

Symbols

- :: namespace alias
 - qualifier 191
- ? modifier 113
- ?? 121
 - See also* null coalescing operator
- .aspx 195
- .Contracts, suffix for contract reference assemblies 469
- .ini, configuration files 414
- .NET 21
 - 2.0 52
 - extension methods 262
 - 3.5
 - attribute supporting
 - extension methods 262
 - extension methods in the framework 265
 - introduction of Func and Action 50
 - SP1 293
 - 4, observable interfaces 351
 - Passport 21
- ‘ generics 89
- * transparent identifiers 300
- /l 385, 409
- /link 385
- /r 385
- /reference 385
- & operator 117
- #pragma 194
 - checksum 195
 - warning 194
- == operator

- overloading and
 - generics 77
- reference type
 - constraints 70
- unconstrained type
 - parameters 77
- value type constraints 70
- => 230
- | operator 117

Numerics

- 64-bit processors 525

A

- aborting threads 398
- about:blank, comparison with null references 104
- abstract base class compared with duck typing 418
- abstract methods 467
- abstract modifier, static
 - classes 188
- abstraction 62
- abuse
 - class inheritance 467
 - general-purpose extension methods 455
- academic license, Code Contracts 472
- access from nested types 159
- access modifiers
 - automatic properties 205
 - defaults 189
 - partial types 182
 - properties 189

- accessors 205
- Action 139
- Action delegate types 229
- action delegates 133
- Active Record 3
- ActiveSheet 409
- actual type
 - dynamic binding 430
- adapter pattern 94, 96
- Add 509
 - collection initializers 289
 - method 216–217
 - method on XContainer 340
- add/remove blocks,
 - events 399
- AddFirst 513
- addition, implicit conversion
 - of operands 417
- AddLast 513
- AddRange 390
- ADO.NET 106, 524
- Aggregate 496
- aggregation 273, 336
 - operators 495
- ahead-of-time compilation 22
- AJAX 524
- Albahari, Joe 24
- aliases 128
 - alternative to implicit typing 209
 - for namespaces and types 190
- aliasing, out parameters 395
- All 505
- AllDefects (and related properties) 285
- alpha geeks 492

- Amazon EC2 492
- ambiguity
 - dynamic conversions 427
 - dynamic typing 420
 - overloading 379
 - terminology 205
- amortized complexity 510, 515
- analysis
 - argument types 367
 - query translations 329
- Ancestors 342
- AncestorsAndSelf 342
- Android 492
- angle brackets 62, 88
 - unspeakable names 140
- annotated specification 25
- Annotations 342
- anonymous functions 208, 229, 429
 - better conversions 254
 - inferred return types 247
 - overloading 253
- anonymous methods 11–12, 50, 138, 172
 - ambiguity 144
 - anonymous types 224
 - as typeless expressions 219
 - capturing variables 144
 - See also* captured variables
 - compared with lambda expressions 229
 - dynamic code 433
 - ignoring parameters 143
 - lack of contravariance 140
 - prohibition on iterator blocks 161
 - readability 227
 - returning values 141
 - type inference 246
- anonymous object
 - initializers 51, 221
- anonymous types 51, 220
 - conversion to XElement
 - and XAttribute 341
 - inner join keys 301
 - keys for grouping 311
 - projections 270
 - ToString 271
 - used for transparent identifiers 299
- Any 505
- Apache 2.0 license 527
- API design 484, 514
- API querying 243
- APIs 453
 - designing to work with LINQ 345
 - parameter names 379
- applicable methods
 - overload resolution 378
 - overloading 253, 379
- application domains 81
- Application, Word 381
- Arbiter 177
- architecture 279
- AreaComparer 391
- ArgumentException 397, 469
- ArgumentNullException 265, 277, 455, 457, 483
- ArgumentOutOfRangeException 499
- arguments 8, 20
 - comparing type and method arguments 63
 - conversions to parameter types 252
 - dynamic values 426
 - evaluation of partial methods 186
 - evaluation order 374
 - terminology 366
 - type inference 74
 - validation 173, 276
 - iterator blocks 164, 174
 - validation in LINQ operators 357
- arithmetic obligations 477
- arithmetic, generics 245
- array bounds obligations 476
- ArrayList 5, 40, 53, 84, 292
- arrays 22, 40, 43, 197, 280, 511
 - CLR terminology 512
 - covariance 40, 92
 - implicit typing 219
 - populating with object initializers 214
- ArrayTypeMismatchException 40, 511
- as operator 120
 - breaking changes 396
 - dynamic types 427
- ASCII 39
- AsEnumerable 498
- AsOrdered 349
- ASP.NET 23, 195, 399, 523–524
- AsParallel 347
- AsQueryable 498
- assemblies
 - anonymous types 221
- COM 385
 - contract reference assemblies 469
 - extern aliases 193
 - references 23
 - rewriting, Code Contracts 464
 - signing 200
- AssemblyInfo.cs 200
- Assert method, Code Contracts 458, 461
- assertions 458, 486
 - Code Contracts 461
 - error windows 470
 - failures 485, 487
- assignment
 - expression trees 239
 - field-like events 399
 - lambda expressions 234
 - unaffected by type inference 74
- associations, LINQ to SQL 323
- Assume method, Code Contracts 458, 461
- assumptions 149, 486
 - Code Contracts 461
- asterisk, transparent identifiers 300
- AsUnordered 349
- asynchronous
 - computation 491
 - delegate invocation 31
 - I/O 356
 - operations 76
 - service access 175
- atomicity, locking 398
- Attributes 342
- attributes 437
 - extension methods 262
 - LINQ to XML 18, 338
- automatic properties 7, 204–205
 - builder pattern implementation 219
 - encouraging mutability 376
 - lambda expression examples 233
- automatic variance 396
- Average 495
- axis methods, LINQ to XML 342
- Axum 492

B

-
- back tick, generics 89
 - back-end technologies 491
 - BackgroundWorker 35, 523
 - backing fields
 - automatically implemented properties 7
 - field-like events 399
 - trivial properties 205
 - backward compatibility 491
 - balance, using implicitly typed local variables 209
 - base types
 - partial types 181
 - static classes 188
 - base, anonymous methods 140
 - baselines 478
 - basic multilingual plane 25
 - BCL 52
 - BeginInvoke, delegate
 - method 31
 - BeginXXX 76
 - behavior 448
 - adding with inheritance 256
 - changes to optional parameter defaults 371
 - DLR rules 424
 - encapsulated in delegates 28
 - behavioral patterns 156
 - benchmarks 121, 347, 488
 - best accessible type 430
 - best practices 469
 - better conversions 253
 - extension methods 263
 - BigInteger 525
 - binary data 104
 - binary operators 117
 - binary representations 395
 - binary rewriter 456, 458, 463–464, 470
 - BinaryExpression 237
 - BinarySearch 511–512
 - binders 424–425
 - parameters to TryXXX methods 444
 - BindGetMember 447
 - binding 52, 403, 406, 513
 - explicit interface implementation 419
 - expression tree parameters 241
 - BindingList 513
 - black magic, COM 380
 - Bloch, Joshua 80
 - block, as a lambda expression
 - body 231
 - blocking, parallelism 176
 - BlockingCollection 518
 - blocks, expression trees in .NET 4 239
 - blogs 492
 - blood spatter pattern 375
 - blueprints 62
 - body of a lambda
 - expression 231
 - boilerplate code 170
 - bool?, behavior of & and | operators 119
 - Boolean, flag indicating nullity 106
 - bottlenecks 46, 488
 - bounds, type inference 249
 - boxing 47, 59, 84, 106, 510
 - avoiding in C# 1 85
 - Hashtable 61
 - Nullable 110
 - using type constraints to avoid 128
 - boxing conversions 71
 - invalid in generic variance 395
 - braces 141, 455
 - removing in lambda expressions 231
 - breaking changes 379, 491
 - capturing loop variables 151
 - caused by generic variance 396
 - delegate variance 137
 - field-like events 399
 - brevity 13
 - bridging
 - fluent interfaces 275
 - static and dynamic code 448
 - browser 43
 - buffering 282, 495
 - custom LINQ operators 358
 - group joins 306
 - grouping 312
 - inner joins 302
 - Parallel LINQ 350
 - buffering data 267
 - bugs 42, 370, 407, 437, 483
 - mutability 108
 - side effects and argument ordering 375
 - build configurations 487
 - builder pattern 218
 - building blocks, concurrency with Parallel LINQ 346
 - Burrows, Chris 399
 - business logic 174
 - business rules 413
 - Button, unqualified name ambiguity 190
 - by, contextual keyword 311
 - byte arrays 258
 - byte, non-nullability 104
 - bytecode 100
-
- C**
-
- C 28, 38
 - C^o 455
 - C# 1 27
 - pain of sorting and filtering 234
 - C# team 25, 396
 - C++ 38, 78, 91
 - compilation model 99
 - const 488
 - templates 99
 - C++0x 99
 - caching 441
 - DLR 422
 - expression trees 237, 244
 - lambda expressions 235
 - multi-level 425
 - Call 242, 450
 - call site validation 488
 - call sites 425, 429
 - DLR 423
 - precondition checking 473
 - callbacks 467
 - iterator execution flow 177
 - callers, affected by variance changes 396
 - caller-specified variance 397
 - CallSite 423
 - Call-site Requires
 - Checking 469
 - cancellation tokens 349, 519
 - Capacity 510
 - captured variables 145, 229, 288
 - behavior 146
 - generating extra classes 149, 152
 - guidelines 153
 - lifetime 148
 - motivation 147
 - See also* anonymous methods
 - Cartesian product 308
 - CAS 526

- case sensitivity 445
 - LINQ queries 318
- Cast 95, 391, 497
- casting 10, 39, 209
 - anonymous methods 144
 - COM 402
 - dynamic types 413, 417, 433
 - generics 58, 96, 393
 - is and as operators 121
 - Java generics 100
 - nullable types 113, 119
 - Office APIs 404
 - overload resolution 379
 - reduced by PIA linking 386
 - reference conversions 389
 - to resolve overload ambiguity 379
 - with Hashtable 61
- catching contract violations 471
- Catchphrase 274
- cccheck 456, 472
- ccdocgen 456
- CCR. *See* Concurrency and Coordination Runtime
- ccrefgen 456, 469
- ccrewite 456
- CDATA, LINQ to XML 338
- Cells 409
- ceremony 3, 5, 216
- chaining 272–273
 - extension methods 257, 268
 - iterators 267
- change tracking, LINQ to SQL 325
- Channel 9 351
- Chars 383
- checked context 478
- checksum pragmas 195
- CHESS 492
- child content, LINQ to XML 338
- child elements 437
- circular buffers 158, 517
- Civilization IV 411
- clarity 451
- class
 - keyword 43
 - reference type constraints 69
- class hierarchy 58
- class libraries 487
- classes, lack of generic variance 394
- Clear 509
- ClearItems 513
- CLI 22, 71, 112, 383
- Click event 132
- ClickOnce 523
- Client Profile 524
- cloning 223
 - LINQ to XML content 340
- closed constructed types, specifying in reflection 90
- closed types 63
 - generics 89
 - static fields 82
- closures 144–145, 155, 170
- cloud computing 493
- CLR 20, 22, 50, 84, 196, 203, 511, 525
 - generic variance 52, 394
 - handling thread aborts 398
 - non-involvement in lifted operators 120
 - parameterized properties 383
 - size of references 45
 - support for nullable value types 110
 - support for ref and out parameters 395
- CLR types, conversions to/from dynamic 427
- CLS compliance 455
- clutter 215
 - code with explicit typing 209
- CoClass 386
- Code Access Security 526
- code as data 236
- Code Contracts 453, 472, 524
 - assertions 461
 - assumptions 461
 - automatic documentation 480
 - baselines 478
 - contract inheritance 466
 - contract reference assemblies 469
 - implicit obligations 475
 - invariants 459
 - legacy contracts 462
 - postconditions 458
 - preconditions 456
 - static checker 472
- code generators 106, 179–180, 183, 192
- code smells 129
- CodeChecksumPragma 195
- CodeDOM 236, 239, 242, 403
- CodeDomProvider 421
- coding standards 178
- cold observables 353
- collation, results in Parallel LINQ 346
- collection initializers 7, 211, 215, 289, 401
 - encouraging mutability 376
 - lambda expression examples 233
 - method 216–217
 - requirements 216
 - within object initializers 217
- Collection<T> 513
- CollectionBase 41
- collections 6, 58, 68, 280
 - populating with collection initializers 215
 - sorting with custom comparisons 142
- colon, named arguments 372
- columns, selecting in SQL queries 225
- COM 19, 207, 380, 402
 - handling in C# binder 424
 - parameterized properties 383
 - Primary Interop Assemblies 385
- Combine 33, 131
 - generic variance 396
- combining delegates 33
- combining type constraints 73
- ComImport 386
- command line 193
- command line options, linking PIAs 385
- commas
 - separating type parameters 62
 - used to indicate number of type parameters 88
- comments 42, 372
- Common Language Infrastructure. *See* CLI
- Common Language Runtime. *See* CLR
- communication 489
 - Code Contracts 453
 - named arguments 373
- community 54, 180, 420
 - debate over extension methods 275
 - influence over CLR boxing behavior 110
 - scientific developers 97
 - uptake of LINQ 361

- Compact Framework 526
- compact profile 22
- compactness 140
- Compare 126
- CompareExchange 399
- Comparer 78, 394
- Comparer.Default 112, 358
- CompareTo 76, 235
- Comparison 142, 233
 - purity 463
- comparisons 126
 - for sorting 271
 - LINQ query operators 318
 - nullable types 14
 - reference type constrained values 70
- compatibility
 - delegate types and methods 29, 137
 - generic variance 93
 - method group
 - conversions 133
 - method signatures and delegates 49
- compilation errors 105
 - invalid casts 39
- compilation, expression
 - trees 238, 244
- Compile 238, 498
- Compile method,
 - LambdaExpression 238
- compile time 38
 - binding 403
 - contract checking 453
- compiler 22, 167, 392
 - binding with dynamic types in C# 4 52
 - code generated using nullable types 115
 - combining source for partial types 181
 - command line for linking PIAs 385
 - creating strongly typed expression trees 239
 - detecting key selector reversal 303
 - embedded for dynamic typing 408
 - errors on dynamic code 432
 - extern aliases 193
 - ignorance of Enumerable and Queryable 334
 - implementation of anonymous methods 149
 - implementation of fixed-size buffers 196
 - importance in C# 3
 - features 204
 - inability to verify
 - comments 372
 - pipeline 491
 - providing more information via casts 58
 - removing unimplemented partial methods 185
 - role in delegate features 49
 - shortcut for obtaining a MethodInfo 242
 - support for object initializers 213
 - translations
 - demonstration with a dummy provider 288
 - query expressions 280, 287
 - verifying variance 396
 - warnings 138, 140
- compiler as a service 491
- CompilerGeneratedAttribute 386
- compile-time checking
 - generics and static typing 59
 - limitations 244
 - LINQ queries 243
- compile-time duck typing 289
- compile-time efficiency 87
- compile-time errors 431
- compile-time type safety 40
- compile-time types 5, 37, 208
 - dynamic binding 431
 - implicitly typed arrays 219
- Complex 97, 525
- complexity 154, 465, 491
 - language design choices 92, 220
 - new C# features 284
 - reducing with LINQ to Rx 356
- ComplexNumber 97
- compliance, C# 4
 - specification 380
- Component Object Model. *See* COM
- component vendors 487
- composition 16, 244, 272
- compound contracts 462
- compression 523
- computer science 145, 491
 - defining pass by reference 46
 - formal specification and verification 454
- Concat 390, 496
 - (strings) comparison with Delegate.Combine 33
- concatenation 44
- concatenation operator 496
- concepts, C++ 99
- concurrency 175
 - collections 508, 519
 - programming 346
- Concurrency and Coordination Runtime 175
- ConcurrentDictionary 514, 519
- conditional code 263
- conditional logical operators 119
- conditional operator 113, 122, 127
- confidence 403
- configuration 411, 413
 - active via the DLR 414
- connection management, LINQ to SQL 325
- consistency 7, 141, 182, 396, 484
 - LINQ 16, 265, 284
 - naming conventions 318
- console 286
- const 369
 - C++ 488
- Constant 332
- constant expressions, C++ template arguments 99
- Constant, method of Expression 332
- constants 15
 - default parameter values 368
 - unchanging collections 218
- constrained type
 - parameters 70
- constraints
 - dynamic 434
 - generic type constraints 69
- constructed types 62, 88
- construction, preconditions for immutable types 484
- constructor type constraints 70
- constructors 211, 441
 - anonymous types 223
 - default provided by compiler 187
 - discrepancy between C# and CLI 71
 - dynamic code 434

- constructors (*continued*)
 - dynamic invocation 428
 - generic type constraints 70
 - generic types 64
 - immutable types 376
 - implicitly typed local variables 210
 - multiple parameters causing confusion 214
 - of Nullable 107
 - shorthand with object initializers 213
 - using arguments for initialization 211
 - utility classes 187
 - XElement 340
 - Contains 505, 509
 - context
 - in anonymous methods 145
 - propagated with range variables 290
 - contextual keywords 16, 290
 - dynamic 406
 - partial 181
 - type constraints 69
 - continuation-passing style 176
 - continuations
 - multiple 316
 - query continuations 314
 - ContinueWith 175
 - continuous build 479
 - Contract class
 - Code Contracts 456
 - last reference marking end of contracts 458
 - contract classes, abstract classes and interfaces 467
 - contract failure handlers 457
 - contract inheritance 466
 - contract reference
 - assemblies 456, 463, 469, 486
 - contract section 464
 - ContractClass 467
 - ContractClassFor 467
 - ContractException 457, 465, 470, 482, 485
 - catching in unit tests 485
 - ContractFailed 471, 485
 - ContractFailedEventArgs 471
 - ContractInvariantMethod-Attribute 460
 - ContractPublicPropertyName-Attribute 457
 - contracts runtime 472
 - CONTRACTS_FULL 464
 - CONTRACTS_
 - PRECONDITIONS 464
 - _ContractsRuntime 465
 - ContractVerificationAttribute 479
 - contradictory behavior, operators on nullable types 120
 - contravariance 52, 95, 388
 - anonymous methods 140
 - delegates 50, 131, 134
 - See also* delegates, contravariance
 - IObserver 351
 - nesting 394
 - parameters 41
 - See also* generic variance
 - contravariance, C# 2 134
 - controversy, extension methods 273
 - convenience 391
 - conventions
 - class contract names 468
 - event handling 136
 - lambda expression parameters 317
 - namespaces and extension methods 276
 - type parameter names 65
 - unspeakable names 140
 - conversion operators 496
 - conversion type constraints 71
 - restrictions 72
 - working around invariance 96
 - conversions
 - "better than" other conversions 253
 - argument types to parameter types 379
 - generic type constraints 71
 - generic variance support in the CLR 394
 - GetVariable 413
 - involving anonymous functions 253
 - LINQ to XML 343
 - method groups and overloading 246
 - Nullable<T> *See* Nullable
 - projections in LINQ 282
 - Convert 450
 - ConvertAll 65, 67, 94, 224, 270, 511–512
 - Converter<TInput, TOutput> variance 393
 - coordination 175–176
 - copying collections, work-around for limitations in generics 94
 - copying values, boxing 47
 - copying, value type and reference type behavior 43
 - copyright 43
 - CopyTo 258, 509
 - CoreCLR 527
 - corner cases 21, 170, 469, 485
 - coroutines 178
 - correctness 59
 - Code Contracts 453
 - contracts 473
 - Count 307, 358, 495, 509
 - property of ICollection and ICollection<T> 418
 - count with SQL for joins 328
 - covariance 52, 92, 388
 - arrays 511
 - delegates 50, 131, 134
 - IObservable 351
 - of arrays 40
 - return types 41
 - See also* generic variance; delegates, covariance
 - CPU, cost of JIT compilation 83
 - CPU-bounded tasks 350
 - CreateInstance 512
 - CreateQuery, method of IQueryable 330
 - cross joins 308
 - CSharpArgumentInfo 430
 - CSharpCodeProvider 403
 - CSS 491
 - cultural issues 275
 - culture 321
 - Current 85, 157, 162, 164, 168, 282, 352
 - cursor iterators 157
 - custom comparisons 358
 - LINQ query operators 318
 - custom iteration types 85
 - custom rewriter methods 485
 - Code Contracts 472
 - cyclic relationships 217
- ## D
-
- data binding 286, 415, 513, 523
 - Data Connections, Visual Studio 323

- data contexts, LINQ to SQL 325
- data extraction, XML 18
- data grids 286
- data integrity 296
- data models, LINQ 280
- data pipelines 156
 - lambda expressions 230
- data processing, pipelines of
 - extension methods 268
- data sources
 - consistency 284
 - LINQ 16, 228
- data structures 491
- data transfer types 45
- databases 17–18, 175, 225, 228, 272, 320, 322
 - joins 301
 - nullable fields 14, 53, 105
 - rules engines 411
- DataContext 325
- DataReader 267
- DataSet 267
- dates, LINQ to XML
 - content 340
- DateTime 45, 76, 108, 343
 - non-nullability 104
- DateTime.MinValue 76
 - magic value pattern 105
- DateTimeOffset 522, 524
- DateTimeRange 306
- DBNull 106
- deadlocks 398
- debug builds, assertions 455
- Debug.Assert 455, 461, 483
- debugger 195, 446, 485
- debugging 213, 411, 470
- decimal 14
- declarations
 - implicitly typed local variables 208
 - out parameters 124
 - partial methods 186
 - using dynamic 434
 - var and dynamic 408
- declarative style 320
 - contracts 455
 - LINQ to XML construction
 - pattern 340
 - programming 16
- decompilation 428
- deep zoom 527
- Default (EqualityComparer<T> property) 112, 358
- default constructors 187
 - Nullable<T> 107
- default members 383
- default operator 417
 - contract classes 468
 - optional parameters 368
- default properties 383
- default value expressions 75
- default values 14–15, 368
 - fields in structs 206
 - optional parameters 366
 - restrictions 368
 - specifying for
 - parameters 367
- default(T) 75
- DefaultIfEmpty 500
- defaulting
 - null coalescing operator 123
- defaults
 - private access modifiers 189
- defect tracking 273, 285
- defensive code 488
- deferred execution 282, 495
 - custom LINQ operators 358
- definite assignment 60, 395
 - output parameters 76
- degenerate query
 - expressions 295
- degree of parallelism 349
- Delegate 29
- delegate
 - ambiguity of term 29
 - keyword 11, 43, 140, 143
- delegate creation
 - expressions 133
 - variance 137
- delegate instances 28
 - caching 235
 - LINQ 288
 - referring to captured variables 148
- delegate parameters 143
- delegate types 28, 43
 - Action<...> 229
 - conversions involving anonymous functions 254
 - expression trees 238
 - Func<...> 229
- delegates 10, 22, 28, 334
 - action 30
 - alternative approach to iteration 170
 - anonymous methods. *See* anonymous methods
 - anonymous methods
 - asynchronous invocation 31
 - C# 2 130
 - combining 33
 - compatibility 137
 - in C# 1 30
 - in C# 2 49
 - compiling from expression trees 238
 - constructing with lambda expressions 50
 - contravariance, C# 2 131, 134–135
 - covariance, C# 2 131, 134, 136
 - exceptions 34
 - ExpandableObject 436
 - field-like events 34
 - garbage collection 30
 - generic 66
 - generic variance 387, 392
 - immutability 33
 - in process data
 - processing 228
 - increased use in .NET 2.0 154
 - invocation 28, 31, 131
 - invocation list 33
 - invoking asynchronously 31
 - LINQ 282
 - meaning of combining
 - null 33
 - method group
 - conversions 67, 133
 - See also* method group conversions
 - motivation 32
 - option for iteration
 - pattern 170
 - order of execution 34
 - removing 33
 - summary of C# 1 35
 - target 30, 32
- delegates, contravariance
 - C# 2 131, 135
- delegates, covariance
 - C# 2 131, 134, 136
- Dependency Injection 414
- deployment 20
- deprecation 277
- Dequeue 517
- derivation, reference types and
 - value types 45
- derived data, anonymous types 225
- DescendantNodes 343
- DescendantNodesAndSelf 343

Descendants 342
 DescendantsAndSelf 342
 descending, contextual
 keyword 296
 deserialization 415
 design 108, 161
 class inheritance 467
 LINQ 173
 LINQ to XML 345
 named indexers 384
 static classes 188
 variance 400
 Design by Contract 483
 design patterns 156
 designers
 code generation 180
 LINQ to SQL 323
 desktop framework 521
 development platforms 23
 DevLabs 350, 456
 diagrams, LINQ to SQL
 models 323
 dictionaries, using nullable
 keys 111
 Dictionary 60
 Dictionary<TKey, TValue> 514
 collection initializers 216
 DictionaryEntry 61
 dir, Python 445
 disabling warnings 195
 dispatch, single and
 multiple 419
 Dispose 22, 87, 167, 171
 invariants 460
 Distinct 506
 distribution, tasks in Parallel
 LINQ 346
 division by zero 477
 DLR 19, 53, 402, 421
 interoperability 410
 using expression trees 244
 document model, LINQ to
 XML 339
 Document Object Model 436
 documentation 445
 Code Contracts 480
 collection mutability 514
 custom LINQ operators 358
 for weakly typed collection
 usage 40
 informal contracts 454
 joins 301
 null first parameters for
 extension methods 277
 type safety 59

DocumentElement 338
 documents, LINQ to XML 338
 DOM 338, 436
 domain specific languages 274,
 405
 dot notation 317, 326, 341
 advantages over query
 expressions 318
 double buffering 523
 double, special values 106
 doubly linked list 512
 drill down 446
 DSLs. *See* domain specific
 languages
 duality, LINQ to Rx 351
 duck typing 3, 210, 289, 404,
 418
 Duffy, Joe 346
 duplicate keys 497
 duplicates 506
 duplication 126
 overloading 367
 Dyer, Wes 274
 dynamic 20, 401, 405
 COM variants 409
 compiler behavior 426
 contextual keyword 52
 dynamic behavior 435
 dynamic calls,
 DynamicObject 440
 dynamic code restrictions 432
 dynamic contextual
 keyword 20
 dynamic expressions,
 conversions 427
 Dynamic Language Runtime.
 See DLR
 dynamic languages 3, 19, 491
 interoperating with C# 4 410
 dynamic methods 98
 dynamic typing 3, 20, 37, 52,
 207, 244
 alternative to reflection 91
 benefits 404
 expression tree support 239
 gotchas 419
 responding dynamically 435
 working around limitations
 in generics 98
 Dynamic View, Visual Studio
 debugger 446
 DynamicAttribute 405, 435
 DynamicMetaObject 445–446
 DynamicMethod 421, 525
 DynamicObject 440

E

eager evaluation 267
 early out, iterator blocks 165
 ECMA 25, 81
 e-commerce 4
 Effective Java 80
 efficiency 282
 generic variance 389
 standard query
 operators 357
 Eiffel 454
 Eini, Oren 275
 elegance 42, 357, 420
 element operators 498
 ElementAt 499
 Elements 343
 elements 437
 LINQ to XML 338
 ElementsAfterSelf 343
 ElementsBeforeSelf 343
 ElementType, property of
 IQueryable 330
 ellipsis in code snippets 23
 embarrassingly parallel
 tasks 346
 Embed Interop Types 409
 embedded collections 217
 embedded objects 214
 with collection
 initializers 217
 embedded sequences, group
 joins 306
 embedding
 languages within C# 411
 Primary Interop
 Assemblies 385
 emphasis 125
 "what" vs "how" 210
 Empty 44, 500
 empty arrays 501
 empty sequences, group joins
 matching no elements 306
 encapsulation 4, 125, 171, 189
 delegates 28
 events 35
 enclosing types 189
 encodings 171
 reading files 366
 EndContractBlock 463
 EndInvoke, delegate
 method 31
 EndXXX 76
 enforcement
 preconditions 484

- Enqueue 517
- Ensures 458, 486
- EnsuresOnThrow 486
- Enter 398
- entities 183, 323
 - change tracking 325
 - See also* partial types
- Entity Framework 322, 524
- entry point, Code Contracts 477
- enum 43
 - restriction on type constraints 72
- Enumerable 265, 280, 334
- enumeration 43
 - mapping in LINQ to SQL 323
 - terminology clash 157
- enumerators 22
- environment 271
 - adapting via extension methods 274
 - closures 145, 155
- Environment.GetCommandLin
 - eArgs 209
- equality 78
 - joins 301
 - Nullable 111
- equality operators 117, 500
- EqualityComparer<T> 78
- Equals 48, 79, 126
 - anonymous types 223
 - Nullable
 - implementation 111
- equals, contextual keyword 302
- Equals() 440
- equijoins 308
- erasure, transparent
 - identifiers 300
- errata 25
- errors
 - checking 258
 - dynamic code 431
 - messages 392
 - pragma directives 194
- escaping string literals 412
- eval 403
- evaluation order
 - named arguments 374
 - null coalescing operator 121
- Evans, Eric 274
- event handlers 30, 50, 131, 352
 - conventions 136
 - lambda expression
 - examples 233
 - lambda expressions 235
- EventArgs 135–136, 236
- EventHandler 131, 135
- events 34, 185, 230, 352
 - avoiding nullity checks 143
 - changes to field-like events in C# 4 399
 - subscription 34
 - subscriptions 50, 136
- everything Code Contracts
 - retention option 486
- evil uses of dynamic typing 420
- evolution 8, 12–13, 15
- Excel 19, 409
- Except 506
- Exception 485
 - catching
 - ContractException 470
- exceptions 105, 231
 - documentation 454
 - during delegate invocation 34
 - dynamic responses 448
 - handling postconditions 459
 - preconditions 456
 - specified in
 - preconditions 470
 - thrown by Cast 293
 - thrown by invalid casts 39
 - thrown for invalid conversions 45
 - TryXXX pattern 76
 - while iterating over a sequence 352
- ExceptWith 516
- exclusive OR operator 119
- executable caches 425
- Execute 412
 - aggregation operators 336
 - method of
 - IQueryProvider 330
- ExecuteFile 412
- execution model, abstracted by
 - expression trees 243
- execution modes, Parallel LINQ 350
- execution order, clarity with
 - extension methods 268
- execution patterns
 - finally blocks 165
 - sequential appearance of
 - iterator blocks 161
- execution plans 327
- execution time
 - array access checks 40
 - contract checking 454
- execution-time types, virtual
 - dispatch 403
- Exists 461
- exit points 142
 - invariant and postcondition checking 473
- ExpandableObject 435, 514
- experimental style 405
- experimentation 60
- experiments with code from
 - the book 23
- explicit conversions
 - LINQ to XML 343
 - method groups 134
 - Nullable<T> 108
 - See also* Nullable<T>, explicit conversions
- explicit interface
 - implementation 41, 86
 - dynamic typing 419
- explicit typing 38, 211
- explicitly typed parameter lists,
 - lambda expressions 230
- explicitly typed range
 - variables 292
- Expression class 237–238, 288
- expression evaluator 413
 - Watch and Immediate windows 403
- expression trees 98, 228, 236, 334
 - behavior in DLR rules 425
 - compiling 238
 - conversion from lambda expressions 239
 - dynamic typing 405, 429, 447
 - enhancements in .NET 4 239, 421
 - LINQ 288
 - restrictions on lambda
 - expression conversions 240
 - used outside LINQ 244
 - visualizer in Visual Studio 2010 242
- Expression, property of
 - IQueryable 330
- expressions
 - in object initializers 213
 - static typing 37, 403
 - value as a reference type or value type 43
- ExpressionType 237
- expressive code 14, 91, 272, 489

extended types
 of an extension method 260
 options around
 namespaces 276
 Extensible Application Markup
 Language 183
 extension methods 11, 51–52,
 134, 257, 455
 calling 261
 compile-time discovery 262
 creating TimeSpan
 values 418
 declaring 259
 dynamic code 433
 extending LINQ to
 Objects 357
 guidance 276
 LINQ to Rx 353
 LINQ to XML 344
 restrictions 275
 used by query
 expressions 280
 ExtensionAttribute 262
 Extensions class
 LINQ to XML 344
 extern aliases 192
 extra classes, generated to hold
 captured variables 149

F

F# 473, 491
 F# Interactive 403
 factory methods 333, 376, 441
 expression trees 237
 factory pattern 71, 172, 434
 FakeQuery 332
 FakeQueryProvider 332
 false operator 117
 field-like events 34–35
 changes in C# 4 399
 fields
 automatically implemented
 properties 7
 backing trivial
 properties 205
 dynamic 408
 field-like events 399
 in Nullable<T> 107
 lack of implicit typing 208
 object initializers 213
 FIFO 517
 file handles 87
 files, iterating over lines
 elegantly 170
 FileStream 258

filtering 9, 141, 173, 267, 273,
 281, 294
 before joins 303
 lambda expressions 234
 LINQ to Rx 354
 operators 505
 finalizers 98
 invariants 460
 finally 87, 161, 171
 FindAll 13, 148, 173–174, 233,
 511–512
 First 499
 LinkedList<T> 513
 first edition, Range class 170
 first phase, type inference 249
 FirstOrDefault 499
 fixed 196
 fixed type variables 249
 FixedBufferAttribute 196
 fixed-size buffers 196
 flags
 for nullable values 106
 indicating missing values 14
 Flash 526
 flattening
 LINQ to Rx 355
 LINQ to XML 344
 flattening sequences 308
 flexibility 182, 371, 391, 487
 LINQ 16, 19
 options for Code
 Contracts 485
 float, special values 106
 floating-point numbers 106
 flow chart, type inference in
 C# 3 250
 flow control 399
 flow of execution, iterators 162
 fluent interfaces 274
 fluent notation 317
 fluff 3, 5, 126, 216
 Foord, Michael 411
 for statements
 capturing the loop
 variable 151
 implicitly typed local
 variables 209
 ForAll 461
 ForEach 13, 140, 233, 511
 foreach 10, 12, 157, 163
 capturing the loop
 variable 151
 disposal of iterator 87, 167,
 358, 522
 generics 85
 implicit casting 58

foreach statements
 anonymous types 222
 implicitly typed local
 variables 209
 foreign keys 304
 formal parameters 366
 formal specification 454
 forward references 181
 Fowler, Martin 274
 framework 521
 non-involvement in lifted
 operators 120
 support for LINQ 280
 uses of dynamic typing 408
 framework libraries 21
 frequently asked questions,
 nullity 104
 friend assemblies 198
 from
 contextual keyword 286
 multiple from clauses 308
 FromEvent 356
 fsi, F# Interactive 403
 fully qualified names 190
 Func<...> delegate types 50,
 229, 393
 function 145
 function names, C++ template
 arguments 99
 function pointers 28
 functional programming 4,
 129, 228, 233, 243, 272,
 376
 declarative style 320
 languages 491
 functionality, adding to existing
 types 258
 functions, interoperability 413
 fundamental units, value
 types 45
 future changes 151

G

garbage collection 22, 106,
 197, 421
 captured variables 148
 delegates 30
 effect of excessive boxing 48
 iterators 168
 garbage collector, generics 84
 generated code 185, 190
 equivalence of dot notation
 and query expressions 319
 iterators 168

- generated files 183
- generation operators 500
- generic collections 292, 508
- generic delegate types 66
 - Action 229
 - Func 229
- generic helper class 95
- generic method definitions,
 - retrieving through reflection 91
- generic methods 62, 65, 74, 96
 - reflection 90
 - type inference 246
- generic type definition 88
 - specifying in reflection 90
- generic type inference. *See* type inference
- generic types 62
 - example with Dictionary <TKey, TValue> 60
 - reflection 415
 - unaided by type inference 74
- generic variance 12, 52, 92, 525
 - C# 4 387
 - explicit declaration requirement 396
 - Java 397
 - restrictions 394
- generics 10, 51, 53, 57
 - anonymous types 222
 - closed types 63, 89
 - constructed types 62
 - Java 100
 - lack of generic properties etc 98
 - limitations 91
 - open types 63, 89
 - operators 245
 - pronunciation 64
 - reflection 88
 - type constraints 69
 - type parameter substitution 63
 - ubiquity within language specification 81
 - unbound types 62
- get_, avoided by named indexers 384
- GetConsoleScreenBufferEx 196
- GetDynamicMemberNames() 440, 445
- GetEnumerator 85, 157–158, 160, 162, 172, 351

- GetGenericArguments 90
- GetGenericTypeDefinition 89
- GetHashCode 48, 79, 440
 - anonymous types 223
 - Nullable<T> implementation 108
- GetInvocationList 34
- GetKeyForItem 513
- GetMemberBinder 444
- GetMetaObject 424, 440
- GetMethods 91
- getters 189, 205
- GetType 48, 90
- GetUnderlyingType 112
- GetValueOrDefault 108
- GetVariable 413
- GetViewBetween 516
- global namespace alias 192
- global scope, IronPython 412
- glue, Queryable methods 335
- gmcs 195
- goes to, lambda expressions 230
- Google 93, 183
 - AppEngine 492
- gotchas, dynamic typing 53
- grammar, domain specific languages 274
- graphing 413
- Gravell, Marc 98, 245
- Groovy 3, 275
- group ... by 311
- group joins 305
- group, contextual keyword 311
- GroupBy 273, 313, 501
- grouping 273
 - LINQ to Rx 354
 - order of results 312
- grouping expressions 311
- grouping operators 501
- GroupJoin 307, 502
- GTK# 23
- guarantees, Code Contracts 453
- guidance
 - anonymous types 225
 - captured variables 153
 - extending LINQ to Objects 357
 - extension methods 276
 - implicit typing 211
- GUIDs, checksum pragma directives 195
- gut feelings 211
 - implicit typing 210

H

- hash collisions 515
- hash tables 514
- hashing
 - generic interfaces 78
 - joins 301
- HashSet 516
- Hashtable 40, 60–61, 124
- HasValue 14, 107
 - See also* Nullable, HasValue property
- heap 44, 46, 106, 149, 197
- hello, world 4
- helper classes 96
- helper methods 187
 - initializing collections 218
- method group conversions 134
- hierarchical structure, code mirroring XML document 340
- higher-order functions 233, 393
- hooks, partial methods 185
- hosting languages
 - fluent interfaces 274
 - within C# 411
- hot observables 353
- HTML 491
- hyperbole 279

I

- ICloneable 41
- ICollection<T> 216, 496, 509–510
 - optimizing LINQ operators 358
- IComparable<T> 76
- IComparer 10, 358, 477
- IComparer and IComparer<T> 9, 78, 95, 142, 389, 515
- IDE 59, 181, 207, 445
- identity 441
 - DataContext 325
- identity conversions 71, 293
- IDictionary 509
 - ExpandoObject 436
- idioms 124
 - code 155, 453
 - delegates 230
- IDispatch 424

- IDisposable 22, 87, 157, 168, 171–172, 351
 - iterator block
 - implementation 167
 - iterator disposal in
 - foreach 522
- IDynamicMetaObjectProvider 424, 446
 - ExpandObject 436
- IEEE-754 106
- IEnumerable 52, 85, 93, 389, 418, 508
 - collection initializers 216
 - iterator blocks 156, 159, 168
 - LINQ 280, 292, 334
 - concurrent collections 519
 - returned from axis
 - methods 343
- IEnumerator 156, 159, 168
- IEqualityComparer 78, 498, 514
- IEquatable 79
- if/else blocks 128
- IgnoreCase 445
- IGroupedObservable 354
- IGrouping 311, 501
- IL 22, 236, 242, 245, 422
 - const and default parameter values 369
 - faked ref arguments 382
 - generated for object
 - initializers 213
 - type system 37, 83, 89, 107, 263
 - See also* Intermediate Language
- ildasm 115, 140, 167, 428
- IList 358, 499, 509
- IList<T> 40, 510
- immediate execution 283, 495
 - custom LINQ operators 358
- Immediate window, Visual Studio 403
- immutability 7, 108, 223, 376, 491
 - collections 501, 514
 - expression trees 237, 244
 - futility of invariants 484
 - method call chaining 268
 - See also* mutability
- impedance mismatch 280, 445
- imperative model 320, 493
- implementation
 - partial methods 186
 - revealing contracts 484
- simplicity of LINQ to Objects 173
- implementation details 167
 - dynamic typing 428
 - generated names 140
 - transparent identifiers 300
- implicit conversions 10, 58, 71, 248, 252
 - array covariance 40, 511
 - conditional operator 127
 - default parameter values 368
 - due to type constraints 96
 - dynamic 406
 - extension methods 262
 - for wrapper types 106
 - generic variance 389
 - method group
 - conversions 134
 - method groups 133
 - Nullable<T> 108
 - role in type inference 249
 - string to XName 338
 - string to XNamespace 338
 - to object 409
- implicit conversions, Nullable<T>. *See* Nullable<T>, implicit conversions
- implicit joins, LINQ to SQL 329
- implicit obligations 475
 - arithmetic 477
 - array bounds 476
 - non-nullity 475
- implicit parameter lists, lambda expressions 252
- implicit typing 38, 51, 207, 215
 - projections 270
 - range variables 292
- implicitly typed arrays 219
 - type inference 246
- implicitly typed local variables 16, 51, 207
- anonymous types 221
- lambda expression examples 233
- implicitly typed parameters
 - anonymous methods 231
 - lambda expressions 292
 - type inference 246
- impure code 462
- in keyword 286
- in modifier,
 - contravariance 389
- inclusive OR operator 119
- inconsistency, naming conventions 318
- indentation 218
- independence, multiple iterators over one
 - collection 159
- independent
 - responsibilities 13
- index, overloads for Select and Where 317
- indexed properties 383
- indexers
 - Dictionary<TKey, TValue> 61
 - named 383
 - optional parameters and named arguments 372
- IndexOf 509, 511
- indirection 28
 - delegates 33, 132
- InDocumentOrder 344
- infer 474
- inference
 - implicitly typed local variables 16, 207
 - type parameters for generic methods 74
- inferred return types
 - anonymous functions 247
 - comparing anonymous function conversions 254
- infinite loops 231
- infinite sequences 280
- infinity 106
- infoof 245
- inheritance 256, 356, 480
 - contracts 466
 - deriving from type parameters in C++ 100
 - method group
 - conversions 137
 - overload resolution 378
 - variance 93, 387, 395
- initialization 7, 211
 - arrays 219
 - immutable types 376
 - implicitly typed local variables 208
 - partial types 182
- initialization expressions 207–210
- inline initialization for "constants" 218
- inline specification of delegate actions 139
- inlining 465

- inner joins 301
 - use in object-oriented code 304
 - inner sequence 301
 - INotifyCollectionChanged 514
 - INotifyPropertyChanged 514
 - input positions,
 - contravariance 388
 - InsertAt 509
 - InsertItem 513
 - instance members 205
 - capturing this reference 145
 - instance methods
 - calling on null references 263
 - delegate example 32
 - preferred over extension methods 262
 - using to create delegate instances 30
 - instance variables, representing local variables in
 - iterators 162
 - instant messaging 285
 - instantiation of local variables 149
 - instincts 277
 - integer literals, potential confusion with implicit typing 210
 - integration, with LINQ 342
 - IntelliSense 59, 379, 410, 482
 - extension methods 261, 276
 - intentions, communicating via
 - Code Contracts 453
 - interfaces 43, 172, 288, 395, 434
 - adding functionality 256
 - anonymous types 223
 - array covariance 92
 - breaking changes 396
 - collections 508
 - comparison with
 - contracts 454
 - comparison with delegates 28
 - comparison with duck typing 418
 - contract classes 467
 - extension methods 52
 - generic variance 387, 389
 - inherited contracts 466
 - interface keyword 43
 - out-of-process LINQ providers 330
 - return type covariance 41
 - static classes 188
 - static members 98
 - Interlocked 206
 - intermediate computations, let
 - clauses 298
 - intermediate format, LINQ queries 243
 - Intermediate Language. *See* IL
 - intermediate variable, method
 - group conversions 134
 - internal accessibility 198
 - internal classes, hiding implementation details 441
 - InternalsVisibleToAttribute 198
 - interoperability 19, 21, 380, 405
 - interpreters 22, 403, 405, 411
 - Intersect 506
 - IntersectWith 516
 - into, contextual keyword 314
 - introspection, LINQ queries 329
 - InvalidCastException 47, 413
 - InvalidOperationException 107, 117, 499, 517–518
 - invariance 92, 388
 - Invariant method 460
 - invariants 459, 486
 - breaking via inheritance 467
 - inherited contracts 466
 - Inversion of Control 88, 414
 - invocation
 - delegates 28
 - field-like events 399
 - invocation list 33
 - Invoke 31, 134
 - delegate method 31–32
 - InvokeMemberBinder 444, 450
 - invoking methods,
 - reflection 91
 - IObservable 351, 525
 - IObserver 351, 525
 - IOrderedEnumerable 298
 - iPad 22
 - iPhone 22, 492
 - iPod Touch 22
 - IProducerConsumerCollection<T> 518
 - IQueryable 329–330, 334
 - IQueryProvider 329
 - IronPython 21, 53, 402
 - binder reuse 425
 - using from C# 4 410
 - IronRuby 402
 - binder reuse 425
 - is operator 121
 - breaking changes 396
 - ISet 510, 516
 - IsGenericMethod 91
 - IsGenericType 89
 - IsGenericTypeDefinition 90
 - IsInterned 244
 - IsNullOrEmpty 264
 - isolating dynamic typing 416
 - IsProperSubsetOf 516
 - IsProperSupersetOf 516
 - IsReadOnly 509
 - IsSubsetOf 516
 - IsSupersetOf 516
 - IStructuralComparable 525
 - IStructuralEquatable 525
 - ITask 176–177
 - Items 513
 - iterable 157
 - iteration pattern 156
 - iteration variables, foreach
 - loops 86
 - iterator blocks 160–161, 357, 429, 522
 - elegance 169
 - iterators 94, 156, 334
 - implementing in C# 1 157
 - real-life examples 169
 - yield type 161
- ## J
-
- Java 3, 92–93
 - generics 100, 397
 - JavaScript 350, 403, 491, 527
 - JIT compiler 22, 77, 421, 447
 - DLR caches 425
 - expression trees 236, 244–245
 - generics 59, 77, 83, 99
 - Join 304, 502, 524
 - custom comparisons 318
 - join ... into, not a
 - continuation 316
 - join operators 502
 - join plans, LINQ to Rx 356
 - join-calculus 356
 - joins
 - LINQ to SQL 327
 - query expressions 301
 - Just-In-Time compilation 22
 - See also* JIT compiler
 - just-in-time, iterator
 - behavior 266

K

kernel profile 22
 Key modifier, VB anonymous types 223
 key selectors 301, 319
 Key, property in
 IGrouping<, > 311
 KeyedCollection 513
 KeyNotFoundException 509
 KeyPress event 132
 KeyPressEventArgs 135
 KeyPressEventHandler 133, 135
 keys
 grouping 311
 joins 301
 keystrokes, saved by implicit typing 209
 KeyValuePair 64
 KeyValuePair<TKey, TValue> 216, 509
 Knuth, Donald 115

L

L0, L1, L2 caches, DLR 425
 Lambda 238, 242
 lambda calculus 228
 lambda expressions 11, 13, 50, 228, 230, 288, 334
 caching 235
 common uses in extension methods 267
 converting to delegates 229
 converting to expression trees 239
 dynamic typing 433
 event handlers 235
 implicitly typed
 parameters 246, 292
 inferring parameter and return types 249
 restrictions when converting to expression trees 240
 static checking 480
 timing of checking the body 252
 type inference 246
 lambda. *See* lambda expression
 LambdaExpression 238
 Langer, Angelika 397
 language complexity 284
 language design 402

language designers 92, 130, 220, 410
 Language Integrated Query 16
 language integration 173
 language specification 25, 113, 134, 194, 205
 iterators 168
 lack of guarantees of memory layout 46
 object initializer terminology 214
 query continuations 314
 query expressions 288
 transparent identifiers 300
 type inference 246
 ubiquity of generics 81
 languages 21
 additional functionality for nullable types 111
 behavior with nullable types 120
 incorporating closures 145
 Last 499
 LinkedList<T> 513
 layout of code, object initializers 215
 lazy evaluation 173–174, 267
 Lazy<T> 525
 LDAP 228
 leaf expressions, expression trees 237
 learning 23
 left outer joins
 generated SQL 328
 simulating with group joins 306
 left sequence 301
 legacy contracts 462, 486
 let
 contextual keyword 298
 LINQ to SQL 326
 libraries 21–22, 76, 256, 274, 411
 DLR 421
 duck typing 418
 generics 59
 licensing 385
 lifetime management 170
 lifetime, captured variables 148
 LIFO 517
 lifted conversions 117
 lifted operators 117
 limitations
 C# 1 type system 39
 generics 91
 line breaks 140
 line count, bad measure of complexity 154
 LineReader 171
 lines, iterating over a file 170
 LinkedList 512
 LinkedListNode 512
 linking, Primary Interop Assemblies 385
 LINQ 16, 52, 170, 173, 203, 279, 322
 anonymous types 225
 data model 280
 dynamic code 434
 key enabling aspects 243
 query styles 317, 319
 standard query operators 495
 third party providers 322
 without query expressions 265
 See also Language Integrated Query
 LINQ providers, building your own 329
 LINQ to Entities 322
 LINQ to Objects 173, 228, 284, 288, 320, 343, 495
 covariance in C# 4 391
 execution path 243
 extended in
 System.Interactive 351
 extending 357
 LINQ to Rx 350
 LINQ to SQL 18, 228, 284, 322
 DataContext 325
 execution path 243
 queries 325
 LINQ to XML 18, 337, 436
 declarative construction 340
 design decisions 345
 LINQPad 24, 265
 Lippert, Eric 30, 161, 245, 400, 511–512
 Liskov's Substitution Principle 466
 List<T> 6, 53, 65, 280, 510
 collection initializers 215
 lambda expression examples 233
 lists 280
 literals 210
 default parameter values 368
 little-endian 39
 local variable declarations, anonymous methods 140

local variables 46, 149, 212, 288, 375
 autogenerated for fake pass-by-value 382
 captured by anonymous methods 145
 dynamic 435
 for pass-by-reference parameters 381
 implicit typing 207
 instantiation 149
 iterator blocks 162
 restrictions on var 408
 traditional postcondition testing 458
 locking 189, 206, 398, 429, 518
 field-like events 399
 log files 225, 310
 Log, DataContext 325
 logging 189, 235, 335
 configuration 414
 fake LINQ provider 331
 logic 170
 nullable Booleans 119
 represented in expression trees 228
 logical AND operator 119
 logical negation operator 119
 logically related conditions
 combining in where clauses 295
 LongCount 495
 loops, captured variables 150, 154
 l-values 46

M

macros 99
 magic value pattern 14, 105, 370
 magic variables 232
 Main method, snippets 23
 mainstream languages 490
 maintainability 4, 154, 256, 375, 491
 maintenance 8, 445
 cost of multiple configurations 487
 MakeGenericType 89
 malformed contracts 458
 managed code 46
 Managed Extensibility Framework 525
 Mandelbrot 346
 mapping, LINQ to SQL 323
 MARS 523
 marshalling 196, 421
 master pages 523
 matching 141
 Math 97
 mathematical code 97
 Max 495
 SortedSet<T> 516
 maybe, nullable logic 120
 MD5 195
 meaning, clarifying arguments 373
 mechanisms, iteration 170
 media player 285
 MEF 525
 Meier, Sid 411
 member invocation 405
 member resolution 430
 members, obtaining without reflection 244
 memory
 cost of JIT compilation 83
 generics 59
 limit on buffering operations 283
 overhead of extra methods 46
 overhead of objects 106
 representation of null 104
 memory fragmentation 515
 memory leaks 30
 MemoryStream 136, 258
 messages, contract failures 471
 metadata 100, 245
 contracts 468
 DynamicAttribute 435
 LINQ to SQL 323
 metaobjects 422, 440, 447–448
 metaprogramming 414
 MetaRumpelstiltskin 448
 method arguments
 compared with type arguments 63
 role in type inference 249
 method calls, specifying arrays as arguments 219
 method group conversions 67, 133
 ambiguity 134
 breaking changes 137
 method groups 133, 208
 complexities in overloading 246
 dynamic code 433
 method invocation, ExpandableObject 436
 method parameters, compared with type parameters 63
 method signatures 29
 MethodInfo 91, 241, 244, 449
 compiler shortcut 242
 MethodInvocation 137, 145
 methods
 compatibility with delegate types 29
 delegate compatibility 134
 generated by anonymous methods 140
 meaning of return statements 164
 using results with var 209
 virtual dispatch 403
 metrics 273
 Meyer, Bertrand 454
 Micro Framework 22, 527
 micro-optimization 268
 Microsoft 21–22, 167, 322, 380, 451, 492
 anonymous type implementation 222
 CCR and DSS Toolkit 175
 choices around exceptions 265
 event pattern guidelines 50
 pragma directives 194
 Reactive Extensions 350
 Robotics Studio 175
 Microsoft.Office.Interop.Excel 409
 migrating, to use extension methods 261
 Min 495
 SortedSet<T> 516
 MinValue 478
 miscellaneous utility library 258
 MiscUtil 170–171, 275, 341
 misinformation 45
 missing data 14
 misuse of helper classes 188
 ML 228
 mocking 275, 408
 model databases 323
 modifiers
 extension methods 260
 out and ref with named arguments 373
 monads 491
 Mondrian 93

monitors, locking changes in
 C# 4 398
 Mono 22–23, 195, 403, 422,
 491, 525
 MonoTouch 22
 Monty Python 218
 Moonlight 527
 MoreLINQ 284, 357
 MouseEventArgs 135
 EventHandler 135
 MoveNext 85, 157, 162–168,
 282, 352
 msbuild 456
 mscorlib 456
 MSDN 284, 301, 343
 expression tree
 documentation 237
 multicast delegates, generic
 variance 396
 MulticastDelegate 29
 multiple active result sets 523
 multiple criteria, sorting 126
 multiple dispatch 419
 multiple from clauses 308
 multiple orderings, in one
 orderby clause 298
 multiple query
 continuations 316
 multiple type parameters
 variance 394
 multitargeting 521
 mutability 7, 11, 108, 223
 collections 510–511, 514
 encouraged by C# 3 376
 GetViewBetween 516
 method call chaining 268
 mutated state,
 postconditions 458
 myths, value types and refer-
 ence types 45

N

name/value pairs 436
 named arguments 8, 20, 214,
 372
 evaluation order 374
 in tandem with optional
 parameters 376
 to resolve overload
 ambiguity 379
 named indexers 20, 383
 design decisions 384
 namespace alias qualifier 191

namespace aliases 190, 193
 namespaces 190
 extension methods 262, 276
 LINQ to XML 338
 naming 140, 373, 437
 avoiding collisions 194
 collisions 205, 440
 compile-time duck
 typing 401
 extension methods 261, 265
 generic reflection
 methods 89
 lambda expression
 parameters 232
 overloading and dynamic
 typing 416
 parameters 379
 readability benefit of object
 initializers 377
 scope in the real world 190
 variables used in
 contracts 482
 naming conventions
 class contracts 468
 extension methods 275
 lambda expression
 parameters 317
 ObjectInvariant 460
 type parameters 65, 89
 NaN. *See* not a number
 native code 22, 196, 408, 421
 generics 83
 natural language, fluent
 interfaces 274
 nested classes 448
 nested subscriptions, LINQ to
 Rx 354
 nested types 159, 188–189
 generics 82
 in snippets 24
 interfaces 468
 nested variance 394
 NetworkStream 258
 new keyword, object
 initializers 215
 new style contracts 463
 Ng, Sam 421
 NGen 22
 Noda Time 524
 Nodes 343
 nodes
 LINQ to XML 338
 of expression trees 237
 NodeType property, expression
 trees 237

none, Code Contracts reten-
 tion option 486
 nongeneric classes, require-
 ment for extension
 methods 260
 nongeneric helper classes 80
 non-nested classes, require-
 ment for extension
 methods 260
 non-null obligations 475
 non-nullable fields, generating
 inner joins 329
 non-nullable reference
 types 455, 488
 non-pointer types 409
 nonrepeatable sequences 358
 nonvirtual calls, instance
 method calls on null
 references 264
 nonvoid return types, lambda
 expressions 231
 no-op select clauses 296
 not a number 106
 notation, transparent
 identifiers 300
 notes 512
 NotificationSubscription, sam-
 ple data model 285
 NULL
 database fields 105
 SQL 120
 null 14, 53, 208, 248, 417
 as a typeless expression 219
 comparisons with generic
 values 77, 118
 default parameter values 368
 extension methods 277
 ignored in LINQ to XML
 construction 340
 language support for
 Nullable<T> 112
 magic value for optional
 parameters 370
 meaning of 104
 meaning when combining
 delegates 33
 nullable value types 114
 parameter passing 47
 reference type constraints 70
 results of
 CompareTo(null) 76
 null coalescing operator 121,
 126
 used in comparisons 127
 null literal 114

null references
 boxing Nullable 110
 extension methods 263
 null values 112
 Hashtable 124
 Nullable class 107, 111
 nullable fields, generating
 outer joins 329
 nullable types
 lifted conversions 117
 meaning in the C#
 specification 113
 nullable value types 14, 53, 103
 compared with "nullable
 type" 113
 conversions in LINQ to
 XML 343
 generic comparisons 77
 optional parameters 370
 underlying type 107
 Nullable<T> 14
 boxing 110
 constructors 107
 conversions 116
 default constructor 107, 114
 Equals method 111
 explicit conversions 108, 116
 GetHashCode method 108
 GetValueOrDefault
 method 108
 HasValue property 107, 112
 implicit conversions 108, 116
 null value 112
 operators 116
 ToString method 108
 unboxing 110
 Value property 107
 nullity 473
 nullity check, events 143
 NullReferenceException 77,
 111, 263, 265, 407
 unboxing 110
 numbers, LINQ to XML
 content 340

O

object
 immediate base of helper
 classes 187
 LINQ to XML construction
 pattern 340
 use before generics 58
 object initializers 7, 211–213,
 285
 encouraging mutability 376

object model, Office 409
 Object Relational
 Mapping 183
 ObjectInvariant,
 conventions 460
 object-oriented code 259
 object-oriented data models,
 inner joins 304
 object-oriented
 programming 33
 object-relational mapping 322
 objects
 distinguishing from
 references 44
 role in parameter passing 47
 Observable<T> 353
 ObservableCollection<T> 513
 observables, hot and cold 353
 observers 351
 off-by-one errors 160
 Office 380, 404, 409
 office suite 285
 OfType 95, 497, 505
 OldValue 459
 on, contextual keyword 302
 OnCompleted 351
 one-to-one correspondence,
 group joins 306
 OnException 351
 OnNext 351
 on-the-fly reconfiguration 414
 open constructed types, reflec-
 tion with generics 89
 open source 322
 DLR 421
 libraries 487
 open types 63
 generics 89
 OpenText 170, 209
 operand types, operators or
 nullable types 117
 operator constraints 97
 operators 117, 416
 default value expression 76
 generics 245
 Nullable<T> 116
 See also Nullable, operators
 overloaded terminology 284
 provided by languages for
 nullable types 111
 purity 463
 static classes 188
 type constraints 97
 optimization 402, 415, 487
 LINQ operators 357, 360,
 496, 500, 505
 optional parameters 14–15, 20,
 366
 declaring 367
 in tandem with named
 arguments 376
 optional values 123
 order of evaluation, null
 coalescing operator 123
 order of execution, multicast
 delegates 34
 OrderBy 11, 270, 297, 507, 511
 OrderByDescending 270, 297,
 507
 OrderedParallelQuery 348
 ordering 78, 296
 arguments and
 parameters 368
 dictionaries 515
 importance in anonymous
 types 222
 initialization 182
 Parallel LINQ 348
 OrdinalIgnoreCase 318
 ORM. *See* Object Relational
 Mapping
 out modifier, covariance 389
 out of process queries 228, 243
 out parameters
 forbidden in partial
 methods 186
 inability to capture 145
 lambda expressions 231
 named arguments 373
 postconditions 458
 prohibited in iterator
 blocks 165
 restrictions on extension
 methods 260
 restrictions on generic
 variance 395
 restrictions on optional
 parameters 368
 OutAttribute 395
 outer joins 301
 LINQ to SQL 328
 outer sequence 301
 outer variables 145
 output parameters 76, 126
 TryXXX pattern 124
 output positions
 covariance 388
 out parameters 395
 Output window, warning
 numbers 194
 overengineering 129

overflows 478
 overhead, memory used by
 objects 106
 Overlaps 516
 overload resolution 246, 289,
 368, 403, 431
 breaking changes due to
 generic variance 396
 extension methods 262
 generic types 78
 generics 78
 lambda expressions and
 dynamic typing 433
 named arguments and
 optional parameters 378
 summary of changes in
 C# 3 254
 overloaded operators, nullable
 types 117
 overloading 86, 246
 alternative to optional
 parameters 366
 changes in C# 3 252
 complicated by
 inheritance 379
 custom LINQ operators 358
 dynamic arguments 426
 Func delegate types 172
 GroupBy 314
 method group
 conversions 137
 method groups 133
 query expressions and dot
 notation 317
 single and multiple
 dispatch 419
 overloads, Execute and
 CreateQuery 330
 overriding 380, 419, 466
 partial types 183
 return type covariance 41
 ownership 172

P

P/Invoke 196, 408
 pain points 126
 Pair<T1,T2> 78
 Parallel Extensions 175, 346,
 493, 518, 524
 Parallel LINQ 346
 ParallelEnumerable 347
 parallelism 175, 346, 493
 ParallelQuery 347–348

parameter array 340
 optional parameters 368
 parameter expressions 241
 appearance in visualizer 242
 parameter list 143
 parameter names
 from range variables 291
 importance of names 379
 parameter passing 46
 parameter type
 contravariance 41
 parameter types
 anonymous methods 140
 conversion from argument
 types 252
 delegates 29
 parameter wildcarding 144
 ParameterExpression 241
 parameterized properties 383
 ParameterizedThreadStart 28,
 144, 147
 parameterless
 constructors 187, 206, 212
 default parameter values 368
 generic type constraints 70
 parameters 8
 anonymous methods 143
 binding in expression
 trees 241
 captured by anonymous
 methods 145
 comparing type and method
 parameters 63
 contravariance 51
 contravariance in
 delegates 135
 documentation 454
 dynamic 408
 dynamic type 426
 implicit typing in lambda
 expressions 231
 initializing arrays 219
 Invoke method on a delegate
 type 31
 iterator blocks 165
 lambda expressions with a
 single parameter 232
 listing explicitly in
 lambdas 230
 optional 366
 readability 214
 required for extension
 methods 260
 terminology 366
 type inference 74
 using lambda expressions for
 logging 236
 ParamName 458
 params 340
 optional parameters 368
 parent nodes, LINQ to XML
 content 340
 parentheses
 constructor calls 213
 query expressions and dot
 notation 318
 removing from lambda
 expressions 232
 parser 403
 partial methods 181, 184
 partial types 180
 initialization 182
 LINQ to SQL 324
 restrictions 181
 partial variance 397
 partial, contextual
 keyword 181
 PartialComparer 127
 partitioning operators 503
 pass by reference 46
 pathological cases 420
 pattern matching 491
 patterns 76, 124, 278
 iterator 156
 language support 177
 nullability 105
 pausing execution, iterator
 blocks 162
 pdb files 464
 Peek
 Queue<T> 517
 Stack<T> 518
 perfection 445
 performance 197, 487, 523
 as operator with nullable
 value types 121
 boxing 48
 Code Contracts 465, 487–
 488
 cost of JIT compilation 83
 dynamic typing 402, 424
 generic comparisons with
 null 77
 generics 59, 91, 98
 Java generics 101
 LINQ 268, 290, 302
 of value and reference
 types 45
 TryXXX pattern 76

- performance
 - characteristics 210
 - permissions 199
 - perspectives, viewing types and objects 395
 - Pex 492
 - phases, type inference 248
 - PIA. *See* Primary Interop Assemblies
 - pinning 197
 - pipelines 268
 - LINQ 228
 - piping data 267
 - placeholders
 - Result 459
 - type parameters 62
 - planning, generic variance 396
 - platform designers 92
 - PLINQ 346
 - Point structures, mutability 108
 - pointer types, restrictions on extension methods 260
 - pointers 39, 196, 427
 - polymorphism 401, 434
 - Pop 518
 - portability 400
 - position, iterators 157
 - postconditions 458, 486
 - inherited contracts 466
 - Power Collections 520
 - power, language design choices 92
 - practices 278
 - pragma directives 194
 - checksum 195
 - warnings 194
 - pragmatism 445
 - pre and post, Code Contracts retention option 486
 - precompilation, ASP.NET 523
 - preconditions 456, 486
 - Code Contracts retention option 486
 - contract reference assemblies 470
 - inherited contracts 466
 - suggested by static checker 477
 - variations in meaning 483
 - predefined conversions 116
 - predefined operators 117
 - Predicate 269
 - purity 463
 - predicates 141, 173, 282, 495
 - predictability 272
 - prefix, convention for type parameter names 65
 - preprocessing directives 194
 - preprocessing, query expressions 280
 - preprocessor symbols 263, 486
 - Code Contracts 464
 - primary constraints 73
 - Primary Interop Assemblies 20, 385, 409
 - primary keys 304
 - primary orderings 297
 - primitive types 196
 - private
 - access modifier 189
 - partial methods 186
 - private accessibility, from nested types 159
 - private constructors, helper methods 187
 - private fields 457
 - private members, access from nested types 189
 - processing models, buffering versus streaming 267
 - ProcessStartInfo 218
 - producer/consumer pattern 518
 - production assemblies 199
 - productivity 410, 420, 491
 - profiles 22
 - program verifier, Spec# 455
 - project files
 - editing for dependencies 184
 - project management 273
 - projection initializers 223
 - projection operators 503
 - projections 269, 282, 287
 - grouping 311
 - LINQ to Rx 354
 - sorting 273
 - Projects, sample data model 285
 - promises 453
 - promotion, addition operands 417
 - pronunciation
 - generics 64
 - lambda expressions 230
 - properties 204, 244
 - access modifiers 5–6
 - anonymous types 222
 - automatically implemented 7
 - default 383
 - ExpandableObject 436
 - getter / setter access 189
 - mutability 377
 - navigation in object-oriented models 304
 - projection initializers 224
 - purity 463
 - setting subproperties with object initializers 214
 - setting with object initializers 212
 - property descriptors 236
 - property names, anonymous types 52
 - protected access, prohibited in static classes 188
 - protected methods 180
 - Protocol Buffers 183
 - provider pattern 172
 - proxying, COM 386
 - public domain 43
 - public interface, unit testing 199
 - public keys, friend assemblies 199
 - public methods, invariants 467
 - publish/subscribe pattern 35
 - pull model, LINQ to Objects 350
 - pure methods 482
 - PureAttribute 463
 - purity 463
 - Push 518
 - push model, LINQ to Rx 350
 - Python 3, 20, 405, 492
-
- Q**
- quantifier operators 505
 - queries
 - LINQ to SQL 325
 - LINQ to XML 342
 - observables 354
 - parallelizing 348
 - query continuations 311, 314
 - multiple 316
 - query expression pattern 288, 356
 - query expressions 16, 173, 265, 269, 280, 495
 - compared to dot notation 317, 319
 - consistency between LINQ providers 284

query expressions (*continued*)
 continuation
 translations 314
 degenerate queries 295
 dynamic code 434
 explicitly typed range
 variables 292
 order of clauses 290
 select clauses 286
 translation 19, 325, 334, 356
 type inference example 291
 query optimizer 327
 Queryable 265, 330–331, 334
 querying 12
 QueryProvider, property of
 IQueryable 330
 Queue<T> 517

R

Rahien, Ayende 275
 random numbers 280, 359
 Random, using carefully 359
 Range 266, 500
 Excel 409
 LINQ to Rx 353
 ParallelEnumerable 350
 range variables 290, 292
 combined with transparent
 identifiers 299
 explicit typing 292
 scope with query
 continuations 315
 Range<T> 170
 raw types (Java) 100
 reachability 474
 Reactive Extensions 175, 284,
 350, 493, 525
 readability 74, 152, 215, 257,
 272
 argument reordering 375
 captured variables 145
 consistency 141
 declarative code 320
 extension method
 naming 261
 extra methods for
 delegates 132
 generics 59
 implicit typing 204, 209–210
 iterator blocks 170
 LINQ 11, 13, 16, 232, 268
 LINQ to XML namespace
 support 339
 measuring objectively 276

method group
 conversions 133
 multiple where clauses 295
 named arguments 372, 380
 naming 379
 nullable types 114
 query expressions and dot
 notation 317
 single exit point 455
 transparent identifiers 319
 using Word before C# 4 381
 ReadAllLines 157, 174
 ReadAllText 523
 ReaderWriterLockSlim 524
 read-evaluate-print loop 403
 ReadFully 258
 reading from streams 258
 ReadLine 170
 read-only fields, anonymous
 types 223
 read-only properties 8, 206
 anonymous types 223
 ReadOnlyCollection 218
 ReadOnlyCollection<T> 94,
 514
 ReadOnlyObservableCollection
 <T> 514
 receivers 424
 recompilation 371, 413
 const and default values 369
 reconfiguration, on the fly 414
 rectangular arrays 476, 512
 recursion 437
 guard in Code Contracts 465
 LINQ to XML
 construction 341
 red-black trees 515–516
 redeployment 413
 redirecting 448
 redistribution 385
 redundant computations 299
 ref parameters
 CLR support 395
 COM 382
 inability to capture 145
 invariance 389
 lambda expressions 231
 named arguments 373
 postconditions 458
 prohibited in iterator
 blocks 165
 restrictions on extension
 methods 260
 restrictions on optional
 parameters 368

refactoring 126, 183, 244, 357,
 375
 reference conversions 71, 135,
 293
 generic variance 389, 395
 reference type constraints 69,
 77
 avoiding boxing 128
 reference types 14, 22, 42
 array covariance 92
 null coalescing operator 123
 nullability 104
 type arguments and generic
 variance 395
 wrapper for nullable value
 types 106
 references 44
 memory size 83
 navigation in object-oriented
 models 304
 referencing, Primary Interop
 Assemblies 385
 reflection 199, 244, 426
 generics 71, 88
 invoking methods 91
 obtaining a MethodInfo 236,
 242
 operators and expression
 trees 245
 retrieving generic method
 definitions 91
 side-stepping with dynamic
 typing 91, 402–403, 415
 Reflector 115, 140, 167, 296,
 386, 428
 reformatting 413
 refreshingly minty code 23
 regions 180
 register optimizations 84
 regular expressions 61
 relational data models, inner
 joins 304
 relational operators 117
 release builds 484, 486
 assertions 455
 release requires, Code Con-
 tracts retention
 option 486
 releasing resources, foreach 87
 Remove 33, 131, 509
 RemoveAt 509
 RemoveFirst 513
 RemoveItem 513
 RemoveLast 513
 RemoveWhere 516

- removing delegates 33
 - Repeat 500
 - repetitive code 226
 - REPL 403, 445
 - Replace 267–268
 - required parameters 367–368
 - requirements 453
 - Requires 456, 486
 - research 454
 - Reset 168
 - resolution, dynamic calls 426
 - resource acquisition, timing in
 - iterators 172
 - resource management,
 - iterators 165
 - resources, released after
 - iterating 87
 - responding dynamically 448
 - Response.Redirect 399
 - RESTful services 408
 - restoring warnings 195
 - restrictions 368
 - default parameter values 368
 - documenting 445
 - dynamic code 432
 - generic variance 394
 - implicitly typed local variables 208
 - nullable type operators 117
 - optional parameters 368
 - partial types 181
 - Result 459
 - return statements 164
 - implicit return types 248
 - lambda expressions 231
 - prohibited in iterator blocks 161
 - return types
 - anonymous methods 142
 - compatibility with delegate types 29
 - compatibility, covariance of delegates in C# 2 50
 - covariance 41, 51
 - covariance in delegates 136
 - delegates 29
 - dynamic 408
 - Func and Action 229
 - implementing IDynamic-MetaObjectProvider 448
 - inferred from anonymous functions 247
 - lambda expressions 231
 - of Invoke method on a delegate type 31
 - overloading 86
 - return values 126
 - covariance 388
 - indicators of success 124
 - postconditions 458
 - return, anonymous
 - methods 142
 - ReturnType 448
 - reuse, native code from JIT
 - compilation 84
 - Reverse 266, 282, 317, 507
 - missing from LINQ to Rx 356
 - SortedSet 517
 - reversing strings 25
 - rewriting, Code Contracts 464
 - right associativity, null coalescing operators 123
 - right sequence 301
 - robustness 272, 451
 - aided by generics 59
 - dynamic typing 419
 - LINQ to XML queries 343
 - static checking 473
 - root element, LINQ to XML 338
 - Root property,
 - XDocument 338
 - RouteValueDictionary 514
 - rows, joins 301
 - RPCs 492
 - RSS 341
 - Ruby 3, 405
 - rules
 - C# 2 type inference 75
 - DLR 424
 - dynamic typing 244
 - rules engines 411
 - Rumpelstiltskin 447
 - runtime 21–22, 521
 - behavior of casting 45
 - RuntimeBinderException 407
- S**
-
- sample data model 285
 - SampleData 285
 - sandboxing 415, 526
 - Sandcastle 482
 - sanity checking 485
 - Java compiler 100
 - SaveAs, Word 381
 - scaling 493
 - schedulers
 - LINQ to Rx 355
 - schema 329
 - LINQ to SQL 323
 - Scheme 145
 - scientific community 97
 - scope 145–146, 152
 - captured variables 149
 - finally blocks 165
 - IronPython 412
 - query continuations 315
 - range variables in joins 302
 - ScriptEngine 411
 - ScriptHost 411
 - scripting 411
 - ScriptRuntime 411
 - ScriptScope 412
 - ScriptSource 411
 - sealed classes, adding
 - functionality 256
 - sealed modifier
 - static classes 188
 - utility classes 187
 - second phase, type
 - inference 249
 - secondary constraints 73
 - security 171, 199, 415
 - Select 269, 282, 317, 503
 - select, contextual keyword 287
 - SelectMany 310, 344, 503
 - semantics
 - clarifying with named arguments 373
 - magic value pattern 105
 - value types and reference types 45
 - semicolons, lambda
 - expressions 231, 234
 - separation of concerns 13
 - sequence diagrams 162
 - sequence of sequences 308
 - SequenceEqual 500
 - sequences 156, 280
 - generating in
 - SelectMany 310
 - joins 301
 - produced and consumed by LINQ to XML 345
 - sequential execution, appearance of iterator blocks 161
 - serialization 183, 223, 245
 - Server Explorer, Visual Studio 323
 - service pack 1 293
 - service-oriented
 - applications 523
 - set-based operators 506

- SetEquals 516
- SetItem 513
- setters 189, 205
- SetUnwind 471
- SetVariable 413
- Seurat, George 93
- SHA-1 195
- Shakespeare 453
- sharing, captured variables 152
- short methods 461
- short-circuiting 128
- shortcomings, first edition
 - Range class 170
- shorthand
 - delegate combination 33
 - delegate invocation 31
- Show 372
- shrink-wrapped
 - applications 487
- side effects 11, 186, 455
 - argument evaluation 374
 - purity 463
- side-by-side execution 526
- side-effect-free code 271
- signatures, custom rewriter
 - methods 472
- signed assemblies 199
- silver bullet 280
- Silverlight 350, 526
- simplicity 154
- simplification, automatically
 - implemented properties 7
- Single 326, 499
- single dispatch 419
- single exit point 455
- single method interfaces,
 - delegates 172
- single parameter lambda
 - expressions 232
- single responsibility
 - principle 159
- single threading, Mandelbrot
 - generation 347
- single-class hierarchy 58
- single-dimensional arrays 476
- single-method interfaces 392
 - comparison with
 - delegates 28
- SingleOrDefault 499
- site containers 429
- SkeetySoft 285
- Skip 503
- SkipWhile 503
- sn 199
- snapshots 518–519
- snippets 23
- Snippy 24, 138, 239, 428
- sock puppets 492
- software engineering 278
- solid state drives 493
- Solution Explorer 184, 194
- Sondheim, Stephen 93
- Sort 10, 142, 233, 271, 511
 - custom comparisons 95
- SortedDictionary<TKey, TValue> 210, 515
- SortedList<TKey, TValue> 210, 515
- SortedSet<T> 510
- sorting 9, 142, 257, 270
 - implementing custom
 - comparisons 126
 - lambda expressions 234
 - stability 511
 - sorting operators 507
- source code 23, 286
- source files, checksums 195
- Spec# 455, 473
- specialization 187, 256
- specification 22, 25, 205
 - C# 4 compliance 380
 - ubiquity of generics 81
- SPOT 527
- SQL 120, 290
 - converted from query
 - expressions 17, 19, 243, 322, 325
 - joins 301–302, 306, 328
- SQL Server 523
- SQL Server 2005 322, 525
- SQL Server Management
 - Studio Express 327
- square brackets, reflection over
 - generics 89
- stack 44, 46, 149
- stack frames 146
- stack overflow, recursion in
 - Code Contracts 465
- Stack<T> 518
- standard query operators 284, 330, 357, 495
 - parallel equivalents 348
- standards 71
- Start 218
- StartsWith 244
- state
 - asynchronous web service
 - calls 176
 - invariants 459
 - iterators 158
 - utility classes 187
- state machines, generated for
 - iterator blocks 161
- statefulness 271
- static checker 456, 461, 472, 487
 - restrictions on invariants 461
- static classes 80, 186, 257
 - call site storage 429
 - extension methods and
 - dynamic code 433
 - Nullable 111
 - requirement for extension
 - methods 260
- static constructors 82, 206, 218
- static fields, generic types 81
- static initializers 82
- static interfaces 98
- static members
 - automatic properties 205
 - thread safety 205
- static methods 52, 256
 - chaining with extension
 - methods 268
 - custom rewriter methods
 - (Code Contracts) 472
 - delegate example 32
 - dynamic code 433–434
 - using to create delegate
 - instances 30
 - utility classes 187
- static modifier, static
 - classes 188
- static types
 - expressions 37
 - targets of method calls 380
- static typing 21, 37, 134, 211, 401, 491
 - expression trees 238
 - implicitly typed local
 - variables 51
 - productivity 410
- static variables,
 - initialization 182
- Stream 136
- streaming 174, 282, 495
 - custom LINQ operators 358
 - flattening and cross
 - joins 310
 - group joins 306
 - inner joins 302
- streaming data 267
- StreamReader 173
- streams 258
- StreamUtil 258
- string literals, Python 412

String, purity 463
 StringCollection 40
 StringComparer 78, 318
 strings
 immutability 33
 reversing 25
 strongly typed collections 6,
 36, 40, 51, 293, 434
 .NET 1.1 40
 Stroustrup, Bjarne 100, 244
 struct
 automatically implemented
 properties 206
 keyword 43
 value type constraints 70
 structural comparisons 525
 structures 196
 style, query expressions and dot
 notation 319
 subclassing, reference types
 and value types 45
 subjective readability 204
 subsequences, grouping 312
 substitution of type parameters
 with type arguments 63
 suffixes
 attributes 198
 I..Contracts 468
 -suggest (command line
 option) 474
 Sum 272, 495
 dynamic
 implementation 416
 purity 463
 super awesome code 23
 supercomputers, string length
 computations 299
 surrogate pairs 25
 switch statements, iterator
 blocks 168
 Symbian S60 526
 SymmetricExceptWith 516
 symmetry 128
 synchronous delegate
 invocation 31
 synchronous service access 175
 SynonymInfo 383
 syntactic sugar, delegates 131
 syntax
 checking in dynamic
 languages 403
 consistency for queries in
 LINQ 16
 domain specific
 languages 274

options for queries 317
 query expressions 16
 System.AddIn 524
 System.Collections 40
 System.Collections.Concurrent
 518
 System.Collections.Generic 78,
 508
 System.Collections.ObjectModel
 513
 System.Collections.Specialized
 40
 System.Delegate 134
 System.Diagnostics.Contracts
 456
 System.Enum 70
 System.Interactive 351
 System.Linq, Parallel
 LINQ 347
 System.Linq.Expressions 237
 System.Math 187
 System.Nullable 111
 System.Numeric 525
 System.Reactive 351
 System.ValueType 70
 System.Xml.Linq 338

T

Table 325
 TableLayoutPanel 523
 tables
 cross joins 308
 LINQ to SQL 323
 navigating via joins 304
 table-valued functions 301
 Take 503
 TakeWhile 503
 target, delegates 30, 32
 Target, DLR cache 425
 Tatham, Simon 178
 team preferences 211, 219, 276
 consistency in
 formatting 141
 tedious coding 153, 226
 template
 metaprogramming 99
 templates 78, 91
 code generation 106
 temporary local variable, used
 for object initializers 213
 temporary values 212
 temporary variable, let
 clauses 326
 terminology 44, 198, 260
 ambiguity of "lazy" and
 "eager" 267
 CLR arrays and vectors 512
 covariance and
 contravariance 388
 dot notation and query
 expressions 317
 generic type constraints 73
 generics 62
 nullable type and nullable
 value type 113
 parameters and
 arguments 366
 variable classifications 145
 wrapping and
 unwrapping 108
 testable code 272
 test-first coding 484
 text encodings 171
 reading files 366
 text nodes, LINQ to XML 338
 TextReader 170–171
 ThenBy 270, 297, 507
 ThenByDescending 270, 297,
 507
 thinking in sequences 281
 third-party libraries 277
 changing contract
 behavior 472
 third-party LINQ providers 322
 this 145, 149
 anonymous methods 140
 constructor chaining 206
 declaring extension
 methods 260
 delegate target 30
 locking 399
 meta-object
 construction 448
 thread pool 147, 176
 thread safety 33, 205, 272, 399
 threading 147, 176, 346, 429
 immutability 108, 223
 LINQ to Rx 355
 ThreadPool (reactive
 extensions) 355
 threads 154, 175, 518
 robust locking 398
 thread-safety 508
 ThreadStart 133, 137, 144
 ThreadStatic 81
 throwaway code 205, 211
 timeouts 519
 times, LINQ to XML
 content 340

- TimeSpan 108, 417
 - timestamp 370
 - TimeZoneInfo 524
 - TIME_ZONE_INFORMATION 196
 - tlbimp. *See* Type Library Importer
 - ToArray 347, 497, 518
 - ToDictionary 317, 497
 - ToList 318, 497
 - ToLookup 497, 501
 - Tony the Pony 492
 - tools 183
 - Code Contracts 456, 464
 - tooltips 207
 - anonymous types 222
 - extension methods 261
 - ToString 48, 340, 440
 - anonymous types 223, 271
 - ExpandableObject 437
 - Nullable<T> 108
 - totals, calculating with LINQ 272
 - Toub, Stephen 519
 - transaction management LINQ to SQL 325
 - transformations
 - automatically implemented properties 205
 - LINQ sequences 281
 - translations
 - query continuations 314
 - query expressions 280, 287
 - transparency 526
 - transparent identifiers 298
 - implementation 300
 - inner joins 305
 - query expression readability 319
 - tree, LINQ to XML 338
 - trial and error 60
 - TrimExcess 510
 - triple quotes, Python string literals 412
 - trivial properties 205, 225–226
 - formatting 141
 - true operator 117
 - truth tables 119
 - try/catch and try/finally, in iterator blocks 161
 - TryAdd 518
 - TryGetMember 444
 - TryGetValue 509
 - TryInvokeMember() 444
 - TryParse 125
 - TryPeek 519
 - TryTake 518
 - TryXXX pattern 76, 125
 - DynamicObject 440, 443
 - T-SQL 323
 - Tuple 491, 525
 - Turing, Alan 115
 - two-phase type inference 248
 - type aliases 190
 - type arguments 62, 74, 98, 246, 391
 - dynamic 434
 - generic variance 395
 - GetVariable 413
 - preconditions 456
 - type constraints 69, 95
 - avoiding boxing 128
 - class 69
 - combining 73
 - conversion 71
 - new() 70
 - numeric 97
 - partial types 181
 - primary 73
 - secondary 73
 - struct 70
 - type parameter constraints 71
 - type declarations
 - dynamic 434
 - partial 181
 - type equivalence 526
 - type erasure 101, 397
 - type inference 228, 246, 287, 391, 491
 - anonymous types 222, 224
 - C# 2 74
 - dynamic code compile-time checking 432
 - generics 69, 80, 112
 - hiding variance workarounds 96
 - implicitly typed local variables 16, 38, 207, 408
 - query expressions 291
 - summary of changes in C# 3 254
 - two-phase (C# 3) 248
 - type initializers 206
 - Type Library Importer 385
 - type parameters 62, 89
 - comparisons with null 118
 - constraints 71
 - covariance 389
 - covariance and contravariance 95
 - dynamic 434
 - Func and Action delegate types 229
 - generic methods 65
 - generic variance 387
 - naming conventions 65
 - tooltips 207
 - Type property, expression trees 237
 - type safety 38, 59, 91–92, 106
 - compile time vs execution time 59
 - contravariance 96
 - generic variance 387
 - type system 36, 488
 - improving robustness 59
 - type variables, bounds 249
 - Type, System.Type 89
 - Type.GetTypeFromCLSID 386
 - Type.Missing 381
 - typedef 97
 - TypedReference 427
 - TypeIdentifier 386
 - typeless expressions 219
 - typeof 48, 63, 76, 88, 90, 242, 244
 - dynamic 435
 - nullable types 113
- ## U
-
- uint 455
 - unary operators 117
 - unbound generic types 62, 88
 - unboxing 47, 59, 106, 413, 510
 - avoiding in C# 1 85
 - choice of nullable or non-nullable target type 110
 - conversions 293
 - uncertainty, ? modifier 114
 - Unconstrained Melody 72
 - unconstrained type parameters 69
 - underlying type 107
 - unfixed type variables 249
 - unimplemented partial methods 185
 - uninitialized fields, default values 75
 - Union 506
 - UnionWith 516
 - unit testing 275, 396
 - frameworks 472
 - unit tests 184, 199, 357
 - contracts 469, 478, 483, 485

- unit tests (*continued*)
 - dynamic typing 403, 407, 432
 - sample data 218
- unknown values 14
- UnmanagedType.ByValArray 197
- unordered queries 349
- unprovable contracts 479
- unqualified names 190
- unreachable code 474
- unsafe code 36, 196, 408
- unsigned types 455
- unspeakable names 140, 205
 - invariants 460
 - iterators 167
- untyped variables 38
- unverifiable code 140
- unwrapping 116
 - nullable value types 108
- URI, XML namespaces 338
- URLs 43
 - comparison with references 104
- user groups 492
- user interfaces 180, 272
- user-defined conversions 116, 427
 - default parameter values 368
 - invalid in generic variance 395
 - Nullable 117
- user-defined operators 117
- Users, sample data model 285
- using directives 23, 128, 173, 190, 193, 433
 - extension methods 262
- using statements 170
 - implicitly typed local variables 209
 - iterator blocks 165
 - iterator disposal 358
- UTC 371
- UTF-16 25
- UTF-8 171, 366
- UTF8 (encoding property) 370
- utility code 52, 170, 186, 257, 274

V

- validation 189, 205, 377
 - arguments 173, 276, 357
 - iterator blocks 164, 174
 - explicit vs implicit 455
 - partial methods 185
- Value 107, 119
 - Excel 409
 - See also* Nullable, Value property
- value type constraints 70, 77
 - Nullable<T> 107
- value type conversions, invalid
 - in generic variance 395
- value types 14, 22, 42, 101
 - adding functionality 256
 - immutability 108
 - null values 104
 - nullability 53
 - Nullable<T> 107
 - satisfying constructor type constraints 70
 - type constraints 70
- ValueAtReturn 459
- var 16, 38, 51
 - anonymous types 221
 - comparison with dynamic 408
 - contextual keyword 207
 - projections 270
- variables
 - arrays 219
 - assigning values in expression trees 239
 - automatically implemented properties 7
 - backing trivial properties 205
 - behavior when captured 146
 - C++ template arguments 99
 - declarations for out parameters 124
 - implicit typing 16
 - location in memory 46
 - naming 482
 - pass-by-reference parameters 382
 - range variables 290
 - value type values 104
- variance 92
 - API design 396
 - C# 4 387
 - different flavors 135
 - nesting 394
- VARIANT 207, 404
- variant conversions 395
- variants 386
- VB. *See* Visual Basic
- VB.NET 23
 - See also* Visual Basic
- VB6 207
- vector<T> 100
- vectors 476
 - CLR terminology 512
- verification 454
- version numbers 521
- versioning 23, 101, 369, 371
 - COM 385
 - extension methods 277
- views 301
- virtual machine, Java 101
- virtual methods 37, 41, 180, 185, 403
 - Collection 513
 - contracts 467
 - partial types 183
- visibility, preconditions 457
- Visual Basic 64, 366, 380, 383, 445
 - COM support 19
 - VB 9 284
- Visual Studio 24, 193–194, 265, 323, 385, 456, 521
 - Code Contracts 482, 492
 - hovering for more information 207, 222, 261
- Visual Studio 2010 446
 - expression tree visualizer 242
 - Premium and Ultimate editions 456
- visualizer, expression trees 242
- void 448
 - return type of Action 229
- void return type, partial methods 186
- _VtblGap 386

W

- warnings 194, 491
 - breaking changes 138
 - hidden extension methods 263
 - nullable comparisons 118
 - pragma directives 194
- Watch window, Visual Studio 403
- WCF 523
- weak typing 36, 207
 - APIs 409
 - collections 40
- web pages, reference type analogy 43
- web services 19–20, 175, 243, 272, 284, 487

WebRequest 259
 WebResponse 259
 WF 524
 what versus how 210–211, 273
 Where 13, 173, 267, 282, 294,
 317, 505
 where clause
 combining 294
 contextual keyword 288
 type constraints 69
 while 157
 whitespace 141, 215
 sensitivity of Python 413
 wildcards, Java 101
 Windows 23
 Windows APIs 196
 Windows Azure 492
 Windows CardSpace 524
 Windows Communication
 Foundation 523
 Windows Forms 23, 108, 135,
 180, 523
 Windows Live ID 21
 Windows Mobile 526
 Windows Phone 7 526
 Windows Presentation
 Foundation 108, 513, 523
 Windows Workflow
 Foundation 524
 Wintellect 520
 WithCancellation 349
 WithDegreeOfParallelism 349

WithExecutionMode 350
 WithMergeOptions 350
 Word 380
 word processor 403
 workaround 397
 Worksheet, Excel 409
 WPF 513, 523
 wrapper classes 14
 wrapping, nullable value
 types 108
 WriteLine 407
 WriteTo 259

X

X, prefix for LINQ to XML
 types 338
 XAML 183, 527
 See also Extensible Applica-
 tion Markup Language
 XAttribute 338
 explicit conversion 343
 Xbox 360 526
 XCDATA 338
 XComment 338
 XContainer 338
 XDeclaration 338
 XDocument 338
 XElement 338, 436
 conversions 343
 XML 17, 218, 280, 337, 436
 comments 23

configuration files 414
 documentation 454, 480
 generated by
 cdocgen 456
 formatting of numbers and
 other data types 340
 XmlDocument 338
 XmlReader.Create, overload-
 ing complexity 367
 XmlReaderSettings 367
 XNA 526
 XName 338
 XNamespace 338
 XNode 338
 XObject 338
 XPath 342, 344
 XText 338
 conversion from object 340

Y

yield return 160, 164, 171
 restrictions 161
 yield statements 160
 yield type 161, 168

Z

zero 417
 Zip 504