7
 - See also* assemblies; members; namespaces; programs; types
- promotions. *See* numeric promotions
- properties
 - arguments list, 162
 - classes, 317–326
 - compared to indexers, 32, 335
 - Console class, 324
 - described, 31
 - enumerating stack elements, 469
 - example of, 29
 - function members as, 157
 - ID string example, 576
 - interfaces, 377
 - member names reserved
 - for, 286–287
 - members of classes, 19
 - type parameters, 500
 - See also* Current property; instance properties; static properties
- property access
 - expressions as, 147

- fully qualified names, 382
 - values stored in, 148
- property access constructs, 159
- property string format, 572
- protected access, instance
 - members, 84–85
- protected accessibility, defined, 19
- protected internal accessibility, defined, 19
- protected internal modifiers, defined, 80
- protected members, generic class
 - declarations, 478
- protected modifiers, defined, 80
- public accessibility, defined, 19
- public modifiers, defined, 80
- punctuators
 - kinds of, 61
 - lexical grammar, 589

R

- rank
 - arrays, 367, 368
 - defined, 36
- reachability
 - described, 230–231
 - do statements, 244
 - for statements, 246
 - labeled statements, 234
 - throw statements, 254
 - while statements, 244
- reachability of function members,
 - blocks, 231
- read-only fields, 21, 292
- real literals, 57
- rectangular shaped arrays, 186
- reference conversions
 - referential identity of object, 141
 - See also* explicit reference conversions
- reference parameters
 - default values, 119

reference type equality operators ■ <seealso> tag

- described, 22, 117, 302–303
 - runtime processing, 163
 - reference type equality operators
 - described, 212–214
 - generics, 509
 - reference types
 - compared to value types, 101
 - described, 7–8, 109–114
 - See also* array types; class type; delegate type; interface type
 - references
 - pointers, 433
 - variables, 133
 - referential identity of object, reference conversions, 141
 - reflection, attribute information, 43
 - region directives, 69–70
 - reimplementation, interfaces, 389–391
 - relational operators
 - described, 13
 - precedence of, 150
 - remainder operators, 201–202
 - <remarks> tag, 569
 - reserved attributes, 422
 - reserved member names, 286–287
 - reserved words
 - simple types, 104
 - See also* explicit keywords; keywords
 - resolution. *See* binary operator overload
 - resolution; overload resolution
 - resource, defined, 260
 - return statement
 - anonymous method blocks, 528
 - definite assignment rules, 125
 - described, 253–254
 - example, 16
 - methods, 24
 - return types
 - binary operators, 341
 - defined, 21
 - entry points, 74
 - method declarations, 300
 - methods, 316
 - termination status codes, 74
 - return values, output parameters, 304
 - <returns >tag, 570
 - right shift operator, 208, 522
 - runtime
 - binding type parameters, 475
 - unboxing conversions, 114
 - values of `System.Array` types, 368
 - runtime checks, enforcing, 478
 - runtime libraries, 5
 - runtime processing
 - array creation expressions, 185
 - delegate creation expressions, 188
 - function member invocations, 163, 168–169
 - object creation expressions, 184
 - postfix increment or decrement operations, 182
 - runtime retrieval, attribute instances, 421
 - runtime types, defined, 308
- S**
- sbyte value type
 - default value, 103
 - defined, 105
 - scope
 - described, 87–93
 - enum members, 397
 - instance constructor declarations, 345
 - local constants, 236
 - local variables, 235, 466
 - parameters, 466, 525
 - type parameters, 474, 486
 - using directives, 266
 - See also* global scope; nested scopes
 - sealed accessors, 325–326, 333
 - sealed classes, 275
 - sealed methods, 313
 - sealed modifier, overriding property declarations, 325
 - <see> tag, 570
 - <seealso> tag, 571

- selection statements
 - defined, 14
 - described, 237–243
- semantics. *See* value semantics
- set accessor
 - defined, 31
 - described, 319
- shift operators
 - described, 13, 207–208
 - precedence of, 150
- short value type
 - default value, 103
 - defined, 105
- short-circuiting logical operators. *See* conditional logical operators
- side effects, ordering of, 99
- signatures
 - anonymous methods, 525
 - binary operators, 341
 - conversion operators, 342
 - defined, 21
 - disambiguation of interface
 - members, 383
 - indexers, 335, 376
 - Main method, 73
 - and overloading, 86–87
 - reserved member names, 286–287
 - unary operators, 340
- signed integral numeric types,
 - precision, 8
- simple assignment operators, 222–224
- simple names
 - described, 171–173
 - expressions, 498
 - generics, 516–517
 - and identical type names, 175–176
- simple types
 - default value, 103
 - defined, 104
 - described, 7, 101
 - reserved words, 104
 - See also specific value type*
- single-dimensional arrays
 - array initializers, 371
 - defined, 367
 - example of, 36
- single-inheritance interfaces, 378
- single-line comments, 49–50
- single-quote escape sequence, Unicode
 - encoding, 58
- single-use attribute classes,
 - defined, 412
- size of operator, pointers, 444
- skipped sections, 67–68
- source files
 - compilation of, 7
 - compilation units, 263
 - conditionally including or
 - excluding portions of, 66
 - #define and #undefine
 - directives, 63
 - described, 45
 - lexical structure, 47
 - partial types, 471
 - preprocessing directives, 61
 - regions, 69–70
 - skipped sections, 67–68
- source operand, in explicit reference
 - conversions, 141
- space. *See* declaration space; global
 - declaration space; local variable
 - declaration space
- stack allocation, unsafe code, 450–451
- Stack class, generics, 458–459
- standard explicit conversions,
 - defined, 142
- standard implicit conversions
 - list of, 142
 - See also* boxing conversions; identity
 - conversions; implicit constant
 - expression conversions; implicit
 - numeric conversions; implicit
 - reference conversions
- statement lists, in blocks, 232

statements ■ symbols

- statements, 229–262
 - blocks, 232–233
 - definite assignment rules, 122–128
 - described, 14–17
 - empty statements, 233
 - end points and reachability, 230–231
 - generics, 508–511
 - labeled statements, 233–234
 - syntactic grammar, 597–601
 - See also* declaration statements; exception statements; expression statements; iteration statements; jump statements; selection statements; *specific statement*
- states
 - enumerator objects, 541, 543
 - exposing through properties, 323
 - See also* definite assignment states; transient states
- static constructors
 - defined, 30
 - described, 349–352
 - function members as, 157
 - generic classes, 478
- static events, 332
- static field initialization, 297
- static fields
 - defined, 20
 - described, 291
 - generic class declarations, 477
- static function members, defined, 168
- static members, 281
- static methods, 24–25, 308
- static modifiers, example of in Hello, World program, 4
- static properties
 - defined, 31
 - described, 318
- static variables
 - described, 116
 - initializing default values, 119
- string class type, defined, 111
- string concatenation, 204
- string equality operators, 214
- string literals
 - described, 59
 - processing comments, 50
 - See also* verbatim string literals
- strings
 - and pointers, 449
 - processing and Unicode encoding, 8
- struct body, 356
- struct constructors, new operator, 35
- struct members, 78, 80
- struct value types
 - declaring parameterized instance constructors, 103
 - default value, 103
 - defined, 9, 104
 - described, 7
 - See also* simple types
- structs
 - described, 34–35, 355–366
 - implementing interfaces, 38
 - instance variables in, 116
 - interface implementations, 375, 380–391
 - interface mapping, 384
 - syntactic grammar, 609
- subtraction operators, 205–207
- suffixes
 - attributes, 418
 - integer literals, 56
 - optional symbols, 46
 - real literals, 57
- <summary> tag, 571
- superseding, overriding, 308
- switch blocks. *See* blocks
- switch statement
 - definite assignment rules, 123
 - described, 238–243
 - example, 15
 - falling through to the next switch section, 231
- symbolic names, constant values, 289, 292
- symbols
 - grammar notation, 45
 - See also* conditional compilation symbols

- syntactic analysis, defined, 45
 - syntactic grammar, 591–614
 - defined, 45
 - described, 47
 - generics and right shift operator, 522
 - `System.ArithmeticException` class, 409
 - `System.Array` type
 - described, 368
 - instantiation, 369
 - `System.ArrayTypeMismatchException` class, 409
 - `System.DivideByZeroException` class, 409
 - `System.Enum` type, defined, 397
 - `System.exception` class, 408
 - `System.IndexOutOfRangeException` class, 409
 - `System.InvalidCastException` class, 409
 - `System.Nullable` type, 492
 - `System.NullReferenceException` class, 409
 - `System.OutOfMemoryException` class, 409
 - `System.OverflowException` class, 410
 - `System.StackOverflowException` class, 410
 - `System.TypeInitializationException` class, 410
 - `System.ValueType` class, 102
- T**
- tab escape sequences, 59
 - tags, documentation
 - comments, 561–572
 - targets, specification of attributes, 417
 - terminal symbols
 - grammar notation, 45
 - lexical grammar, 47
 - termination. *See* application termination
 - termination status codes, return types, 74
 - terminators. *See* line terminators
 - ternary operator. *See* conditional operator
 - `this`
 - instance methods, 24
 - structs, 360
 - `this` access
 - described, 180–181
 - and iterator blocks, 540
 - nested types, 283–285
 - `throw` statement
 - definite assignment rules, 125
 - described, 254–255
 - example, 17
 - exceptions, 407
 - generics, 510
 - tokens
 - kinds of, 51–61
 - lexical grammar, 585
 - and preprocessing directives, 62
 - top level types, accessibility
 - domains, 81
 - transformation, defined, 45
 - transient states, sharing between enumerators, 470
 - try blocks, return statements, 253
 - try statement
 - and jump statements, 249
 - definite assignment rules, 126–127
 - example, 17
 - generics, 510
 - handling exceptions, 408–409
 - transfer of control in `throw` statements, 254
 - try-catch statement
 - defined, 14
 - definite assignment rules, 125
 - type arguments
 - generics, 488

type casts ■ unary operators

- inference of, 496–498
- type casts
 - example of, 40
 - See also* dynamic type casts
- type declarations
 - defined, 9
 - described, 271–272
 - fully qualified names, 95
 - See also* class types; delegate types; enum types; interface types; nested type declarations; struct value types
- type inferences, defined, 496
- type interfacing, defined, 463
- type names
 - and identical simple names, 175–176
 - and namespaces, 93–95, 512–514
- type parameters
 - base classes, 475
 - boxing, 504–506
 - constraints, 462
 - conversions, 506–507
 - described, 474
 - generics, 459, 476
 - member lookups, 503
 - parameter array methods and generics, 480
 - partial types, 554
 - properties, events, indexers, and operators, 500
- type string format, 572
- type testing operators
 - described, 13
 - precedence of, 150
- typeof operator
 - described, 189–190
 - generics, 509
- types, 101–114
 - accessibility constraints, 85
 - accessibility domains, 81–84
 - application domains, 73
 - arrays, 10, 367–368
 - class constraints, 501
 - constant declarations, 288
 - creating new instances, 183
 - declared accessibility, 80
 - described, 7–10
 - event declarations, 328
 - expressions as, 147
 - indexer declarations, 334
 - of integers, 56
 - objects as, 10, 101
 - syntactic grammar, 592
 - See also* attribute parameter types; classes; closed types; compile-time types; constituent types; constructed types; delegate types; element types; enum types; generic type instantiation; governing types; instance types; interfaces; nested types; open types; partial types; pointers; reference types; runtime types; `System.Nullable` type; value types

U

- uint value type
 - default value, 103
 - defined, 105
- ulong value type
 - default value, 103
 - defined, 106
- unary minus operators, 194
- unary numeric promotions, 154
- unary operators
 - described, 12, 193–197, 340–341
 - ID string example, 576
 - notation, 152
 - overloading, 151, 153, 182
 - pointer indirection, 439
 - precedence of, 150
 - precision, 106
 - See also* bitwise complement operator; logical negation operator; prefix increment and decrement operators; unary minus operator; unary plus operator

- unary plus operators, 193
 - unboxing
 - example, 10
 - struct types, 360
 - unboxing conversions
 - defined, 142
 - described, 112, 114
 - unchecked operators, 190–193
 - unchecked statement
 - defined, 14
 - described, 258
 - example, 17
 - underlying types, enums, 39, 393
 - Unicode encoding
 - character and string processing, 8
 - escape sequences, 51–52, 58–59, 585
 - identifiers, 52
 - lexical grammar, 47
 - unified type system, defined, 3
 - unintended derivations, sealed modifier, 275
 - unmanaged types, defined, 433
 - unsafe code, 429–453
 - dynamic memory
 - allocation, 451–453
 - fixed and moveable variables, 436–437
 - fixed statements, 446–449
 - grammar extensions for, 614–617
 - pointer conversions, 437–438
 - pointers in expressions, 438–445
 - stack allocation, 450–451
 - unsafe contexts, 429–432
 - unsigned integral numeric types, precision, 9
 - user-defined conditional logical operators, 219
 - user-defined conversions
 - conversion operator
 - declarations, 341
 - described, 143–146
 - user-defined explicit conversions
 - defined, 142
 - evaluating, 145
 - user-defined implicit conversions
 - defined, 137
 - evaluating, 144
 - user-defined operator declarations, and overloading, 152
 - user-defined operators
 - argument list, 162
 - function members as, 157
 - ushort value type
 - default value, 103
 - defined, 105
 - using alias directives
 - compared to using namespace directives, 269
 - described, 266–268
 - using directive
 - described, 265–271
 - global attributes and namespace member declarations, 263
 - in Hello, World program, 4
 - using namespace directives, 269–271
 - using statement, 511
 - defined, 14
 - definite assignment rules, 127
 - described, 260–262
 - example, 17
- V**
- value parameters
 - default values, 119
 - defined, 22
 - described, 117, 302
 - runtime processing, 163
 - value semantics, structs, 357
 - <value> tag, 571
 - value type variables, possible contents, 11
 - value types
 - array covariance, 370
 - described, 7, 101–109

values ■ yield types

- See also* delegate types; enum types; enumeration types; simple types; *specific value type*
 - values
 - array types, 368
 - enums, 397
 - of expressions, 148
 - expressions as, 147
 - fields, 291
 - local constants, 236
 - of local variables, 235
 - of variables, 115
 - variable initializers, 295–298
 - variables, 115–133
 - categories, 115–119
 - default values, 119–132
 - definite assignment states, 533
 - described, 7–10
 - expressions as, 147
 - kinds of, 10
 - references, 133
 - syntactic grammar, 593
 - values stored in, 148
 - See also* elements; fields; instance variables; local variables; outer variables; output parameters; parameters; reference parameters; static variables; value parameters
 - verbatim identifier, defined, 53
 - verbatim string literals, 59–61
 - versioning
 - constants and static read-only fields, 293
 - described, 3
 - vertical tab escape sequence, Unicode encoding, 59
 - virtual accessors, 325–326, 333
 - virtual generic methods, 495
 - virtual methods, 308–311
 - described, 25–28
 - interface mappings, 388
 - void return type, termination status code, 74
 - volatile fields, 293–295
-
- ## W
- warnings, hiding inherited names, 92
 - while statement
 - definite assignment rules, 124
 - described, 243–244
 - example, 15
 - white space
 - defined, 49
 - lexical grammar, 584
 - Win 32 components, attributes for interoperation, 427
 - words. *See* explicit keywords; keywords; reserved words
-
- ## X
- XML
 - documentation tags, 561
 - example, 580–582
-
- ## Y
- yield return statement, definite assignment, 547
 - yield statement, iterators, 467, 539, 545–547
 - yield types, iterator blocks, 540



















