

DETAILED CONTENTS

PREFACE

WELCOME – AND THANK YOU!	xlvi
TARGET AUDIENCE	xlvi
APPROACH(ES)	xlvi
PEDAGOGY – I MEAN, HOW THIS BOOK'S ARRANGED	xlvi
LEARNING OBJECTIVES	xlvi
INTRODUCTION	xlvi
CONTENT	xlvi
QUICK REVIEWS	xlvi
SUMMARY	xlvi
SKILL-BUILDING EXERCISES	xlvi
SUGGESTED PROJECTS	xlvi
SELF-TEST QUESTIONS	xlvi
REFERENCES	xlvi
NOTES	xlvi
TYPOGRAPHICAL FORMATS	xlvi
THIS IS AN EXAMPLE OF A FIRST LEVEL SUBHEADING	xlvi
<i>This Is An Example Of A Second Level Subheading</i>	xlvi
SOURCE CODE FORMATTING	xlvi
SUPPORTSITE™ WEBSITE	xlvi
PROBLEM REPORTING	xlvi
ABOUT THE AUTHOR	xlvi
ACKNOWLEDGMENTS	xlvi

I AN APPROACH TO THE ART OF PROGRAMMING

INTRODUCTION	4
THE DIFFICULTIES YOU WILL ENCOUNTER LEARNING C#	4
<i>REQUIRED Skills</i>	4
<i>THE PLANETS WILL COME INTO ALIGNMENT</i>	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

<i>EXECUTING THE APPLICATION</i>	30
Quick Review	31
CREATING PROJECTS WITH MICROSOFT VISUAL C# EXPRESS	32
Download and Install Visual C# Express	32
Quick Tour Of Visual C# Express	33
<i>SELECT PROJECT TYPE</i>	33
<i>SAVING THE PROJECT</i>	36
<i>BUILD THE PROJECT</i>	36
<i>LOCATING THE PROJECT EXECUTABLE FILE</i>	36
<i>EXECUTE THE PROJECT</i>	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
<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
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
<i>IMPLEMENTING STATE TRANSITION DIAGRAMS</i>	60
<i>IMPLEMENTING THE PRINTFLOOR() METHOD</i>	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
Notes	74

4 Computers, Programs, And Algorithms

Introduction	76
What Is A Computer?	76
Computer vs. Computer System	76
<i>Computer System</i>	76
<i>Processor</i>	78
Three Aspects of Processor Architecture	79
<i>Feature Set</i>	79
<i>Feature Set Implementation</i>	79
<i>Feature Set Accessibility</i>	79
Memory Organization	79
Memory Basics	80
<i>Memory Hierarchy</i>	80
<i>Bits, Bytes, Words</i>	80
Alignment and Addressability	81
What Is A Program?	82
Two Views of A Program	82
<i>The Human Perspective</i>	82
<i>The Computer Perspective</i>	82
The Processing Cycle	82
Fetch	83
Decode	83
Execute	83
Store	83
<i>Why A Program Crashes</i>	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
<i>Four Parts Of The Common Language Infrastructure</i>	87
<i>The Cross Platform Promise</i>	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
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

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
<i>Message Array</i>	171
<i>Calculating Averages</i>	173
<i>Histogram: Letter Frequency Counter</i>	173
Quick Review	175
Creating And Using Multidimensional Arrays	176
Rectangular Arrays	176
<i>Initializing Rectangular Arrays With Array Literals</i>	178
Ragged Arrays	178
Multidimensional Arrays In Action	179
<i>Weighted Grade Tool</i>	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
<i>Fields</i>	195
<i>Constants</i>	197
<i>The Difference Between const and readonly; Compile-Time vs. Runtime Constants</i>	197
<i>Properties</i>	198
<i>Methods</i>	199
<i>Instance Constructors</i>	199

<i>Static Constructors</i>	200
<i>Events</i>	200
<i>Operators</i>	200
<i>Indexers</i>	200
<i>Nested Type Declarations</i>	200
<i>Finalizers</i>	200
Access Modifiers	201
<i>Public</i>	201
<i>Private</i>	201
<i>Protected</i>	201
<i>Internal</i>	201
<i>Protected Internal</i>	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
Maximize Method Cohesion	203
Structure Of A Method Definition	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
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
<i>Adding Properties</i>	208
<i>Adding A Constructor Method</i>	208
Testing The Person Class: A Miniature Test Plan	209
<i>Use The PeopleManagerApplication Class As A Test Driver</i>	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
<i>Value Parameters: The Default Parameter Passing Mode</i>	219
<i>Reference Parameters: Using The ref Parameter Modifier</i>	220
The <i>out</i> Parameter Modifier	223
Parameter Arrays: Using The <i>params</i> 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
<i>The Relationship BETWEEN Aggregation And Object Lifetime.....</i>	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

II 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
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
<i>Chained Hash Table vs. Open-Address Hash Table</i>	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
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
<i>IMPLEMENTING SYSTEM.ICOMPARABLE<T></i>	357
<i>EXTENDING COMPARE<T></i>	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
<i>First Line of Defense: Use Defensive Coding</i>	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

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
<i>ThreadStart Delegate</i>	389
<i>ParameterizedThreadStart Delegate: Passing Arguments To Threads</i>	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 DOQ DATA TO A TEXT FILE	418
Quick Review	420
WORKING WITH BINARY DATA	420
Quick Review	422
RANDOM ACCESS FILE I/O	422
TOWARDS AN APPROACH TO THE ADAPTER PROJECT	422
<i>Start Small And Take Baby Steps</i>	423
OTHER PROJECT CONSIDERATIONS	424
<i>Locking A Record For Updates And DELETES</i>	424
<i>MONITOR. ENTER()/MONITOR.EXIT() vs. The lock keyword</i>	425
<i>TRANSLATING LOW-LEVEL EXCEPTIONS INTO HIGHER-LEVEL EXCEPTION ABSTRACTIONS</i>	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
<i>HOMOGENEOUS VS. HETEROGENEOUS NETWORKS</i>	451
<i>THE UNIFYING NETWORK PROTOCOLS: TCP/IP</i>	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
<i>RUNNING MULTIPLE CLIENTS ON THE SAME COMPUTER</i>	454
<i>ADDRESSING THE LOCAL MACHINE</i>	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
<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
WHAT YOU NEED TO KNOW	460
Quick Review	460
SUMMARY	460
SKILL-BUILDING EXERCISES	461
SUGGESTED PROJECTS	462
SELF-TEST QUESTIONS	462
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	500
TERMINOLOGY	501
STRUCTURED QUERY LANGUAGE (SQL)	502
DATA DEFINITION LANGUAGE (DDL)	502
<i>CREATING THE EMPLOYEE TRAINING DATABASE</i>	502
<i>CREATING A DATABASE WITH A SCRIPT</i>	503
<i>CREATING TABLES</i>	504
SQL SERVER DATABASE TYPES	505
DATA MANIPULATION LANGUAGE (DML)	506
<i>USING THE INSERT COMMAND</i>	507
<i>USING THE SELECT COMMAND</i>	507
<i>USING THE UPDATE COMMAND</i>	509
<i>USING THE DELETE COMMAND</i>	510
Quick Review	511
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
<i>JOIN OPERATIONS</i>	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
<i>CODING THE EMPLOYEEVO AND EMPLOYEEDAO</i>	520
<i>APPLICATION CONFIGURATION FILE</i>	528
<i>CREATING TEST APPLICATION</i>	528
SECOND ITERATION	531
<i>TESTING THE CODE - SECOND ITERATION</i>	543
<i>REALITY CHECK</i>	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 EMPLOYEE TRAINING CLIENT 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
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

<i>Overloaded CONSTRUCTORS</i>	616
MEMBER ACCESSIBILITY	616
<i>HORIZONTAL MEMBER ACCESS</i>	616
<i>VERTICAL MEMBER ACCESS</i>	617
OVERRIDING <code>OBJECT.TOSTRING()</code>	617
STATIC VS. INSTANCE MEMBERS	617
SERIALIZATION	618
<i>CUSTOM SERIALIZATION EXAMPLE</i>	618
Quick Review	623
Copy/Assignment Behavior	623
VALUE OBJECT VS. REFERENCE OBJECT ASSIGNMENT	624
<i>RULE OF THUMB – FAVOR THE CLASS CONSTRUCT FOR COMPLEX TYPES</i>	624
SHALLOW COPY VS. DEEP COPY	624
COPY CONSTRUCTORS	625
SYSTEM.ICLONEABLE VS. OBJECT.MEMBERWISECLONE()	627
Quick Review	629
Equality Behavior	629
REFERENCE EQUALITY VS. VALUE EQUALITY	629
RULES FOR OVERRIDING THE <code>OBJECT.EQUALS()</code> METHOD	630
OVERRIDING THE <code>OBJECT.GETHASHCODE()</code> METHOD	630
<i>BLOCH'S HASH CODE GENERATION ALGORITHM</i>	631
<i>ASHMORE'S HASH CODE GENERATION ALGORITHM</i>	631
OVERRIDING <code>OBJECT.EQUALS()</code> AND <code>OBJECT.GETHASHCODE()</code> METHODS IN THE <code>PERSONVO</code> CLASS	632
Quick Review	634
Comparison/Ordering Behavior	634
IMPLEMENTING <code>SYSTEM.ICOMPARABLE<T></code>	634
<i>RULES FOR IMPLEMENTING THE <code>COMPARETO(T OTHER)</code> METHOD</i>	635
EXTENDING THE <code>COMPARER<T></code> CLASS	636
Quick Review	638
SUMMARY	638
SKILL-BUILDING EXERCISES	638
SUGGESTED PROJECTS	638
SELF-TEST QUESTIONS	639
REFERENCES	639
NOTES	640

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
<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
DESIGNING WITH THE LSP/DbC IN MIND	644
<i>CLASS DECLARATIONS VIEWED AS BEHAVIOR SPECIFICATIONS</i>	644
Quick Review	644
PRECONDITIONS, POSTCONDITIONS, AND CLASS INVARIANTS	644
CLASS INVARIANT	644
PRECONDITION	644

Postcondition	645
An Example	645
<i>A Note On Using The Debug.Assert() Method To Enforce Pre- And Postconditions</i>	646
Using Incrementer As A Base Class	646
<i>Changing The Preconditions Of Derived Class Methods</i>	648
Changing The Postconditions Of Derived Class Methods	652
Special Cases Of Preconditions And Postconditions	652
<i>Method Argument Types</i>	653
<i>Method Return Types</i>	654
Three Rules Of The Substitution Principle	654
<i>Signature Rule</i>	655
<i>Methods Rule</i>	655
<i>Properties Rule</i>	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
<i>Flexible Application Architectures</i>	669
<i>Modularity And Reliability</i>	669
<i>Architectural Stability Via Managed Dependencies</i>	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
<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
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
<i>Dominant Roles</i>	674

<i>Collateral Roles</i>	675
<i>Dynamic Roles</i>	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
<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
Running The Application	721
Summary	721
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	729
--------------------------	------------

Appendix C: Identifier Naming: Writing Self-Commenting Code

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

