Detailed Contents

Preface

Welcome – And Thank You! ..... **xlv**
Target Audience ..... **xlv**
Approach(es) ..... **xlv**
Organization ..... **xlvi**
Part I: The Java Student Survival Guide ..... *xlvi*
Chapter 1 - An Approach To The Art Of Programming ..... *xlvi*
Chapter 2 - Small Victories ..... *xlvi*
Chapter 3 - Project Walkthrough: A Complete Example ..... *xlvii*
Chapter 4 - Computers, Programs, And Algorithms ..... *xlvii*
Part II: Language Fundamentals ..... *xlvii*
Chapter 5 - Overview Of The Java API ..... *xlvii*
Chapter 6 - Simple Java Programs: Using Primitive And Reference Data Types ..... *xlvii*
Chapter 7 - Controlling The Flow Of Program Execution ..... *xlvii*
Chapter 8 - Arrays ..... *xlvii*
Chapter 9 - Toward Problem Abstraction: Creating New Data Types ..... *xlviii*
Chapter 10 - Compositional Design ..... *xlviii*
Chapter 11 - Extending Class Behavior Through Inheritance ..... *xlviii*
Part III: Graphical User Interface Programming ..... *xlviii*
Chapter 12 – Java Swing API Overview ..... *xlviii*
Chapter 13 – Handling GUI Events ..... *xlviii*
Chapter 14 – An Advanced GUI Project ..... *xlviii*
Part IV: Intermediate Concepts ..... *xlix*
Chapter 15 – Exceptions ..... *xlix*
Chapter 16 – Threads ..... *xlix*
Chapter 17 – Collections ..... *xlix*
Chapter 18 – File I/O ..... *xlix*
Part V: Network Programming ..... *xlix*
Chapter 19 – Introduction To Networking And Distributed Applications ..... *l*
Chapter 20 – Client-Server Applications ..... *l*
Chapter 21 – Applets And JDBC ..... *l*
Part VI: Object-Oriented Programming ..... *l*
Chapter 22 – Inheritance, Composition, Interfaces, and Polymorphism ..... *l*
Chapter 23 – Well-Behaved Objects ..... *li*
Chapter 24 – Three Design Principles ..... *li*
Chapter 25 – Helpful Design Patterns ..... *li*
Pedagogy ..... **li**
Learning Objectives ..... *li*
Introduction ..... *li*
Content ..... *li*
Quick Reviews ..... *li*
Summary ..... *lii*
Skill-Building Exercises ..... *lii*
Suggested Projects ..... *lii*
Self-Test Questions ..... *lii*
References ..... *lii*
Notes ..... *lii*
Typographical Formats ..... **lii**
This Is An Example Of A First-Level Subheading ..... *lii*

This Is An Example Of A Second-Level Subheading ..... *lii*
Source Code Formatting ..... *liii*
CD-ROM ..... **liii**
SupportSite™ Website ..... **liii**
Problem Reporting ..... **liii**
About The Authors ..... **liii**
Acknowledgments ..... **liv**

Part I: The Java Student Survival Guide

1 An Approach To The Art Of Programming

Introduction **4**
The Difficulties You Will Encounter Learning Java *4*
Required Skills *4*
The Planets Will Come Into Alignment *4*
How This Chapter Will Help You *5*
Project Management **5**
Three Software Development Roles *5*
Analyst *5*
Architect *5*
Programmer *6*
A Project-Approach Strategy *6*
You Have Been Handed A Project – Now What? *6*
Strategy Areas of Concern *6*
Think Abstractly *8*
The Strategy In A Nutshell *8*
Applicability To The Real World *8*
The Art of Programming **8**
Don't Start At The Computer *9*
Inspiration Strikes At The Weirdest Time *9*
Own Your Own Computer *9*
You Either Have Time and No Money, or Money and No Time *9*
The Family Computer Is Not Going To Cut It! *9*
Set The Mood *10*
Location, Location, Location *10*
Concept Of The Flow *10*
The Stages of Flow *10*
Be Extreme *11*
The Programming Cycle *11*
The Programming Cycle Summarized *12*
A Helpful Trick: Stubbing *12*
Fix The First Compiler Error First *12*
Managing Project Complexity **12**
Conceptual Complexity *12*
Managing Conceptual Complexity *13*
Physical Complexity *13*
The Relationship Between Physical and Conceptual Complexity *14*
Managing Physical Complexity *14*
Maximize Cohesion – Minimize Coupling *14*
Java Source File Structure **14**
SampleClass In Action *15*
General Rules For Creating Java Source Files *16*

Rule-of-Thumb: One Class Per File 17
Avoid Anonymous, Nested, And Inner Classes 17
Create A Separate Main Application File 17
Packages **17**
Commenting **18**
Single-Line Comments 18
Multi-Line Comments 18
Javadoc Comments 19
Generating Javadoc Example Output 19
Identifier Naming - Writing Self-Commenting Code **19**
Benefits of Self-Commenting Code 20
Coding Convention 20
Class Names 21
Constant Names 21
Variable Names 21
Method Names 22
Summary **22**
Skill-Building Exercises **22**
Suggested Projects **24**
Self-Test Questions **24**
References **25**
Notes **25**

2 Small Victories: Creating Java Projects

Introduction **28**
Java Project Creation Process **28**
Source File Creation Stage 29
Source File Compilation Stage 29
What If A Source File Contains Errors? 29
Main Application Execution Stage 29
Automating The Java Project Creation Process 30
Integrated Development Environments **30**
TextPad™ – J2SDK Combination 30
Sun's NetBeans™ 30
Borland® JBuilder® 30
Eclipse® 30
Creating Java Projects Using Microsoft® Windows® 2000/XP **31**
Sun's Java 2 Standard Edition Software Development Kit (J2SDK) 31
Step 1: Download and Install the J2SDK 31
Steps 2 & 3: Setting Microsoft Windows Environment Variables 31
Another Helpful Hint: Display Files Suffixes And Full Path Names 36
Steps 4 & 5: Select A Text Editor And Create The Source Files 36
Step 6: Compiling The Source Files 38
Step 7: Executing The Main Application Class 38
Borland's JBuilder 39
Steps To Creating Java Projects In JBuilder 39
Quick Review 45
Creating Java Projects Using Linux™ **45**
A Brief Note On The Many Flavors Of Linux 46
Obtaining and Installing Sun's J2SDK For Linux 46
Selecting An Installation Location 47
Setting The PATH Environment Variable (bash shell) 48
Setting The CLASSPATH Environment Variable (bash) 49
Creating, Compiling, And Running Java Programs 49
Creating Java Source Files 49

Compiling Java Source Files 50
Running The Application 50
Quick Review 51
Creating Java Projects Using Macintosh OS X Developer Tools **51**
Creating A Java Project With Xcode 51
Quick Review 52
Creating And Running Executable Jar Files **53**
Steps To Create An Executable Jar File 53
Step 1: Create Your Java Program 53
Step 2: Create Manifest File 54
Step 3: Use The jar Command 54
Executing The jar File 54
Quick Review 54
Summary **55**
Skill-Building Exercises **55**
Suggested Projects **56**
Self-Test Questions **56**
References **57**
Notes **57**

3 Project Walkthrough: A Complete Example

Introduction **60**
The Project-Approach Strategy Summarized **60**
Development Cycle **61**
Project Specification **62**
Analyzing The Project Specification 63
Application Requirements Strategy Area 63
Problem Domain Strategy Area 64
Language Features Strategy Area 66
Design Strategy Area 68
Development Cycle First Iteration **69**
Plan (First Iteration) 69
Code (First Iteration) 70
Test (First Iteration) 70
Integrate/Test (First Iteration) 70
Development Cycle Second Iteration **70**
Plan (Second Iteration) 71
Code (Second Iteration) 71
Test (Second Iteration) 71
Integrate/Test (Second Iteration) 72
Development Cycle Third Iteration **72**
Plan (Third Iteration) 72
Code (Third Iteration) 73
Test (Third Iteration) 75
Integrate/Test (Third Iteration) 75
Development Cycle Fourth Iteration **75**
Plan (Fourth Iteration) 75
Implementing State Transition Diagrams 76
Implementing The printFloor() Method 77
Code (Fourth Iteration) 77
Test (Fourth Iteration) 79
Integrate/Test (Fourth Iteration) 79
Development Cycle Fifth Iteration **79**
Plan (Fifth Iteration) 80
Code (Fifth Iteration) 81

Test (Fifth Iteration) *82*
Integrate/Test (Fifth Iteration) *82*
Some Final Considerations **82**
Complete RobotRat.java Source Code Listing **84**
Summary **88**
Skill-Building Exercises **88**
Suggested Projects **88**
Self-Test Questions **88**
References **89**
Notes **89**

4 Computers, Programs, and Algorithms

Introduction **92**
What Is A Computer? **92**
Computer vs. Computer System *92*
Computer System *92*
Processor *94*
Three Aspects of Processor Architecture *95*
Feature Set *95*
Feature Set Implementation *95*
Feature Set Accessibility *95*
Memory Organization **95**
Memory Basics *96*
Memory Hierarchy *96*
Bits, Bytes, Words *96*
Alignment and Addressability *97*
What Is A Program? **98**
Two Views of a Program *98*
The Human Perspective *98*
The Computer Perspective *98*
The Processing Cycle **98**
Fetch *99*
Decode *99*
Execute *99*
Store *99*
Why A Program Crashes *99*
Algorithms **99**
Good vs. Bad Algorithms *100*
Don't Reinvent The Wheel! *102*
The Java HotSpot™ Virtual Machine **102**
Obtaining The Java HotSpot™ Virtual Machine *102*
Client & Server Virtual Machines *102*
Classic VM vs. JIT vs. HotSpot™ *103*
Java HotSpot™ Virtual Machine Architecture *103*
Services Provided By The Java HotSpot™ Virtual Machine and the JRE **103**
Summary **104**
Skill-Building Exercises **104**
Suggested Projects **104**
Self-Test Questions **105**
References **105**
Notes **105**

Part II: Language Fundamentals

5 Overview Of The Java Platform API

Introduction **110**
Java Platform API Packages **110**
Obtaining Detailed Java Platform API Information *111*
Navigating A Class Inheritance Hierarchy **112**
Deprecated Methods *115*
Java Platform Packages Used In This Book **116**
Summary **117**
Skill-Building Exercises **117**
Suggested Projects **118**
Self-Test Questions **118**
References **118**
Notes **119**

6 Simple Java Programs: Using Primitive And Reference Data Types

Introduction **122**
Terminology **122**
Definition Of A Java Program **123**
Application Objects *123*
Talking About Applications *123*
Applet Objects *123*
Creating Simple Java Programs (Applications) **123**
Structure Of A Java Application *124*
Compiling and Executing SimpleApplication *125*
Quick Review *125*
Building Bigger Applications *126*
Identifiers and Identifier Naming Rules **126**
Well-Named Identifiers Will Simplify Your Life *127*
Java Reserved Keywords **127**
Java Type Categories **129**
Primitive Data Types *129*
Reference Data Types *130*
Array Types *130*
Working With Primitive Types **130**
Application Class Structure *130*
Definition Of The Term "Variable" *131*
Declaring And Using Variables In The Main() Method *131*
Definition Of The Term "Constant" *132*
Declaring And Using Constants In The Main() Method *132*
Getting Simple Input Into Your Program *133*
Using The main() Method's String Array To Accept Input At Program Startup *133*
Working With Reference Types **134**
Definition Of A Reference Variable *135*
Definition Of An Object *135*
Creating Objects with the new Operator *135*
Garbage Collection *136*
Accessing Class Members From the Main() Method **136**
The main() Method Is A Static Method *137*
Definition Of The Term Class Variable/Constant *137*

Definition Of The Term Instance Variable/Constant *137*
An Example *137*
Statements And Expressions **138**
Operators **139**
How To Read The Operator Table *139*
Use Parentheses To Explicitly Force Precedence *141*
Java Operators In Detail *141*
*Arithmetic Operators +, -, *, /* *141*
String Concatenation Operator + ... *142*
Modulus (Integer Remainder) Operator % *143*
Greater-Than/Less-Than Operators >, >=, <, <= *143*
Equality and Inequality Operators ==, != *144*
Conditional AND and OR Operators &&, || *144*
Boolean AND, OR, XOR, and NOT Operators &, |, ^, ! *145*
Ternary Conditional Operator ?: *145*
Left Shift and Right Shift Operators <<, >>, >>> *146*
instanceof Operator *147*
Unary Prefix And Postfix Increment And Decrement Operators ++, - *147*
Member Access Operator *147*
Bitwise AND, OR, XOR, and Complement Operators &, |, ^, ~ *148*
*Combination Assignment Operators +=, -=, *=, /=, <<=, >>=, >>>=, &=, |=, ^=* *148*
Java Primitive Type Wrapper Classes **149**
Summary **150**
Skill-Building Exercises **150**
Suggested Projects **151**
Self-Test Questions **152**
References **152**
Notes **153**

7 Controlling The Flow Of Program Execution

Introduction **156**
Selection Statements **156**
if Statement *156*
Executing Code Blocks in if Statements *157*
Executing Consecutive if Statements *157*
if/else Statement *158*
Chained if/else Statements *159*
switch Statement *159*
Nested Switch Statement *161*
Quick Review *162*
Iteration Statements **162**
while Statement *162*
Personality of the while Statement *163*
do/while Statement *164*
Personality of the do/while Statement *164*
for Statement *165*
How The for Statement is Related to the While Statement *165*
Personality of the for Statement *165*
Nesting Iteration Statements *166*
Mixing Selection & Iteration Statements: A Powerful Combination *166*
Quick Review *168*
break and continue **168**
Unlabeled break *168*
Labeled break *169*
Unlabeled Continue *170*

Labeled Continue *170*
Quick Review *171*
Selection and Iteration Statement Selection Table **171**
Summary **173**
Skill-Building Exercises **173**
Suggested Projects **175**
Self-Test Questions **176**
References **177**
Notes **177**

8 Arrays

Introduction **180**
What Is An Array? **180**
Specifying Array Types *181*
Quick Review *182*
Functionality Provided By Java Array Types **182**
Array-Type Inheritance Hierarchy *182*
The java.lang.reflect.Array Class *182*
Special Properties Of Java Arrays *183*
Quick Review *183*
Creating And Using Single-Dimensional Arrays **183**
Arrays Of Primitive Types *183*
How Primitive Type Array Objects Are Arranged In Memory *184*
Calling Object and Class Methods on Array References *185*
Creating Single-Dimensional Arrays Using Array Literal Values *186*
Differences Between Arrays Of Primitive Types And Arrays Of Reference Types *187*
Single-dimensional Arrays In Action *187*
Message Array *188*
Calculating Averages *191*
Histogram: Letter Frequency Counter *192*
Quick Review *193*
Creating And Using Multidimensional Arrays **194**
Multidimensional Array Declaration Syntax *194*
Memory Representation Of A Two Dimensional Array *196*
Creating Multidimensional Arrays Using Array Literals *196*
Ragged Arrays *197*
Multidimensional Arrays In Action *198*
Weighted Grade Tool *198*
Quick Review *199*
The main() method's String Array **200**
Purpose And Use Of The main() Method's String Array *200*
Manipulating Arrays With The java.util.Arrays Class **201**
Java API Classes Used In This Chapter **202**
Summary **203**
Skill-Building Exercises **203**
Suggested Projects **204**
Self-Test Questions **206**
References **207**
Notes **207**

9 Toward Problem Abstraction: Creating New Data Types

Introduction **210**
Abstraction: Amplify The Essential – Eliminate The Irrelevant **210**

Abstraction Is The Art Of Programming 210
Where Problem Abstraction Fits Into The Development Cycle 210
Creating Your Own Data Types 211
Case-Study Project: Write A People Manager Program 211
Quick Review 213
The UML Class Diagram **213**
Quick Review 214
Overview Of The Java Class Construct **214**
Four Categories Of Class Members 214
Static or Class-Wide Fields 214
Non-Static or Instance Fields 215
Static or Class-Wide Methods 215
Non-Static or Instance Methods 215
Access Modifiers 215
public 215
private 215
protected 216
package 216
The Concepts Of Horizontal Access, Interface, and Encapsulation 216
Quick Review 217
Methods **217**
Method Naming – Use Action Words That Indicate The Method's Purpose 217
Maximize Method Cohesion 217
Structure Of A Method Definition 217
Method Modifiers (optional) 218
Return Type or void (optional) 219
Method Name (mandatory) 219
Parameter List (optional) 219
Method Body (optional for abstract or native methods) 219
Example Method Definitions 219
Method Signatures 220
Overloading Methods 220
Constructor Methods 220
Quick Review 220
Building And Testing The Person Class **221**
Start By Creating The Source File And Class Definition Shell 221
Defining Person Instance Fields 221
Defining Person Instance Methods 221
The Constructor Method 222
Adding A Few Accessor Methods 222
Testing The Person Class 223
Use The PeopleManagerApplication Class As A Test Driver 223
Adding Features To The Person Class – Calculating Age 224
Adding Features To The Person Class – Convenience Methods 225
Adding Features To The Person Class – Finishing Touches 226
Quick Review 227
Building And Testing The PeopleManager Class **228**
Defining The PeopleManager Class Shell 228
Defining PeopleManager Fields 228
Defining PeopleManager Constructor Methods 228
Defining Additional PeopleManager Methods 229
Testing The PeopleManager Class 230
Adding Features To The PeopleManager Class 230
Quick Review 232
More About Methods **232**
How Arguments Are Passed To Methods 232
Local Variable Scoping 233
Anywhere An Object Of <type> Is Required, A Method That Returns <type> Can Be Used 233

Quick Review *234*
Special Topics **234**
Static Initializers *234*
Data Encapsulation - The Naked Truth *235*
Quick Review *236*
Summary **236**
Skill-Building Exercises **237**
Suggested Projects **238**
Self-Test Questions **239**
References **241**
Notes **241**

10 Compositional Design

Introduction **244**
Managing Conceptual And Physical Complexity **244**
Compiling Multiple Source Files Simultaneously with javac *244*
Quick Review *245*
Dependency vs. Association **245**
Aggregation **245**
Simple vs. Composite Aggregation *246*
The Relationship Between Aggregation and Object Lifetime *246*
Quick Review *246*
Expressing Aggregation In A UML Class Diagram **246**
Simple Aggregation *246*
Composite Aggregation *247*
Aggregation Example Code **247**
Simple Aggregation Example *248*
Composite Aggregation Example *249*
Quick Review *250*
Sequence Diagrams **250**
Quick Review *251*
The Aircraft Engine Simulation: An Extended Example **251**
The Purpose of the Engine Class *252*
Engine Class Attributes and Methods *252*
PartStatus –A Typesafe Enumeration Pattern *254*
Aircraft Engine Simulation Sequence Diagrams *254*
Running The Aircraft Engine Simulation Program **256**
Quick Review *256*
Complete Aircraft Engine Simulation Code Listing **257**
Summary **260**
Skill-Building Exercises **260**
Suggested Projects **261**
Self-Test Questions **262**
References **263**
Notes **263**

11 Extending Class Behavior Through Inheritance

Introduction **266**
Three Purposes Of Inheritance **266**
Implementing The "is a" Relationship *267*
The Relationship Between The Terms Type, Interface, and Class *267*
Meaning Of The Term Interface *267*

Meaning Of The Term Class 267
Quick Review 268
Expressing Generalization & Specialization In The UML 268
A Simple Inheritance Example 269
The UML Diagram 269
BaseClass Source Code 269
DerivedClass Source Code 270
DriverApplication Program 270
Quick Review 271
Another Inheritance Example: Person - Student 271
The Person - Student UML Class Diagram 271
Person - Student Source Code 272
Casting 274
Use Casting Sparingly 275
Quick Review 275
Overriding Base Class Methods 275
Quick Review 276
Abstract Methods & Abstract Base Classes 276
The Primary Purpose Of An Abstract Base Class 276
Expressing Abstract Base Classes In UML 277
Quick Review 279
Interfaces 279
The Purpose Of Interfaces 279
Authorized Interface Members 279
The Differences Between An Interface And An Abstract Class 280
Expressing Interfaces In UML 280
Expressing Realization In A UML Class Diagram 280
An Interface Example 281
Quick Review 282
Controlling Horizontal & Vertical Access 283
Quick Review 285
Final Classes & Methods 285
Quick Review 285
Polymorphic Behavior 285
Quick Review 285
Inheritance Example: Employee 286
Inheritance Example: Aircraft Engine Simulation 288
Aircraft Engine Simulation UML Diagram 288
Simulation Operational Description 290
Take A Deep Breath And Relax! 290
Compiling The Aircraft Engine Simulation Code 290
Complete Aircraft Engine Simulation Code Listing 290
Terms & Definitions 297
Summary 297
Skill-Building Exercises 298
Suggested Projects 299
Self-Test Questions 300
References 301
Notes 302

Part III: Graphical User Interface Programming

12 Java Swing API Overview

Introduction **306**
AWT and Swing **306**
Naming Conventions *307*
The Mathematics of GUIs **307**
Coordinate Systems *307*
Components and Their Bounds *308*
Top-Level Containers **308**
Window and JWindow *309*
Frame and JFrame *309*
Dialog and JDialog *309*
Our First GUI Program *310*
Explanation of the Code *311*
Top-Level Containers API *312*
Top-Level Container Constructors *312*
Dialogue with a Skeptic *313*
Top-Level Container Methods *314*
Quick Review *316*
Organizing Components into Containers **316**
Absolute Positioning *316*
Layout Managers *318*
FlowLayout *318*
GridLayout *320*
BorderLayout *321*
GridBagLayout *323*
Combining LayoutManagers Using JPanels *327*
Quick Review *328*
The Components **329**
Brief Descriptions *329*
Component, Container, and JComponent Methods *331*
Component Method Tables *332*
Container Method Tables *335*
JComponent Method Tables *336*
Quick Review *337*
The Final GUI **337**
Highlights of the Final GUI *341*
Layout of the Final GUI *342*
Summary **344**
Skill-Building Exercises **345**
Suggested Projects **346**
Self-Test Questions **346**
References **347**
Notes **347**

13 Handling GUI Events

Introduction **350**
Event-Handling Basics **350**
The Event *350*
The Event Listener *351*
Registering an Event Listener with a Component *351*

Using the API 352
Choosing the Right Event 352
Quick Review 354
Creating Event Listeners **354**
ListeningMainFrame0 354
ListeningMainFrame1 355
Step 1. Application Behavior 355
Step 2. The Source 355
Step 3. The Event 355
Step 4. The Listener 357
Step 5: The Design Approach: External Class 358
Step 6: Registration 358
ListeningMainFrame2 358
Step 1: Application Behavior 359
Step 2: The Source 359
Step 3: The Event 359
Step 4: The Listener 361
Step 5: The Design Approach: External class with fields 362
Step 6: Registration 363
ListeningMainFrame3 364
Step 1: Application Behavior 364
Step 2: The Source 364
Step 3: The Event 364
Step 4: The Listener 365
Step 5: The Design Approach: Inner Class (field-level) 366
Step 6: Registration 367
ListeningMainFrame4 367
Step 1: Application Behavior 367
Step 2: The Source 367
Step 3: The Event 367
Step 4: The Listener 367
Step 5: The Design Approach: Inner Class (local-variable-level) 368
Step 6: Registration 368
ListeningMainFrame5 368
Step 1: Application Behavior 369
Step 2: The Source 369
Step 3: The Event 369
Step 4: The Listener 369
Step 5: The Design Approach: Existing Class 369
Step 6: Registration 371
ListeningMainFrame6 371
Step 1: Application Behavior 371
Step 2: The Source 371
Step 3: The Event 371
Step 4: The Listener 372
Step 5: The Design Approach: Anonymous Class 372
Step 6: Registration 372
ListeningMainFrame7 373
Step 1: Application Behavior 373
Step 2: The Source 373
Step 3: The Event 373
Step 4: The Listener 373
Step 5: The Design Approach: Anonymous Class and Instance 374
Step 6: Registration 374
Adapters **374**
Summary **375**
Skill-Building Exercises **376**
Suggested Projects **376**

Self-Test Questions **377**
References **377**
Notes **377**

14 An Advanced GUI Project

Introduction **380**
The Project *380*
The Approach *380*
Step 1 – Laying the Groundwork **381**
The Garment Class *381*
The DressingBoard Class *382*
Graphics and the AWT/Swing Framework *382*
Graphics (The "How" of Drawing) *382*
The AWT/Swing Framework (The "When" of Drawing) *384*
Calling Repaint *385*
Obtaining a Graphics2D *386*
The DressingBoard Class (continued) *386*
Loading an Image or Resource *387*
The MainFrame Class *388*
Quick Review *389*
Step 2 – Swing's Separable Model Architecture **389**
Using a ListModel *389*
The MainFrame Class *390*
Quick Review *392*
Step 3 – Writing a Custom Renderer **392**
Using a Renderer *393*
The CheckboxListCell Class *394*
The MainFrame Class *396*
Quick Review *398*
Step 4 –The Art of Illusion **398**
The CheckboxListCell Class *401*
The MainFrame Class *402*
Quick Review *403*
Step 5 – Defining Your Own Events and EventListeners **403**
The CheckboxListCell Class *403*
The MainFrame Class *406*
Quick Review *408*
Interlude – Writing a Custom Editor **408**
The DemoTreeOrTableCellHandler Class *410*
The DemoFrame Class *412*
Quick Review *413*
Step 6 – A Good Component Gets Better **413**
The DragList Class – Overview *414*
Algorithm *414*
Use DefaultListModel (Lines 41 - 65) *414*
ReorderEvent and ReorderListener (Lines 184 - 203) *414*
The DragList Class – Dragging *415*
dragIndex, dragItem and dragRect Attributes *415*
dragStart, dragThreshold and allowDrag Attributes *415*
deltaY and inDrag Attributes *415*
How the inDrag Attribute is Used *416*
The DragList Class – Painting *416*
The createDragImage() Method (Lines 66 - 94) *416*
The paintComponent() Method (Lines 95 - 108) *417*
The DragList Class – Code *417*
The MainFrame Class *419*

Summary 421
Skill-Building Exercises 422
Suggested Projects 422
Self-Test Questions 423
References 423
Notes 423

Part IV: Intermediate Concepts

15 Exceptions

Introduction 428
The Throwable Class Hierarchy 428
Errors *429*
Exceptions *429*
RuntimeExceptions *429*
Checked vs. Unchecked Exceptions *430*
Quick Review *430*
Handling Exceptions 430
Try/Catch Statement *431*
Multiple Catch Blocks *431*
The Importance Of Exception Type And Order In Multiple Catch Blocks *432*
Manipulating A Throwable Object *433*
Try/Catch/Finally Statement *435*
Try/Finally Statement *435*
Quick Review *436*
Throwing Exceptions 436
The Throws Clause *436*
The throw Keyword *437*
Quick Review *438*
Creating Custom Exceptions 438
High & Low Level Abstractions *438*
Extending The Exception Class *438*
Some Advice On Naming Your Custom Exceptions *440*
Quick Review *440*
Summary 440
Skill-Building Exercises 441
Suggested Projects 441
Self-Test Questions 441
References 441
Notes 442

16 Threads

Introduction 444
What is a Thread? 444
Quick Review *446*
Building a Clock Component 446
currentThread(), sleep(), interrupt(), interrupted() and isInterrupted() *447*
Creating Your Own Threads *449*
Quick Review *451*

Computing Pi **452**
Quick Review *455*
Thread Priority and Scheduling **455**
Your Results May Vary *457*
Quick Review *458*
Race Conditions **459**
The Mechanics of Synchronization *460*
Three Basic Rules of Synchronization *462*
Synchronization Rule 1 ..... *462*
Synchronization Rule 2 ..... *463*
Synchronization Rule 3 ..... *465*
Synchronizing Methods *466*
Quick Review *467*
The Producer-Consumer Relationship **468**
Quick Review *473*
Deadlock **474**
Quick Review *476*
About the Pi-Generating Algorithm **477**
Summary **477**
Skill-Building Exercises **478**
Suggested Projects **479**
Self-Test Questions **479**
References **480**
Notes **480**

17 Collections

Introduction **482**
Case Study: Building A Dynamic Array **482**
Evaluating DynamicArray *484*
The ArrayList Class To The Rescue *484*
The Power Of Collections – Polymorphic Code *485*
Java Collections Framework Overview **486**
The Purpose Of The Collections Framework *486*
Framework Organization *486*
Core Collections Interfaces *486*
General Purpose Implementation Classes *487*
Mapping An Implementation Class To Its Underlying Data Structure *492*
Algorithms *492*
Infrastructure *493*
Quick Review *493*
Java 1.4.2 Style Collections **493**
Creating A Set From A List *494*
PeopleManager Revisited *494*
Casting Retrieved Objects *496*
Creating New Data Structures From Existing Collections *497*
Quick Review *499*
Java 5 Style Collections: Generics **499**
Java 5 Collection Framework Core Interfaces *499*
Java 5 Collections Framework Sample Programs *500*
SetTestApp Program Revised *500*
When To Use The Enhanced For Loop *501*
Static Polymorphism – Generic Methods *501*
Quick Review *502*
Moving Forward *502*
Summary **503**

Skill-Building Exercises **503**
Suggested Projects **504**
Self-Test Questions **504**
References **505**
Notes **505**

18 File I/O

Introduction **508**
Java I/O Package Overview **508**
File Class *508*
InputStream Classes *508*
OutputStream Classes *510*
Reader Classes *510*
Writer Classes *510*
RandomAccessFile Class *510*
How Do You Choose Between Byte Streams, Character Streams, and RandomAccessFile? *510*
Another Way To Categorize The Java I/O Classes *510*
Quick Review *512*
Using The File Class **513**
Quick Review *515*
Sequential Byte Stream File I/O Using Input- & OutputStreams **515**
OutputStreams *515*
FileOutputStream *515*
BufferedOutputStream *516*
DataOutputStream *517*
ObjectOutputStream *518*
PrintStream *520*
InputStreams *520*
FileInputStream *521*
BufferedInputStream *521*
DataInputStream *522*
ObjectInputStream *523*
Quick Review *523*
Sequential Character Stream File I/O Using Readers & Writers **524**
Writers *524*
FileWriter *524*
BufferedWriter *525*
OutputStreamWriter *526*
PrintWriter *526*
Readers *527*
FileReader *527*
InputStreamReader *527*
BufferedReader *528*
Quick Review *529*
A Practical File I/O Example: The java.util.Properties Class **529**
Quick Review *531*
Random Access File I/O **531**
Towards An Approach To The Adapter Project *533*
Start Small And Take Baby Steps *533*
Other Project Considerations *534*
Locking A Record For Updates And Deletes *534*
Translating Low-Level Exceptions Into Higher-Level Exception Abstractions *534*
Where To Go From Here *534*
Complete RandomAccessFile Legacy Datafile Adapter Source Code Listing *535*
Quick Review *544*

Quick File I/O Combination Reference 545
Summary 546
Skill-Building Exercises 547
Suggested Projects 547
Self-Test Questions 548
References 549
Notes 549

Part V: Network Programming

19 Introduction to Networking And Distributed Applications

Introduction 554
What Is A Computer Network? 554
Purpose Of A Network 554
The Role Of Network Protocols 555
Homogeneous Vs. Heterogeneous Networks 555
The Unifying Network Protocols: TCP/IP 555
What's So Special About The Internet? 556
Quick Review 556
Servers & Clients 557
Server Hardware And Software 557
Client Hardware And Software 557
Quick Review 558
Application Distribution 558
Physical Distribution On One Computer 558
Running Multiple Java Virtual Machines On The Same Computer 558
Starting Multiple JVMs In Windows 558
Starting Multiple JVMs In Unix And Linux 559
Running Multiple Clients On The Same Computer 561
Addressing The Local Machine 561
Physical Distribution Across Multiple Computers 561
Multiple Java Virtual Machines 562
Quick Review 562
Multi-Tiered Applications 562
Logical Application Tiers 562
Physical Tier Distribution 563
Quick Review 564
Internet Networking Protocols – Nuts & Bolts 564
The Internet Protocols: TCP, UDP, And IP 564
The Application Layer 564
Transport layer 565
Network Layer 565
Data Link & Physical Layers 565
Putting It All Together 566
What You Need To Know 566
Quick Review 566
Java Support For Network Programming 567
A Network Programming Scenario Using Sockets – Overview 567
The URL Class 568
Quick Review 569
Remote Method Invocation (RMI) Overview 569
Creating & Running An RMI Application In Java 1.4.2 570

Required Steps *570*
Step 1: Define A Remote Interface *570*
Step 2: Define A Remote Interface Implementation *571*
Step 3: Compile The Remote Interface & Remote Interface Implementation Source Files *571*
Step 4: Use The rmic Tool To Generate Stub Class *571*
Step 5: Create Server Application *572*
Step 6: Create Client Application *573*
Step 7: Start The RMI Registry *573*
Step 8: Run The Server Application *574*
Step 9: Run The Client Application *574*
Creating & Running An RMI Application In Java 1.5 ..... *575*
Considerations When Deploying RMI Applications In Mixed JVM Version Environments *575*
Server JVM 1.5 - Client JVM 1.5 ..... *575*
Server JVM 1.5 - Client JVM 1.4.2 ..... *575*
Server JVM 1.4.2 - Client JVM 1.5 ..... *575*
Server JVM 1.4.2 - Client JVM 1.4.2 ..... *575*
Quick Review *576*
Summary **576**
Skill-Building Exercises **577**
Suggested Projects **578**
Self-Test Questions **578**
References **579**
Notes **579**

20 Client-Server Applications

Introduction **582**
A Simple Socket-Based Client-Server Example **582**
Simple Server Application *583*
Simple Client Application *585*
Running The Examples *586*
A Few Words About Protocol *588*
Quick Review *588*
Project Specification **588**
Client-Server Project Initial Design Considerations **589**
Noun-Verb Analysis *589*
High-Level Application Operations *590*
How To Proceed *592*
Quick Review *592*
Implementing NetRatServer – First Iteration **592**
First Iteration Code *593*
Testing The Fist Iteration Code *595*
Quick Review *596*
Implementing NetRatServer - Second Iteration **596**
Second Iteration Code *597*
Testing Second Iteration Code *599*
Quick Review *599*
The RMI-Based Client **600**
Testing Multiple RMI Clients *602*
Quick Review *602*
NetRatServer Implementation – Third Iteration **602**
Updated Server Application Class Diagram *602*
Third Iteration Code *603*
Testing The Third Iteration Code *604*
Testing Multiple RMI Client Applications *606*
Quick Review *607*

NetRatServer Implementation – Fourth Iteration 607
Fourth Iteration Code 608
Testing Fourth Iteration Code 610
Testing The Fourth Iteration Code 613
Quick Review 613
NetRatServer Implementation – Final Iteration 613
Final Iteration Code 615
Testing The Final Iteration Code 619
Parting Comments 619
Quick Review 620
Summary 620
Skill-Building Exercises 621
Suggested Projects 622
Self-Test Questions 622
References 623
Notes 623

21 Applets & JDBC

Introduction 626
Applet Overview 626
The Benefits Of Using Applets 626
<applet> Vs. <object> Tags 627
Browser JVM Vs. Java Plug-In 627
A Basic Applet Example 627
Applet Code 627
HTML Page Code 628
Packaging And Distribution 628
Running The Basic Applet Example 628
Applet Life-Cycle Stages 629
The <applet> Tag In Detail 630
Quick Review 631
Applet Security Restrictions 631
Miscellaneous Applet Security Restrictions 634
Security Policies & Signed Applets 634
Quick Review 634
Using Applet Parameters 634
Quick Review 636
Extended Applet Example – Poetry In Browser 636
Java Database Connectivity (JDBC™) Overview 639
JDBC™ Packages 639
JDBC™ Specification 639
Using JDBC – A Macro View 640
Properly Configured Database 640
JDBC Driver Class 640
Knowledge of Relational Database Design 640
Knowledge of the Database Schema 640
knowledge of SQL 640
The Java JDBC Program 640
General Steps Required To Employ JDBC 641
Moving Forward 641
Quick Review 641
JDBC Project Description 641
Overall System Architecture 641
Detailed System Architecture 641
Role Of The Persister Class 643

Quick Review *643*
MySQL Database & JDBC **643**
Setting Up MySQL To Run The Example Code *643*
Obtaining and Installing MySQL *644*
The mysqladmin Program *644*
The mysql Monitor Program *645*
Helpful MySQL Monitor Commands *646*
Creating The chapter_21 Database *647*
Establishing Database Security Via MySQL Access Control Tables *647*
General Strategy For Managing Permissions Using The MySQL Access Control Tables *649*
Creating The Employee Training Management System Tables *650*
Populating Tables with Data and Running Queries *651*
Inserting Data Into Tables *651*
Querying a Table with the select Statement *652*
Updating Data In A Table *652*
Joining Tables With A Select Statement *652*
Accessing chapter_21 Database Tables Via JDBC *654*
A Short JDBC Example Program *654*
Accessing ResultSet MetaData *655*
Quick Review *656*
Extended Applet & JDBC Example **656**
The Code *656*
Compiling And Packaging The Applet Code For Distribution *667*
Running The Example *668*
Start MySQL Database Application *668*
Start The DBServerApp Program *668*
Access The employeetraining.html Page *668*
Using The Employee Training Applet *668*
Code Discussion *670*
Employee and EmployeeTraining Classes *670*
AddNewEmployeeDialog and AddTrainingRecordDialog Classes *671*
EmployeeTrainingApplet Class *671*
DBServerApp Class *671*
Persister Class *671*
DBServerProperties *671*
Statements vs. Prepared Statements *672*
Summary **672**
Skill-Building Exercises **673**
Suggested Projects **674**
Self-Test Questions **675**
References **675**
Notes **676**

Part VI: Object-Oriented Design

22 Inheritance, Composition, Interfaces, Polymorphism

Introduction **680**
Inheritance Vs. Composition: The Great Debate **680**
What's The End Game? *681*
Flexible Application Architectures *681*
Modularity And Reliability *681*
Architectural Stability Via Managed Dependencies *681*

Knowing When To Accept A Design That's Good Enough *682*
Quick Review *682*
Inheritance-Based Design **682**
Three Good Reasons To Use Inheritance *682*
As A Means To Reason About Code Behavior *682*
To Gain A Measure Of Code Reuse *682*
To Facilitate Incremental Development *682*
Forms Of Inheritance: Meyer's Inheritance Taxonomy *683*
Coad's Inheritance Criteria *684*
Person - Employee Example Revisited *685*
Quick Review *685*
The Role Of Interfaces **686**
Reducing or Limiting Intermodule Dependencies *686*
Modeling Dominant, Collateral, and Dynamic Roles *686*
Dominant Roles *687*
Collateral Roles *687*
Dynamic Roles *687*
Quick Review *687*
Applied Polymorphism **687**
Quick Review *688*
Composition-Based Design As A Force Multiplier **688**
Two Types Of Aggregation *688*
Polymorphic Containment *688*
An Extended Example *689*
Quick Review *693*
Summary **693**
Skill-Building Exercises **694**
Suggested Projects **695**
Self-Test Questions **696**
References **696**
Notes **697**

23 Well-Behaved Objects

Introduction **700**
Consider Object Usage From The Start **700**
Applying The Object Usage Scenario Checklist *701*
Quick Review *702*
Overriding java.lang.Object Methods **703**
Overriding toString() *703*
Overriding equals() And hashCode() *703*
equals() Method *703*
Testing Person.toString() and Person.equals() Methods *705*
hashCode() Method *705*
Overriding hashCode() In The Person Class *707*
Testing Person.hashCode() *707*
Overriding clone() *708*
Shallow Copy vs. Deep Copy *708*
Overriding Person.clone() *709*
Overriding finalize() – And An Alternative Approach *710*
Quick Review *710*
Implementing The java.lang.Comparable Interface **711**
Quick Review *712*
Using Person Objects In A Collection **713**
Implementing java.util.Comparator Interface **713**
Quick Review *715*

Implementing The Serializable Interface **716**
Person Class – The Final Version **716**
Summary **717**
Skill-Building Exercises **718**
Suggested Projects **718**
Self-Test Questions **718**
References **719**
Notes **719**

24 Three Design Principles

Introduction **722**
The Preferred Characteristics of an Object-Oriented Architecture **722**
Easy to Understand – (How does this thing work?) *722*
Easy to Reason About – (What are the effects of change?) *722*
Easy to Extend – (Where do I add functionality?) *722*
The Liskov Substitution Principle & Design by Contract **723**
Reasoning About the Behavior of Supertypes and Subtypes *723*
Relationship Between the LSP and DbC *723*
The Common Goal of the LSP and DbC *723*
Java Support for the LSP and DbC *723*
Designing with the LSP/DbC in Mind *724*
Class Declarations Viewed as Behavior Specifications *724*
Quick Review *724*
Preconditions, Postconditions, and Class Invariants **724**
Class Invariant *724*
Precondition *724*
Postcondition *725*
An Example *725*
A Note On Using Assertions To Enforce Pre- and Postconditions *726*
Using Incrementer As A Base Class *726*
Changing the Preconditions of Derived Class Methods *728*
Changing the Postconditions of Derived Class Methods *731*
Special Cases of Preconditions and Postconditions *732*
Method Argument Types *732*
Method Return Types *735*
Three Rules of the Substitution Principle *735*
Signature Rule *735*
Methods Rule *735*
Properties Rule *735*
Quick Review *735*
The Open-Closed Principle **735**
Achieving The Open-Closed Principle *736*
An OCP Example *736*
Quick Review *739*
The Dependency Inversion Principle **740**
Characteristics of Bad Software Architecture *740*
Characteristics of Good Software Architecture *741*
Selecting The Right Abstractions Takes Experience *741*
Quick Review *741*
Terms and Definitions **741**
Summary **742**
Skill-Building Exercises **743**
Suggested Projects **743**
Self-Test Questions **743**
References **744**

Notes **744**

25 Helpful Design Patterns

Introduction **746**
Software Design Patterns And How They Came To Be **746**
What Exactly Is A Software Design Pattern? *746*
Origins *746*
Pattern Specification *747*
Applying Software Design Patterns *747*
Quick Review *747*
The Singleton Pattern **748**
Quick Review *749*
The Factory Pattern **749**
The Dynamic Factory *750*
Advantages Of The Dynamic Factory Pattern *751*
Quick Review *751*
The Model-View-Controller Pattern **751**
Potential Issues With One ActionListener *753*
Quick Review *753*
The Command Pattern **753**
Quick Review *756*
A Comprehensive Pattern-Based Example **756**
Code Listing By Package Name *756*
com.pulpfreepress.commands *756*
com.pulpfreepress.controller *759*
com.pulpfreepress.exceptions *760*
com.pulpfreepress.interfaces *760*
com.pulpfreepress.model *761*
com.pulpfreepress.utils *767*
com.pulpfreepress.view *769*
Summary **776**
Skill-Building Exercises **777**
Suggested Projects **777**
Self-Test Questions **777**
References **778**
Notes **778**

Appendices

Appendix A: Helpful Checklists And Tables

Project-Approach Strategy Check-off List **781**
Development Cycle **782**

Appendix B: ASCII Table

ASCII Table **783**

List of Tables

Table 1-1: Java Source File Rules Summary 17
Table 1-2: Class Naming Examples 21
Table 1-3: Constant Naming Examples 21
Table 1-4: Variable Naming Examples 21
Table 1-5: Method Naming Examples 22
Table 2-1: Helpful Operating System Commands 56
Table 3-1: Project Approach Strategy 60
Table 3-2: Development Cycle 61
Table 3-3: Project Specification 62
Table 3-4: Robot Rat Nouns and Verbs 64
Table 3-5: Language Feature Study Check-Off List For Robot Rat Project 66
Table 3-6: First Iteration Design Considerations 69
Table 3-7: Second Iteration Design Considerations 71
Table 3-8: Third Iteration Design Considerations 72
Table 3-9: Fourth Iteration Design Considerations 76
Table 3-10: Fifth Iteration Design Considerations 80
Table 3-11: Final Project Review Check-List 83
Table 5-1: Java Platform Packages Used In This Book 116
Table 6-1: Terms And Definitions To Get You Started 122
Table 6-2: Reserved Java Keywords 127
Table 6-3: Java Primitive Data Types 129
Table 6-4: Java Operators 139
Table 6-5: Primitive Type Wrapper Classes 149
Table 7-1: Java Selection And Iteration Statement Selection Guide 171
Table 8-1: Java Array Properties 183
Table 8-2: Java API Classes and Interfaces Referenced in Chapter 8 202
Table 8-3: EISCS Machine Instructions 205
Table 9-1: People Manager Program Class Responsibilities 212
Table 9-2: Java Method Modifier Keywords 218
Table 11-1: Differences Between Abstract Classes And Interfaces 280
Table 11-2: Chapter 11 Terms and Definitions 297
Table 12-1: java.awt.Component.setBounds() Methods 311
Table 12-2: Top-Level Containers Constructor Chart 312
Table 12-3: Methods Available to All Descendants of Window 314
Table 12-4: Methods Available to Frame, JFrame, Dialog, JDialog Only 314
Table 12-5: Methods Available to Frame, JFrame Only 315
Table 12-6: Methods Available to JWindow, JFrame, JDialog Only 315
Table 12-7: Methods Available to JFrame, JDialog Only 315
Table 12-8: GridBagConstraints Fields and Their Default Values 326
Table 12-9: Top-Level Components For Containing Other Components 330
Table 12-10: Non Top-Level Components For Containing Other Components 330
Table 12-11: Components that Allow the Selection of a Value from a Discrete Set of Values 330
Table 12-12: Components that Allow the Selection of a Value from a Virtual Continuum of Values 330
Table 12-13: Components that Allow the User to Initiate an Action 331
Table 12-14: Components that Represent a Boolean Value 331
Table 12-15: Components for Entering Text 331
Table 12-16: View-Only Components 331
Table 12-17: Appearance-Related Component Methods 332
Table 12-18: Size- and Location-Related Component Methods 333
Table 12-19: Visibility-Related Component Methods 334

Table 12-20: Containment Hierarchy-Related Component Methods 334
Table 12-21: Other Property-Related Component Methods 334
Table 12-22: Appearance-Related Container Methods 335
Table 12-23: Containment Hierarchy-Related Container Methods 335
Table 12-24: Appearance-Related JComponent Methods 336
Table 12-25: Size- and Location-Related JComponent Methods 336
Table 12-26: Visibility-Related JComponent Methods 336
Table 12-27: Containment Hierarchy-Related JComponent Methods 337
Table 12-28: Other Property-Related JComponent Methods 337
Table 13-1: EventObject Methods 351
Table 13-2: Component Methods for Managing Event Listeners 351
Table 13-3: JButton's EventListener Registration Methods 352
Table 13-4: Listeners and Event Types for JMenuItem 355
Table 13-5: ActionEvent Event IDs 356
Table 13-6: ActionEvent Constants 356
Table 13-8: ActionListener Methods 357
Table 13-7: ActionEvent Properties 357
Table 13-9: ComponentEvent Methods 359
Table 13-10: InputEvent Methods 359
Table 13-11: InputEvent Convenience Methods 359
Table 13-12: MouseEvent Event IDs 360
Table 13-13: MouseEvent-Specific Methods 360
Table 13-14: SwingUtilities Helper Methods 361
Table 13-15: MouseListener Methods 361
Table 13-16: MouseMotionListener Methods 362
Table 13-17: MouseWheelListener Methods 362
Table 13-18: KeyEvent Event IDs 365
Table 13-19: KeyEvent Methods 365
Table 13-20: KeyListener Methods 366
Table 13-21: ChangeListener Methods 372
Table 13-22: ListSelectionEvent Methods 373
Table 13-23: ListSelectionListener Methods 374
Table 14-1: Component's Painting Method 384
Table 14-2: JComponent's Painting Methods 385
Table 14-3: Repaint Methods Defined by Component 385
Table 14-4: Repaint Methods Defined by JComponent 386
Table 14-5: ListModel Methods 390
Table 14-6: JList's ListModel Methods 390
Table 14-7: JList's, JTree's and JComboBox's Renderer-Related Methods 393
Table 14-8: JTable's Renderer-Related Methods 393
Table 14-9: JTree's Editor-Related Methods 409
Table 14-10: JTable's and TableColumn's Editor-Related Methods 409
Table 14-11: javax.swing.CellEditor Methods 409
Table 14-12: dragIndex, dragItem and dragRect Attributes 415
Table 14-13: dragStart, dragThreshold and allowDrag Attributes 415
Table 14-14: deltaY and inDrag Attributes 416
Table 14-15: How the inDrag Attribute is Used 416
Table 15-1: Helpful Throwable Methods 433
Table 16-1: Getting the Current Thread 447
Table 16-2: Sleeping and Interrupting 447
Table 16-3: Checking the Interrupted Status 448
Table 16-4: Thread Constructors 449
Table 16-5: Starting a Thread 449
Table 16-6: Calling the Thread.yield() Method 455
Table 16-7: Thread's Priority-Related Methods 455
Table 16-8: ThreadGroup's Priority-Related Methods 456
Table 16-9: Thread's join() Methods 458
Table 16-10: Object's Wait() and Notify() Methods 470
Table 17-1: Core Collection Interface Characteristics 487

Table 17-2: New Java 5 Core Collection Interfaces 500
Table 18-1: Java File I/O Classes By Constructor Argument Type 511
Table 18-2: Java I/O Classes Organized By File-Terminal, Intermediate, Or User-Fronting Characteristic 512
Table 18-3: Handy java.io Class Combination Reference 545
Table 20-1: Client-Server Project Specification 588
Table 20-3: Class Responsibility Assignment 590
Table 20-2: Client-Server Project Noun-Verb Analysis 590
Table 20-4: First Iteration Design Considerations and Decisions 592
Table 20-5: Second Iteration Design Considerations and Decisions 596
Table 20-6: Third Iteration Design Considerations and Decisions 602
Table 20-7: Third Iteration Design Considerations and Decisions 607
Table 20-8: Final Iteration Design Considerations and Decisions 613
Table 21-1: Helpful MySQL Monitor Commands 646
Table 21-2: Employee Training Management System Class Descriptions 657
Table 22-1: Inheritance Form Descriptions 683
Table 23-1: Object Usage Scenario Evaluation Checklist 700
Table 23-2: Applying The Object Usage Scenario Evaluation Checklist 701
Table 23-3: equals() Method Equivalence Relation 703
Table 23-4: Bloch's equals() Method Criteria 704
Table 23-5: The hashCode() General Contract 706
Table 24-1: Terms and Definitions Related to the LSP 741
Table 25-1: Pattern Specification Template 747
Table Appendix A-1: Project Approach Strategy 781
Table Appendix A-2: Development Cycle 782
Table Appendix B-1: ASCII Table 783