

- (private) visibility indicator, meaning, 2
- # (protected) visibility indicator, meaning, 2
- + (public) visibility indicator, meaning, 2

A

- ABCJournalRecordFactory class, Factory Method pattern, 104
- Abstract Base Class pattern
 - abstract base classes, 61
 - class roles, 63–64
 - code example, 64–67
 - concrete classes, 64
 - dependencies, 64
 - IEnumerator interface, 61
 - implementation, 64
 - Interface and Abstract Class pattern and, 67
 - .NET Framework and, 64
 - Template Method pattern and, 67
- abstract base classes, behavior, 61
- Abstract Class pattern, Interface and Abstract Class pattern, 72
- abstract classes
 - extension, 99
 - Template Method pattern, 407
- abstract factory class
 - abstract class instances and, 109
 - instantiation and, 132
- Abstract Factory pattern
 - Bridge pattern and, 223
 - class roles, 111
 - Client role, 112
 - code example, 114–119
 - ConcreteFactory1 role, 112
 - ConcreteFactory2 role, 112
 - Factory Method pattern and, 107, 119
 - IFactory interface, 112
 - implementation, 113
 - instances of classes, 110
 - IWidgetA interface, 112
 - IWidgetB interface, 112
 - Kit, 109
 - object creation and, 94
 - Product1WidgetA role, 112
 - Product2WidgetA role, 112
 - Prototype pattern and, 139
 - Singleton pattern and, 119
 - Toolkit, 109
- Abstract Superclass, 61
- AbstractBaseclass, 63–64
- AbstractBuilder class, Builder pattern, 125
- AbstractCommandHandler class, Chain of Responsibility pattern, 297
- AbstractComposite class, Composite pattern, 177
- AbstractDoorControllerWrapper class, Decorator pattern, 266–267

- AbstractElement class, Visitor pattern, 419
- AbstractFlyweight class, 238
- Abstraction class, Bridge pattern, 216
- AbstractionImpl interface, Bridge pattern, 217
- abstractions
 - Bridge pattern, 213
 - hierarchies, combining, 216
 - implementations, creating, 216
 - logic reuse, 216
- AbstractLoadableClass class
 - Dynamic Linkage pattern, 248
 - subclasses, 249
- AbstractPullFilter class, Filter pattern, 167
- AbstractPushFilter class, Filter pattern, 169
- AbstractTemplate class, Template Method pattern, 409
- AbstractVisitor class, Visitor pattern, 419
- AbstractWrapper class, Decorator pattern, 268
- accessor methods, attributes, 35
- Action class, Hashed Adapter Objects pattern and, 431
- ActionKey class, Hashed Adapter Objects pattern and, 431
- active buffer, 500
- Active Object pattern
 - Asynchronous Processing pattern and, 527
 - Future pattern and, 540
- Adaptee classes, Adapter pattern, 197
- Adapter classes, Adapter pattern, 197
- adapter classes, interfaces, 195
- adapter objects, Hashed Adapter Objects pattern, 427
- Adapter pattern
 - Adaptee classes, 197
 - Adapter classes, 197
 - adapter classes, 195
 - Client class, 197
 - code example, 199–206
 - Facade pattern, 204, 231
 - Hashed Adapter Objects pattern and, 435
 - implementation, 198–199
 - Interface pattern, 59
 - ITarget interface, 197
 - Iterator pattern, 204, 211
 - Observer pattern and, 382
 - overview, 193
 - Proxy pattern, 204
 - Read-Only interface pattern, 192
 - Strategy pattern, 204, 400
- aggregations
 - class relationships, 8
 - composite, 9
- algorithms, encapsulation (Strategy pattern), 395–400
- application-independent objects, 97–98
- application-specific classes, creation, 96
- arbitrary objects, Hashed Adapter Objects pattern, 427
- ArrayCopier class, 195
- arrowheads, asynchronous calls, 20–21
- ASP.NET, Interface and Abstract Class pattern and, 70–71
- assemblies, packages and, 11
- associations
 - association names, 7
 - classes, 7
 - dependencies, 10
 - multiplicity indicators, 8
 - navigation arrows, 7
 - role names, 7
- assumptions, requirements and, 29
- asynchronous calls, arrowheads, 20–21
- asynchronous interactions, collaborative diagram, 19
- Asynchronous Processing pattern
 - Active Object pattern and, 527
 - code example, 525–527
 - Facade pattern and, 527
 - Future pattern and, 527, 540
 - implementation, 522–524
 - introduction, 438
 - .NET, 525
 - overview, 519
 - Producer-Consumer pattern and, 527
 - request management, 522–524
 - Scheduler pattern and, 527
 - thread allocation, 522–524
 - Thread Pool pattern and, 527

B

- Backus-Naur Form (BNF), tokens, 320
- Balking pattern
 - code example, 469–470
 - Guarded Suspension pattern and, 465, 470
 - implementation, 469
 - introduction, 438
 - Mediator pattern and, 354
 - Single Threaded Execution pattern and, 470
 - state and, 467–470
- batching notifications, Observer pattern, 378
- behavioral patterns
 - categories, 293
 - Chain of Responsibility pattern, 295–304
 - Command pattern, 305–315
 - Hashed Adapter Objects pattern, 427–435
 - Little Language pattern, 317–342
 - Mediator pattern, 343–354
 - Null Object, 401–405
 - Observer pattern, 373–382
 - Snapshot pattern, 355–372
 - State pattern, 383–394
 - Strategy pattern, 395–400
 - Template Method, 407–413
 - Visitor, 415–425
- behaviors
 - delegation and, 51
 - IEnumerator interface, 61
 - operations, 1
 - Strategy pattern and, 396
- Bentley, Jon, 317
- BNF (Backus-Naur Form), tokens, 320
- Both Dirty state, 385
- Bridge pattern
 - Abstract Factory pattern and, 223
 - Abstraction class, 216
 - abstraction hierarchies, 213
 - AbstractionImpl interface, 217
 - code example, 218–223
 - Decorator pattern and, 223
 - Factory Method object, 219
 - hierarchy extension, 218
 - implementation, 218

- Layered Architecture pattern and, 223
- .NET Framework, 218
- overview, 193
- SpecializedAbstraction class, 216
- SpecializedAbstractionImpl interface, 217

buffers

- active, 500
- Double Buffering pattern and, 502–503
- reserve, 500

Builder pattern

- AbstractBuilder class, 125
- client classes, 125
- code example, 128–129
- Composite pattern and, 129
- Concrete Builder class, 125
- consequences, 127
- Director object, 125
- external data representation, 124
- Factory Method pattern and, 129
- implementation, 126–127
- Interface pattern and, 129
- IProduct interface, 124
- .NET Framework, 128
- Null Object pattern and, 129
- object creation and, 94
- overview, 121
- Product class, 124
- Strategy pattern and, 129
- Template Method pattern and, 129
- Visitor pattern and, 129

C

- cache, object limits, 280
- Cache class, Cache Management pattern, 275
- Cache Management pattern
 - ArrayList, 278
 - Cache class, 275
 - CacheManager class, 275
 - Client classes, 275
 - code example, 282–292
 - Ephemeral Cache Item pattern and, 292
 - Facade pattern and, 292
 - Flyweight pattern and, 244

- Cache Management pattern (*continued*)
 - GetHashCode() method, 281
 - Hashtable, 279
 - hit rate, 277
 - implementation, 275–281
 - object access, 273
 - Object Pool pattern and, 161
 - Object Replication pattern and, 292
 - ObjectCreator class, 275
 - ObjectKey class, 275
 - Optimistic Concurrency pattern and, 292
 - overview, 193
 - performance tuning and, 277–281
 - prefetching objects, 277
 - ProductCacheManager class and, 274
 - ProductInfoFetcher object, 274
 - read consistency, 282
 - Singleton pattern and, 147
 - Template Method pattern and, 292
 - Virtual Proxy pattern and, 292
 - write consistency, 282
- CacheManager class
 - Cache Management pattern, 275
 - hit rate and, 280
 - ObjectCreator class and, 275
- Caretaker class, Snapshot pattern, 360
- CAS (code access security), Dynamic Linkage Pattern and, 249
- case studies
 - business case in deployment, 28–29
 - essential use cases, 30–32
 - object-oriented analysis, 32–34
 - object-oriented design, 34–43
 - requirements for deployment, 29–30
- Chain of Responsibility pattern
 - AbstractCommandHandler class, 297
 - behavior and, 293
 - code example, 299–304
 - Command pattern and, 304
 - commands, 295
 - CommandSender class, 297
 - Composite pattern and, 185, 304
 - ICommandHandler interface, 297
 - implementation, 297–298
 - .NET API, 299
 - object coupling, 299
 - Template Method pattern and, 304
- class diagrams
 - classes and, 6
 - definition, 2
 - interfaces and, 6
 - objects, 12
- classes
 - ABCJournalRecordFactory, 104
 - Abstract Base Class pattern, 63–64
 - abstract factory class, 109
 - AbstractBaseclass, 63–64
 - AbstractPushFilter, 169
 - adapter classes, 195
 - aggregations, 8
 - application-specific, 96
 - ArrayCopier, 195
 - associations, 7
 - attributes, 1
 - class diagrams and, 6
 - Client, 56
 - Client, Object Pool pattern, 153
 - compartments, 4
 - compatibility, Dynamic Linkage pattern and, 248
 - concrete, Abstract Base Class pattern, 64
 - ConcretePushFilter, 169
 - constructors, interfaces and, 56
 - CSharpCodeProvider, 128
 - data-driven, Factory Method pattern, 100–101
 - data filter classes, 166
 - delegation and, 47
 - DoubleLinkedList, 71–72
 - ellipsis (...), 3
 - Facade pattern, 225
 - factory method, 132
 - Game Snapshot, 356
 - hiding, 69
 - inheritance, 47
 - LazyCloneDictionary, 86–87
 - loading, deferred, 259–260
 - loading arbitrary, 245–253
 - Mediator pattern, 345
 - nested, 11
 - proxy objects, 81
 - RecordFactoryFactory, 104
 - rectangles, 2
 - relationships, 7

- SaleLineItem, 104
- simplified, 4
- Singleton pattern, 142
- StartOfSale, 104
- State pattern, 386–388
- two-compartment classes, 4
- UIInterface, 356
- VBCodeProvider, 128
- word combination language, 327
- wrapper classes, 121
- WrapperGenerator, 123–124
- XYZJournalRecordFactory, 104
- Client class
 - Abstract Factory pattern, 112
 - Adapter pattern, 197
 - Builder pattern, 125
 - Cache Management pattern, 275
 - concrete widget classes and, 113
 - Future pattern, 531
 - Hashed Adapter Objects pattern and, 431
 - IFactory interface and, 113
 - Indirection interface, 56
 - Little Language pattern, 328
 - Object Pool pattern, 153
 - problem domain and, 50
 - Prototype pattern, 132
 - Strategy pattern, 396
 - Virtual Proxy pattern, 257–258
 - Visitor pattern and, 418
- client objects, complex object creation,
 - Builder pattern, 121–129
- Clone function
 - deep copies, 84, 133
 - shallow copies, 84, 133
- code example
 - Abstract Base Class pattern, 64–67
 - Abstract Factory pattern, 114–119
 - Adapter pattern, 199–206
 - Asynchronous Processing pattern, 525–527
 - Balking pattern, 469–470
 - Bridge pattern, 218–223
 - Builder pattern, 128–129
 - Cache Management pattern, 282–292
 - Chain of Responsibility pattern, 299–304
 - Command pattern, 310–315
 - Composite pattern, 180–185
 - Decorator pattern, 270–271
 - delegation example, 52–54
 - Double Buffering pattern, 504–518
 - Dynamic Linkage pattern, 250–253
 - Façade pattern, 228–231
 - Factory Method pattern, 102–107
 - Filter pattern, 170–174
 - Flyweight pattern, 239–244
 - Future pattern, 536–540
 - Guarded Suspension pattern, 464–465
 - Hashed Adapter Objects pattern, 433–435
 - Immutable pattern, 78–79
 - Interface and Abstract Class pattern, 71–72
 - Interface pattern example, 58–59
 - Iterator pattern, 209–211
 - Little Language pattern, 331
 - Lock Object pattern, 457–458
 - Mediator pattern, 349–354
 - Null Object pattern, 404–405
 - Object Pool pattern, 156–161
 - Observer pattern, 379–381
 - Producer-Consumer pattern, 496–498
 - Prototype pattern, 135–139
 - Proxy pattern, 83–91
 - Read-Only interface pattern, 190–192
 - Read/Write Lock pattern, 487–492
 - Scheduler pattern, 476–481
 - Single Threaded Execution pattern, 444–446
 - Singleton pattern, 146–147
 - Snapshot pattern, 370–371
 - State pattern, 389–394
 - Static Locking Order pattern, 450–451
 - Strategy pattern, 398–399
 - Template Method pattern, 410–413
 - Virtual Proxy pattern, 261–263
 - Visitor pattern, 422–425
- coding, deployment and, 27
- collaboration diagram
 - example, 14
 - interactions, 14
 - links, 14
- collaborations, 14
- Colleague classes, Mediator pattern, 349
- Collection class, Iterator pattern and, 207

- Collection interface, Iterator pattern and, 207
- collections
 - access, Iterator pattern, 206
 - object modification during iteration, 208
 - remote, access, 206
- Command pattern
 - behavior and, 293
 - Chain of Responsibility pattern and, 304
 - code example, 310–315
 - CommandManager class, 306
 - commands, encapsulating, 305
 - ConcreteCommand class, 306
 - do function, 305
 - Factory Method pattern and, 315
 - ICommand interface, 306
 - implementation, 307–309
 - interface dependencies, 308–309
 - Invoker class, 306
 - Little Language pattern and, 315
 - Marker Interface pattern and, 315
 - .NET, 310
 - Snapshot pattern and, 315, 372
 - Template Method pattern and, 315
 - undo function, 305
- CommandManager class
 - Command pattern, 306
 - objects, undo/redo, 307–308
- CommandSender class, Chain of Responsibility pattern, 297
- comments, Delegation pattern, 52
- compartments, classes, 2, 4
- compatibility of classes, Dynamic Linkage pattern and, 248
- composite aggregations, classes, 9
- Composite pattern
 - AbstractComposite class, 177
 - Builder pattern and, 129
 - Chain of Responsibility pattern and, 185, 304
 - code example, 180–185
 - Element1 class, 177
 - Filter pattern and, 174
 - Flyweight pattern and, 244
 - Frame objects, 175
 - hierarchy and, 177
 - IElement interface, 177
 - implementation, 179
 - introduction, 163
 - Little Language pattern and, 342
 - .NET Framework, 180
 - Page objects, 175
 - Prototype pattern and, 139
 - recursion, 175
 - Recursive Composition pattern, 175
 - Visitor pattern and, 185, 425
- Concrete Builder class, Builder pattern, 125
- concrete classes
 - Abstract Base Class pattern, 64
 - interface implementation, 109
 - widget classes, client classes and, 113
- ConcreteCommand class, Command pattern, 306
- ConcreteLoadableClass class, Dynamic Linkage pattern and, 248
- ConcretePullFilter class, Filter pattern, 168
- ConcretePushFilter class, Filter pattern, 169
- ConcreteService class, Decorator pattern, 268
- ConcreteTemplate class, Template Method pattern, 409
- concurrency patterns
 - Asynchronous Processing, 519–527
 - Balking pattern, 467–470
 - Double Buffering pattern, 499–518
 - Future pattern, 529–540
 - Guarded Suspension pattern, 459–465
 - Lock Object pattern, 453–458
 - Producer-Consumer pattern, 493–498
 - Read/Write Lock pattern, 483–492
 - Scheduler pattern, 471–481
 - sequence of operations and, 437
 - shared resources and, 437
 - Single Threaded Execution, 439–446
 - Static Locking Order, 447–451
- connections
 - dynamic, 165–174
 - IDbConnection interface and, 149–150
- constructors
 - Immutable pattern and, 77
 - interfaces, 56, 57–58
 - .NET Framework, 57–58

- singleton classes, 143
 - wrapper classes, 121
 - Consumer class, Producer-Consumer pattern, 495
 - Context class, State pattern, 387
 - Controller pattern, Mediator pattern and, 354
 - copy, serialization and, 144
 - creational patterns
 - Abstract Factory pattern, 94, 109–119
 - Builder pattern, 94, 121–129
 - Factory Method pattern, 94, 95–108
 - object creation and, 93
 - Object Pool pattern, 94
 - Prototype pattern, 94, 131–139
 - Singleton pattern, 94, 141–147
 - CreationRequester role class, Factory Method pattern, 98
 - CSharpCodeProvider class, Builder pattern, 128
- D**
- data analyses
 - data filter classes, 166
 - implementation, 166
 - data-driven classes, Factory Method pattern and, 100–101
 - data filter classes
 - analysis, 166
 - transformations, 166
 - data structure
 - Hashed Adapter Objects pattern, 433
 - object pool, 155
 - data transformations. *See* transformations
 - deadlocked threads
 - Single Threaded Execution pattern and, 443
 - Static Locking Order pattern, 447–449
 - deadly embrace, threads (Single Threaded Execution pattern), 444–446
 - decorator classes, implementation, 121–122
 - Decorator pattern
 - AbstractWrapper class, 268
 - Bridge pattern and, 223
 - code example, 270–271
 - ConcreteService class, 268
 - delegation, 54
 - Delegation pattern and, 272
 - Filter pattern and, 174, 272
 - IAbstractService interface, 268
 - implementation, 269
 - inheritance, 269
 - object functionality, 265
 - overview, 193
 - Prototype pattern and, 139
 - Proxy pattern and, 91
 - Strategy pattern and, 272
 - Template Method pattern and, 272
 - wrapper classes, 121
 - Wrapper pattern, 265
 - deep copies, objects
 - Clone function, 84
 - Prototype pattern, 133
 - deferred class loading, Virtual Proxy pattern, 259–260
 - delegation of object creation, Object Pool pattern, 155
 - Delegation pattern
 - behaviors and, 51
 - behaviors and, run time, 52
 - classes and, 47
 - code example, 52–54
 - comments, 52
 - Decorator pattern and, 54, 272
 - disadvantages, 52
 - implementation, 51
 - indirect, 52
 - indirect delegations, 52
 - inheritance and, 47–54
 - Interface pattern and, 59
 - .NET and, 52
 - Observer pattern and, 382
 - Proxy pattern and, 54
 - roles, 48
 - Delegator class, Null Object pattern, 403
 - dependencies
 - Abstract Base Class pattern, 64
 - dialog boxes, Mediator pattern, 344–345
 - dynamically registering, 373
 - Façade pattern, 38, 226
 - interfaces, Command pattern, 308–309
 - logic, Mediator pattern, 349
 - minimizing, 38

- dependencies (*continued*)
 - Observer pattern, 373
 - requirements and, 29
 - state-related, Mediator pattern classes, 345–346
 - dependent objects, classes, 56
 - deployment
 - activities leading up to, 26
 - coding and, 27
 - essential use cases, 27
 - high-level system architecture, 27
 - object-oriented analysis and, 27
 - object-oriented design, 27
 - planning, 26
 - prototype creation, 27
 - requirements, 26–27
 - testing and, 27
 - Deserializer class, Snapshot pattern, 358
 - dialog boxes, Mediator Pattern and, 343–345
 - Director object, Builder pattern, 125
 - DirectShow API, filters, 170
 - do function, Command pattern, 305
 - DocChar class, Flyweight pattern, 234
 - documentation, IEnumerator interface, 61
 - DocumentContainer class, Flyweight pattern, 234
 - DocumentManager object, Factory Method pattern and, 95–97
 - DocumentVisitor class, Visitor pattern, 417
 - Double Buffering pattern
 - code example, 504–518
 - DoubleBufferedStream class, 499
 - exception handling, 503
 - Guarded Suspension pattern and, 518
 - implementation, 502–503
 - introduction, 438
 - multiple buffers, 502
 - .NET, 504
 - Producer-Consumer pattern and, 518
 - threads, 502–503
 - DoubleLinkedList class, 71–72
 - doubly linked lists, inserting/deleting objects, 73
 - dynamic connections, Filter pattern, 165
 - dynamic linkage, instantiation and, 132
 - Dynamic Linkage pattern
 - AbstractLoadableClass class and, 248
 - arbitrary classes, loading, 245
 - ConcreteLoadableClass class, 248
 - Environment class, 247
 - IEnvironment interface, 247
 - implementation, 248–249
 - loading time, 250
 - .NET Framework and, 250
 - overview, 193
 - Protection Proxy pattern and, 253
 - security and, 249
 - Virtual Proxy pattern and, 253
 - dynamic registration of dependencies, 373
- E**
- ellipsis (...), classes and, 3
 - encapsulation, algorithms (Strategy pattern), 395–400
 - Environment class, Dynamic Linkage pattern and, 247
 - Ephemeral Cache Item pattern, Cache Management pattern and, 292
 - essential use cases
 - case study, 30–32
 - deployment and, 27
 - events
 - recursive, Mediator pattern, 348
 - state and, 22
 - exception handling, Double Buffering pattern and, 503
 - execution order of threads, Schedule pattern and, 471–481
 - execution suspension, Guarded Suspension pattern, 459–465
 - Expect subroutine, Parser class, 337
 - external data representation, Builder pattern and, 124
- F**
- Façade pattern
 - Adapter pattern and, 204, 231
 - Asynchronous Processing pattern and, 527
 - Cache Management pattern and, 292
 - code example, 228–231

- dependencies and, 38, 226
- implementation, 227–228
- Interface pattern and, 231
- .NET Framework, 228
- object access, 225
- Object Pool pattern and, 161
- overview, 193
- Prototype pattern and, 139
- Proxy pattern and, 91
- Virtual Proxy pattern and, 263
- wrapper classes, 121
- Factory class
 - Factory Method pattern, 99
 - journal-file-related, 103
- factory method classes, instantiation and, 132
- Factory Method object, Bridge pattern, 219
- Factory Method pattern
 - ABCJournalRecordFactory class, 104
 - abstract class extension, 99
 - Abstract Factory pattern and, 107, 119
 - Builder pattern and, 129
 - class determination, data-driven, 100–101
 - Command pattern and, 315
 - consequences, 101
 - CreationRequester class, 98
 - Factory class, 99
 - Flyweight pattern and, 244
 - Hashed Adapter Objects pattern and, 107
 - IFactory interface, 98–99
 - IJournalRecord interface, 104
 - IJournalRecordFactory interface, 104
 - implementation, 99
 - IProduct interface, 98
 - Iterator pattern and, 211
 - .NET Framework, 101–102
 - object creation and, 94
 - Object Pool pattern and, 161
 - overview, 95
 - Prototype pattern and, 108, 139
 - RecordFactoryFactory class, 104
 - StartOfSale class, 104
 - Strategy pattern and, 108
 - Template Method pattern and, 108
 - XYZJournalRecordFactory class, 104
- File Dirty state, 385
- FileStream class, Snapshot pattern, 358
- Filter pattern
 - AbstractPullFilter class, 167
 - AbstractPushFilter class, 169
 - code example, 170–174
 - Composite pattern and, 174
 - ConcretePullFilter class, 168
 - ConcretePushFilter class, 169
 - connecting objects dynamically, 165–174
 - data analyses, 166
 - data transformations, 166
 - Decorator pattern and, 174, 272
 - implementation, 169
 - introduction, 163
 - ISink interface, 169
 - ISource interface, 167
 - .NET Framework, 170
 - Pipe pattern and, 174
 - pull filters, 166
 - Sink class, 168, 169
 - Source class, 167
 - state and, 170
- Flyweight pattern
 - AbstractFlyweight class, 238
 - Cache Management pattern and, 244
 - code example, 239–244
 - Composite pattern and, 244
 - DocChar class, 234
 - DocumentContainer class, 234
 - Factory Method pattern and, 244
 - FlyweightFactory class, 238
 - IDocumentElement interface, 234
 - Immutable pattern and, 244
 - implementation, 238
 - instances, multiple, 233
 - .NET Framework and, 239
 - objects, shared, 239
 - overview, 193
 - SharedConcreteFlyweight class, 238
 - State pattern and, 394
 - Strategy pattern and, 400
 - UnsharedConcreteFlyweight class, 238
- FlyweightFactory class, 238
- For Each statement (.NET), Iterator pattern and, 206
- formal parameters, 3
- Frame objects, Composite pattern, 175

function calls, Single Threaded Execution pattern, 439

functions

Clone, 84

interfaces and, 196

Iterator pattern, 208

multiple parameters, 3

subroutines and, 1

threads, multiple, 18

visibility indicators, 2

Future class, Future pattern, 531

Future pattern

Active Object pattern and, 540

Asynchronous Processing pattern and, 527, 540

Client class, 531

code example, 536–540

encapsulation, 529

exceptions, 534

Future class, 531

implementation, 532–534

introduction, 438

.NET, 534–535

Observer pattern and, 540

Promise pattern, 529

Proxy pattern and, 533, 540

rendezvous, 534

Requester class, 531

Result class, 531

synchronous computation launch, 533

Virtual Proxy pattern and, 540

G

Game Snapshot classes, 356

GameModel class, Snapshot pattern, 357

generics, Iterator pattern, 207

GetClassname method, singleton classes, 143

GetHashCode() function

Cache Management pattern, 281

Hashed Adapter Objects pattern, 432

GetInstance method

concurrent calls, 144–145

singleton classes, 143

grammar, language

definition, 317

tokens, 320

Guarded Suspension pattern

Balking pattern, 465

Balking pattern and, 470

code example, 464–465

Double Buffering pattern and, 518

execution suspension, 459–465

implementation, 461–463

introduction, 438

.NET, 464

Producer-Consumer pattern and, 498

H

hash tables, data structure, 432–433

Hashed Adapter Objects pattern

Action class, 431

ActionKey class, 431

adapter objects, 427

Adapter pattern and, 435

arbitrary objects, 427

behaviors and, 293

Client class, 431

code example, 433–435

data structures, alternate, 433

Factory Method pattern and, 107

GetHashCode() function, 432

hash table data structure, 432–433

IAction interface, 431

implementation, 432–433

Lookup Table pattern and, 435

Polymorphism pattern and, 435

Single Threaded Execution pattern and, 435

Strategy pattern and, 435

Hashtable, Cache Management pattern, 279

hierarchy

Composite pattern and, 177

extending, Bridge pattern, 218

high-level system architecture, deployment, 27

hit rate, cache manager and, 277

I

IAbstractService interface, Decorator pattern, 268

IAction interface, Hashed Adapter Objects pattern and, 431

- IBuilder interface, Builder pattern, 125
- ICloneable interface, 84, 131
- ICommand interface, Command pattern, 306
- ICommandHandler interface, Chain of Responsibility pattern, 297
- ICopyFilter interface, 195
- IDbConnection interface, data source connections, 149–150
- IDescription interface
 - browser object, 55
 - code example, 58–59
- IDictionary object, cloning, 85
- IDocumentElement interface, Flyweight pattern, 234
- IElement interface, Composite pattern, 177
- IEnumerable interface, Iterator pattern and, 207
- IEnumerator interface
 - behavior implementation, 61
 - documentation, 61
 - Iterator pattern and, 207
- IEnvironment interface, Dynamic Linkage pattern and, 247
- IFactory interface
 - Abstract Factory pattern, 112
 - client classes and, 113
 - Factory Method pattern, 98–99
- IFormatter interface, Snapshot pattern, 361
- IIndirection interface, Client class, 56
- IJournalRecord interface, Factory Method pattern, 104
- IJournalRecordFactory interface, Factory Method pattern, 104
- IMemento interface, Snapshot pattern, 360
- IMilestoneMemento interface, Snapshot pattern, 357
- Immutable pattern
 - code example, 78–79
 - constructors and, 77
 - Flyweight pattern and, 244
 - implementation, 77
 - instance variables, 77
 - .NET and, 78
 - Read-Only interface pattern and, 79
 - robustness and, 75
 - shared references and, 75
- Single Threaded Execution pattern and, 79
- state information, 75, 76
- String class and, 78
- threads, 76
- value objects and, 75, 76
- implementation, Read/Write Lock pattern, 486
- implementation
 - Abstract Base Class pattern, 64
 - Abstract Factory pattern, 113
 - Adapter pattern, 198–199
 - Asynchronous Processing pattern, 522–524
 - Balking pattern, 469
 - Bridge pattern, 218
 - Builder pattern, 126–127
 - Cache Management pattern, 275–281
 - Chain of Responsibility pattern, 297–298
 - Command pattern, 307–309
 - Composite pattern, 179
 - decorator classes, 121–122
 - Decorator pattern, 269
 - delegation, 51
 - Double Buffering pattern, 502–503
 - Dynamic Linkage pattern, 248–249
 - Façade pattern, 227–228
 - Filter pattern, 169
 - Flyweight pattern, 238
 - Future pattern, 532–534
 - Guarded Suspension pattern, 461–463
 - Hashed Adapter Object pattern, 432–433
 - Immutable pattern, 77
 - Iterator pattern, 207–209
 - Little Language pattern, 329–330
 - Lock Object pattern, 455–456
 - Mediator pattern, 347–348
 - Null Object pattern, 403
 - Object Pool pattern, 154–156
 - Observer pattern, 376–378
 - parser, recursive descent, 334
 - Producer-Consumer pattern, 496
 - Prototype pattern, 133–134
 - Proxy pattern, 83
 - Read-Only interface pattern, 190–192

- implementation (*continued*)
 - Scheduler pattern, 476
 - Single Threaded Execution pattern, 442–443
 - Singleton pattern, 143–145
 - Snapshot pattern, 363–370
 - State pattern, 388–389
 - Static Locking Order pattern, 449
 - Strategy pattern, 397
 - Template Method pattern, 410
 - Virtual Proxy pattern and, 259–260
 - Visitor pattern, 421
- indirect delegations, intermediate classes and, 52
- inheritance
 - classes and, 47
 - Decorator pattern, 269
 - delegation, 47–54
 - inappropriate use, 51
 - roles, 48
- INonterminal interface, Little Language pattern, 328
- instance variables, Immutable pattern and, 77
- instances
 - Abstract Factory pattern and, 110
 - Immutable pattern and, 76
 - interfaces and, 55
 - loaded classes, 247
 - maximum number, Object Pool pattern, 155
 - modification, 189
 - multiple, singleton classes, 143
 - Object Pool pattern, 151
 - object reuse, 151
 - singleton classes, 145
 - Strategy pattern, 396
- instantiation
 - abstract factory classes, 132
 - dynamic linkage and, 132
 - factory method classes and, 132
 - singleton classes, 143
 - Singleton pattern and, 141–147
 - Virtual Proxy pattern, 255–263
- interactions
 - asynchronous, 19
 - collaborations, 14
 - collaborative diagrams, 14
 - multilevel sequence numbers, 15
 - preconditions, 20
 - repeated, 16
 - sequence numbers, 14
 - sequence numbers, multilevel, 15
- Interface and Abstract Class pattern
 - Abstract Base Class pattern and, 67
 - Abstract Class pattern and, 72
 - ASP.NET and, 70–71
 - code example, 71–72
 - Interface pattern and, 72
 - overview, 69
- Interface pattern
 - Adapter pattern, 59
 - Builder pattern and, 129
 - code example, 58–59
 - constructors, 57–58
 - delegates, 57
 - Delegation pattern and, 59
 - dependent objects, 56
 - Façade pattern and, 231
 - IDescription, 55
 - IDescription interface, 55
 - IIndirection interface, 56
 - implementation, 56–58
 - instances and, 55
 - Interface and Abstract Class pattern, 72
 - .NET delegates, 57
 - Read-Only interface pattern, 192
 - Reference Interface, 55
 - Strategy pattern, 59
- interfaces
 - class diagrams and, 6
 - compartments, 5
 - constructors, 56, 57–58
 - dependencies, Command pattern, 308–309
 - function calls, 196
 - IAbstractService, 268
 - IAction, 431
 - IBuilder interface, 125
 - ICloneable interface, 84
 - ICommandHandler, 297
 - ICopyFilter, 195
 - IDbConnection, 149–150
 - IDocumentElement, 234

- IEnvironment, 247
 - IFactory interface, 98–99
 - IFormatter, 361
 - IJournalRecord, 104
 - IJournalRecordFactory, 104
 - IMilestoneMemento, 357
 - implementation, concrete classes and, 109
 - INonterminal, 328
 - IObservable, 375
 - IObserver, 375
 - IOperation, 403
 - IReadOnly interface, 190
 - IScheduleOrdering, 474
 - ISerializable, 365
 - IService, 259
 - ISink, 169
 - IStrategy, 396
 - ITarget interface, 197
 - Mediator pattern, 345
 - stereotype, 5
 - subroutine calls, 196
 - internal ToC table, Visitor pattern, 415
 - Invoker class, Command pattern, 306
 - IObservable interface, 376–377
 - IObserver interface, Observer pattern, 375
 - IOperation interface, Null Object pattern, 403
 - IProduct interface, Builder pattern, 124
 - IPrototype interface, Prototype pattern and, 132
 - Iprototype interface, Prototype pattern and, 133
 - IReadOnly interface, Read-Only interface pattern, 190
 - is-a-part-of relationships, 36
 - is-a relationships, 35
 - IScheduleOrdering interface, 474
 - ISerializable interface, 365
 - IService interface, Virtual Proxy pattern, 259
 - ISink interface, Filter pattern, 169
 - IStrategy interface, Strategy pattern, 396
 - ITarget interface, Adapter pattern, 197
 - Iterator class, Iterator pattern and, 207
 - Iterator pattern
 - Adapter pattern, 204, 211
 - code example, 209–211
 - collection access, 206
 - Collection class, 207
 - Factory Method pattern, 211
 - functions, 208
 - generics, 207
 - ICollection interface, 207
 - IEnumerable interface, 207
 - IEnumerator interface, 207
 - implementation, 207–209
 - Iterator class, 207
 - multiple orderings, 208
 - .NET Framework, 209
 - null iterator, 208
 - Null Object pattern, 211
 - overview, 193
 - sequential access, 205
 - Visitor pattern and, 425
 - IWidgetA interface, Abstract Factory pattern, 112
 - IWidgetB interface, Abstract Factory pattern, 112
- L**
- languages, 317. *See also* Little Language pattern
 - combinations of words, 318
 - lexical rules for word combination language, 326
 - parse tree, 323
 - precedence rule, 319
 - regular expression, 325
 - semantics, 317
 - syntax, 317
 - tokens, grammar, 320
 - white space, 324
 - Layered Architecture pattern
 - Bridge pattern and, 223
 - Object Pool pattern and, 161
 - lazy instantiation, Virtual Proxy pattern and, 255
 - LazyCloneDictionary class, 86–87
 - lexical rules for word combination language, 326

- LexicalAnalyzer, Little Language pattern, 328
 - links, collaborative diagrams, 14
 - Little Language pattern
 - behavior and, 293
 - Client class, 328
 - code example, 331
 - combinations of words, 318
 - Command pattern and, 315
 - Composite pattern and, 342
 - implementation, 329–330
 - INonterminal interface, 328
 - Interpreter pattern and, 317
 - lexical rules for word combination language, 326
 - LexicalAnalyzer, 328
 - little languages (Jon Bentley), 317
 - .NET, 331
 - Parser class, 328
 - reserved words, 318
 - TerminalToken class, 329
 - tokens, grammar, 320
 - Visitor pattern and, 342, 425
 - word combination language classes, 327
 - WordCombination class, 339
 - Lock Object pattern
 - code example, 457–458
 - implementation, 455–456
 - introduction, 438
 - Object Pool pattern and, 161
 - Single Threaded Execution pattern and, 458
 - Static Locking Order pattern and, 451, 458
 - threads, object locks and, 454
 - lock objects, threads, 454
 - logic, Visitor pattern, 415–425
 - Lookup Table pattern, Hashed Adapter Objects pattern and, 435
 - Low Coupling/High Cohesion pattern, Mediator pattern and, 354
- M**
- Marker Interface pattern, Command pattern and, 315
 - Mediator pattern
 - Balking pattern and, 354
 - classes, 345
 - code example, 349–354
 - Colleague classes, 349
 - Controller pattern and, 354
 - dialog boxes, dependencies, 344–345
 - dialog boxes and, 343–344
 - implementation, 347–348
 - interfaces and, 345
 - Low Coupling/High Cohesion pattern and, 354
 - Mediator object, 346–347
 - Observer pattern and, 354, 382
 - recursive events, 348
 - state and, 343
 - State pattern and, 394
 - White Box Testing pattern and, 354
 - MemberwiseClone function, 84
 - Memento class, Snapshot pattern, 360
 - Memento objects
 - serialization comparison, 362
 - Snapshot pattern, 359–360
 - state, 362
 - Memento pattern, Snapshot pattern, 355
 - MilestoneMemento class, Snapshot pattern, 357
 - MilestoneMementoManager class, Snapshot pattern, 357
 - Multicaster class, Observer pattern and, 376, 377
 - multilevel sequence numbers, 15
 - interactions, 15
 - multiobjects, 15
 - multiplicity indicators, associations, 8
 - Mutable class, Read-Only interface pattern, 190
 - MutatorClient class, Read-Only interface pattern, 190
- N**
- names, association names, 7
 - namespaces, packages and, 11
 - navigation arrows, associations, 7
 - nested classes, 11
 - .NET
 - Abstract Base Class pattern and, 64
 - Asynchronous Processing pattern, 525
 - Bridge pattern, 218
 - Builder pattern, 128
 - Chain of Responsibility pattern, 299

- Command pattern, 310
 - Composite pattern, 180
 - constructors, 57–58
 - delegate, stereotype, 5
 - delegation and, 52
 - Double Buffering pattern and, 504
 - Dynamic Linkage pattern, 250
 - Façade pattern, 228
 - Factory Method pattern and, 101–102
 - Filter pattern, 170
 - Flyweight pattern, 239
 - Future pattern, 534–535
 - Guarded Suspension pattern, 464
 - Immutable pattern and, 78
 - Iterator pattern, 209
 - Little Language pattern, 331
 - Observer pattern, 379
 - Producer-Consumer pattern, 496
 - Prototype pattern, 135
 - Read/Write Lock pattern, 487
 - Singleton pattern, 145
 - Strategy pattern, 397
 - .NET delegates, Interface pattern, 57
 - non-terminal tokens, language, 320
 - Not Dirty state, 384
 - notifications, batching, Observer pattern, 378
 - null iterator, Iterator pattern, 208
 - Null Object pattern
 - Builder pattern and, 129
 - code example, 404–405
 - Delegation class, 403
 - implementation, 403
 - IOperation interface, 403
 - Iterator pattern and, 211
 - NullOperation class, 403
 - object absence, 401
 - RealOperation class, 403
 - Singleton pattern and, 405
 - Strategy pattern and, 400, 405
 - NullOperation class, Null Object pattern, 403
- O**
- object diagrams, example, 13
 - object-oriented analysis
 - case study, 32–34
 - deployment and, 27
 - object-oriented design
 - case study, 34–43
 - deployment, 27
 - object pool
 - hiding, 154–155
 - size limit, 155–156
 - stateful objects, 156
 - structure, 155
 - Object Pool pattern
 - Cache Management pattern and, 161
 - Client class, 153
 - code example, 156–161
 - Façade pattern and, 161
 - Factory Method pattern and, 161
 - implementation, 154–156
 - instances, 151
 - instances, maximum number, 155
 - Layered Architecture pattern and, 161
 - Lock Object pattern and, 161
 - object creation, 94, 156
 - object reuse, 149–161
 - Reusable object, 153
 - ReusablePool object, 154
 - Singleton pattern and, 147, 161
 - Thread Pool pattern and, 161
 - Virtual Proxy pattern and, 263
 - Object Replication pattern, Cache Management pattern and, 292
 - Object Request Broker pattern, Proxy pattern and, 91
 - object reuse
 - instances, 151
 - Object Pool pattern, 149–161
 - ObjectCreator class, Cache Management pattern, 275
 - ObjectKey class, Cache Management pattern, 275
 - objects
 - cache limits, 280
 - class diagrams, 12
 - commands, Chain of Responsibility pattern, 295
 - commands, Command pattern, 305
 - creation, delegating, 155
 - creation, Object Pool pattern and, 156
 - creational patterns and, 93
 - custom, Prototype pattern and, 131
 - deep copies, 84

objects (*continued*)

- deep copies, Prototype pattern and, 133
- deleting, doubly linked lists and, 73
- dependent, 56
- flyweight, shared, 239
- inserting, double linked lists and, 73
- modifying, iteration and, 208
- modifying, Read-Only interface pattern, 187–192
- multiobjects, 15
- prototypes, 131, 132–139 (*See also* Prototype pattern)
- proxy objects (*See* proxy objects)
- read-only status, 189
- read/write access, 483–492
- Reusable, Object Pool pattern, 153
- serialization, 144
- ServiceProxy, 82–83
- shallow copies, 84
- shallow copies, Prototype pattern and, 133
- shared references, Immutable pattern and, 75
- shared service objects, Virtual Proxy pattern, 259
- temporary, 15

ObjectStructure class, Visitor pattern and, 418

Observer pattern

- Adapter pattern and, 382
- batching notifications, 378
- behavior and, 293
- code example, 379–381
- Delegation pattern and, 382
- dependencies, 373
- Future pattern and, 540
- implementation, 376–378
- IObservable interface and, 375
- IObserver interface and, 375
- Mediator pattern and, 354, 382
- Multicaster class and, 376, 377
- .NET, 379
- Observable class, 375, 376–377
- Observer class, 375
- Publish-Subscribe pattern and, 382
- veto changes, 378

operations

- behaviors, 1
- meaning, 1
- sequence, concurrency patterns and, 437

Optimistic Concurrency pattern, Cache Management pattern and, 292

Originator class, Snapshot pattern, 360

P

packages

- assemblies, 11
- namespaces, 11
- visibility indicators, 11

Page objects, Composite pattern, 175

Param Dirty state, 385

parameters

- formal, 3
- multiple, functions, 3
- multiple, subroutines, 3

parse tree

- introduction, 323
- token sequences, 323

parser, recursive descent, 334

Parser class

- Expect subroutine, 337
- Little Language pattern, 328

partitioning patterns

- Composite pattern, 163, 175–185
- Filter pattern, 163, 165–174
- Read-Only interface pattern, 187–192
- Read-Only pattern, 163

performance tuning, Cache Management pattern and, 277–281

persistence, serialization and, 144

Pipe pattern

- Filter pattern and, 174
- Producer-Consumer pattern and, 498

planning, deployment and, 26

polling, Future pattern, 532–533

Polymorphism pattern

- Hashed Adapter Objects pattern and, 435
- State pattern and, 394

precedence rule, language, 319

preconditions, interactions, 20

prefetching objects, Cache Management pattern and, 277

- private (-) visibility indicator, meaning, 2
 - problem domain, client classes, 50
 - Processor class, Scheduler pattern, 473
 - Producer class, Producer-Consumer pattern, 495
 - Producer-Consumer pattern
 - Asynchronous Processing pattern and, 527
 - code example, 496–498
 - Consumer class, 495
 - Double Buffering pattern and, 518
 - Guarded Suspension pattern and, 498
 - implementation, 496
 - introduction, 438
 - .NET, 496
 - Pipe pattern and, 498
 - Producer class, 495
 - Queue class, 495
 - Scheduler pattern and, 498
 - Product class, Builder pattern, 124
 - ProductCacheManager class, Cache Management pattern, 274
 - ProductInfoFetcher object, Cache Management pattern, 274
 - protected (#) visibility indicator, meaning, 2
 - Protection Proxy pattern
 - Dynamic Linkage pattern and, 253
 - Proxy pattern and, 91
 - Prototype Builder class, roles, 133
 - prototype, deployment and, 27
 - Prototype pattern
 - Abstract Factory pattern and, 139
 - code, 135–139
 - Composite pattern and, 139
 - consequences, 134–135
 - custom objects, 131
 - Decorator pattern and, 139
 - deep copying, 133
 - Façade pattern and, 139
 - Factory Method pattern and, 108, 139
 - ICloneable interface, 131
 - implementation, 133–134
 - IPrototype interface and, 132
 - Iprototype, 133
 - .NET Framework and, 135
 - object creation and, 94
 - overview, 131
 - shallow copying, 133
 - symbol objects, 131
 - SymbolBuilder objects, 132
 - PrototypeBuilder objects, Prototype pattern implementation, 133–134
 - proxy objects
 - classes, 81
 - description, 81
 - function calls, 81–82
 - Proxy pattern and, 81
 - uses, 81–82
 - Proxy pattern
 - Adapter pattern, 204
 - code example, 83–91
 - Decorator pattern and, 91
 - delegation, 54
 - Façade pattern and, 91
 - Future pattern and, 532–533, 540
 - implementation, 83
 - Object Request Broker pattern and, 91
 - organization, 82–83
 - Protection Proxy pattern and, 91
 - proxy objects and, 81
 - Virtual Proxy pattern and, 91, 263
 - wrapper classes, 121
 - public (+) visibility indicator, meaning, 2
 - Publish-Subscribe pattern, Observer pattern and, 382
 - pull filters
 - AbstractPullFilter class, 167
 - introduction, 166
 - pull method, Queue class, 459
 - push method, Queue class, 459
- Q**
- Queue class
 - Producer-Consumer pattern, 495
 - pull method, 459
 - push method, 459
- R**
- read consistency, cache management and, 282
 - read-only access, IEnumerator interface, 61

- Read-Only Interface pattern
 - Adapter pattern and, 192
 - code example, 190–192
 - Immutable pattern and, 79
 - implementation, 190–192
 - Interface pattern and, 192
 - IReadOnly interface, 190
 - Mutable class, 190
 - MutatorClient class, 190
 - object modification, 187–192
 - ReadOnly Client class, 190
 - Snapshot pattern and, 372
 - Read-Only pattern, 163
 - Read/Write Lock pattern
 - code example, 487–492
 - concurrent read access, 483–492
 - implementation, 486
 - introduction, 438
 - .NET, 487
 - Scheduler pattern and, 481, 492
 - Single Threaded Execution pattern and, 492
 - ReadOnlyClient class, Read-Only interface pattern, 190
 - RealOperation class, Null Object pattern, 403
 - RecordFactoryFactory class, Factory Method pattern, 104
 - rectangles, classes, 2
 - recursion
 - Composite pattern, 175
 - Mediator pattern, 348
 - tail recursion, 335
 - Recursive Composition pattern, 175
 - recursive descent parsers, 334
 - redo commands, 307–308
 - Reference Interface, indirection, 55
 - references, shared (Immutable pattern), 75
 - regular expression languages, 325
 - relationships
 - is-a, 35
 - is-a-part-of, 36
 - remote procedure calls, serialization and, 144
 - renderer filter, DirectShow API filters, 170
 - rendezvous, Future pattern, 534
 - ReorgVisitor class, Visitor pattern, 417
 - Request class, Scheduler pattern, 473
 - request management, Asynchronous Processing pattern, 522–524
 - Requester class, Future pattern, 531
 - requirements
 - assumptions, 29
 - case study, 29–30
 - dependencies, 29
 - deployment and, 26–27
 - risks, 29
 - reserve buffer, 500
 - reserved words, Little Language pattern, 318
 - Result class, Future pattern, 531
 - Reusable object, Object Pool pattern, 153
 - ReusablePool object, Object Pool pattern, 154
 - reusing objects, Object Pool pattern, 149–161
 - risks, requirements and, 29
 - role names, associations, 7
 - roles
 - delegation, 48
 - inheritance, 48
- S**
- Scheduler class, Scheduler pattern, 473
 - Scheduler pattern
 - Asynchronous Processing pattern and, 527
 - code example, 476–481
 - implementation, 476
 - IScheduleOrdering interface, 474
 - Processor class, 473
 - Producer-Consumer pattern and, 498
 - Read/Write Lock pattern and, 481, 492
 - Request class, 473
 - Scheduler class, 473
 - thread execution, 471–481
 - security
 - Dynamic Linkage pattern, 249
 - object modification, 187–192
 - semantics, language, 317
 - sequence numbers, interactions
 - introduction, 14
 - multilevel, 15

- sequence of operations, concurrency patterns and, 437
- sequential access
 - IEnumerator interface, 61
 - Iterator pattern, 205
- serialization
 - Memento object comparison, 362
 - objects, 144
 - persistence and, 144
 - remote procedure calls and, 144
 - SOAP and, 144
 - XML (Extensible Markup Language) and, 144
- Serialize method, Snapshot pattern and, 363–365
- Serializer class, Snapshot pattern, 358
- Service class, Virtual Proxy pattern, 257
- ServiceProxy class, Virtual Proxy pattern, 258–259
- ServiceProxy objects, 82–83
- shallow copies, objects
 - Clone function, 84
 - Prototype pattern, 133
- shared references, Immutable pattern and, 75
- shared resources, concurrency patterns and, 437
- SharedConcreteFlyweight class, 238
- Single Threaded Execution pattern
 - Balking pattern and, 470
 - code example, 444–446
 - deadlocked threads, 443
 - function calls, concurrent, 439
 - Hashed Adapter Objects pattern and, 435
 - Immutable pattern and, 79
 - implementation, 442–443
 - Lock Object pattern and, 458
 - Read/Write Lock pattern and, 492
 - shared resources and, 437
 - TrafficSensor class, 440
 - TrafficSensorController class, 440
 - TrafficTransmitter class, 440
- singleton classes
 - access, 143
 - constructors, 143
 - instances, 145
 - instantiation, 143
 - object copies, 143
 - subclassing, 145
 - variables, 143
- Singleton pattern
 - Abstract Factory pattern and, 119
 - Cache Management pattern and, 147
 - classes, 142
 - code example, 146–147
 - implementation, 143–145
 - instantiation and, 141–147
 - .NET API, 145
 - Null Object pattern and, 405
 - object creation and, 94
 - Object Pool pattern and, 147, 161
 - State pattern and, 394
- Sink class, Filter pattern, 168, 169
- Snapshot pattern
 - Caretaker class, 360
 - code example, 370–371
 - Command pattern and, 315, 372
 - Deserializer class, 358
 - FileStream class, 358
 - Game Snapshot classes, 356
 - GameModel class, 357
 - IFormatter interface, 361
 - IMemento interface, 360
 - IMilestoneMemento interface, 357
 - implementation, 363–370
 - Memento class, 360
 - Memento objects and, 359–360
 - Memento pattern and, 355
 - MilestoneMemento class, 357
 - MilestoneMementoManager class, 357
 - Originator class, 360
 - Read-Only Interface pattern and, 372
 - Serializer class, 358
 - Stream class, 361
 - Target object, 361
 - TextFileReader class, 369
 - UserInterface class, 356
- SOAP (Simple Object Access Protocol), serialization and, 144
- software, deployment, activities leading up to, 26
- software lifecycle, 25–26
- Source class, Filter pattern, 167
- source filter, DirectShow API filters, 170

- SpecializedAbstraction class, Bridge pattern, 216
- SpecializedAbstractionImpl interface, Bridge pattern, 217
- StartOfSale class, Factory Method pattern, 104
- state
 - Balking pattern, 467–470
 - Both Dirty state, 385
 - dependencies, Mediator pattern, 345–346
 - encapsulation, State pattern, 383–394
 - events, 22
 - File Dirty state, 385
 - filter objects, 170
 - Immutable pattern and, 75
 - Mediator pattern, 343–354
 - Memento objects, 362
 - Not Dirty state, 384
 - notification batching, Observer pattern, 378
 - object pool, 156
 - Param Dirty state, 385
 - serialization and, 362
 - Snapshot pattern, 355–372
 - transitions, 23
- State class, State pattern, 387
- state machines
 - implementation, 42
 - statechart diagrams, 22
- State pattern
 - Both Dirty state, 385
 - classes, 386–388
 - code example, 389–394
 - Context class, 387
 - File Dirty state, 385
 - Flyweight pattern and, 394
 - implementation, 388–389
 - Mediator pattern and, 394
 - Not Dirty state, 384
 - Param Dirty state, 385
 - Polymorphism pattern and, 394
 - Singleton pattern and, 394
 - State class, 387
 - state encapsulation, 383–394
- statechart diagrams
 - state machines and, 22
 - transition lines, 23
- stateful objects, 383
- Static Locking Order pattern
 - code example, 450–451
 - deadlocks, 447–449
 - implementation, 449
 - introduction, 438
 - Lock Object pattern and, 451, 458
- static relationships, inheritance and, 50
- stereotypes
 - definition, 3
 - interfaces, 5
 - .NET delegate, 5
- Strategy pattern
 - Adapter pattern, 204, 400
 - algorithm encapsulation, 395
 - Builder pattern and, 129
 - Client class, 396
 - code example, 398–399
 - Decorator pattern and, 272
 - Factory Method pattern and, 108
 - Flyweight pattern and, 400
 - Hashed Adapter Objects pattern and, 435
 - implementation, 397
 - Interface pattern and, 59
 - IStrategy interface, 396
 - .NET API usage, 397
 - Null Object pattern and, 400, 405
 - Template Method pattern and, 400, 413
- Stream class, Snapshot pattern, 361
- String class, Immutable pattern and, 78
- structural patterns
 - Adapter pattern, 195–206
 - Bridge pattern, 213–223
 - Cache Management, 273–292
 - Decorator pattern, 265–272
 - Dynamic Linkage pattern, 245–253
 - Facade pattern, 225–228
 - Flyweight pattern, 233–244
 - Iterator pattern, 205–211
 - Virtual Proxy pattern, 255–263
- subclassing, singleton classes, 145
- subroutines
 - functions and, 1
 - interfaces and, 196
 - multiple parameters, 3
 - threads, multiple, 18
 - visibility indicators, 2

- suspended execution, Guarded Suspension pattern, 459–465
 - Symbol objects, SymbolBuilder objects, 132
 - symbol objects, Prototype pattern and, 131
 - SymbolBuilder objects, 132
 - synchronous design, 519
 - syntax, language
 - combinations of words, 318
 - definition, 317
 - grammar, 317
 - system architecture, deployment and, 27
- T**
- tail recursion, 335
 - Target object, Snapshot pattern, 361
 - Template Method pattern
 - Abstract Base Class pattern and, 67
 - abstract classes, 407
 - AbstractTemplate class and, 409
 - behavior and, 294
 - Builder pattern and, 129
 - Cache Management pattern and, 292
 - Chain of Responsibility pattern and, 304
 - code example, 410–413
 - Command pattern and, 315
 - ConcreteTemplate class and, 409
 - Decorator pattern and, 272
 - Factory Method pattern and, 108
 - implementation, 410
 - Strategy pattern and, 400, 413
 - temporary objects, collaboration, 15
 - terminal tokens, language, 320
 - TerminalToken class, Little Language pattern, 329
 - testing, deployment and, 27
 - TextFileReader class, Snapshot pattern, 369
 - Thread Pool pattern
 - Asynchronous Processing pattern and, 527
 - Object Pool pattern and, 161
 - threads
 - Asynchronous Processing pattern, 522–524
 - deadlocked, Single Threaded Execution pattern, 443
 - deadlocked, Static Locking Order pattern, 447–449
 - deadly embrace, Single Threaded Execution pattern, 444–446
 - Double Buffering pattern and, 502–503
 - execution order, Schedule pattern and, 471–481
 - functions, 18
 - Immutable pattern and, 76
 - lock objects and, 454
 - multiple, 18
 - subroutines, 18
 - TOCVisitor class, Visitor pattern, 417
 - tokens, grammar, 320, 323
 - transform filter, DirectShow API filters, 170
 - transformations
 - data filter classes, 166
 - implementation, 166
 - transition lines, state charts, 23
 - transitions, state, State pattern, 388
 - two-compartment classes, example, 4
- U**
- UML (Unified Modeling Language), 1
 - undo function, 305, 307–308
 - UnsharedConcreteFlyweight class, 238
 - user interface, windowing systems, 109
 - UserInterface class, Snapshot pattern, 356
- V**
- value objects
 - Immutable pattern and, 75, 76
 - purpose, 75
 - variables
 - singleton classes, 143
 - visibility indicators, 2
 - VBCodeProvider class, Builder pattern, 128
 - VB.NET. *See also* .NET
 - assumptions about, 1
 - interfaces and, 57
 - veto changes, Observer pattern, 378
 - Virtual Proxy pattern
 - Cache Management pattern and, 292
 - Client class, 257–258
 - code example, 261–263
 - deferred class loading, 259–260

Virtual Proxy pattern (*continued*)

- Dynamic Linkage pattern and, 253
 - Facade pattern, 263
 - Future pattern and, 540
 - implementation, 259–260
 - instantiation and, 255–263
 - IService interface, 259
 - Object Pool pattern, 263
 - overview, 193
 - Proxy pattern, 91, 263
 - Service class, 257
 - ServiceProxy class, 258–259
 - shared service objects, 259
- visibility indicators
- meanings, 2
 - packages, 11
- Visitor pattern
- AbstractElement class and, 419
 - AbstractVisitor class and, 419
 - Builder pattern and, 129
 - Client class and, 418
 - code example, 422–425
 - Composite pattern and, 185, 425
 - DocumentVisitor class, 417
 - implementation, 421
 - Iterator pattern and, 425
 - Little Language pattern and, 342, 425
 - logic and, 415
 - ObjectStructure class and, 418
 - ReorgVisitor class, 417
 - TOCVisitor class, 417
 - WordProcessor class, 416

W

- White Box Testing pattern, Mediator pattern and, 354
- white space, 324
- word combination language classes, 327
- word combination language lexical rules, 326
- WordCombination class, Little Language pattern, 339
- WordProcessor class, Visitor pattern, 416
- words. *See* languages; Little Language pattern
- wrapper classes
 - constructors, 121
 - Decorator pattern, 121
 - definition, 121
 - Facade pattern and, 121
 - Proxy pattern and, 121
 - source creation, 123
- wrapper objects, inheritance and, 269
- Wrapper pattern. *See* Decorator pattern
- WrapperGenerator class, 123–124
- write consistency, cache management and, 282

X–Y–Z

- XML (Extensible Markup Language), serialization and, 144
- XYZJournalRecordFactory class, Factory Method pattern, 104

