Detailed Contents

Preface

Welcome – And Thank You! vii
Target Audience vii
Approach(es) vii
Pedagogy – I Mean, How This Book's Arranged viii
Learning Objectives viii
Introduction viii
Content viii
Quick Reviews viii
Summary viii
Skill-Building Exercises viii
Suggested Projects viii
Self-Test Questions ix
References ix
Notes ix
Typographical Formats ix
This Is An Example Of A First Level Subheading ix
This Is An Example Of A Second Level Subheading ix
Source Code Formatting ix
SupportSite™ Website ix
Problem Reporting x
About The Author x
Acknowledgments x

1 An Approach To The Art Of Programming

Introduction 4
The Difficulties You Will Encounter Learning C# 4
Required Skills 4
The Planets Will Come Into Alignment 4
How This Chapter Will Help You 5
Personality Traits Found In Great Programmers 5
Creative 5
Tenacious 5
Resilient 5
Methodical 5
Meticulous 6
Honest 6
Proactive 6
Humble 6
Be a Generalist and a Just-in-Time Specialist 6
Project Management 6
Three Software Development Roles 6
Analyst 6
Architect 7
Programmer 7
A Project-Approach Strategy 7

- You Have Been Handed A Project – Now What? 7
- Strategy Areas of Concern 8
- Think Abstractly 9
- The Strategy In A Nutshell 10
- Applicability To The Real World 10

The Art Of Programming 10

- Don't Start At The Computer 10
- Inspiration Strikes At The Weirdest Time 10
- Own Your Own Computer 11
  - You Either Have Time and No Money, or Money and No Time 11
  - The Family Computer Is Not Going To Cut It! 11
- Set The Mood 11
  - Location, Location, Location 11
- Concept Of The Flow 11
  - The Stages of Flow 12
- Be Extreme 12
  - The Programming Cycle 12
  - The Programming Cycle Summarized 13
- A Helpful Trick: Stubbing 13
- Fix The First Compiler Error First 14

Managing Project Complexity 14

- Conceptual Complexity 14
  - Managing Conceptual Complexity 14
  - The Unified Modeling Language (UML) 15
- Physical Complexity 15
  - Managing Physical Complexity 15
- The Relationship Between Physical and Conceptual Complexity 15
- Maximize Cohesion – Minimize Coupling 15

Summary 16

Skill-Building Exercises 16

Suggested Projects 16

Self-Test Questions 17

References 17

Notes 17

2 Small Victories: Creating C# Projects

Introduction 20

Creating Projects With Microsoft C#.NET Command-Line Tools 20

- Downloading And Installing The .NET Framework 20
- Downloading And Installing Notepad++ 22
- Configuring Your Development Environment 22
  - Environment Variables 22
  - Creating A Project Folder 25
  - Setting Folder Options 25
  - Creating A Shortcut To The Command Console And Setting Its Properties 26
- Testing The Configuration 29
  - Creating The Source File 29
  - Compiling The Source File 29
  - Executing The Application 30
- Quick Review 31

Creating Projects With Microsoft Visual C# Express 32

- Download and Install Visual C# Express 33
- Quick Tour Of Visual C# Express 33
  - Select Project Type 33
  - Saving The Project 36
  - Build The Project 36

Locating The Project Executable File 36
Execute The Project 38
Where To Go For More Information About Visual C# Express 38
Quick Review 38
Summary **38**
Skill-Building Exercises **39**
Suggested Projects **39**
Self-Test Questions **39**
References **40**
Notes **40**

3 Project Walkthrough

Introduction **42**
The Project-Approach Strategy Summarized **42**
Development Cycle **43**
Project Specification **44**
Analyzing The Project Specification 45
Application Requirements Strategy Area 45
Problem-Domain Strategy Area 46
Language-Features Strategy Area 48
Design Strategy Area 50
Development Cycle: First Iteration **51**
Plan (First Iteration) 51
Code (First Iteration) 52
Test (First Iteration) 52
Integrate/Test (First Iteration) 52
Development Cycle: Second Iteration **52**
Plan (Second Iteration) 53
Code (Second Iteration) 53
Test (Second Iteration) 53
Integrate/Test (Second Iteration) 54
Development Cycle: Third Iteration **54**
Plan (Third Iteration) 54
Code (Third Iteration) 55
Integrate/Test (Third Iteration) 57
A Bug In The Program 57
Development Cycle: Fourth Iteration **59**
Plan (Fourth Iteration) 59
Implementing State Transition Diagrams 60
Implementing The PrintFloor() Method 60
Code (Fourth Iteration) 61
Test (Fourth Iteration) 62
Integrate/Test (Fourth Iteration) 63
Development Cycle: Fifth Iteration **63**
Plan (Fifth Iteration) 63
Code (Fifth Iteration) 64
Test (Fifth Iteration) 65
Integrate/Test (Fifth Iteration) 65
Final Considerations **66**
Complete RobotRat.cs Source Code Listing **67**
Summary **73**
Skill-Building Exercises **73**
Suggested Projects **73**
Self-Test Questions **73**
References **74**

© 2008 Rick Miller — All Rights Reserved

Notes **74**

4 Computers, Programs, And Algorithms

Introduction **76**
What Is A Computer? **76**
Computer vs. Computer System 76
Computer System 76
Processor 78
Three Aspects of Processor Architecture 79
Feature Set 79
Feature Set Implementation 79
Feature Set Accessibility 79
Memory Organization **79**
Memory Basics 80
Memory Hierarchy 80
Bits, Bytes, Words 80
Alignment and Addressability 81
What Is A Program? **82**
Two Views of a Program 82
The Human Perspective 82
The Computer Perspective 82
The Processing Cycle **82**
Fetch 83
Decode 83
Execute 83
Store 83
Why A Program Crashes 83
Algorithms **83**
Good vs. Bad Algorithms 83
Don't Reinvent The Wheel! 86
Virtual Machines And The Common Language Infrastructure **86**
Virtual Machines 87
The Common Language Infrastructure (CLI) 87
Four Parts Of The Common Language Infrastructure 87
The Cross Platform Promise 89
Summary **90**
Skill-Building Exercises **90**
Suggested Projects **91**
Self-Test Questions **91**
References **92**
Notes **92**

5. Navigating .NET Framework Documentation

Introduction **94**
MSDN: The Definitive Source For API Information **94**
Discovering Information About Classes **96**
General Overview Page 96
Class Member Page 97
Getting Information On Other Class Members 98
Quick Review 100
The Base Class Libraries (BCL) **100**
Quick Review 101
Navigating An Inheritance Hierarchy **101**

© 2008 Rick Miller — All Rights Reserved

Quick Review ... 102
Beware Obsolete APIs ... **102**
Summary ... **103**
Skill-Building Exercises ... **103**
Suggested Projects ... **104**
Self-Test Questions ... **104**
References ... **104**
Notes ... **105**

6 Simple C# Programs

Introduction ... **110**
What Is A C# Program? ... **110**
A Simple Console Application ... **111**
Definition Of Terms: Application, Assembly, Module, and Entry Point ... 111
Structure Of A Simple Application ... 111
Purpose Of The Main() Method ... 112
Main() Method Signatures ... 112
Quick Review ... 113
Identifiers And Reserved Keywords ... **113**
Identifier Naming Rules ... 114
Quick Review ... 115
Types ... **115**
Value Type Variables vs. Reference Type Variables ... 116
Value Type Variables ... 116
Reference Type Variables ... 116
Maybe Some Pictures Will Help ... 117
Mapping Predefined Types To System Structures ... 118
Quick Review ... 119
Statements, Expressions, and Operators ... **119**
Statement Types ... 119
Operators And Their Use ... 120
Operator Precedence And Associativity ... 121
Forcing Operator Precedence and Associativity Order With Parentheses ... 121
Operators and Operands ... 121
Operator Usage Examples ... 122
Primary Expression Operators ... 122
Unary Expression Operators ... 122
Multiplicative Expression Operators ... 123
Additive Expression Operators ... 124
Shift Expression Operators ... 124
Relational, Type-Testing, and Equality Expression Operators ... 125
Logical AND, OR, and XOR Expression Operators ... 126
Conditional AND and OR Expression Operators ... 129
Conditional (Ternary) Expression Operator ... 129
Assignment Expression Operators ... 130
Quick Review ... 131
Summary ... **131**
Skill-Building Exercises ... **131**
Suggested Projects ... **132**
Self-Test Questions ... **132**
References ... **133**
Notes ... **133**

© 2008 Rick Miller — All Rights Reserved

7 Controlling The Flow Of Program Execution

Introduction **136**
Selection Statements **136**
If Statement 136
Handling Program Error Conditions 137
Executing Code Blocks In If Statements 139
Executing Consecutive If Statements 139
If/Else Statement 140
Chained If/Else Statements 141
Switch Statement 142
Implicit Case Fall-Through 143
Nested Switch Statement 144
Quick Review 145
Iteration Statements **145**
While Statement 145
Personality Of The While Statement 145
Do/While Statement 146
Personality Of The Do/While Statement 147
For Statement 148
How The For Statement Is Related To The While Statement 148
Personality Of The For Statement 148
Nesting Iteration Statements 149
Mixing Selection And Iteration Statements: A Powerful Combination 150
Quick Review 151
Break, Continue, And Goto **151**
Break Statement 152
Continue Statement 152
Goto Statement 153
Quick Review 153
Selection And Iteration Statement Selection Table **154**
Summary **155**
Skill-Building Exercises **155**
Suggested Projects **157**
Self-Test Questions **158**
References **159**
Notes **159**

8 Arrays

Introduction **162**
What Is An Array? **162**
Specifying Array Types 163
Quick Review 164
Functionality Provided By C# Array Types **164**
Array-Type Inheritance Hierarchy 164
Special Properties Of C# Arrays 165
Quick Review 165
Creating And Using Single-Dimensional Arrays **166**
Arrays Of Value Types 166
How Value-Type Array Objects Are Arranged In Memory 166
Finding An Array's Type, Rank, And Total Number of Elements 167
Creating Single-Dimensional Arrays Using Array Literal Values 168
Differences Between Arrays Of Value Types And Arrays Of Reference Types 169
Single-dimensional Arrays In Action 171
Message Array 171

© 2008 Rick Miller — All Rights Reserved

Calculating Averages 173
Histogram: Letter Frequency Counter 173
Quick Review 175
Creating And Using Multidimensional Arrays **176**
Rectangular Arrays 176
Initializing Rectangular Arrays With Array Literals 178
Ragged Arrays 178
Multidimensional Arrays In Action 179
Weighted Grade Tool 179
Quick Review 181
The Main() Method's String Array **181**
Purpose And Use Of The Main() Method's String Array 181
Manipulating Arrays With The System.Array Class **182**
Numeric Formatting **183**
Summary **183**
Skill-Building Exercises **184**
Suggested Projects **184**
Self-Test Questions **187**
References **188**
Notes **188**

9 Toward Problem Abstraction: Creating New Data Types

Introduction **190**
Abstraction: Amplify The Essential, Eliminate The Irrelevant **190**
Abstraction Is The Art Of Programming 190
Where Problem Abstraction Fits Into The Development Cycle 191
Creating Your Own Data Types 191
Case-Study Project: Write A People Manager Program 191
Quick Review 193
The UML Class Diagram **193**
Quick Review 194
Overview Of The Class Construct **194**
Eleven Categories Of Class Members 194
Fields 195
Constants 197
The Difference Between `const` *and* `readonly`*; Compile-Time vs. Runtime Constants* 197
Properties 198
Methods 199
Instance Constructors 199
Static Constructors 200
Events 200
Operators 200
Indexers 200
Nested Type Declarations 200
Finalizers 200
Access Modifiers 201
Public 201
Private 201
Protected 201
Internal 201
Protected Internal 201
The Concepts Of Horizontal Access, Interface, and Encapsulation 201
Quick Review 202
Methods **202**
Method Naming: Use Action Words That Indicate The Method's Purpose 203

© 2008 Rick Miller — All Rights Reserved

Maximize Method Cohesion 203
Structure Of A Method Definition 203
Method Modifiers (optional) 203
Return Type Or Void (optional) 204
Method Name (mandatory) 205
Parameter List (optional) 205
Method Body (optional for abstract or external methods) 205
Method Definition Examples 205
Method Signatures 206
Overloading Methods 206
Constructor Methods 206
Quick Review 206
Building And Testing The Person Class 207
Start By Creating The Source File And Class Definition Shell 207
Defining Person Instance Fields 207
Defining Person Properties And Constructor Method 208
Adding Properties 208
Adding A Constructor Method 208
Testing The Person Class: A Miniature Test Plan 209
Use The PeopleManagerApplication Class As A Test Driver 209
Adding Features To The Person Class: Calculating Age 210
Adding Features To The Person Class: Convenience Properties 211
Adding Features To The Person Class: Finishing Touches 213
Quick Review 214
Building and Testing The PeopleManager Class 215
Defining The PeopleManager Class Shell 215
Defining PeopleManager Fields 215
Defining PeopleManager Constructor Methods 215
Defining Additional PeopleManager Methods 216
Testing The PeopleManager Class 217
Adding Features To The PeopleManager Class 217
Quick Review 219
More About Methods 219
Value Parameters And Reference Parameters 219
Value Parameters: The Default Parameter Passing Mode 219
Reference Parameters: Using The `ref` Parameter Modifier 220
The `out` Parameter Modifier 223
Parameter Arrays: Using The `params` Modifier 223
Local Variable Scoping 224
Anywhere An Object Of <type> Is Required, A Method That Returns <type> Can Be Used 224
Quick Review 225
Structures vs. Classes 225
Value Semantics vs. Reference Semantics 225
Ten Authorized Members vs. Eleven 226
Default Variable Field Values 226
Behavior During Assignment 226
`this` Behaves Differently 226
Inheritance Not Allowed 226
Boxing And Unboxing 226
When To Use Structures 227
Summary 227
Skill-Building Exercises 228
Suggested Projects 229
Self-Test Questions 231
References 232
Notes 232

10 Compositional Design

Introduction **234**
Managing Conceptual And Physical Complexity **234**
Compiling Multiple Source Files Simultaneously With csc 234
Quick Review 235
Dependency vs. Association **235**
Aggregation **235**
Simple vs. Composite Aggregation 236
The Relationship Between Aggregation And Object Lifetime 236
Quick Review 236
Expressing Aggregation In A UML Class Diagram **236**
Simple Aggregation Expressed In UML 237
Composite Aggregation Expressed In UML 237
Aggregation Example Code **237**
Simple Aggregation Example 238
Composite Aggregation Example 239
Quick Review 240
Sequence Diagrams **240**
Magic Draw 241
Quick Review 241
The Engine Simulation: An Extended Example **242**
The Purpose Of The Engine Class 243
Engine Class Attributes And Methods 244
Engine Simulation Sequence Diagrams 244
Running The Engine Simulation Program 244
Quick Review 246
Complete Engine Simulation Code Listing **246**
Summary **250**
Skill-Building Exercises **250**
Suggested Projects **252**
Self-Test Questions **252**
References **253**
Notes **253**

11 Inheritance and Interfaces

Introduction **256**
Three Purposes Of Inheritance **256**
Implementing The "is a" Relationship 257
The Relationship Between The Terms Type, Interface, and Class 257
Meaning Of The Term Interface 257
Meaning Of The Term Class 257
Quick Review 258
Expressing Generalization And Specialization In The UML **258**
A Simple Inheritance Example **259**
The UML Diagram 259
BaseClass Source Code 259
DerivedClass Source Code 260
DriverApplication Program 260
Quick Review 261
Another Inheritance Example: Person - Student **261**
The Person - Student UML Class Diagram 261
Person - Student Source Code 262
Casting 264
Use Casting Sparingly 265

Quick Review 265
Overriding Base Class Methods **266**
Quick Review 267
Abstract Methods and Abstract Base Classes **267**
The Primary Purpose Of An Abstract Base Class 268
Expressing Abstract Base Classes In UML 268
Quick Review 270
Interfaces **270**
The Purpose Of Interfaces 270
Authorized Interface Members 270
The Differences Between An Interface And An Abstract Class 271
Expressing Interfaces In UML 271
Expressing Realization In A UML Class Diagram 271
An Interface Example 272
Quick Review 274
Controlling Horizontal And Vertical Access **274**
Quick Review 274
Sealed Classes And Methods **274**
Quick Review 274
Polymorphic Behavior **275**
Quick Review 275
Inheritance Example: Employee **275**
Inheritance Example: Engine Simulation **278**
Engine Simulation UML Diagram 278
Simulation Operational Description 278
Compiling The Engine Simulation Code 280
Complete Engine Simulation Code Listing **280**
Summary **284**
Skill-Building Exercises **285**
Suggested Projects **285**
Self-Test Questions **287**
References **287**
Notes **288**

12 Windows Forms Programming

Introduction **292**
The Form Class **292**
Form Class Inheritance Hierarchy 292
A Simple Form Program 293
Quick Review 294
Application Messages, Message Pump, Events, And Event Loop **294**
Message Categories 295
Messages In Action: Trapping Messages With IMessageFilter 296
Final Thoughts On Messages 296
Quick Review 297
Screen And Window (Client) Coordinate System **297**
Quick Review 299
Manipulating Form Properties **299**
Quick Review 301
Adding Components To Windows: Button, TextBox, And Label **301**
Quick Review 302
Registering Event Handlers With GUI Components **303**
Delegates And Events 303
Quick Review 305
Handling GUI Component Events In Separate Objects **305**

© 2008 Rick Miller — All Rights Reserved

Quick Review 307
Layout Managers **307**
FlowLayoutPanel 308
TableLayoutPanel 310
Quick Review 311
Menus **312**
Quick Review 315
A Little More About TextBoxes **315**
Quick Review 316
The Rhythm Of Coding GUIs **317**
Summary **317**
Skill-Building Exercises **318**
Suggested Projects **319**
Self-Test Questions **319**
References **320**
Notes **320**

13 Custom Events

Introduction **322**
C# Event Processing Model: An Overview **322**
Quick Review 323
Custom Events Example: Minute Tick **323**
Custom Events Example: Automated Water Tank System **326**
Naming Conventions **331**
Final Thoughts On Extending The EventArgs Class **332**
Summary **332**
Skill-Building Exercises **333**
Suggested Projects **333**
Self-Test Questions **333**
References **334**
Notes **334**

14 Collections

Introduction **338**
Case Study: Building A Dynamic Array **338**
Evaluating DynamicArray 340
The ArrayList Class To The Rescue 340
A Quick Peek At Generics 341
Quick Review 341
Data Structure Performance Characteristics **342**
Array Performance Characteristics 342
Linked List Performance Characteristics 343
Hash Table Performance Characteristics 345
Chained Hash Table vs. Open-Address Hash Table 345
Red-Black Tree Performance Characteristics 346
Stacks And Queues 347
Quick Review 347
Navigating The .NET Collections API **348**
System.Collections 348
System.Collections.Generic 348
System.Collections.ObjectModel 349
System.Collections.Specialized 349

© 2008 Rick Miller — All Rights Reserved

Mapping Non-Generic To Generic Collections 349
Quick Review 350
Using Non-Generic Collection Classes - Pre .NET 2.0 350
Objects In – Objects Out: Casting 101 351
Extending ArrayList To Create A Strongly-Typed Collection 352
Using Generic Collection Classes – .NET 2.0 and Beyond 354
List<T>: Look Ma, No More Casting! 354
Implementing KeyedCollection<TKey, TItem> 355
Quick Review 356
Special Operations On Collections 357
Sorting A List 357
Implementing System.IComparable<T> 357
Extending Comparer<T> 359
Converting A Collection Into An Array 361
Quick Review 361
Summary 362
Skill-Building Exercises 362
Suggested Projects 363
Self-Test Questions 363
References 364
Notes 364

15 Exceptions: Writing Fault-Tolerant Software

Introduction 366
What Is An Exception 366
.NET CLR Exception Handling Mechanism 366
Unhandled Exceptions 366
The Exception Information Table 367
Quick Review 367
Exception Class Hierarchy 367
Application vs. Runtime Exceptions 368
Runtime Exception Listing 368
Determining What Exceptions A .NET Framework Method Throws 369
Quick Review 369
Exception Class Properties 370
Quick Review 370
Creating Exception Handlers: Using Try/Catch/Finally Blocks 371
Using A Try/Catch Block 371
First Line of Defense: Use Defensive Coding 372
Using Multiple Catch Blocks 372
Using A Finally Block 373
Quick Review 374
Creating Custom Exceptions 374
Extending The Exception Class 374
Manually Throwing An Exception With The throw Keyword 375
Translating Low-Level Exceptions Into High-Level Exceptions 375
Quick Review 376
Documenting Exceptions 376
Summary 377
Skill-Building Exercises 377
Suggested Projects 378
Self-Test Questions 378
References 378
Notes 379

© 2008 Rick Miller — All Rights Reserved

16 Multithreaded Programming

Introduction **382**
Multithreading Overview: The Tale Of Two Vacations **382**
Single-Threaded Vacation 382
Multithreaded Vacation 382
The Relationship Between A Process And Its Threads 383
Vacation Gone Bad 384
Quick Review 385
Creating Managed Threads With The Thread Class **385**
Single-Threaded Vacation Example 386
Multithreaded Vacation Example 386
Thread States 389
Creating And Starting Managed Threads 389
ThreadStart Delegate 389
ParameterizedThreadStart Delegate: Passing Arguments To Threads 390
Blocking A Thread With Thread.Sleep() 391
Blocking A Thread With Thread.Join() 392
Foreground vs. Background Threads 394
Quick Review 395
Creating Threads With The BackgroundWorker Class **396**
Quick Review 399
Thread Pools **399**
Quick Review 400
Asynchronous Method Calls **400**
Obtaining Results From An Asynchronous Method Call 402
Providing A CallBack Method To BeginInvoke() 402
Quick Review 403
Summary **404**
Skill-Building Exercises **405**
Suggested Projects **405**
Self-Test Questions **406**
References **406**
Notes **407**

17 File I/O

Introduction **410**
Manipulating Directories And Files **410**
Files, Directories, And Paths 411
Manipulating Directories And Files 411
Verbatim String Literals 412
Quick Review 413
Serializing Objects To Disk **413**
Serializable Attribute 413
Serializing Objects With BinaryFormatter 414
Serializing Objects With XMLSerializer 416
Quick Review 418
Working With Text Files **418**
Some Issues You Must Consider 418
Saving Dog Data To A Text File 418
Quick Review 420
Working With Binary Data **420**
Quick Review 421
Random Access File I/O **422**
Towards An Approach To The Adapter Project 422

© 2008 Rick Miller — All Rights Reserved

Start Small And Take Baby Steps 423
Other Project Considerations 424
Locking A Record For Updates And Deletes 424
Monitor.Enter()/Monitor.Exit() vs. The lock Keyword 424
Translating Low-Level Exceptions Into Higher-Level Exception Abstractions 425
Where To Go From Here 425
Complete RandomAccessFile Legacy Datafile Adapter Source Code Listing 425
Quick Review 437
Working With Log Files **438**
Quick Review 440
Using FileDialogs **440**
Quick Review 442
Summary **443**
Skill-Building Exercises **444**
Suggested Projects **444**
Self-Test Questions **444**
References **445**
Notes **445**

18 Network Programming Fundamentals

Introduction **450**
What Is A Computer Network? **450**
Purpose Of A Network 450
The Role Of Network Protocols 451
Homogeneous Vs. Heterogeneous Networks 451
The Unifying Network Protocols: TCP/IP 451
What's So Special About The Internet? 452
Quick Review 452
Servers & Clients **453**
Server Hardware And Software 453
Client Hardware And Software 453
Quick Review 454
Application Distribution **454**
Physical Distribution On One Computer 454
Running Multiple Clients On The Same Computer 454
Addressing The Local Machine 455
Physical Distribution Across Multiple Computers 455
Quick Review 455
Multitiered Applications **456**
Logical Application Tiers 456
Physical Tier Distribution 456
Quick Review 456
Internet Networking Protocols: Nuts & Bolts **457**
The Internet Protocols: TCP, UDP, And IP 457
The Application Layer 458
Transport layer 458
Network Layer 459
Data Link And Physical Layers 459
Putting It All Together 459
What You Need To Know 460
Quick Review 460
Summary **460**
Skill-Building Exercises **461**
Suggested Projects **462**
Self-Test Questions **462**

© 2008 Rick Miller — All Rights Reserved

References 462
Notes 463

19 Networked Client -Server Applications

Introduction 466
Building Client-Server Applications With .NET Remoting 466
The Three Required Components of A .NET Remoting Application 466
A Simple .NET Remoting Application 467
SingleCall vs. Singleton 469
Accessing A Remote Object Via An Interface 470
Using Configuration Files 472
Passing Objects Between Client And Server 474
Quick Review 477
Client-Server Applications With TcpListener And TcpClient 478
TCP/IP Client-Server Overview 478
A Simple Client-Server Application 479
Building A Multithreaded Server 480
Listening On Multiple IP Addresses 482
Sending Objects Between Client And Server 484
Quick Review 488
Summary 489
Skill-Building Exercises 490
Suggested Projects 491
Self-Test Questions 491
References 492
Notes 492

20 Database Access & Multitiered Applications

Introduction 494
What You Are Going To Build 494
Preliminaries 495
Installing SQL Server Express Edition 495
Installing Microsoft SQL Server Management Studio Express 496
Installing Microsoft Enterprise Library 498
A Simple Test Application 499
Introduction To Relational Databases And SQL 501
Terminology 501
Structured Query Language (SQL) 502
Data Definition Language (DDL) 502
Creating The EmployeeTraining Database 502
Creating A Database With A Script 503
Creating Tables 504
SQL Server Database Types 505
Data Manipulation Language (DML) 506
Using The Insert Command 506
Using The Select Command 507
Using The Update Command 509
Using The Delete Command 510
Quick Review 510
Complex SQL Queries 511
Creating A Related Table With A Foreign Key 511
Inserting Test Data Into The tbl_employee_training Table 512
Selecting Data From Multiple Tables 514

© 2008 Rick Miller — All Rights Reserved

Join Operations ... 514
Testing The Cascade Delete Constraint ... 515
Quick Review ... 515
The Server Application ... 516
Project Folder Organization ... 516
Using Microsoft Build To Manage And Build the Project ... 517
First Iteration ... 519
Coding The EmployeeVO And EmployeeDAO ... 520
Application Configuration File ... 528
Creating Test Application ... 528
Second Iteration ... 532
Testing The Code - Second Iteration ... 543
Reality Check ... 551
Third Iteration ... 551
The Client Application ... 556
Third Iteration (continued) ... 556
Fourth Iteration ... 558
Fifth Iteration ... 564
Sixth Iteration ... 569
Compiling And Running The Modified EmployeeTrainingClient Project ... 580
Where To Go From Here ... 582
Summary ... 583
Skill-Building Exercises ... 583
Suggested Projects ... 584
Self-Test Questions ... 584
References ... 585
Notes ... 585

21 Operator Overloading

Introduction ... 590
Operator Overloading ... 590
Overloadable Operators ... 590
Quick Review ... 591
Overloading Unary Operators ... 591
+,- Operators ... 591
! Operator ... 592
true, false Operators ... 593
++ --, Operators ... 595
Quick Review ... 597
Overloading Binary Operators ... 597
+, - Operators ... 597
*, / Operators ... 599
&, | Operators ... 601
Quick Review ... 603
Overloading Comparison Operators ... 603
==, !=,<,>,<=,>= Operators ... 604
Quick Review ... 607
Creating Implicit And Explicit Cast Operators ... 607
Implicit vs. Explicit Cast ... 607
Overloaded Cast Operators Example ... 607
Quick Review ... 610
The Assignment Operators: Things You Get For Free ... 610
Quick Review ... 610
Summary ... 610
Skill-Building Exercises ... 611

© 2008 Rick Miller — All Rights Reserved

Suggested Projects **611**
Self-Test Questions **611**
References **612**
Notes **612**

22 Well-Behaved Objects

Introduction **614**
Object Behavior Defined **614**
Fundamental Behavior 614
Copy/Assignment Behavior 614
Equality Behavior 615
Comparison/Ordering Behavior 615
Seven Object Usage Scenarios 615
Fundamental Behavior **616**
Object Creation – Constructors 616
Default Constructor 616
Private Constructors 616
Overloaded Constructors 616
Member Accessibility 616
Horizontal Member Access 616
Vertical Member Access 617
Overriding Object.ToString() 617
Static vs. Instance Members 617
Serialization 618
Custom Serialization Example 618
Quick Review 623
Copy/Assignment Behavior **623**
Value Object vs. Reference Object Assignment 624
Rule Of Thumb – Favor The Class Construct For Complex Types 624
Shallow Copy vs. Deep Copy 624
Copy Constructors 624
System.ICloneable vs. Object.MemberwiseClone() 627
Quick Review 629
Equality Behavior **629**
Reference Equality vs. Value Equality 629
Rules For Overriding The Object.Equals() Method 629
Overriding The Object.GetHashCode() Method 630
Bloch's hash Code Generation Algorithm 631
Ashmore's Hash Code Generation Algorithm 631
Overridng Object.Equals() and Object.GetHashCode() Methods In The PersonVO Class 631
Quick Review 633
Comparison/Ordering Behavior **633**
Implementing System.IComparable<T> 634
Rules For Implementing The CompareTo(T other) Method 635
Extending The Comparer<T> Class 636
Quick Review 637
Summary **637**
Skill-Building Exercises **638**
Suggested Projects **638**
Self-Test Questions **638**
References **639**
Notes **639**

23 Three Design Principles

Introduction **642**
The Preferred Characteristics Of An Object-Oriented Architecture **642**
Easy To Understand: How does this thing work? 642
Easy To Reason About: What are the effects of change? 642
Easy To Extend: Where do I add functionality? 642
The Liskov Substitution Principle & Design by Contract **643**
Reasoning About The Behavior Of Supertypes And Subtypes 643
Relationship Between The LSP And DbC 643
The Common Goal Of The LSP And DbC 643
C# Support For The LSP And DbC 643
Designing With The LSP/DbC In Mind 644
Class Declarations Viewed As Behavior Specifications 644
Quick Review 644
Preconditions, Postconditions, And Class Invariants **644**
Class Invariant 644
Precondition 644
Postcondition 645
An Example 645
A Note On Using The Debug.Assert() Method To Enforce Pre- and Postconditions 646
Using Incrementer As A Base Class 646
Changing The Preconditions Of Derived Class Methods 648
Changing The Postconditions Of Derived Class Methods 652
Special Cases Of Preconditions And Postconditions 652
Method Argument Types 653
Method Return Types 654
Three Rules Of The Substitution Principle 654
Signature Rule 655
Methods Rule 655
Properties Rule 655
Quick Review 655
The Open-Closed Principle **656**
Achieving The Open-Closed Principle 656
An OCP Example 656
Quick Review 661
The Dependency Inversion Principle **661**
Characteristics Of Bad Software Architecture 661
Characteristics Of Good Software Architecture 662
Selecting The Right Abstractions Takes Experience 662
Quick Review 662
Terms and Definitions **663**
Summary **663**
Skill-Building Exercises **664**
Suggested Projects **664**
Self-Test Questions **664**
References **665**
Notes **666**

24 Inheritance, Composition, Interfaces, Polymorphism

Introduction **668**
Inheritance Vs. Composition: The Great Debate **668**
What's The End Game? 669
Flexible Application Architectures 669
Modularity And Reliability 669

Architectural Stability Via Managed Dependencies 669
Knowing When To Accept A Design That's Good Enough 670
Quick Review 670
Inheritance-Based Design **670**
Three Good Reasons To Use Inheritance 670
As A Means To Reason About Code Behavior 670
To Gain A Measure Of Code Reuse 670
To Facilitate Incremental Development 670
Forms Of Inheritance: Meyer's Inheritance Taxonomy 671
Coad's Inheritance Criteria 672
Person - Employee Example Revisited 673
Quick Review 673
The Role Of Interfaces **674**
Reducing Or Limiting Intermodule Dependencies 674
Modeling Dominant, Collateral, and Dynamic Roles 674
Dominant Roles 674
Collateral Roles 675
Dynamic Roles 675
Quick Review 675
Applied Polymorphism **675**
Quick Review 676
Composition-Based Design As A Force Multiplier **676**
Two Types Of Aggregation 676
Polymorphic Containment 676
An Extended Example **677**
Quick Review 682
Summary **682**
Skill-Building Exercises **683**
Suggested Projects **684**
Self-Test Questions **685**
References **685**
Notes **686**

25 Helpful Design Patterns

Introduction **688**
Software Design Patterns And How They Came To Be **688**
What Exactly Is A Software Design Pattern? 688
Origins 688
Pattern Specification 689
Applying Software Design Patterns 689
Quick Review 689
The Singleton Pattern **690**
Quick Review 693
The Factory Pattern **693**
The Dynamic Factory 693
Advantages Of The Dynamic Factory Pattern 695
Quick Review 695
The Model-View-Controller Pattern **695**
Quick Review 697
The Command Pattern **697**
Quick Review 702
A Comprehensive Pattern-Based Example **702**
Complete Code Listing 702
Com.PulpFreePress.Exceptions 702
Com.PulpFreePress.Common 702

© 2008 Rick Miller — All Rights Reserved

Com.PulpFreePress.Utils 707
Com.PulpFreePress.Commands 710
Com.PulpFreePress.Model 713
Com.PulpFreePress.View 714
Com.PulpFreePress.Controller 720
Running The Application 721
Summary 722
Skill-Building Exercises 722
Suggested Projects 723
Self-Test Questions 723
References 723
Notes 724

Appendix A: Helpful Checklists And Tables

Project-Approach Strategy Check-off List 727
Development Cycle 728
Final Project Review Checklist 728

Appendix B: ASCII Table

ASCII Table 731

Appendix C: Identifier Naming: Writing Self-Commenting Code

Identifier Naming: Writing Self-Commenting Code 735
Benefits of Self-Commenting Code 735
Coding Convention 735
Class Names 735
Constant Names 736
Variable Names 736
Method Names 736
Property Names 737

© 2008 Rick Miller — All Rights Reserved