



Index



A

Abstract Factory Pattern 156. *See also* Factory Pattern

Adapter Pattern

advantages 242

class adapters 244

class diagram 243

combining patterns 504

defined 243

duck magnets 245

Enumeration Iterator Adapter 248

exercise 251

explained 241

fireside chat 247, 252–253

introduction 237

object adapters 244

Alexander, Christopher 602

annihilating evil 606

Anti-Patterns 606–607

Golden Hammer 607

application patterns 604

architectural patterns 604

B

Bridge Pattern 612–613

Builder Pattern 614–615

bullet points 32, 74, 105, 162, 186, 230, 270, 311, 380,
423, 491, 560, 608

business process patterns 605

C

CD Cover Viewer 463

Chain of Responsibility Pattern 616–617

change 339

anticipating 14

constant in software development 8

identifying 53

Choc-O-Holic, Inc. 175

class explosion 81

code magnets 69, 179, 245, 350

cohesion 339–340

Combining Patterns 500

Abstract Factory Pattern 508

Adapter Pattern 504

class diagram 524

Composite Pattern 513

Decorator Pattern 506

Observer Pattern 516

Command Pattern

class diagram 207

command object 203

defined 206–207

introduction 196

loading the Invoker 201

Command Pattern, continued

- logging requests 229
- macro command 224
- Null Object 214
- queuing requests 228
- undo 216, 220, 227

Composite Pattern

- and Iterator Pattern 368
- class diagram 358
- combining patterns 513
- composite behavior 363
- default behavior 360
- defined 356
- interview 376–377
- safety 367
- safety versus transparency 515
- transparency 367, 375
- composition 23, 85, 93, 247, 309
- compound pattern 500, 522
- controlling access 460. *See also* Proxy Pattern
- creating objects 134
- crossword puzzle 33, 76, 163, 187, 231, 271, 310, 378, 490
- cubicle conversation 55, 93, 195, 208, 387, 397, 433, 583–584

D

Decorator Pattern

- and Proxy Pattern 472–473
- class diagram 91
- combining patterns 506
- cubicle conversation 93
- defined 91
- disadvantages 101, 104

fireside chat 252–253

interview 104

introduction 88

in Java I/O 100–101

structural pattern 591

Dependency Inversion Principle 139–143

and the Hollywood Principle 298

Design Patterns

Abstract Factory Pattern 156

Adapter Pattern 243

benefits 599

Bridge Pattern 612–613

Builder Pattern 614–615

categories 589, 592–593

Chain of Responsibility Pattern 616–617

class patterns 591

Command Pattern 206

Composite Pattern 356

Decorator Pattern 91

defined 579, 581

discover your own 586–587

Facade Pattern 264

Factory Method Pattern 134

Flyweight Pattern 618–619

Interpreter Pattern 620–621

Iterator Pattern 336

Mediator Pattern 622–623

Memento Pattern 624–625

Null Object 214

object patterns 591

Observer Pattern 51

organizing 589

Prototype Pattern 626–627

Proxy Pattern 460

Simple Factory 114
 Singleton Pattern 177
 State Pattern 410
 Strategy Pattern 24
 Template Method Pattern 289
 use 29
 versus frameworks 29
 versus libraries 29
 Visitor Pattern 628–629
 Design Principles. *See* Object Oriented Design Principles
 Design Puzzle 25, 133, 279, 395, 468, 542
 Design Toolbox 32, 74, 105, 162, 186, 230, 270, 311,
 380, 423, 491, 560, 608
 DJ View 534
 domain specific patterns 604

E

Elvis 526
 encapsulate what varies 8–9, 75, 136, 397, 612
 encapsulating algorithms 286, 289
 encapsulating behavior 11
 encapsulating iteration 323
 encapsulating method invocation 206
 encapsulating object construction 614–615
 encapsulating object creation 114, 136
 encapsulating requests 206
 encapsulating state 399

F

Facade Pattern
 advantages 260
 and Principle of Least Knowledge 269
 class diagram 264

defined 264
 introduction 258

Factory Method Pattern 134. *See also* Factory Pattern

Factory Pattern

Abstract Factory
 and Factory Method 158–159, 160–161
 class diagram 156–157
 combining patterns 508
 defined 156
 interview 158–159
 introduction 153

Factory Method

advantages 135
 and Abstract Factory 160–161
 class diagram 134
 defined 134
 interview 158–159
 introduction 120, 131–132
 up close 125

Simple Factory

defined 117
 introduction 114

family of algorithms. *See* Strategy Pattern

family of products 145

favor composition over inheritance 23, 75

fireside chat 62, 247, 252, 308, 418, 472–473

Five minute drama 48, 478

Flyweight Pattern 618–619

forces 582

Friedman, Dan 171

G

Gamma, Erich 601

Gang of Four 583, 601

Gamma, Erich 601

Helm, Richard 601

Johnson, Ralph 601

Vlissides, John 601

global access point 177

gobble gobble 239

Golden Hammer 607

guide to better living with Design Patterns 578

Gumball Machine Monitor 431

H

HAS-A 23

Head First learning principles xxx

Helm, Richard 601

Hillside Group 603

Hollywood Principle, The 296

and the Dependency Inversion Principle 298

Home Automation or Bust, Inc. 192

Home Sweet Home Theater 255

Hot or Not 475

I

inheritance

disadvantages 5

for reuse 5–6

versus composition 93

interface 12

Interpreter Pattern 620–621

inversion 141–142

IS-A 23

Iterator Pattern

advantages 330

and collections 347–349

and Composite Pattern 368

and Enumeration 338

and Hashtable 343, 348

class diagram 337

code magnets 350

defined 336

exercise 327

external iterator 338

for/in 349

internal iterator 338

introduction 325

java.util.Iterator 332

Null Iterator 372

polymorphic iteration 338

removing objects 332

J

Johnson, Ralph 601

K

KISS 594

L

Law of Demeter. *See* Principle of Least Knowledge

lazy instantiation 177

loose coupling 53

M

magic bullet 594

master and student 23, 30, 85, 136, 592, 596

Matchmaking in Objectville 475

Mediator Pattern 622–623
 Memento Pattern 624–625
 middleman 237
 Mighty Gumball, Inc. 386
 Model-View-Controller
 Adapter Pattern 546
 and design patterns 532
 and the Web 549
 Composite Pattern 532, 559
 introduction 529
 Mediator Pattern 559
 Observer Pattern 532
 ready-bake code 564–576
 song 526
 Strategy Pattern 532, 545
 up close 530
 Model 2 549. *See also* Model-View-Controller
 and design patterns 557–558
 MVC. *See* Model-View-Controller

N

Null Object 214, 372

O

Objectville Diner 26, 197, 316, 628
 Objectville Pancake House 316, 628
 Object Oriented Design Principles 9, 30–31
 Dependency Inversion Principle 139–143
 encapsulate what varies 9, 111
 favor composition over inheritance 23, 243, 397
 Hollywood Principle 296
 one class, one responsibility 185, 336, 339, 367
 Open-Closed Principle 86–87, 407

Principle of Least Knowledge 265
 program to an interface, not an implementation 11,
 243, 335
 strive for loosely coupled designs between objects that
 interact 53
 Observable 64, 71
 Observer Pattern
 class diagram 52
 code magnets 69
 combining patterns 516
 cubicle conversation 55
 defined 51–52
 fireside chat 62
 Five minute drama 48
 introduction 44
 in Swing 72–73
 Java support 64
 pull 63
 push 63
 one-to-many relationship 51–52
 OOPSLA 603
 Open-Closed Principle 86–87
 oreo cookie 526
 organizational patterns 605

P

 part-whole hierarchy 356. *See also* Composite Pattern
 patterns catalog 581, 583, 585
 Patterns Exposed 104, 158, 174, 377–378
 patterns in the wild 299, 488–489
 patterns zoo 604
 Pattern Honorable Mention 117, 214
 Pizza shop 112
 Portland Patterns Repository 603

- Principle of Least Knowledge 265–268
 - disadvantages 267
- program to an implementation 12, 17, 71
- program to an interface 12
- program to an interface, not an implementation 11, 75
- Prototype Pattern 626–627
- Proxy Pattern
 - and Adapter Pattern 471
 - and Decorator Pattern 471, 472–473
 - Caching Proxy 471
 - class diagram 461
 - defined 460
 - Dynamic Proxy 474, 479, 486
 - and RMI 486
 - exercise 482
 - fireside chat 472–473
 - java.lang.reflect.Proxy 474
 - Protection Proxy 474, 477
 - Proxy Zoo 488–489
 - ready-bake code 494
 - Remote Proxy 434
 - variants 471
 - Virtual Proxy 462
 - image proxy 464
 - publisher/subscriber 45

Q

- Quality, The. *See* Quality without a name
- Quality without a name. *See* Quality, The

R

- refactoring 354, 595
- remote control 193, 209

- Remote Method Invocation. *See* RMI
- remote proxy 434. *See also* Proxy Pattern
- reuse 13, 23, 85
- RMI 436

S

- shared vocabulary 26–28, 599–600
- sharpen your pencil 5, 42, 54, 61, 94, 97, 99, 124, 137, 148, 176, 183, 205, 225, 242, 268, 284, 322, 342, 396, 400, 406, 409, 421, 483, 511, 518, 520, 589
- Simple Factory 117
- SimUDuck 2, 500
- Singleton Pattern
 - advantages 170, 184
 - and garbage collection 184
 - and global variables 185
 - and multithreading 180–182
 - class diagram 177
 - defined 177
 - disadvantages 184
 - double-checked locking 182
 - interview 174
 - up close 173
- Single Responsibility Principle 339. *See also* Object Oriented Design Principles: one class, one responsibility
- skeleton 440
- Starbuzz Coffee 80, 276
- state machines 388–389
- State Pattern
 - and Strategy Pattern 411, 418–419
 - class diagram 410
 - defined 410

- disadvantages 412, 417
- introduction 398
- sharing state 412
- static factory 115
- Strategy Pattern 24
 - and State Pattern 411, 418–419
 - and Template Method Pattern 308–309
 - encapsulating behavior 22
 - family of algorithms 22
 - fireside chat 308
- stub 440

T

- Template Method Pattern
 - advantages 288
 - and Applet 307
 - and `java.util.Arrays` 300
 - and Strategy Pattern 305, 308–309
 - and Swing 306
 - and the Hollywood Principle 297
 - class diagram 289
 - defined 289
 - fireside chat 308–309
 - hook 292, 295
 - introduction 286
 - up close 290–291
- The Little Lisper 171
- thinking in patterns 594–595
- tightly coupled 53

U

- undo 216, 227
- user interface design patterns 605

V

- varies. *See* encapsulate what varies
- Visitor Pattern 628–629
- Vlissides, John 601

W

- Weather-O-Rama 38
- when not to use patterns 596–598
- Who Does What? 202, 254, 298, 379, 422, 487, 588
- Why a duck? 500
- wrapping objects 88, 242, 252, 260, 473, 508. *See also* Adapter Pattern, Decorator Pattern, Facade Pattern, Proxy Pattern

Y

- your mind on patterns 597