

# DETAILED CONTENTS

## PREFACE

<b>WELCOME – And Thank You!</b> .....	<b>vii</b>
<b>TARGET AUDIENCE</b> .....	<b>vii</b>
<b>Approach(es)</b> .....	<b>vii</b>
<b>Pedagogy – I MEAN, How This Book’s ARRANGED</b> .....	<b>viii</b>
<i>LEARNING OBJECTIVES</i> .....	<i>viii</i>
<i>INTRODUCTION</i> .....	<i>viii</i>
<i>CONTENT</i> .....	<i>viii</i>
<i>Quick Reviews</i> .....	<i>viii</i>
<i>SUMMARY</i> .....	<i>viii</i>
<i>Skill-Building EXERCISES</i> .....	<i>viii</i>
<i>SUGGESTED PROJECTS</i> .....	<i>viii</i>
<i>Self-Test QUESTIONS</i> .....	<i>ix</i>
<i>REFERENCES</i> .....	<i>ix</i>
<i>NOTES</i> .....	<i>ix</i>
<b>Typographical FORMATS</b> .....	<b>ix</b>
<i>This Is An Example Of A First Level Subheading</i> .....	<i>ix</i>
<i>This Is An Example Of A Second Level Subheading</i> .....	<i>ix</i>
<i>SOURCE CODE FORMATTING</i> .....	<i>ix</i>
<b>SupportSite™ Website</b> .....	<b>ix</b>
<b>Problem Reporting</b> .....	<b>x</b>
<b>ABOUT THE AUTHOR</b> .....	<b>x</b>
<b>Acknowledgments</b> .....	<b>x</b>

## 1 AN APPROACH TO THE ART OF PROGRAMMING

<b>INTRODUCTION</b> .....	<b>4</b>
<i>THE DIFFICULTIES YOU WILL ENCOUNTER LEARNING C#</i> .....	<i>4</i>
<i>REQUIRED SKILLS</i> .....	<i>4</i>
<i>THE PLANETS WILL COME INTO ALIGNMENT</i> .....	<i>4</i>
<i>How This Chapter Will Help You</i> .....	<i>5</i>
<b>PERSONALITY TRAITS FOUND IN GREAT PROGRAMMERS</b> .....	<b>5</b>
<i>CREATIVE</i> .....	<i>5</i>
<i>TENACIOUS</i> .....	<i>5</i>
<i>RESILIENT</i> .....	<i>5</i>
<i>METHODICAL</i> .....	<i>5</i>
<i>METICULOUS</i> .....	<i>6</i>
<i>HONEST</i> .....	<i>6</i>
<i>PROACTIVE</i> .....	<i>6</i>
<i>HUMBLE</i> .....	<i>6</i>
<i>BE A GENERALIST AND A JUST-IN-TIME SPECIALIST</i> .....	<i>6</i>
<b>PROJECT MANAGEMENT</b> .....	<b>6</b>
<i>THREE SOFTWARE DEVELOPMENT ROLES</i> .....	<i>6</i>
<i>ANALYST</i> .....	<i>6</i>
<i>ARCHITECT</i> .....	<i>7</i>
<i>PROGRAMMER</i> .....	<i>7</i>
<i>A Project-Approach Strategy</i> .....	<i>7</i>

<i>You Have Been Handled A Project – Now What?</i> .....	7
<i>Strategy Areas of Concern</i> .....	8
<i>Think Abstractly</i> .....	9
<i>The Strategy In A Nutshell</i> .....	10
<i>Applicability To The Real World</i> .....	10
<b>The Art Of Programming</b> .....	<b>10</b>
<i>Don't Start At The Computer</i> .....	10
<i>Inspiration Strikes At The Weirdest Time</i> .....	10
<i>Own Your Own Computer</i> .....	11
<i>You Either Have Time And No Money, Or Money And No Time</i> .....	11
<i>The Family Computer Is Not Going To Cut It!</i> .....	11
<i>Set The Mood</i> .....	11
<i>Location, Location, Location</i> .....	11
<i>Concept Of The Flow</i> .....	11
<i>The Stages of Flow</i> .....	12
<i>Be Extreme</i> .....	12
<i>The Programming Cycle</i> .....	12
<i>The Programming Cycle Summarized</i> .....	13
<i>A Helpful Trick: Stubbing</i> .....	13
<i>Fix The First Compiler Error First</i> .....	14
<b>Managing Project Complexity</b> .....	<b>14</b>
<i>Conceptual Complexity</i> .....	14
<i>Managing Conceptual Complexity</i> .....	14
<i>The Unified Modeling Language (UML)</i> .....	15
<i>Physical Complexity</i> .....	15
<i>Managing Physical Complexity</i> .....	15
<i>The Relationship Between Physical And Conceptual Complexity</i> .....	15
<i>Maximize Cohesion – Minimize Coupling</i> .....	15
<b>Summary</b> .....	<b>16</b>
<b>Skill-Building Exercises</b> .....	<b>16</b>
<b>Suggested Projects</b> .....	<b>16</b>
<b>Self-Test Questions</b> .....	<b>17</b>
<b>References</b> .....	<b>17</b>
<b>Notes</b> .....	<b>17</b>

## 2 Small Victories: Creating C# Projects

<b>Introduction</b> .....	<b>20</b>
<b>Creating Projects With Microsoft C#.NET Command-Line Tools</b> .....	<b>20</b>
<i>Downloading And Installing The .NET Framework</i> .....	20
<i>Downloading And Installing Notepad++</i> .....	22
<i>Configuring Your Development Environment</i> .....	22
<i>Environment Variables</i> .....	22
<i>Creating A Project Folder</i> .....	25
<i>Setting Folder Options</i> .....	25
<i>Creating A Shortcut To The Command Console And Setting Its Properties</i> .....	26
<i>Testing The Configuration</i> .....	29
<i>Creating The Source File</i> .....	29
<i>Compiling The Source File</i> .....	29
<i>Executing The Application</i> .....	30
<i>Quick Review</i> .....	31
<b>Creating Projects With Microsoft Visual C# Express</b> .....	<b>32</b>
<i>Download And Install Visual C# Express</i> .....	32
<i>Quick Tour Of Visual C# Express</i> .....	33
<i>Select Project Type</i> .....	33
<i>Saving The Project</i> .....	34
<i>Build The Project</i> .....	36

<i>Locating The Project Executable File</i> .....	36
<i>Execute The Project</i> .....	38
<i>Where To Go For More Information About Visual C# Express</i> .....	38
<i>Quick Review</i> .....	38
<b>SUMMARY</b> .....	<b>38</b>
<b>Skill-Building Exercises</b> .....	<b>39</b>
<b>SUGGESTED PROJECTS</b> .....	<b>39</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>39</b>
<b>REFERENCES</b> .....	<b>40</b>
<b>NOTES</b> .....	<b>40</b>

## 3 PROJECT WALKTHROUGH

<b>INTRODUCTION</b> .....	<b>42</b>
<b>THE PROJECT-APPROACH STRATEGY SUMMARIZED</b> .....	<b>42</b>
<b>DEVELOPMENT CYCLE</b> .....	<b>43</b>
<b>PROJECT SPECIFICATION</b> .....	<b>44</b>
<i>Analyzing The Project Specification</i> .....	45
<i>Application Requirements Strategy Area</i> .....	45
<i>Problem-Domain Strategy Area</i> .....	46
<i>Language-Features Strategy Area</i> .....	48
<i>Design Strategy Area</i> .....	50
<b>DEVELOPMENT CYCLE: FIRST ITERATION</b> .....	<b>51</b>
<i>Plan (First Iteration)</i> .....	51
<i>Code (First Iteration)</i> .....	52
<i>Test (First Iteration)</i> .....	52
<i>Integrate/Test (First Iteration)</i> .....	52
<b>DEVELOPMENT CYCLE: SECOND ITERATION</b> .....	<b>52</b>
<i>Plan (Second Iteration)</i> .....	53
<i>Code (Second Iteration)</i> .....	53
<i>Test (Second Iteration)</i> .....	53
<i>Integrate/Test (Second Iteration)</i> .....	54
<b>DEVELOPMENT CYCLE: THIRD ITERATION</b> .....	<b>54</b>
<i>Plan (Third Iteration)</i> .....	54
<i>Code (Third Iteration)</i> .....	55
<i>Integrate/Test (Third Iteration)</i> .....	57
<i>A Bug In The Program</i> .....	57
<b>DEVELOPMENT CYCLE: FOURTH ITERATION</b> .....	<b>59</b>
<i>Plan (Fourth Iteration)</i> .....	59
<i>Implementing State Transition Diagrams</i> .....	60
<i>Implementing The PrintFloor() Method</i> .....	60
<i>Code (Fourth Iteration)</i> .....	61
<i>Test (Fourth Iteration)</i> .....	62
<i>Integrate/Test (Fourth Iteration)</i> .....	63
<b>DEVELOPMENT CYCLE: FIFTH ITERATION</b> .....	<b>63</b>
<i>Plan (Fifth Iteration)</i> .....	63
<i>Code (Fifth Iteration)</i> .....	64
<i>Test (Fifth Iteration)</i> .....	65
<i>Integrate/Test (Fifth Iteration)</i> .....	65
<b>FINAL CONSIDERATIONS</b> .....	<b>66</b>
<b>COMPLETE ROBOTRAT.CS SOURCE CODE LISTING</b> .....	<b>67</b>
<b>SUMMARY</b> .....	<b>73</b>
<b>Skill-Building Exercises</b> .....	<b>73</b>
<b>SUGGESTED PROJECTS</b> .....	<b>73</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>73</b>
<b>REFERENCES</b> .....	<b>74</b>

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

<i>Quick Review</i> .....	102
<b>BEWARE OBSOLETE APIs</b> .....	<b>102</b>
<b>SUMMARY</b> .....	<b>103</b>
<b>SKILL-BUILDING EXERCISES</b> .....	<b>103</b>
<b>SUGGESTED PROJECTS</b> .....	<b>104</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>104</b>
<b>REFERENCES</b> .....	<b>104</b>
<b>NOTES</b> .....	<b>105</b>

## 6 Simple C# Programs

<b>INTRODUCTION</b> .....	<b>110</b>
<b>WHAT IS A C# PROGRAM?</b> .....	<b>110</b>
<b>A SIMPLE CONSOLE APPLICATION</b> .....	<b>111</b>
<i>Definition Of Terms: Application, Assembly, Module, and Entry Point</i> .....	111
<i>STRUCTURE OF A SIMPLE APPLICATION</i> .....	111
<i>PURPOSE OF THE MAIN() METHOD</i> .....	112
<i>MAIN() METHOD SIGNATURES</i> .....	112
<i>Quick Review</i> .....	113
<b>IDENTIFIERS AND RESERVED KEYWORDS</b> .....	<b>113</b>
<i>Identifier Naming Rules</i> .....	114
<i>Quick Review</i> .....	115
<b>Types</b> .....	<b>115</b>
<i>VALUE TYPE VARIABLES VS. REFERENCE TYPE VARIABLES</i> .....	116
<i>VALUE TYPE VARIABLES</i> .....	116
<i>REFERENCE TYPE VARIABLES</i> .....	116
<i>MAYBE SOME PICTURES WILL HELP</i> .....	117
<i>MAPPING PREDEFINED TYPES TO SYSTEM STRUCTURES</i> .....	118
<i>Quick Review</i> .....	119
<b>STATEMENTS, EXPRESSIONS, AND OPERATORS</b> .....	<b>119</b>
<i>STATEMENT TYPES</i> .....	119
<i>OPERATORS AND THEIR USE</i> .....	120
<i>OPERATOR PRECEDENCE AND ASSOCIATIVITY</i> .....	121
<i>FORCING OPERATOR PRECEDENCE AND ASSOCIATIVITY ORDER WITH PARENTHESES</i> .....	121
<i>OPERATORS AND OPERANDS</i> .....	121
<i>OPERATOR USAGE EXAMPLES</i> .....	122
<i>PRIMARY EXPRESSION OPERATORS</i> .....	122
<i>UNARY EXPRESSION OPERATORS</i> .....	122
<i>MULTIPLICATIVE EXPRESSION OPERATORS</i> .....	123
<i>ADDITIVE EXPRESSION OPERATORS</i> .....	124
<i>SHIFT EXPRESSION OPERATORS</i> .....	124
<i>RELATIONAL, TYPE-TESTING, AND EQUALITY EXPRESSION OPERATORS</i> .....	125
<i>LOGICAL AND, OR, AND XOR EXPRESSION OPERATORS</i> .....	126
<i>CONDITIONAL AND AND OR EXPRESSION OPERATORS</i> .....	129
<i>CONDITIONAL (TERNARY) EXPRESSION OPERATOR</i> .....	129
<i>ASSIGNMENT EXPRESSION OPERATORS</i> .....	130
<i>Quick Review</i> .....	131
<b>SUMMARY</b> .....	<b>131</b>
<b>SKILL-BUILDING EXERCISES</b> .....	<b>131</b>
<b>SUGGESTED PROJECTS</b> .....	<b>132</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>132</b>
<b>REFERENCES</b> .....	<b>133</b>
<b>NOTES</b> .....	<b>133</b>

## 7 CONTROLLING THE FLOW OF PROGRAM EXECUTION

<b>INTRODUCTION .....</b>	<b>136</b>
<b>SELECTION STATEMENTS .....</b>	<b>136</b>
<i>If STATEMENT .....</i>	<i>136</i>
<i>Handling PROGRAM ERROR Conditions.....</i>	<i>137</i>
<i>EXECUTING Code Blocks IN If STATEMENTS.....</i>	<i>139</i>
<i>EXECUTING CONSECUTIVE If STATEMENTS.....</i>	<i>139</i>
<i>If/Else STATEMENT .....</i>	<i>140</i>
<i>CHAINED If/Else STATEMENTS.....</i>	<i>141</i>
<i>Switch STATEMENT .....</i>	<i>142</i>
<i>Implicit CASE Fall-Through.....</i>	<i>143</i>
<i>NESTED Switch STATEMENT.....</i>	<i>144</i>
<i>Quick Review .....</i>	<i>145</i>
<b>ITERATION STATEMENTS .....</b>	<b>145</b>
<i>While STATEMENT .....</i>	<i>145</i>
<i>PERSONALITY OF THE While STATEMENT.....</i>	<i>145</i>
<i>Do/While STATEMENT .....</i>	<i>146</i>
<i>PERSONALITY OF THE Do/While STATEMENT.....</i>	<i>147</i>
<i>FOR STATEMENT .....</i>	<i>148</i>
<i>HOW THE FOR STATEMENT IS RELATED TO THE While STATEMENT.....</i>	<i>148</i>
<i>PERSONALITY OF THE FOR STATEMENT .....</i>	<i>148</i>
<i>NESTING Iteration STATEMENTS .....</i>	<i>149</i>
<i>Mixing SELECTION AND Iteration STATEMENTS: A POWERFUL COMBINATION .....</i>	<i>150</i>
<i>Quick Review .....</i>	<i>151</i>
<b>BREAK, CONTINUE, AND GOTO .....</b>	<b>151</b>
<i>BREAK STATEMENT .....</i>	<i>152</i>
<i>CONTINUE STATEMENT .....</i>	<i>152</i>
<i>GOTO STATEMENT .....</i>	<i>153</i>
<i>Quick Review .....</i>	<i>153</i>
<b>SELECTION AND ITERATION STATEMENT SELECTION TABLE .....</b>	<b>154</b>
<b>SUMMARY .....</b>	<b>155</b>
<b>SKILL-BUILDING EXERCISES .....</b>	<b>155</b>
<b>SUGGESTED PROJECTS .....</b>	<b>157</b>
<b>SELF-TEST QUESTIONS .....</b>	<b>158</b>
<b>REFERENCES .....</b>	<b>159</b>
<b>NOTES .....</b>	<b>159</b>

## 8 ARRAYS

<b>INTRODUCTION .....</b>	<b>162</b>
<b>WHAT IS AN ARRAY? .....</b>	<b>162</b>
<i>Specifying ARRAY Types .....</i>	<i>163</i>
<i>Quick Review .....</i>	<i>164</i>
<b>FUNCTIONALITY PROVIDED BY C# ARRAY TYPES .....</b>	<b>164</b>
<i>ARRAY-TYPE INHERITANCE Hierarchy .....</i>	<i>164</i>
<i>SPECIAL PROPERTIES OF C# ARRAYS .....</i>	<i>165</i>
<i>Quick Review .....</i>	<i>165</i>
<b>CREATING AND USING SINGLE-DIMENSIONAL ARRAYS .....</b>	<b>166</b>
<i>ARRAYS OF VALUE Types .....</i>	<i>166</i>
<i>HOW VALUE-TYPE ARRAY OBJECTS ARE ARRANGED IN MEMORY .....</i>	<i>166</i>
<i>FINDING AN ARRAY'S TYPE, RANK, AND TOTAL NUMBER OF ELEMENTS .....</i>	<i>167</i>
<i>CREATING SINGLE-DIMENSIONAL ARRAYS USING ARRAY LITERAL VALUES .....</i>	<i>168</i>
<i>DIFFERENCES BETWEEN ARRAYS OF VALUE Types AND ARRAYS OF REFERENCE Types .....</i>	<i>169</i>
<i>SINGLE-DIMENSIONAL ARRAYS IN ACTION .....</i>	<i>171</i>
<i>MESSAGE ARRAY.....</i>	<i>171</i>

Calculating Averages .....	173
Histogram: Letter Frequency Counter .....	173
Quick Review .....	175
<b>Creating And Using Multidimensional Arrays .....</b>	<b>176</b>
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
<b>The Main() Method's String Array .....</b>	<b>181</b>
Purpose And Use Of The Main() Method's String Array .....	181
<b>Manipulating Arrays With The System.Array Class .....</b>	<b>182</b>
<b>Numeric Formatting .....</b>	<b>183</b>
<b>Summary .....</b>	<b>183</b>
<b>Skill-Building Exercises .....</b>	<b>184</b>
<b>Suggested Projects .....</b>	<b>184</b>
<b>Self-Test Questions .....</b>	<b>187</b>
<b>References .....</b>	<b>188</b>
<b>Notes .....</b>	<b>188</b>

## 9 TOWARD PROBLEM ABSTRACTION: CREATING NEW DATA TYPES

<b>Introduction .....</b>	<b>190</b>
<b>Abstraction: Amplify The Essential, Eliminate The Irrelevant .....</b>	<b>190</b>
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
<b>The UML Class Diagram .....</b>	<b>193</b>
Quick Review .....	194
<b>Overview Of The Class Construct .....</b>	<b>194</b>
Eleven Categories Of Class Members .....	194
Fields .....	195
Constants .....	197
The Difference Between <code>const</code> and <code>readonly</code> ; 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
<b>Methods .....</b>	<b>202</b>
Method Naming: Use Action Words That Indicate The Method's Purpose .....	203

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



## 10 Compositional Design

<b>INTRODUCTION</b>	<b>234</b>
<b>MANAGING CONCEPTUAL AND PHYSICAL COMPLEXITY</b>	<b>234</b>
<i>Compiling Multiple Source Files Simultaneously With csc</i>	234
<i>Quick Review</i>	235
<b>DEPENDENCY VS. ASSOCIATION</b>	<b>235</b>
<b>AGGREGATION</b>	<b>235</b>
<i>Simple vs. Composite Aggregation</i>	236
<i>The Relationship Between Aggregation And Object Lifetime</i>	236
<i>Quick Review</i>	236
<b>EXPRESSING AGGREGATION IN A UML CLASS DIAGRAM</b>	<b>236</b>
<i>Simple Aggregation Expressed In UML</i>	237
<i>Composite Aggregation Expressed In UML</i>	237
<b>AGGREGATION EXAMPLE CODE</b>	<b>237</b>
<i>Simple Aggregation Example</i>	238
<i>Composite Aggregation Example</i>	239
<i>Quick Review</i>	240
<b>SEQUENCE DIAGRAMS</b>	<b>240</b>
<i>Magic Draw</i>	241
<i>Quick Review</i>	241
<b>THE ENGINE SIMULATION: AN EXTENDED EXAMPLE</b>	<b>242</b>
<i>The Purpose Of The Engine Class</i>	243
<i>Engine Class Attributes And Methods</i>	244
<i>Engine Simulation Sequence Diagrams</i>	244
<i>Running The Engine Simulation Program</i>	244
<i>Quick Review</i>	246
<b>COMPLETE ENGINE SIMULATION CODE LISTING</b>	<b>246</b>
<b>SUMMARY</b>	<b>250</b>
<b>SKILL-BUILDING EXERCISES</b>	<b>250</b>
<b>SUGGESTED PROJECTS</b>	<b>252</b>
<b>SELF-TEST QUESTIONS</b>	<b>252</b>
<b>REFERENCES</b>	<b>253</b>
<b>NOTES</b>	<b>253</b>

## 11 INHERITANCE AND INTERFACES

<b>INTRODUCTION</b>	<b>256</b>
<b>THREE PURPOSES OF INHERITANCE</b>	<b>256</b>
<i>Implementing The “is A” Relationship</i>	257
<i>The Relationship Between The Terms Type, Interface, And Class</i>	257
<i>Meaning Of The Term Interface</i>	257
<i>Meaning Of The Term Class</i>	257
<i>Quick Review</i>	258
<b>EXPRESSING GENERALIZATION AND SPECIALIZATION IN THE UML</b>	<b>258</b>
<b>A SIMPLE INHERITANCE EXAMPLE</b>	<b>259</b>
<i>The UML Diagram</i>	259
<i>BaseClass Source Code</i>	259
<i>DerivedClass Source Code</i>	260
<i>DriverApplication Program</i>	260
<i>Quick Review</i>	261
<b>ANOTHER INHERITANCE EXAMPLE: PERSON - STUDENT</b>	<b>261</b>
<i>The Person - Student UML Class Diagram</i>	261
<i>Person - Student Source Code</i>	262
<i>Casting</i>	264
<i>Use Casting Sparingly</i>	265

<i>Quick Review</i> .....	265
<b>OVERRIDING BASE CLASS METHODS</b> .....	<b>266</b>
<i>Quick Review</i> .....	267
<b>ABSTRACT METHODS AND ABSTRACT BASE CLASSES</b> .....	<b>267</b>
<i>The Primary Purpose Of An Abstract Base Class</i> .....	268
<i>Expressing Abstract Base Classes In UML</i> .....	268
<i>Quick Review</i> .....	270
<b>INTERFACES</b> .....	<b>270</b>
<i>The Purpose Of Interfaces</i> .....	270
<i>Authorized Interface Members</i> .....	270
<i>The Differences Between An Interface And An Abstract Class</i> .....	271
<i>Expressing Interfaces In UML</i> .....	271
<i>Expressing Realization In A UML Class Diagram</i> .....	271
<i>An Interface Example</i> .....	272
<i>Quick Review</i> .....	274
<b>CONTROLLING HORIZONTAL AND VERTICAL ACCESS</b> .....	<b>274</b>
<i>Quick Review</i> .....	274
<b>SEALED CLASSES AND METHODS</b> .....	<b>274</b>
<i>Quick Review</i> .....	274
<b>POLYMORPHIC BEHAVIOR</b> .....	<b>275</b>
<i>Quick Review</i> .....	275
<b>INHERITANCE EXAMPLE: EMPLOYEE</b> .....	<b>275</b>
<b>INHERITANCE EXAMPLE: ENGINE SIMULATION</b> .....	<b>278</b>
<i>Engine Simulation UML Diagram</i> .....	278
<i>Simulation Operational Description</i> .....	278
<i>Compiling The Engine Simulation Code</i> .....	280
<b>COMPLETE ENGINE SIMULATION CODE LISTING</b> .....	<b>280</b>
<b>SUMMARY</b> .....	<b>284</b>
<b>SKILL-BUILDING EXERCISES</b> .....	<b>285</b>
<b>SUGGESTED PROJECTS</b> .....	<b>285</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>287</b>
<b>REFERENCES</b> .....	<b>287</b>
<b>NOTES</b> .....	<b>288</b>

## 12 Windows Forms Programming

<b>INTRODUCTION</b> .....	<b>292</b>
<b>THE FORM CLASS</b> .....	<b>292</b>
<i>Form Class Inheritance Hierarchy</i> .....	292
<i>A Simple Form Program</i> .....	293
<i>Quick Review</i> .....	294
<b>APPLICATION MESSAGES, MESSAGE PUMP, EVENTS, AND EVENT LOOP</b> .....	<b>294</b>
<i>Message Categories</i> .....	295
<i>Messages In Action: Trapping Messages With IMessageFilter</i> .....	296
<i>Final Thoughts On Messages</i> .....	296
<i>Quick Review</i> .....	297
<b>SCREEN AND WINDOW (CLIENT) COORDINATE SYSTEM</b> .....	<b>297</b>
<i>Quick Review</i> .....	299
<b>MANIPULATING FORM PROPERTIES</b> .....	<b>299</b>
<i>Quick Review</i> .....	301
<b>ADDING COMPONENTS TO WINDOWS: BUTTON, TEXTBOX, AND LABEL</b> .....	<b>301</b>
<i>Quick Review</i> .....	302
<b>REGISTERING EVENT HANDLERS WITH GUI COMPONENTS</b> .....	<b>303</b>
<i>Delegates And Events</i> .....	303
<i>Quick Review</i> .....	305
<b>HANDLING GUI COMPONENT EVENTS IN SEPARATE OBJECTS</b> .....	<b>305</b>

<i>Quick Review</i> .....	307
<b>LAYOUT MANAGERS</b> .....	<b>307</b>
<i>FlowLayoutPanel</i> .....	308
<i>TableLayoutPanel</i> .....	310
<i>Quick Review</i> .....	311
<b>MENUS</b> .....	<b>312</b>
<i>Quick Review</i> .....	315
<b>A LITTLE MORE ABOUT TEXTBOXES</b> .....	<b>315</b>
<i>Quick Review</i> .....	316
<b>THE RHYTHM OF CODING GUIs</b> .....	<b>317</b>
<b>SUMMARY</b> .....	<b>317</b>
<b>SKILL-BUILDING EXERCISES</b> .....	<b>318</b>
<b>SUGGESTED PROJECTS</b> .....	<b>319</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>319</b>
<b>REFERENCES</b> .....	<b>320</b>
<b>NOTES</b> .....	<b>320</b>

## 13 CUSTOM EVENTS

<b>INTRODUCTION</b> .....	<b>322</b>
<b>C# EVENT PROCESSING MODEL: AN OVERVIEW</b> .....	<b>322</b>
<i>Quick Review</i> .....	323
<b>CUSTOM EVENTS EXAMPLE: MINUTE TICK</b> .....	<b>323</b>
<b>CUSTOM EVENTS EXAMPLE: AUTOMATED WATER TANK SYSTEM</b> .....	<b>326</b>
<b>NAMING CONVENTIONS</b> .....	<b>331</b>
<b>FINAL THOUGHTS ON EXTENDING THE EVENTARGS CLASS</b> .....	<b>332</b>
<b>SUMMARY</b> .....	<b>332</b>
<b>SKILL-BUILDING EXERCISES</b> .....	<b>333</b>
<b>SUGGESTED PROJECTS</b> .....	<b>333</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>333</b>
<b>REFERENCES</b> .....	<b>334</b>
<b>NOTES</b> .....	<b>334</b>

## 14 COLLECTIONS

<b>INTRODUCTION</b> .....	<b>338</b>
<b>CASE STUDY: BUILDING A DYNAMIC ARRAY</b> .....	<b>338</b>
<i>Evaluating DynamicArray</i> .....	340
<i>The ArrayList Class To The Rescue</i> .....	340
<i>A Quick Peek At Generics</i> .....	341
<i>Quick Review</i> .....	341
<b>DATA STRUCTURE PERFORMANCE CHARACTERISTICS</b> .....	<b>342</b>
<i>Array Performance Characteristics</i> .....	342
<i>Linked List Performance Characteristics</i> .....	343
<i>Hash Table Performance Characteristics</i> .....	345
<i>Chained Hash Table vs. Open-Address Hash Table</i> .....	345
<i>Red-Black Tree Performance Characteristics</i> .....	346
<i>Stacks And Queues</i> .....	347
<i>Quick Review</i> .....	347
<b>NAVIGATING THE .NET COLLECTIONS API</b> .....	<b>348</b>
<i>System.Collections</i> .....	348
<i>System.Collections.Generic</i> .....	348
<i>System.Collections.ObjectModel</i> .....	349
<i>System.Collections.Specialized</i> .....	349

Mapping Non-Generic To Generic Collections .....	349
Quick Review .....	350
<b>Using Non-Generic Collection Classes - Pre .NET 2.0 .....</b>	<b>350</b>
Objects In – Objects Out: Casting 101 .....	351
Extending ArrayList To Create A Strongly-Typed Collection .....	352
<b>Using Generic Collection Classes – .NET 2.0 And Beyond .....</b>	<b>354</b>
List<T>: Look Ma, No More Casting! .....	354
Implementing KeyedCollection<TKey, TItem> .....	355
Quick Review .....	356
<b>Special Operations On Collections .....</b>	<b>357</b>
Sorting A List .....	357
Implementing System.IComparable<T> .....	357
Extending Comparer<T> .....	359
Converting A Collection Into An Array .....	361
Quick Review .....	361
<b>SUMMARY .....</b>	<b>362</b>
<b>Skill-Building Exercises .....</b>	<b>362</b>
<b>SUGGESTED PROJECTS .....</b>	<b>363</b>
<b>SELF-TEST QUESTIONS .....</b>	<b>363</b>
<b>REFERENCES .....</b>	<b>364</b>
<b>NOTES .....</b>	<b>364</b>

## 15 EXCEPTIONS: Writing Fault-Tolerant Software

<b>INTRODUCTION .....</b>	<b>366</b>
<b>WHAT IS AN EXCEPTION .....</b>	<b>366</b>
<b>.NET CLR Exception Handling Mechanism .....</b>	<b>366</b>
Unhandled Exceptions .....	366
The Exception Information Table .....	367
Quick Review .....	367
<b>Exception Class Hierarchy .....</b>	<b>367</b>
Application vs. Runtime Exceptions .....	368
Runtime Exception Listing .....	368
Determining What Exceptions A .NET Framework Method Throws .....	369
Quick Review .....	369
<b>Exception Class Properties .....</b>	<b>370</b>
Quick Review .....	370
<b>CREATING EXCEPTION HANDLERS: Using Try/Catch/Finally Blocks .....</b>	<b>371</b>
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
<b>CREATING CUSTOM EXCEPTIONS .....</b>	<b>374</b>
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
<b>DOCUMENTING EXCEPTIONS .....</b>	<b>376</b>
<b>SUMMARY .....</b>	<b>377</b>
<b>Skill-Building Exercises .....</b>	<b>377</b>
<b>SUGGESTED PROJECTS .....</b>	<b>378</b>
<b>SELF-TEST QUESTIONS .....</b>	<b>378</b>
<b>REFERENCES .....</b>	<b>378</b>
<b>NOTES .....</b>	<b>379</b>

## 16 Multithreaded Programming

<b>INTRODUCTION .....</b>	<b>382</b>
<b>MULTITHREADING OVERVIEW: THE TALE OF TWO VACATIONS .....</b>	<b>382</b>
<i>Single-Threaded Vacation .....</i>	<i>382</i>
<i>Multithreaded Vacation .....</i>	<i>382</i>
<i>The Relationship Between A Process And Its Threads .....</i>	<i>383</i>
<i>Vacation Gone Bad .....</i>	<i>384</i>
<i>Quick Review .....</i>	<i>385</i>
<b>CREATING MANAGED THREADS WITH THE THREAD CLASS .....</b>	<b>385</b>
<i>Single-Threaded Vacation Example .....</i>	<i>386</i>
<i>Multithreaded Vacation Example .....</i>	<i>386</i>
<i>Thread States .....</i>	<i>389</i>
<i>Creating And Starting Managed Threads .....</i>	<i>389</i>
<i>ThreadStart Delegate .....</i>	<i>389</i>
<i>ParameterizedThreadStart Delegate: Passing Arguments To Threads.....</i>	<i>390</i>
<i>Blocking A Thread With Thread.Sleep() .....</i>	<i>391</i>
<i>Blocking A Thread With Thread.Join() .....</i>	<i>392</i>
<i>Foreground vs. Background Threads .....</i>	<i>394</i>
<i>Quick Review .....</i>	<i>395</i>
<b>CREATING THREADS WITH THE BACKGROUNDWORKER CLASS .....</b>	<b>396</b>
<i>Quick Review .....</i>	<i>399</i>
<b>THREAD POOLS .....</b>	<b>399</b>
<i>Quick Review .....</i>	<i>400</i>
<b>ASYNCHRONOUS METHOD CALLS .....</b>	<b>400</b>
<i>Obtaining Results From An Asynchronous Method Call .....</i>	<i>402</i>
<i>Providing A Callback Method To BeginInvoke() .....</i>	<i>402</i>
<i>Quick Review .....</i>	<i>403</i>
<b>SUMMARY .....</b>	<b>404</b>
<b>SKILL-BUILDING EXERCISES .....</b>	<b>405</b>
<b>SUGGESTED PROJECTS .....</b>	<b>405</b>
<b>SELF-TEST QUESTIONS .....</b>	<b>406</b>
<b>REFERENCES .....</b>	<b>406</b>
<b>NOTES .....</b>	<b>407</b>

## 17 File I/O

<b>INTRODUCTION .....</b>	<b>410</b>
<b>MANIPULATING DIRECTORIES AND FILES .....</b>	<b>410</b>
<i>Files, Directories, And Paths .....</i>	<i>411</i>
<i>Manipulating Directories And Files .....</i>	<i>411</i>
<i>Verbatim String Literals .....</i>	<i>412</i>
<i>Quick Review .....</i>	<i>413</i>
<b>SERIALIZING OBJECTS TO DISK .....</b>	<b>413</b>
<i>Serializable Attribute .....</i>	<i>413</i>
<i>Serializing Objects With BinaryFormatter .....</i>	<i>414</i>
<i>Serializing Objects With XmlSerializer .....</i>	<i>416</i>
<i>Quick Review .....</i>	<i>418</i>
<b>WORKING WITH TEXT FILES .....</b>	<b>418</b>
<i>Some Issues You Must Consider .....</i>	<i>418</i>
<i>Saving Dog Data To A Text File .....</i>	<i>418</i>
<i>Quick Review .....</i>	<i>420</i>
<b>WORKING WITH BINARY DATA .....</b>	<b>420</b>
<i>Quick Review .....</i>	<i>421</i>
<b>RANDOM ACCESS FILE I/O .....</b>	<b>422</b>
<i>Towards An Approach To The Adapter Project .....</i>	<i>422</i>

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

## 18 NETWORK PROGRAMMING FUNDAMENTALS

<b>INTRODUCTION</b> .....	<b>450</b>
<b>WHAT IS A COMPUTER NETWORK?</b> .....	<b>450</b>
<i>Purpose Of A Network</i> .....	450
<i>The Role Of Network Protocols</i> .....	451
<i>Homogeneous Vs. Heterogeneous Networks</i> .....	451
<i>The Unifying Network Protocols: TCP/IP</i> .....	451
<i>What's So Special About The Internet?</i> .....	452
<i>Quick Review</i> .....	452
<b>SERVERS &amp; CLIENTS</b> .....	<b>453</b>
<i>Server Hardware And Software</i> .....	453
<i>Client Hardware And Software</i> .....	453
<i>Quick Review</i> .....	454
<b>Application Distribution</b> .....	<b>454</b>
<i>Physical Distribution On One Computer</i> .....	454
<i>Running Multiple Clients On The Same Computer</i> .....	454
<i>Addressing The Local Machine</i> .....	455
<i>Physical Distribution Across Multiple Computers</i> .....	455
<i>Quick Review</i> .....	455
<b>MULTITIERED APPLICATIONS</b> .....	<b>456</b>
<i>Logical Application Tiers</i> .....	456
<i>Physical Tier Distribution</i> .....	456
<i>Quick Review</i> .....	456
<b>INTERNET NETWORKING PROTOCOLS: NUTS &amp; BOLTS</b> .....	<b>457</b>
<i>The Internet Protocols: TCP, UDP, And IP</i> .....	457
<i>The Application Layer</i> .....	458
<i>Transport Layer</i> .....	458
<i>Network Layer</i> .....	459
<i>Data Link And Physical Layers</i> .....	459
<i>Putting It All Together</i> .....	459
<i>What You Need To Know</i> .....	460
<i>Quick Review</i> .....	460
<b>SUMMARY</b> .....	<b>460</b>
<b>Skill-Building Exercises</b> .....	<b>461</b>
<b>SUGGESTED PROJECTS</b> .....	<b>462</b>
<b>Self-Test Questions</b> .....	<b>462</b>

<b>REFERENCES .....</b>	<b>462</b>
<b>NOTES .....</b>	<b>463</b>

## 19 NETWORKED CLIENT -SERVER Applications

<b>INTRODUCTION .....</b>	<b>466</b>
<b>BUILDING CLIENT-SERVER Applications With .NET REMOTING .....</b>	<b>466</b>
<i>THE THREE REQUIRED COMPONENTS OF A .NET REMOTING Application .....</i>	<i>466</i>
<i>A SIMPLE .NET REMOTING Application .....</i>	<i>467</i>
<i>SINGLECALL VS. SINGLETON .....</i>	<i>469</i>
<i>ACCESSING A REMOTE OBJECT VIA AN INTERFACE .....</i>	<i>470</i>
<i>USING CONFIGURATION FILES .....</i>	<i>472</i>
<i>PASSING OBJECTS BETWEEN CLIENT AND SERVER .....</i>	<i>474</i>
<i>QUICK REVIEW .....</i>	<i>477</i>
<b>CLIENT-SERVER Applications With TcpListener And TcpClient .....</b>	<b>478</b>
<i>TCP/IP CLIENT-SERVER OVERVIEW .....</i>	<i>478</i>
<i>A SIMPLE CLIENT-SERVER Application .....</i>	<i>479</i>
<i>BUILDING A MULTITHREADED SERVER .....</i>	<i>480</i>
<i>LISTENING ON MULTIPLE IP ADDRESSES .....</i>	<i>482</i>
<i>SENDING OBJECTS BETWEEN CLIENT AND SERVER .....</i>	<i>484</i>
<i>QUICK REVIEW .....</i>	<i>488</i>
<b>SUMMARY .....</b>	<b>489</b>
<b>SKILL-BUILDING EXERCISES .....</b>	<b>490</b>
<b>SUGGESTED PROJECTS .....</b>	<b>491</b>
<b>SELF-TEST QUESTIONS .....</b>	<b>491</b>
<b>REFERENCES .....</b>	<b>492</b>
<b>NOTES .....</b>	<b>492</b>

## 20 DATABASE ACCESS & MULTITIERED Applications

<b>INTRODUCTION .....</b>	<b>494</b>
<b>WHAT YOU ARE GOING TO BUILD .....</b>	<b>494</b>
<b>PRELIMINARIES .....</b>	<b>495</b>
<i>INSTALLING SQL SERVER EXPRESS EDITION .....</i>	<i>495</i>
<i>INSTALLING MICROSOFT SQL SERVER MANAGEMENT STUDIO EXPRESS .....</i>	<i>496</i>
<i>INSTALLING MICROSOFT ENTERPRISE LIBRARY .....</i>	<i>498</i>
<i>A SIMPLE TEST APPLICATION .....</i>	<i>499</i>
<b>INTRODUCTION TO RELATIONAL DATABASES AND SQL .....</b>	<b>501</b>
<i>TERMINOLOGY .....</i>	<i>501</i>
<i>STRUCTURED QUERY LANGUAGE (SQL) .....</i>	<i>502</i>
<i>DATA DEFINITION LANGUAGE (DDL) .....</i>	<i>502</i>
<i>CREATING THE EMPLOYEETRAINING DATABASE .....</i>	<i>502</i>
<i>CREATING A DATABASE WITH A SCRIPT .....</i>	<i>503</i>
<i>CREATING TABLES .....</i>	<i>504</i>
<i>SQL SERVER DATABASE TYPES .....</i>	<i>505</i>
<i>DATA MANIPULATION LANGUAGE (DML) .....</i>	<i>506</i>
<i>USING THE INSERT COMMAND .....</i>	<i>506</i>
<i>USING THE SELECT COMMAND .....</i>	<i>507</i>
<i>USING THE UPDATE COMMAND .....</i>	<i>509</i>
<i>USING THE DELETE COMMAND .....</i>	<i>510</i>
<i>QUICK REVIEW .....</i>	<i>510</i>
<b>COMPLEX SQL QUERIES .....</b>	<b>511</b>
<i>CREATING A RELATED TABLE WITH A FOREIGN KEY .....</i>	<i>511</i>
<i>INSERTING TEST DATA INTO THE tbl_EMPLOYEE_TRAINING Table .....</i>	<i>512</i>
<i>SELECTING DATA FROM MULTIPLE TABLES .....</i>	<i>514</i>

Join Operations .....	514
Testing The Cascade Delete Constraint .....	515
Quick Review .....	515
<b>The Server Application .....</b>	<b>516</b>
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
<b>The Client Application .....</b>	<b>556</b>
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
<b>Summary .....</b>	<b>583</b>
<b>Skill-Building Exercises .....</b>	<b>583</b>
<b>Suggested Projects .....</b>	<b>584</b>
<b>Self-Test Questions .....</b>	<b>584</b>
<b>References .....</b>	<b>585</b>
<b>Notes .....</b>	<b>585</b>

## 21 Operator Overloading

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



<b>SUGGESTED PROJECTS</b> .....	<b>611</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>611</b>
<b>REFERENCES</b> .....	<b>612</b>
<b>NOTES</b> .....	<b>612</b>

## 22 Well-Behaved Objects

<b>INTRODUCTION</b> .....	<b>614</b>
<b>Object BEHAVIOR Defined</b> .....	<b>614</b>
<i>FUNDAMENTAL BEHAVIOR</i> .....	614
<i>COPY/ASSIGNMENT BEHAVIOR</i> .....	614
<i>EQUALITY BEHAVIOR</i> .....	615
<i>COMPARISON/ORDERING BEHAVIOR</i> .....	615
<i>SEVEN OBJECT USAGE SCENARIOS</i> .....	615
<b>FUNDAMENTAL BEHAVIOR</b> .....	<b>616</b>
<i>Object CREATION – CONSTRUCTORS</i> .....	616
<i>Default CONSTRUCTOR</i> .....	616
<i>PRIVATE CONSTRUCTORS</i> .....	616
<i>Overloaded CONSTRUCTORS</i> .....	616
<i>Member Accessibility</i> .....	616
<i>HORIZONTAL MEMBER ACCESS</i> .....	616
<i>VERTICAL MEMBER ACCESS</i> .....	617
<i>Overriding Object.ToString()</i> .....	617
<i>Static vs. Instance MEMBERS</i> .....	617
<i>SERIALIZATION</i> .....	618
<i>CUSTOM SERIALIZATION Example</i> .....	618
<i>Quick Review</i> .....	623
<b>Copy/ASSIGNMENT BEHAVIOR</b> .....	<b>623</b>
<i>Value Object vs. Reference Object ASSIGNMENT</i> .....	624
<i>RULE OF THUMB – FAVOR THE CLASS CONSTRUCT FOR Complex Types</i> .....	624
<i>Shallow Copy vs. Deep Copy</i> .....	624
<i>COPY CONSTRUCTORS</i> .....	624
<i>SYSTEM.ICLONEABLE vs. Object.MEMBERWISEClone()</i> .....	627
<i>Quick Review</i> .....	629
<b>EQUALITY BEHAVIOR</b> .....	<b>629</b>
<i>REFERENCE Equality vs. Value Equality</i> .....	629
<i>RULES FOR OVERRIDING THE Object.Equals() METHOD</i> .....	629
<i>Overriding THE Object.GetHashCode() METHOD</i> .....	630
<i>Bloch's hash CODE GENERATION ALGORITHM</i> .....	631
<i>ASHMORE'S Hash CODE GENERATION ALGORITHM</i> .....	631
<i>Overriding Object.Equals() AND Object.GetHashCode() METHODS IN THE PERSONVO CLASS</i> .....	631
<i>Quick Review</i> .....	633
<b>COMPARISON/ORDERING BEHAVIOR</b> .....	<b>633</b>
<i>IMPLEMENTING SYSTEM.ICOMPARABLE&lt;T&gt;</i> .....	634
<i>RULES FOR IMPLEMENTING THE CompareTo(T other) METHOD</i> .....	635
<i>EXTENDING THE Comparer&lt;T&gt; CLASS</i> .....	636
<i>Quick Review</i> .....	637
<b>SUMMARY</b> .....	<b>637</b>
<b>SKILL-BUILDING EXERCISES</b> .....	<b>638</b>
<b>SUGGESTED PROJECTS</b> .....	<b>638</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>638</b>
<b>REFERENCES</b> .....	<b>639</b>
<b>NOTES</b> .....	<b>639</b>

## 23 THREE DESIGN PRINCIPLES

<b>INTRODUCTION .....</b>	<b>642</b>
<b>THE PREFERRED CHARACTERISTICS OF AN OBJECT-ORIENTED ARCHITECTURE .....</b>	<b>642</b>
<i>EASY TO UNDERSTAND: HOW DOES THIS THING WORK?</i> .....	642
<i>EASY TO REASON ABOUT: WHAT ARE THE EFFECTS OF CHANGE?</i> .....	642
<i>EASY TO EXTEND: WHERE DO I ADD FUNCTIONALITY?</i> .....	642
<b>THE LISKOV SUBSTITUTION PRINCIPLE &amp; DESIGN BY CONTRACT .....</b>	<b>643</b>
<i>REASONING ABOUT THE BEHAVIOR OF SUPERTYPES AND SUBTYPES</i> .....	643
<i>RELATIONSHIP BETWEEN THE LSP AND DbC</i> .....	643
<i>THE COMMON GOAL OF THE LSP AND DbC</i> .....	643
<i>C# SUPPORT FOR THE LSP AND DbC</i> .....	643
<i>DESIGNING WITH THE LSP/DbC IN MIND</i> .....	644
<i>CLASS DECLARATIONS VIEWED AS BEHAVIOR SPECIFICATIONS</i> .....	644
<i>QUICK REVIEW</i> .....	644
<b>PRECONDITIONS, POSTCONDITIONS, AND CLASS INVARIANTS .....</b>	<b>644</b>
<i>CLASS INVARIANT</i> .....	644
<i>PRECONDITION</i> .....	644
<i>POSTCONDITION</i> .....	645
<i>AN EXAMPLE</i> .....	645
<i>A NOTE ON USING THE DEBUG.ASSERT() METHOD TO ENFORCE PRE- AND POSTCONDITIONS</i> .....	646
<i>USING INCREMENTER AS A BASE CLASS</i> .....	646
<i>CHANGING THE PRECONDITIONS OF DERIVED CLASS METHODS</i> .....	648
<i>CHANGING THE POSTCONDITIONS OF DERIVED CLASS METHODS</i> .....	652
<i>SPECIAL CASES OF PRECONDITIONS AND POSTCONDITIONS</i> .....	652
<i>METHOD ARGUMENT TYPES</i> .....	653
<i>METHOD RETURN TYPES</i> .....	654
<i>THREE RULES OF THE SUBSTITUTION PRINCIPLE</i> .....	654
<i>SIGNATURE RULE</i> .....	655
<i>METHODS RULE</i> .....	655
<i>PROPERTIES RULE</i> .....	655
<i>QUICK REVIEW</i> .....	655
<b>THE OPEN-CLOSED PRINCIPLE .....</b>	<b>656</b>
<i>ACHIEVING THE OPEN-CLOSED PRINCIPLE</i> .....	656
<i>AN OCP EXAMPLE</i> .....	656
<i>QUICK REVIEW</i> .....	661
<b>THE DEPENDENCY INVERSION PRINCIPLE .....</b>	<b>661</b>
<i>CHARACTERISTICS OF BAD SOFTWARE ARCHITECTURE</i> .....	661
<i>CHARACTERISTICS OF GOOD SOFTWARE ARCHITECTURE</i> .....	662
<i>SELECTING THE RIGHT ABSTRACTIONS TAKES EXPERIENCE</i> .....	662
<i>QUICK REVIEW</i> .....	662
<b>TERMS AND DEFINITIONS .....</b>	<b>663</b>
<b>SUMMARY .....</b>	<b>663</b>
<b>SKILL-BUILDING EXERCISES .....</b>	<b>664</b>
<b>SUGGESTED PROJECTS .....</b>	<b>664</b>
<b>SELF-TEST QUESTIONS .....</b>	<b>664</b>
<b>REFERENCES .....</b>	<b>665</b>
<b>NOTES .....</b>	<b>666</b>

## 24 INHERITANCE, COMPOSITION, INTERFACES, POLYMORPHISM

<b>INTRODUCTION .....</b>	<b>668</b>
<b>INHERITANCE VS. COMPOSITION: THE GREAT DEBATE .....</b>	<b>668</b>
<i>WHAT'S THE END GAME?</i> .....	669
<i>FLEXIBLE APPLICATION ARCHITECTURES</i> .....	669
<i>MODULARITY AND RELIABILITY</i> .....	669

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

## 25 Helpful Design Patterns

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

<i>Com.PulpFreePress.Utils</i> .....	707
<i>Com.PulpFreePress.Commands</i> .....	710
<i>Com.PulpFreePress.Model</i> .....	713
<i>Com.PulpFreePress.View</i> .....	714
<i>Com.PulpFreePress.Controller</i> .....	720
<i>Running The Application</i> .....	721
<b>SUMMARY</b> .....	<b>722</b>
<b>Skill-Building Exercises</b> .....	<b>722</b>
<b>SUGGESTED PROJECTS</b> .....	<b>723</b>
<b>SELF-TEST QUESTIONS</b> .....	<b>723</b>
<b>REFERENCES</b> .....	<b>723</b>
<b>NOTES</b> .....	<b>724</b>

## Appendix A: Helpful Checklists And Tables

<b>Project-Approach Strategy Check-off List</b> .....	<b>727</b>
<b>Development Cycle</b> .....	<b>728</b>
<b>Final Project Review Checklist</b> .....	<b>728</b>

## Appendix B: ASCII Table

<b>ASCII Table</b> .....	<b>729</b>
--------------------------	------------

## Appendix C: Identifier Naming: Writing Self-Commenting Code

<b>Identifier Naming: Writing Self-Commenting Code</b> .....	<b>733</b>
<i>Benefits of Self-Commenting Code</i> .....	733
<i>Coding Convention</i> .....	733
<i>Class Names</i> .....	733
<i>Constant Names</i> .....	734
<i>Variable Names</i> .....	734
<i>Method Names</i> .....	734
<i>Property Names</i> .....	735