

Index

Note: Italicized page locator indicates figure/table.

A

Abort approach, 202
About Face 2.0 (Cooper), 286
 Abstraction, 3
 AccessController, 352
 Access transparency, 109
 Administration support,
 building in, 87–95
 Administrators, 356
 Affinity settings, 194–195
 Aggregate functions, 38, 40
 Alert setting, with profiling
 tools, 71
 Alice, Bob, Mallory, Eve, and
 Trent, 324
 Amazon.com, 309–310
 ANSI SQL standard, 77
 Apache Group, 260
 Applets, presentation and,
 291–292
 Application exceptions, system
 exceptions *versus*, 57
 Application server tier, 33
 Applied cryptography, 326
Applied Cryptography
 (Schneier), 324
 Architecture, 19–99
 administration support and,
 87–95
 ClassLoader, 379
 client/server, 104
 components and, 19–26
 data/processors kept close
 together with, 35–40

 deployment and, 96–99
 EJB restricted to
 transactional processing
 and, 64–67
 hook points and, 47–53
 identity, contention, and,
 40–47
 layers differentiated from
 tiers within, 31–35
 loose coupling and, 26–31
 Model-View-Controller,
 305
 monitoring support and,
 78–87
 optimizing and, 67–74
 peer-to-peer, 104
 performance and scalability
 goals for, 59–64
 robustness and, 53–59
 scalable, 60
 vendor neutrality costs and,
 74–78
 Archiving, log messages to files,
 84
 Arena-based garbage collection,
 417, 418–419
 Arenas, 418
Art of Deception, The (Mitnick),
 327
 Ask approach, 202
 Aspect (Eclipse), 11
 Aspect-oriented programming,
 11
 AspectWorks, 11

Asymmetric key cryptography,
 323
 Asynchronous calls, 104
 Asynchronous JMS message
 processing, with EJB, 67
 Asynchronous processing,
 139
 Asynchronous response, 104
 “Attack trees,” 334
 Attributes, 237
 Authentication, 322–323,
 344
 Authorization, 323, 344
 checks, 94
 role-based, 356–365
 Axis bits, 80

B

Backing stores, 91, 92
 Bandwidth, 16, 288–289, 297
 Bank security, 339
 Bank updates
 results with, 211–212
 statements for, 175–176
 Bean-Managed Persistence
 (BMP) entity beans, 66
 Beans, transactional affinity
 interaction between,
 194–195
 Behavior, 237
 hook points and, 48
 of objects, 41

- Behavior-driven
 - communication, data-driven communication *versus*, 128–135
- Binary Large Object (BLOB), 265
- Bloch, Joshua, 1, 431
- Broadcast messages, 104
- Browsers, 286, 287, 288, 307
- Buffer Overflows, 342
- Buffer overrun attack, 345
- Bugs, deployment and, 99

- C**
- C++, 17
- Caching
 - entity bean's identity and, 46
 - performance and, 62, 63
 - pregeneration and, 311
 - SoftReference objects and, 434–437
- Capacity, 61
- Cascading Style Sheets, 288, 306
- Castor (Exolab), 255
- Causality, 192
- Chatty distributed persistent objects, 151, 156
- CipherInputStream/CipherOutputStream pair, 372
- Ciphertext, 323
- ClassLoader(s), 11, 23, 24, 379, 447
 - isolation, 403
 - rules about, 401–402
 - versioning, 405
- ClassLoader boundaries
 - isolation and, 403–404
 - recognizing, 400–408
 - versioning and, 404–408
- CleanThisUp instance, 442, 443, 444
- Client/server architecture, 104
- Client-side scripting, 290, 342
- Closure, 239
- Cloudscape, 260
- Clustering, 33, 113, 140
- Coarse-grained security controls, 357
- Coarse-grained synchronization
 - example, 183, 185
- Codd, E. F., 238
- Code
 - diagnostic-logging behavior added into, 49–52
 - failure in, 53, 54
 - generation, 12
 - JNI, 445–448
 - profiler run over, 69, 70
 - wedding ceremony, 171, 173–174
- Column orders, SELECT statements and, 269
- Columns, 237, 269
- Command injection attacks, SQL statements and, 344–349
- Command Injection Flaws, 342
- Command pattern, 179
- Commercial profilers, 73
- Committing transactions, 170
- Common Object Request Broker Architecture, 105
- Communication, 9, 101–157
 - bulk passing of data and, 150–155
 - context-complete style of, 120–128
 - data-driven over behavior-driven, 128–135
 - lookup and, 108–114
 - network access costs and, 114–120
 - options, 101–108
 - partitioning components and, 140–147
 - proxies, 155–157
 - remote service requests and, 136–140
 - request-response, 103
 - Web services for open integration, 147–150
- Compensating transactions, 99, 178–179
- Complex solutions, problems with, 161–162
- Complex state
 - evaluation/execution, rules engines for, 163–169
- Component-based design, 17
- Component boundaries, loose coupling across, 26–31
- Component Development for the Java Platform* (Halloway), 415, 446
- Components
 - architecture and, 19–26
 - hot deployment of, 447
 - locks and ceding control outside of, 187–193
 - partitioning, 140–147
- ConcatName function, 39
- Concerns
 - crosscutting, 7–10
 - separation of, 8
- Concurrency, 159–160
 - eager-loading and, 272
 - optimistic, 200–206
 - pessimistic, 206–211
 - transparency, 109
- Concurrent load, 61
- Concurrent Programming in Java* (Lea), 160
- Confidentiality, 324
 - SealedObject and, 370–372
 - of Serializable objects, 370–372
- Configuration data, 88–89
- ConnectException, 55
- Connection management, 384–386

- ConnectionPool class, 404
 - Connection pooling, 429–430
 - Container-managed error-handling facilities, 56
 - Container-Managed Persistence (CMP) entities, 47
 - Container-managed resource management, 426–430
 - Content, 305
 - pregenerating, to minimize processing, 309–311
 - style separated from, 305–309
 - Content-Based Router pattern, 134
 - Contention, 310
 - identity and, 40–47
 - scalability and, 186
 - Context-complete
 - communication, preference for, 120–128
 - Control
 - HTML and, 288
 - locks and, 187–193
 - options, by system
 - administrators, 93–95
 - Cooper, Alan, 286
 - Copying collector, 417, 419
 - CORBA. *See* Common Object Request Broker Architecture
 - Costs
 - network access, 114–120
 - vendor neutrality, 74–78
 - Coupling, 20
 - Credentials, security, 322
 - CreditCard class, 413–415
 - Credit card fraud, 312
 - Credit cards
 - fraud issue, 312
 - security issue, 327
 - transactions, 136–137, 138, 139, 145
 - Crosscutting concerns, 7–11, 13
 - Cross-platform differences, HTML and, 288
 - Cross-site scripting attack, 349, 351
 - Cross-Site Scripting Flaws, 342
 - Cryptoanalysis, 324
 - Cryptography, 323, 326
 - CSS. *See* Cascading Style Sheets
 - Curl, 296
 - Customizations, hook points and, 47–53
 - Cyclic references, 251–252, 254
- D**
- Data
 - confidentiality, 325
 - don't assume ownership of, 263–266
 - fragmentation, 142, 143
 - framing, 102
 - frequently used, eager-loading of, 271–273
 - infrequently used, lazy-loading of, 266–271
 - integrity, 325
 - keeping processors close to, 35–40
 - leasing, 221
 - passing in bulk, 150–155
 - relational model of, 238
 - Data access patterns, 249
 - Database
 - attacks, 327
 - don't assume ownership of, 263–266
 - DatabaseMetaData class, 279
 - Database profiler tools, 72–73, 282–283
 - Data-driven communication,
 - behavior-driven communication *versus*, 128–135
 - Data Transfer Objects, 22, 52–53, 135, 152, 224, 249, 268
 - Data types, defining with XML Schema, 257
 - Data validation
 - semantic, 313, 318
 - syntactic, 312, 318
 - Date, Chris, 238
 - Deadlocks, thread, 188–193
 - Debugging, complex solutions and, 161
 - Declarative attributes, 12, 13
 - Decryption, 372
 - Defaults, security and, 337
 - Defense-in-depth strategy, 339, 340–341
 - Defensive programming, 53
 - DELETE statements, batching and, 276
 - Demilitarized zone, 327, 340
 - Denial-of-service attacks, 337
 - Deployment
 - components as key element of, 19–26
 - script, 98, 336
 - simplifying, 96–99
 - Design of Sites, The* (Van Duyne/Landay/Hong), 298
 - Detection, secure systems and, 330, 331, 332
 - Deutsch, Peter, 15
 - Development, components as key element of, 19–26
 - DHTML. *See* Dynamic HTML
 - Diagnostic-logging behavior, adding into code, 49–52
 - Diagnostic logs, 82–87
 - Digital certificates, 369
 - Digital signatures, 366–367
 - Dirty reads, 211
 - DistributedHttpSession class, 229–230
 - Distributed systems,
 - transparency forms in, 109
 - Distributed transactionally aware middleware, 67

- Distributed transactional processing, 200
 - Distributed transaction controllers, 67
 - Distributed transaction protocol, 197
 - Distributed transactions, 175
 - local transactions *versus*, 196–200
 - Distributed two-phase commit protocol, 172
 - Distribution, 140
 - DMZ. *See* Demilitarized zone
 - DNS. *See* Domain Name System
 - Document-oriented style, 128
 - Domain logic, 32–33, 34
 - Domain model, objects-first persistence and preservation of, 232–236
 - Domain Name System, 141
 - Domain Store pattern, 249
 - Downtime, 2
 - Drools, 166
 - Droplets, 296
 - DTCs. *See* Distributed transaction controllers
 - DTOs. *See* Data Transfer Objects
 - Durable state, 8, 225, 226
 - Dynamic HTML, 289, 290
 - Dynamic proxies, 48, 49, 52
- E**
- Eager-loading, of frequently used data, 271–273
 - e-commerce, 7
 - e-commerce shopping cart, transient state and, 225
 - e-commerce sites, checkout stages of, 136–137, 138, 139
 - Eden space, 420
 - Effective C++* (Meyers), 20
 - Effective Java* (Bloch), 1, 20, 160, 170, 180, 187, 223, 374, 415, 431, 432
 - Effectiveness, 1
 - 80/20 rule, 67–68
 - EJB. *See* Enterprise Java Beans
 - EJB containers
 - authorization and, 357
 - middleware and, 13
 - EJB Entity Beans, 12
 - EJBException class, 57
 - EJBQL, 233, 234
 - EJB Specification, 53
 - strengths with, 66–67
 - transaction isolation and, 215
 - vendor neutrality and, 75
 - Encapsulation, 21, 22
 - layer, 246–249, 265
 - Encryption, 372, 413–415
 - Enterprise applications, uptime requirements with, 58
 - Enterprise computing, ten fallacies of, 14–17
 - Enterprise Java Beans, 6
 - exception-handling with, 56–57
 - identity in, 44–47
 - transactional affinity, 193–196
 - transactional processing and, 64–67
 - Enterprise Java system, qualities of, 1–3
 - Enterprise security, two laws of, 338
 - Enterprise systems, loose coupling and, 28
 - Entity beans, 46–47, 52
 - Error Handling Problems, 342
 - errorPage directives, setting, 56
 - Evolution, 132–133
 - Exception handling, 2, 54–59
 - executeBatch method, 275, 276
 - Explicit concurrency control, pessimistic concurrency for, 206–211
 - Explicit transactional locks, 272
 - Exploration testing, 280
 - Extreme programming, 334
- F**
- Failed login attempts, 86
 - Failing securely, 58
 - Failover, 140
 - Failure
 - robustness in face of, 53–59
 - transparency, 109
 - False positives, with detection systems, 331
 - Fast Lane pattern, 283, 302
 - Filemon, 72
 - Finalizers, 380, 431–432
 - finally block, 382–384
 - Fine-grained security controls, 358
 - Fine-grained synchronization
 - example, 181, 184, 185, 186
 - Finite object managers, 424
 - Fire-and-forget, 104
 - Firehose cursor support, 277
 - Firewalls, 16, 331, 339–340
 - communicating across, 107
 - man-in-the-middle attacks within, 327
 - “Five Nines Availability,”
 - enterprise systems and, 2
 - Flags, garbage collection, 421–422
 - Flex, 291
 - Frames, for separating portions of Web pages, 298
 - Framing data, 102
 - Full collection, 420, 421

- Full table scan, 73
 Functionality, hook points and, 47–53
- G**
- Garbage collection
 reference objects and, 430–434
 tuning JVM and, 393–394
Garbage Collection (Jones/Lims), 419
 Garbage collector, 416–426
 arena-based (or copying) collector, 417
 generational collector, 418, 419
 Generation, 418
 Generational collector, 418, 419
 GetJREPath, 396
 Global caches, performance and, 62, 63
 GNUtella, 112
 Gosling, James, 15
 Guarded Object, 372
 GuardedObject, for providing access control on objects, 372–378
 Guests, 356
- H**
- Hackers, 16
 Hacker Web sites, consumer data on, 335
 Handle counts, tracking, 71
 Handler-implementing classes, interceptors as, 53
 happyaxis.jsp, 80
 “happy bit” tests, 80–81, 86, 336
 “happy heartbeat” process, 86
 happysystem.jsp page, 80, 81
- Heap size, tuning JVM and, 393
 Heavy tags, minimizing use of, 297
 Hierarchical data models, 250
 Hierarchical styles, CSS support for, 306
 Hierarchies, ClassLoaders and formation of, 401–402
 HOLDS-A relationships, avoiding, 258–259
 Homogeneity, 17
 Hook points, 24, 47–53, 63, 282
 Horizontal partitioning, 143, 144
 Hot-configurable configuration data, logging level as, 84
 Hot deployment of components, 447–448
 Hotspot, 387, 400
 hprof (Sun), 73–74
 HSQLDB, 260
 HTML
 advantages/disadvantages with, 286–289
 dynamic, 290
 presentation and minimizing of, 296–299
 HttpServletRequest object, 229
 HttpServletResponse object, 229
 HttpSession, sparing use of, 227–231
 Hyper/J (IBM), 11
- I**
- IApi interface, 114–118
 IBM, 40
 Identity
 authorization based on, 359
 contention and, 40–47
 of objects, 41
 Identity Map, 236, 283
- IDL. *See* Interface Description Language
 IIOP. *See* Internet Inter-Orb Protocol
 Images, reusing, 298
 Immutable objects, favoring, 222–224
 Impersonation, 362
 Industrial espionage, 336
 Infinite recursion, 252
 avoiding, 378
 Inheritance, 10
 Initialization, 125
Inmates Are Running the Asylum, The (Cooper), 286
 In-process storage, using to avoid network, 259–263
 Insecurity, assuming, 336–342
 INSERT statements, batching and, 276
Inside Java 2 Platform Security (Gong), 356, 359
 Installation defaults, 337
 Instantiation, 125
 Integration, 6, 147
 Integrity, 325
 Interception pattern, 11, 48
 Interceptors, 52, 53
 Interface Description Language, 106
 Intermediaries, 132, 133–134
 Internet, 7, 288, 325
 Internet Inter-Orb Protocol, 103
 Internet service providers, 228
 Interoperability, across platforms, 147–150
 Interprocess communication, 101
Introduction to Database Systems (Date), 142
 Intrusion risk assessments, 335
 Inventory checking, 263

- IPC. *See* Interprocess communication
- isCallerInRole method, 357
- Isolation
 - ClassLoader, 403
 - between Web applications, 403–404
- Isolation levels, for transactional throughput, 211–216
- J**
- JAAS. *See* Java Authentication and Authorization Service
- JAAS configuration, 359
- JACE, 446
- Jakarta Apache Web site, 305
- Jakarta Taglibs project, 319
- Java API
 - for XML Binding, 250, 255
 - for XML Messaging, 106–107
 - for XML RPC, 53, 106
- Java applets, presentation and, 291–292
- Java Authentication and Authorization Service, 16, 359, 361, 365
- Java Community Process, 85, 349
- Java Connector API, 6
- Java Cryptography Extension Reference Guide*, 371
- Java Data Objects Specification, 11, 48
- Java Expert System Shell, 166
- java.io.Serializable interface, 57, 408
- Java Language Specification, 430, 433
- java launcher, 387–391
- Java Management Extensions, 73, 85–86
- Java Message Service, 6, 106
- Java Naming and Directory Interface, 24
- Java Native Interface (JNI), 102
 - code on server, 445–448
- Java Native Interface, The* (Liang), 446
- Java Network Launch Protocol, 33, 293–296
- Java Object Serialization, 102, 252, 379, 407, 408–415
 - customization (writeObject and readObject) with, 412–413
 - replacement (writeReplace and readResolve) with, 413–415
 - serialVersionUID field in, 411–412
- Java Object Serialization Specification, 365
- Java platform security model, benefits with, 352–356
- Java.policy file, 363
- java.rmi.RemoteException, 54
- JavaScript, 288, 318
- Java Security Manager, 353, 354, 356
- Java Security* (Oaks), 356, 360, 415
- java.security package,
 - KeyPairGenerator class with, 368
- JavaServerPages, 6, 299–301, 303–305
 - accidental use of sessions and, 231
 - exception-handling policies inside of, 56
 - login page, 344–345
- Javassist, 11
- Java threading model, 379
- Java Transaction API, 6
- java.util.logging API, 82
- java.util.prefs.Preferences classes, 90
- java.util.Properties class, 88
- Java Virtual Machine, 10, 73, 379
 - ClassLoader and, 402
 - Profiler Interface, 73
 - Tools Interface, 73
 - tuning, 387–394
- Java Web Start, 292, 293–296
- Jawin, 446
- JAXB. *See* Java API for XML Binding
- JAXM. *See* Java API for XML Messaging
- JAX-RPC. *See* Java API for XML RPC
- JBoss, 85
- JCA. *See* Java Connector API
- JDBC, 6
- JDBC providers, knowing, 277–281
- JDBC Specification, 277
- JDK, permission checks and, 373–374
- JDK 1.2, security mechanisms within, 352–353
- JDO Specification. *See* Java Data Objects Specification
- JESS. *See* Java Expert System Shell
- Jini, 106, 112, 113
- JITA. *See* Just-in-time activation
- JMS. *See* Java Message Service
- JMX. *See* Java Management Extensions
- JNDI. *See* Java Naming and Directory Interface
- JNI. *See* Java Native Interface
- JNLP. *See* Java Network Launch Protocol
- JREs, independent, for side-side versioning, 395–400
- JRockit, 392

JScript, 288, 318
 JSP Specification, 23
 JSR-94, 169
 JSR 174, 73
 JTA. *See* Java Transaction API
 J2EE, 379
 goals of, 3–5
 implementation of, 10–14
 middleware and, 5–10
 profiler tools with, 69
 resource management and, 428
 serialization and, 410
 vendor neutrality and, 74–75
 J2EE containers, hot deployment of components with, 447–448
 J2EE servlet, authorization and, 357
 J2EE Specification, 1, 97–99
 Just-in-time activation, 385
 Just-in-time (JIT) compiler, 28
 JVMPI. *See* Java Virtual Machine Profiler Interface
 JVMPI API, 74
 JVMTI. *See* Java Virtual Machine Tools Interface
 JXTA, 106, 112

K

Kazaa, 112
 KeyPairGenerator class, 368

L

Late-bound code, 27
 Latency, 15, 60, 61, 140, 310
 Layers, tiers differentiated from, 31–35
 Lazy-loading
 ClassLoader and, 402

 of infrequently used data, 266–271
 Lazy Load pattern, 267
 Leasing data, 221
 Least privilege principle, 341, 354
 Linux, 69, 72
 Load, 61, 140–147
 Local storage, using to avoid network, 259–263
 Local transactions, distributed transactions *versus*, 196–200
 Location transparency, 108, 109, 110
 Lock files, 210
 Lock regions, replicating resources and avoidance of, 219–222
 Locks
 ceding control and, 187–193
 purpose of, 43
 Lock windows, 43
 eager-loading and, 272
 minimizing, 180–187
 Log4J logging streams, 283
 Log4j open-source project, 82
 LoginBean, ClassLoader and, 404–407
 LoggingProxy, 49–50
 Logging verbosity levels, keeping low, 85
 Login attempts, failed, 86
 LoginContext, creating, 360–361
 Login credentials, security and, 340
 LoginModule objects, 365
 Log messages, 82–83
 Lookup, 3, 9, 108–114
 Loopback, 192
 Loose coupling, 26–31, 52
 Loosely bound code, 27
 Lost updates, 211

M

Macromedia Flash, presentation and, 290–291
 Maintenance, complex solutions and, 162
 Man-in-the-middle attacks, within corporate firewall, 327
 Mark-and-sweep collectors, 419
 Marshaling, 102, 103
 MBeans, 73, 85
 MDBs. *See* Message-Driven Beans
 Memory, performance and, 71
 Merge approach, 202
 MessageDrivenBean interface, 67
 Message-Driven Beans, 67, 139
 Message-oriented middleware, 104
 Message Router pattern, 134
 Messaging communications APIs, 104
 Messaging systems, 104
 Method interception approach, 52
 Microsoft Internet Explorer, 288
 Microsoft SQL Server, 38
 Middleware
 centralization and, 141
 distributed transactionally aware, 67
 J2EE and, 5–10, 13, 14
 message-oriented, 104
 Migration, 147
 Migration transparency, 109, 110
 Mini-profilers, 74
 Minor collection, 420, 421
 Mitnick, Kevin, 327
 Model 2 architecture, 305

Model-View-Controller
 abstraction, 21
 architecture, 305
 Monitoring support, building
 in, 78–87
 Monitoring tools, 336

N

Nanning, 11
 Napster, 112
 .NET, 17, 395
 Netscape Navigator, 288
 Network File System, 109
 Networks
 access costs, 114–120
 in-process or local storage
 usage and avoidance of,
 259–263
 self-healing, 113–114
 NFS. *See* Network File System
 Nonatomic failure scenarios,
 transactional processing
 for, 169–175
 $N + 1$ query problem, 234, 267,
 271, 273
 Nonrepeatable reads, 211
 NoSuchObjectException, 55
 Nursery, 420

O

OASIS, 148
 Object banks, 424
 Object factories, 240, 424, 425
 Object-hierarchical impedance
 mismatch, recognizing,
 249–259
 Object identifier, 233
 Object identity
 importance of, 236
 preserving, 251
 Object lifecycle management, 9

Object-oriented database
 management systems, 233,
 236
 Object-oriented design, 42
 Object-oriented software
 development, 10
 Object-per-client model,
 properties of, 45–46
 Object pools, 424–425
 Object pooling, 423–424, 425
 Object Query Language, 233
 Object-relational impedance
 mismatch, 9, 237
 Object-relational mapping,
 problems with, 237–238
 Object Request Broker, 105
 Objects, 237
 GuardedObject and access
 control on, 372–378
 this parameter and, 41
 Objects-first persistence
 layers, 156
 for preserving domain
 model, 232–236
 Object-to-object references,
 235
 OID. *See* Object identifier
 Old generation, of garbage
 collection, 420, 421, 422
 Once-and-Only-Once Rule,
 302, 307
 “One security check”
 optimization, 377
 One-way calls, 104
 OODBMS. *See* Object-oriented
 database management
 systems
 Open integration, Web services
 and, 147–150
 OpenJava, 11
 Open Web Application Security
 Project, 338
 Top Ten Web Application
 Security Vulnerabilities
 document by, 338, 342

Operating system profilers, 71,
 72
 Optimistic concurrency
 for better scalability, 200–206
 models, 272
 Optimistic locking, 206
 Optimistic Offline Locks, 201
 Optimization
 complex solutions and, 161
 hook points and, 47–53
 “one security check,” 377
 after profiling, 67–74
 tuning SQL, 282–284
 Optimizing optimally, 63
 OQL. *See* Object Query
 Language
 Oracle, 40, 42
 ORB. *See* Object Request
 Broker
 Overwrite approach, 202
 OWASP. *See* Open Web
 Application Security
 Project

P

Page faults, performance and,
 71
 Page navigation, excessive, 298
 Parameter validation errors, 343
 Pareto Principle (80/20 rule),
 67–68
 Parsers, schema-validating, 258
 Partitioning, 140
 Partitioning components,
 excessive load avoided
 with, 140–147
 Pass-by-reference approach,
 lazy-loading and, 272
 Pass-by-value approach, eager-
 loading and, 272
 Passivation, 44
 Payload, 102
 Peer to peer (P2P), 104, 112

- Performance
 - goals, 59–64
 - vendor neutrality and, 78
- Performance Monitor (PerfMon), 69, 70–71
- Perl, 17, 349
- Permanent space, 420
- Permission classes, 353
- Permissions, 323
 - checks, 357, 373
 - default, 337
 - for user roles, 360, 363–364
- Persistence
 - objects-first, 232–236
 - procedural-first, 246–249
 - relational-first, 236–245
- PersistenceManager, 48
- Persistence transparency, 109
- Persistent data, durable state and, 226
- Person class, 40
- PersonManager/Person code, 42, 43
- Pessimistic concurrency
 - for explicit concurrency control, 206–211
 - models, 272
- Pessimistic Lock, 207
- Pessimistic Offline Lock, 207
- Phantom reads, 211–212
- PhantomReference objects, 433, 439–445
- Ping, 80, 340
- PingBean, 80
- Plain old Java objects, 29
- Plaintext, 323
- Platforms
 - communications across, 107–108
 - interoperability across, 147–150
- Platform security, turning on, 352–356
- Platform shifts, handling, 76–77
- PointBase, 260
- POJOS. *See* Plain old Java objects
- Policy classes, 353
- Portability, 75–76, 147
 - HTML and, 287
 - vendor value-added features *versus*, 77–78
- Portal pages, pregenerated, 310
- Preferences API, 90–92
- PreparedStatement, 279–280
- Presentation, 285–319
 - applets and, 291–292
 - dynamic HTML and, 290
 - Java Network Launch Protocol, Java Web Start and, 293–296
 - Macromedia Flash and, 290–291
 - minimizing HTML and, 296–299
 - pregenerating content to minimize processing, 309–311
 - rich-client UI technologies and, 286–290
 - separating from processing, 299–305
 - style separated from content in, 305–309
 - URLClassLoader class and, 292–293
 - validation and, 311–319
- Presentation elements, HTML and lack of, 289
- Prevention, security and, 329–333, 341
- Principal, 322
- Principle of least privilege, 341, 354
- Private keys, 366, 367, 368, 369
- Procedural-first persistence, for creating encapsulation layer, 246–249
- Procedural model, 246–249
- Procedures, 246
- ProcessExplorer, 72
- Processing, 9, 159–224
 - control and, 187–193
 - EJB transactional affinity and, 193–196
 - immutable objects and, 222–224
 - local transactions *versus* distributed transactions and, 196–200
 - lock windows and, 180–187
 - lower isolation levels and, 211–216
 - optimistic concurrency and, 200–206
 - pessimistic concurrency and, 206–211
 - pregenerating content and minimizing of, 309–311
 - presentation separated from, 299–305
 - resource replication and, 219–222
 - rules engines and, 163–169
 - savepoints and, 216–219
 - simplicity with, 160–163
 - transactional, 169–175
 - user transactions *versus* system transactions, 175–180
- Processors, keeping data close to, 35–40
- /proc filesystem, 69
- Profilers, 62, 69
 - commercial, 73
 - running over code with, 69, 70
- Profiling, optimizing after, 67–74
- Programming
 - extreme, 334
 - rules-based, 165
- Programming Windows Security* (Brown), 342

Properties files, 88, 90
 Proxies, 103, 155–157
 P6Spy driver, 283, 284
 Public-key/private-key
 cryptography, 323
 Public key/Private key pair,
 368–369, 370
 Public keys, 368, 369
 Public static factory methods,
 52
 Python, 17

Q

Query Data Gateway, 242–243
 Query-execution profiler tool,
 280
 Query Objects, 233, 234

R

Rational Unified Process, 334
 Reaction, secure systems and,
 330, 331, 332
 READ COMMITTED isolation
 level, 213, 214
 readObject, 412–413
 readResolve, 413–415
 READ UNCOMMITTED
 isolation level, 213, 214
 Recoverable operations, 132
 Recursion, 252
 infinite, 378
 Refactoring, 29–30
 Reference-counting collectors,
 419
 Reference objects, 433
 for augmenting garbage
 collection behavior,
 430–434
 Java, 433
 PhantomReference objects,
 433, 439–445

SoftReference objects, 433,
 434–437
 WeakReference objects, 433,
 437–439
 ReferenceQueue, 434, 437
 Referent, 433
 Relational databases, 1, 236
 Relational-first approach,
 strengths/weaknesses with,
 245
 Relational-first persistence,
 for exposing power of
 relational model,
 236–245
 Relational model, 236–245
 Reliability, 15
 Relocation transparency, 109,
 110
 RemoteException, 54, 55, 57
 Remote Method Invocation, 6,
 105, 394
 Remote procedure call, 103
 Remote service requests,
 avoiding waiting for
 responses from,
 136–140
 Remote Singleton, 44, 46
 Remote stubs, 43
 Remoting and communica-
 tion, 9
 REPEATABLE READ isolation
 level, 213
 Repeated failures, 58
 Replicating resources, avoiding
 lock regions by, 219–222
 Replication
 partitioning and, 144–145
 transparency, 109, 110
 Report running, administrative
 control and, 95
 Request-response communi-
 cation, 103, 136
 Required-marked transactional
 bean, throwing exception
 out of, 56

Resource churn, 385
 Resource management, 6, 10, 31
 container-managed, 426–430
 Resource Manager, 197
 Resource object management,
 categories of, 424–425
 Resources
 aggressive release of, 379–386
 replicating to avoid lock
 regions, 219–222
 Response time, 60–61, 136
 ResultSet, 37, 277–279,
 380–382, 385, 386
 ResultSetMetaData class, 279
 Reuse
 complex solutions and, 161
 components as key element
 of, 19–26
 Rich-client applications
 transient state and, 225
 Web Services and, 352
 Rich clients, advantages with,
 289
 Rich client technologies,
 286–289
 applets, 291–292
 choosing, 295–296
 dynamic HTML, 290
 Java Network Launch
 Protocol and Java Web
 Start, 293–295
 Macromedia Flash, 290–291,
 295
 URLClassLoader class and,
 292–293, 295
 “Ripple-loading,” 271
 Risk assessment studies, 335
 RM. *See* Resource Manager
 RMI. *See* Remote Method
 Invocation
 RMI exceptions, catching, 55
 Robustness, 2, 53–59
 Role-based authorization, 16,
 356–365
 Roles, 323, 357

- Rollback, savepoints and, 216–219
 - Rolling back transactions, 170, 177
 - Root privileges, 335
 - Round-trips
 - batching SQL work and avoidance of, 274–276
 - objects-first approach and, 234
 - validation and, 314
 - ROWID, 42
 - Rows, 237
 - RowSet, 154, 260–261, 386
 - RPC. *See* Remote procedure call
 - Rule languages, 168
 - Rules-based programming, 165
 - Rules engines
 - business logic and, 168–169
 - for complex state evaluation/execution, 163–169
 - purposes of, 165–166
 - RuleServiceProvider, 166
 - RuleSession types, kinds of, 167
 - RuntimeException,
 - EJBException class inherited from, 57
- S**
- Sash, 296
 - Savepoints, for keeping partial work in face of rollback, 216–219
 - Scalability, 60
 - contention and, 186
 - goals, 59–64
 - optimistic concurrency for, 200–206
 - problems with, 62–63
 - vendor neutrality and, 78
 - Schema-validating parsers, 258
 - Schneier, Bruce, 325, 326, 329, 334, 338
 - Scripting languages, 318
 - SealedObject, 375, 376
 - for providing confidentiality of Serializable objects, 370–372
 - SecretKey, 370, 371
 - Secret key cryptography, 323
 - Secure Sockets Layer, 326
 - Security, 16, 287–288, 321–378
 - assume insecurity, 336–342
 - exception-handling policies and, 56
 - GuardedObject and, 372–378
 - not just for prevention, 329–333
 - platform, 352–356
 - as process, not product, 325–329
 - role-based authorization and, 356–365
 - SealedObject and, 370–372
 - SignedObject and, 365–370
 - terminology, 322–325
 - threat model and, 333–336
 - user-based, 356
 - user input validation and, 342–352
 - Security assessments, calculating, 335
 - SecurityManager, 352
 - Security manager, turning on, 354, 355
 - Security policy, 355
 - SELECT COUNT(), 37
 - SELECT statements
 - batching and, 276, 279
 - column orders and, 269
 - Self-healing networks, 113–114
 - Semantically bound, 27
 - Semantic data corruption, 152
 - Semantic data validation, 313, 318
 - Separation of concerns, 8
 - Serializable access, 212
 - Serializable interface, 57
 - SERIALIZABLE isolation level, 213
 - Serializable objects,
 - SealedObject and confidentiality of, 370–372
 - Serialization, 102
 - hooks, 378
 - Java Object Serialization, 408–415
 - specification, 52
 - Serialized objects, SignedObject and integrity of, 365–370
 - serialVersionUID field, 411–412
 - Server, JNI code on, 445–448
 - Server-Based Java Programming* (Neward), 415
 - Server-side validation, 319
 - Service Data Objects, 152
 - Service time, 61
 - Servlet containers
 - interception and, 12
 - security and, 354–355
 - threading and, 426–428
 - ServletException, 56
 - Servlets, 6, 56, 105–106
 - Servlet 2.2 Specification, 228
 - Servlet 2.3 Specification, 229
 - Session Façade pattern, 194, 195, 249, 268
 - Sessions, accidental use of, 231
 - Session space, 227
 - Session-within-clusters
 - problem, 228
 - setFetchSize, 270
 - setMaxRows, 270
 - “7 Fallacies of Distributed Computing” (Deutsch), 15
 - Side-by-side versioning, independent JREs for, 395–400
 - SignedObject, 372, 375, 376
 - for providing integrity of Serialized objects, 365–370

- Simple Mail Transport Protocol, 107
- Simple Object Access Protocol, 103
- Simple types, XML Schema and, 257
- Simplicity, 160–163
- Singletons, 44, 45
- “Slashdot effect,” avoiding, 138, 385
- Smalltalk, 70
- Smart client front ends, 35
- Smart proxies, 157
- SMTP. *See* Simple Mail Transport Protocol
- Sniffers, 327, 336, 365
- SOAP. *See* Simple Object Access Protocol
- SOAP encoding, 254
- Social engineering attack, 327
- SoftReference objects, 433, 434–437, 445
- Solicit-notify, 104
- Spoofing, 365
- SQL, 4, 246
 - batching, to avoid round-trips, 274–276
 - power of, 239
 - tuning, 281–284
- SQLExceptions, 55, 56, 345, 349
- SQL injection attacks, 345–352
- SQL/J clauses, 244
- SQL/J Specification, 244, 245
- SQL-92 Specification, 77
- SQL-99 Specification, 244
- SQL/1 preprocessor, 244
- SQL Performance Tuning* (Gulutzan/Pelzer), 279
- SQL statements, command injection attacks and, 344–349
- SQLWarning instances, 56
- SSL. *See* Secure Sockets Layer
- State
 - kinds of, 225
 - for objects, 41
- State evaluation/execution, rules engines for, 163–169
- Stateful RuleSession type, 167
- Stateful session beans, 52
- Stateless RuleSession type, 167
- Stateless session beans, 45, 127
- State management, 8–9, 225–284
 - batching SQL work and, 274–276
 - data or database and, 263–266
 - eager-loading of data and, 271–273
 - HTML and, 287–288
 - HttpSession usage and, 227–231
 - in-process or local storage usage and, 259–263
 - JDBC provider and, 277–281
 - lazy-loading of data and, 266–271
 - object-hierarchical impedance mismatch and, 249–259
 - objects-first persistence usage and, 232–236
 - procedural-first persistence usage and, 246–249
 - relational-first persistence and, 236–245
 - rich clients and, 289
 - SQL tuning and, 281–284
 - transient and durable state and, 225–227
- Static factory method, 51
- Steganography, 324, 325
- Stored procedures, 246, 248–249
- Stovepipe systems, 6–7
- String concatenation, log messages and, 83–84
- Struts, 305
- Struts library, 319
- Stubs, 103
- Style, separating from content, 305–309
- Sun, 73, 349
- Sun Hotspot JVM garbage collection mechanism, 420
- Support
 - administration, 87–95
 - monitoring, 78–87
- Survivor spaces, 420
- Symmetric key cryptography, 323
- Synchronization, 9, 186
- Synchronous communication, 107
- Syntactically bound, 27
- Syntactic data validation, 312, 318
- Sysinternals.com, 72
- System, 379–448
 - aggressively release resources, 379–386
 - container-managed resource management and, 426–430
 - garbage collector and, 416–426
 - independent JREs used for side-by-side versioning in, 395–340
 - Java Object Serialization and, 408–415
 - JNI code on server and, 445–448
 - recognize ClassLoader boundaries, 400–408
 - reference objects and augmenting garbage collection behavior, 430–445
 - tuning the JVM, 387–394

System administrators, 16, 87
 control options needed by,
 93–95
 deployment and, 96–97
 System.err, 86
 System exceptions, application
 exceptions *versus*, 57
 System.gc calls, 423
 System.out, 86
 System transactions, user
 transactions *versus*,
 175–180

T

Table Data Gateway, 241–243
 Tables, 237, 273
 Tag libraries, validation and,
 317–319
 “Tainted” string, 349
 TaintedString class, 349–352
 Telnet, security and, 340, 343
 Thick-client applications,
 transient state and, 225,
 226
 Thinlets, 296
 this parameter, 41, 42
 Thread, deadlocks, 188–193
 Threading, resource
 management and,
 426–428
 Threat model, 333–336, 341
 Throughput, 60, 61
 Throwable class, 57
 Throwable exception, catching,
 56
 Tiers, layers differentiated from,
 31–35
 Tight coupling, 20, 26, 28
 Timer service, 429
 Time-to-live (TTL) value, 221
 TLS. *See* Transport Layer
 Security

TM. *See* Transaction Manager
 Tomcat, 85, 97
 Topology changes, 17
 Transactional affinity
 with EJB, 193–196
 setting effects, 194
 Transactional COM+ (Ewald),
 45
 Transactional execution,
 11–12
 Transactional processing
 EJB restricted to, 64–67
 for implicitly nonatomic
 failure scenarios, 169–175
 Transactional throughput,
 lower isolation levels for,
 211–216
 Transaction Manager, 197
 Transaction processing, 4,
 160
 Transaction Processing
 Monitor, 5
 TransactionRolledBack
 Exception, 57
 TransactionRolledBack
 LocalException, 57
 Transient state, 8, 225
 Transparency
 forms of, 109
 location, 108
 Transport costs, 17
 Transport layer, 102
 Transport Layer Security, 326
 Travel agency application,
 176–178
 Traveling salesman problem,
 262
 Triggers, 266
 Tuples, 237
 Two-phase commit (TPC)
 protocol, 197–198
 Type definitions, relativity of,
 258
 Type safety, 153

U

UDP/IP, 113
 Unsynchronized thread
 example, 181–182, 185
 Unvalidated Parameters, 342
 Update propagation, 144–145,
 146, 220
 UPDATE statements, batching
 and, 276
 URLClassLoader class,
 presentation and, 292–293
 URL rewriting, 287
 User-based security, 259, 356
 User-defined function, SQL
 Server syntax for, 38
 User input
 security and, 16
 validating, 342–352
 User interface, building,
 285–286
 User login page, 343–345
 User role, authorization policy
 configured for, 362–364
 Users, types of, 356
 User transactions, system
 transactions *versus*,
 175–180

V

Validation, 342–343, 348
 and presentation, 311–319
 of user input, 342–352
 validationErrorList tag handler,
 318
 validationErrors tag handler,
 318
 validationErrorText tag handler,
 318
 Value-add, 75, 76
 VBScript, 288, 318
 Vendor neutrality, cost of,
 74–78

470 | *Index*

Verbosity levels, logging, 85
 Versioning, and ClassLoader
 boundaries, 404–408
 Version Number, 201
 Vertical partitioning, 143, 144
 Views, 243
 Virtual memory, performance
 and, 71

W

Wait time, 61, 296
 WeakReference objects, 433,
 437–439, 445
 Web, enterprise systems
 and, 5
 web-app deployment
 descriptor, 308
 Web application, 22, 23, 24, 97
 WebLogic, 229
 WebRowSet, 154
 Web Services, 254
 for open integration,
 147–150
 rich-client applications and,
 352

Web Services Description
 Language, 106
 Web sites, design of, 297–298
 WebSphere, 229
 Wedding ceremony code, 171,
 173–174
 Windows Registry, 91, 92
 Win32 JVM, 447
 Win32 security, 341, 342
 Wizard-style interfaces,
 discarding, 298
 Workflow, 163
 Working Set/Working Set Peak
 sizes, 71
 World Wide Web Consortium,
 148, 288, 306
 writeObject, 412
 writeReplace, 413–415
Writing Secure Code
 (Howard/LeBlanc), 349
 WS-Addressing, 150
 WSDL. *See* Web Services
 Description Language
 WS-Reliable Messaging, 150
 WS-Routing, 53
 WS-Security, 53, 150
 WS-Transaction, 150

X

XCON system (DEC), 165
 XHTML, 298, 299
 XML
 marshaling into, 102, 103
 object-hierarchical
 impedance mismatch and,
 249–259
 XML Infoset Specification, 249,
 254
 XML Schema, 103
 data types defined with,
 257
 -Xmx option, 393
 XSLT, 307–309

Y

Young generation, of garbage
 collection, 418, 420, 421,
 422

Z

Zero Deployment, 32, 287, 289