

Table of Contents

Preface

WELCOME – AND THANK YOU!	vii
TARGET AUDIENCE	vii
SUPPORTSITE™ WEBSITE	vii
PROBLEM REPORTING	vii
ABOUT THE AUTHOR	vii
ACKNOWLEDGMENTS	viii

1 Collections Quick Start

INTRODUCTION	2
Why Use a Collection?	2
ArrayList Class	2
Polymorphic Behavior	5
Casting to a Specific Type	5
Quick Review	6
List<T> Class	6
Quick Review	7
Manipulating Lists	7
Sorting, Searching, and Reversing a List<T> Collection	7
<i>Default Sorting Behavior</i>	7
<i>Sort Before Calling The BinarySearch() Method</i>	7
Quick Review	8
WHERE TO GO FROM HERE	9
SUMMARY	9
REFERENCES	9
NOTES	10

2 Collections Framework Overview

INTRODUCTION	12
THE MICROSOFT DEVELOPER NETWORK DOCUMENTATION	12
THE MSDN WEBSITE – www.msdn.com	12
USING GOOGLE TO QUICKLY LOCATE DOCUMENTATION	14
WHERE TO GO FROM HERE	14
Quick Review	16
NAVIGATING AN INHERITANCE HIERARCHY	16
EXTENSION METHODS	17
Quick Review	18
THE .NET COLLECTIONS FRAMEWORK NAMESPACES	18
SYSTEM.COLLECTIONS	18
SYSTEM.COLLECTIONS.GENERIC	18
SYSTEM.COLLECTIONS.OBJECTMODEL	19
SYSTEM.COLLECTIONS.SPECIALIZED	19
MAPPING NON-GENERIC TO GENERIC COLLECTIONS	19

Quick Review	20
SUMMARY	20
REFERENCES	20
NOTES	21

3 ARRAYS

INTRODUCTION	24
WHAT IS AN ARRAY?	24
Specifying Array Types	25
Quick Review	26
FUNCTIONALITY PROVIDED BY C# ARRAY TYPES	26
ARRAY-TYPE INHERITANCE HIERARCHY	26
SPECIAL PROPERTIES OF C# ARRAYS	27
Quick Review	27
CREATING AND USING SINGLE-DIMENSIONAL ARRAYS	28
ARRAYS OF VALUE TYPES	28
HOW VALUE-TYPE ARRAY OBJECTS ARE ARRANGED IN MEMORY	28
FINDING AN ARRAY'S TYPE, RANK, AND TOTAL NUMBER OF ELEMENTS	29
CREATING SINGLE-DIMENSIONAL ARRAYS USING ARRAY LITERAL VALUES	30
DIFFERENCES BETWEEN ARRAYS OF VALUE TYPES AND ARRAYS OF REFERENCE TYPES	31
SINGLE-DIMENSIONAL ARRAYS IN ACTION	33
MESSAGE ARRAY	33
CALCULATING AVERAGES.....	35
HISTOGRAM: LETTER FREQUENCY COUNTER	35
Quick Review	37
CREATING AND USING MULTIDIMENSIONAL ARRAYS	38
RECTANGULAR ARRAYS	38
Initializing Rectangular Arrays With Array Literals.....	40
RAGGED ARRAYS	40
MULTIDIMENSIONAL ARRAYS IN ACTION	41
Weighted Grade Tool.....	41
Quick Review	43
THE MAIN() METHOD'S STRING ARRAY	43
PURPOSE AND USE OF THE MAIN() METHOD'S STRING ARRAY	43
MANIPULATING ARRAYS WITH THE SYSTEM.ARRAY CLASS	44
NON-SUPPORTED IList OPERATIONS	45
SUMMARY	45
REFERENCES	46
NOTES	46

4 FUNDAMENTAL DATA STRUCTURES

INTRODUCTION	48
ARRAY PERFORMANCE CHARACTERISTICS	48
LINKED LIST PERFORMANCE CHARACTERISTICS	49
HASH TABLE PERFORMANCE CHARACTERISTICS	51
CHAINED HASH TABLE vs. OPEN-ADDRESS HASH TABLE.....	51
RED-BLACK TREE PERFORMANCE CHARACTERISTICS	52
STACKS AND QUEUES	53
SUMMARY	54

REFERENCES54
 NOTES55

5 UNDERSTANDING GENERICS

INTRODUCTION58
 CREATING GENERIC TYPES58
 USING A SINGLE TYPE PARAMETER58
 USING MULTIPLE TYPE PARAMETERS59
 UNBOUNDED TYPE PARAMETERS60
 Quick Review60
 CREATING GENERIC METHODS60
 GENERIC TYPE INFERENCE61
 Quick Review61
 GENERIC TYPE CONSTRAINTS61
 DEFAULT CONSTRUCTOR CONSTRAINT62
 REFERENCE/VALUE TYPE CONSTRAINTS63
 REFERENCE TYPE CONSTRAINT.....64
 VALUE TYPE CONSTRAINT.....65
 CLASS/INTERFACE DERIVATION/IMPLEMENTATION CONSTRAINTS66
 INTERFACE IMPLEMENTATION CONSTRAINT66
 CLASS DERIVATION CONSTRAINT.....67
 NAKED CONSTRAINT70
 Limited Utility of the Naked Constraint.....71
 CONSTRAINT SUMMARY TABLE72
 Quick Review72
 BENEFITS OF USING GENERIC TYPES72
 INCREASED TYPE SAFETY72
 GENERICS SAVE SPACE73
 GENERICS IMPROVE PERFORMANCE73
 GENERICS ELIMINATE WORK AND IMPROVE CODE QUALITY74
 Quick Review74
 SUMMARY74
 REFERENCES75
 NOTES75

6 Lists

INTRODUCTION78
 ARRAY-BASED LIST COLLECTIONS – HOW THEY WORK78
 A HOME-GROWN DYNAMIC ARRAY78
 EVALUATING DYNAMICARRAY80
 THE ARRAYLIST CLASS TO THE RESCUE80
 Quick Review81
 THE NON-GENERIC ARRAYLIST: OBJECTS IN – OBJECTS OUT81
 ARRAYLIST INHERITANCE HIERARCHY81
 Functionality Provided by the IENUMERABLE AND IENUMERATOR INTERFACES.....82
 Functionality Provided by the ICOLLECTION INTERFACE84
 Functionality Provided by the IList INTERFACE.....84
 Functionality Provided by the ICLONEABLE INTERFACE84
 Functionality Provided by the SERIALIZABLEATTRIBUTE.....84
 Functionality Provided by the COMVISIBLEATTRIBUTE(TRUE).....84
 EXTENSION METHODS84

DEFENSIVE CODING USING THE FAÇADE SOFTWARE PATTERN	85
Quick Review	88
THE GENERIC LIST<T> COLLECTION	88
LIST<T> INHERITANCE HIERARCHY	88
<i>Functionality Provided by the IENUMERABLE<T> AND IENUMERATOR<T> INTERFACES</i>	<i>89</i>
<i>Functionality Provided by the ICOLLECTION<T> INTERFACE</i>	<i>89</i>
<i>Functionality Provided by the ILIST<T> INTERFACE</i>	<i>89</i>
BENEFITS OF USING LIST<T> VS. ARRAYLIST	89
Quick Review	90
LINKED LIST COLLECTIONS – HOW THEY WORK	90
LINKED LISTS VS. ARRAY-BASED LISTS	90
LINKED LIST OPERATION - THE CIRCULAR LINKED LIST	91
Quick Review	96
THE GENERIC LINKEDLIST<T> COLLECTION	96
LINKEDLIST<T> IS NON-CIRCULAR!	97
THE LINKEDLIST<T> INHERITANCE HIERARCHY	97
<i>Functionality Provided By The IENUMERABLE AND IENUMERABLE<T> INTERFACES</i>	<i>97</i>
<i>Functionality Provided by the ICOLLECTION AND ICOLLECTION<T> INTERFACES.....</i>	<i>97</i>
<i>Functionality Provided by the ISERIALIZABLE AND IDESERIALIZATIONCALLBACK INTERFACES.....</i>	<i>98</i>
LINKEDLIST<T> COLLECTION IN ACTION	98
Quick Review	99
SUMMARY	99
REFERENCES	100
NOTES	100

7 Stacks

INTRODUCTION	102
STACK OPERATIONS	102
CHARACTERISTIC STACK OPERATIONS	102
<i>Push.....</i>	<i>102</i>
<i>Pop.....</i>	<i>102</i>
<i>Peek.....</i>	<i>102</i>
AN ILLUSTRATION WILL HELP	102
WHAT'S ACTUALLY BEING PUSHED AND POPPED?	103
<i>PUSHING A VALUE TYPE OBJECT ONTO A STACK</i>	<i>103</i>
<i>PUSHING A REFERENCE TYPE OBJECT ONTO A STACK</i>	<i>103</i>
<i>VALUE TYPE BOXING IN ACTION</i>	<i>103</i>
<i>Disassembling Example 7.1.....</i>	<i>104</i>
Quick Review	106
A HOME GROWN STACK	107
Quick Review	108
THE STACK CLASS	109
STACK CLASS INHERITANCE HIERARCHY	109
<i>Functionality Provided by the IENUMERABLE INTERFACE.....</i>	<i>109</i>
<i>Functionality Provided by the ICOLLECTION INTERFACE</i>	<i>109</i>
<i>Functionality Provided by the ICLONEABLE INTERFACE</i>	<i>109</i>
BALANCED SYMBOL CHECKER	109
Quick Review	111
THE STACK<T> CLASS	112
STACK<T> CLASS INHERITANCE HIERARCHY	112
<i>Functionality Provided by the IENUMERABLE INTERFACE.....</i>	<i>112</i>
<i>Functionality Provided by the ICOLLECTION INTERFACE</i>	<i>112</i>
<i>Functionality Provided by the IENUMERABLE<T> INTERFACE.....</i>	<i>112</i>

WHAT HAPPENED TO ICollection<T>? 112
 COMMAND LINE POSTFIX CALCULATOR 113
 Quick Review 115
 SUMMARY 115
 REFERENCES 116
 NOTES 116

8 QUEUES

INTRODUCTION 118
 QUEUE OPERATIONS 118
 CHARACTERISTIC QUEUE OPERATIONS 118
 ENQUEUE 118
 DEQUEUE 118
 AN ILLUSTRATION WILL HELP 118
 Quick Review 119
 A HOME GROWN QUEUE BASED ON A CIRCULAR ARRAY 119
 Quick Review 125
 THE QUEUE CLASS 125
 QUEUE CLASS INHERITANCE HIERARCHY 125
 FUNCTIONALITY PROVIDED BY THE IENUMERABLE INTERFACE 125
 FUNCTIONALITY PROVIDED BY THE ICollection INTERFACE 125
 FUNCTIONALITY PROVIDED BY THE ICloneable INTERFACE 125
 PALINDROME CHECKER 126
 Quick Review 127
 THE QUEUE<T> CLASS 127
 QUEUE<T> CLASS INHERITANCE HIERARCHY 127
 FUNCTIONALITY PROVIDED BY THE ICollection INTERFACE 127
 FUNCTIONALITY PROVIDED BY THE IENUMERABLE INTERFACE 128
 FUNCTIONALITY PROVIDED BY THE IENUMERABLE<T> INTERFACE 128
 WHAT HAPPENED TO ICollection<T>? 128
 STORE SIMULATION 128
 Quick Review 129
 SUMMARY 130
 REFERENCES 130
 NOTES 131

9 HASHTABLES AND DICTIONARIES

INTRODUCTION 134
 HASHTABLE OPERATIONS 134
 HASHTABLE COLLISIONS 134
 HOMEGROWNHASHTABLE 135
 Quick Review 140
 HASHTABLE CLASS 140
 FUNCTIONALITY PROVIDED BY THE IENUMERABLE INTERFACE 140
 FUNCTIONALITY PROVIDED BY THE ICollection INTERFACE 141
 FUNCTIONALITY PROVIDED BY THE IDictionary INTERFACE 141
 FUNCTIONALITY PROVIDED BY THE ISerializable AND IDeserializationCallback INTERFACES 141
 FUNCTIONALITY PROVIDED BY THE ICloneable INTERFACE 141
 HASHTABLE IN ACTION 141
 Quick Review 143
 DICTIONARY<TKey, TValue> CLASS 143

<i>Functionality Provided by the IENUMERABLE AND IENUMERABLE<KEYVALUEPAIR<TKEY, TVALUE>> INTERFACES</i>	143
<i>Functionality Provided by the ICOLLECTION AND ICOLLECTION<KEYVALUEPAIR<TKEY, TVALUE>> INTERFACES</i>	143
<i>Functionality Provided by the IDICTIONARY AND IDICTIONARY<KEYVALUEPAIR<TKEY, TVALUE>> INTERFACES</i>	144
<i>Functionality Provided by the ISERIALIZABLE AND IDESERIALIZATIONCALLBACK INTERFACES</i>	144
<i>DICTIONARY<TKEY, TVALUE> EXAMPLE</i>	144
Quick Review	145
SUMMARY	145
REFERENCES	145
NOTES	146

10 Coding For Collections

INTRODUCTION	148
CODING FOR EQUALITY OPERATIONS	148
REFERENCE EQUALITY VS. VALUE EQUALITY	148
OVERRIDING OBJECT.EQUALS() AND OBJECT.GETHASHCODE()	149
<i>RULES FOR OVERRIDING THE OBJECT.EQUALS() METHOD</i>	149
<i>RULES FOR OVERRIDING THE OBJECT.GETHASHCODE() METHOD</i>	149
<i>BLOCH'S HASH CODE GENERATION ALGORITHM</i>	150
<i>ASHMORE'S HASH CODE GENERATION ALGORITHM</i>	150
AN EXAMPLE: THE PERSON CLASS	151
OVERLOADING THE == AND != OPERATORS	153
Quick Review	155
CODING FOR COMPARISON OPERATIONS	156
NATURAL ORDERING	156
<i>ICOMPARABLE AND ICOMPARABLE<T> INTERFACES</i>	156
CUSTOM ORDERING: CREATING SEPARATE COMPARER OBJECTS	159
<i>ICOMPARER AND ICOMPARER<T> INTERFACES</i>	159
<i>AN EXAMPLE: PERSONAGECOMPARER</i>	160
Quick Review	161
USING OBJECTS AS KEYS	161
RULES FOR OBJECTS USED AS KEYS	161
OBJECT IMMUTABILITY	162
EXAMPLE: PERSONKEY CLASS	163
Quick Review	166
SUMMARY	166
REFERENCES	166
NOTES	167

11 SORTED COLLECTIONS

INTRODUCTION	170
RED-BLACK TREE	170
WHY RED-BLACK TREES?	170
RED-BLACK TREE OPERATION	171
<i>THE RULES OF THE GAME</i>	172
RED-BLACK TREE CODE	174
Quick Review	184
SORTEDDICTIONARY<TKEY, TVALUE>	185
IENUMERABLE<KEYVALUEPAIR<TKEY, TVALUE>> AND IENUMERABLE INTERFACES	185
ICOLLECTION<KEYVALUEPAIR<TKEY, TVALUE>> AND ICOLLECTION INTERFACES	185
IDICTIONARY<TKEY, TVALUE> AND IDICTIONARY INTERFACES	185
SORTEDDICTIONARY<TKEY, TVALUE> EXAMPLE PROGRAM	185

Quick Review	189
SORTEDLIST<TKEY, TVALUE>	190
PERFORMANCE DIFFERENCES BETWEEN SORTEDLIST AND SORTEDDICTIONARY	190
SORTEDLIST<TKEY, TVALUE> EXAMPLE	190
Quick Review	191
SUMMARY	191
REFERENCES	192
NOTES	192

12 SETS

INTRODUCTION	194
HASHSET<T> VS. SORTEDSET<T>	194
HASHSET<T> INHERITANCE HIERARCHY	194
IENUMERABLE<T> AND IENUMERABLE	194
ICOLLECTION<T>	194
ISET<T>	195
ISERIALIZABLE AND IDESERIALIZATIONCALLBACK	195
SORTEDSET<T> INHERITANCE HIERARCHY	195
IENUMERABLE<T> AND IENUMERABLE	195
ICOLLECTION<T> AND ICOLLECTION	195
ISET<T>	195
ISERIALIZABLE AND IDESERIALIZATIONCALLBACK	195
Quick Review	195
SET OPERATIONS	196
INTERSECTWITH()	196
UNIONWITH()	197
ISSUBSETOF()	198
ISPROPERSUBSETOF()	200
ISSUPERSETOF()	201
ISPROPERSUPERSETOF()	203
OVERLAPS()	205
SYMMETRICEXCEPTWITH()	208
Quick Review	209
SUMMARY	209
REFERENCES	209
NOTES	209

13 THREAD PROGRAMMING

INTRODUCTION	212
MULTITHREADING OVERVIEW: THE TALE OF TWO VACATIONS	212
SINGLE-THREADED VACATION	212
MULTITHREADED VACATION	212
THE RELATIONSHIP BETWEEN A PROCESS AND ITS THREADS	213
VACATION GONE BAD	214
Quick Review	215
CREATING MANAGED THREADS WITH THE THREAD CLASS	215
SINGLE-THREADED VACATION EXAMPLE	216
MULTITHREADED VACATION EXAMPLE	216
THREAD STATES	219
CREATING AND STARTING MANAGED THREADS	219
THREADSTART DELEGATE	219

<i>PARAMETERIZEDTHREADSTART DELEGATE: PASSING ARGUMENTS TO THREADS</i>	220
Blocking A Thread With Thread.Sleep()	221
Blocking A Thread With Thread.Join()	222
Foreground vs. Background Threads	224
Quick Review	225
CREATING THREADS WITH THE BACKGROUNDWORKER CLASS	226
Quick Review	229
THREAD POOLS	229
Quick Review	230
ASYNCHRONOUS METHOD CALLS	230
Obtaining Results From An Asynchronous Method Call	232
Providing A Callback Method To BeginInvoke()	232
Quick Review	233
SUMMARY	234
REFERENCES	235
NOTES	236

14 Collections And Threads

INTRODUCTION	238
THE NEED FOR THREAD SYNCHRONIZATION	238
Quick Review	240
Using THE C# lock KEYWORD	240
Quick Review	241
ANATOMY OF .NET THREAD SYNCHRONIZATION	242
Old School – SyncRoot, IsSynchronized, and Synchronized()	242
Quick Review	245
MONITOR.ENTER() AND MONITOR.EXIT()	246
Using MONITOR.ENTER() AND MONITOR.EXIT()	248
Using OVERLOADED MONITOR.ENTER() METHOD	249
NON-BLOCKING MONITOR.TRYENTER()	251
Quick Review	253
SYNCHRONIZING ENTIRE METHODS	254
Quick Review	255
SYNCHRONIZED COLLECTIONS IN THE SYSTEM.COLLECTIONS.GENERIC NAMESPACE	255
THREAD SYNCHRONIZATION – RECOMMENDATIONS FOR USAGE	255
THREAD SYNCHRONIZATION USAGE TABLE	256
SUMMARY	259
REFERENCE	260
NOTES	260

15 EVENTS AND EVENT PROCESSING

INTRODUCTION	262
C# EVENT PROCESSING MODEL: AN OVERVIEW	262
Quick Review	263
CUSTOM EVENTS EXAMPLE: MINUTE TICK	263
CUSTOM EVENTS EXAMPLE: AUTOMATED WATER TANK SYSTEM	266
NAMING CONVENTIONS	271
FINAL THOUGHTS ON EXTENDING THE EVENTARGS CLASS	272

SUMMARY272
 REFERENCES273
 NOTES273

16 Collections And Events

INTRODUCTION276
OBSERVABLECOLLECTION<T>276
 FUNCTIONALITY PROVIDED BY THE COLLECTION<T> CLASS276
 FUNCTIONALITY PROVIDED BY THE INOTIFYCOLLECTIONCHANGED INTERFACE277
 FUNCTIONALITY PROVIDED BY THE INOTIFYPROPERTYCHANGED INTERFACE277
 OBSERVABLECOLLECTION<T> EXAMPLE PROGRAM277
 Quick Review281
BINDINGLIST<T>281
 FUNCTIONALITY PROVIDED BY THE COLLECTION<T> CLASS282
 FUNCTIONALITY PROVIDED BY THE ICANCELADDNEW INTERFACE282
 FUNCTIONALITY PROVIDED BY THE IRAISEITEMCHANGEDEVENTS INTERFACE282
 ONE-WAY DATABINDING EXAMPLE282
 TWO-WAY DATABINDING EXAMPLE289
 Quick Review295
 SUMMARY295
 REFERENCES296
 NOTES296

17 Collections And I/O

INTRODUCTION298
MANIPULATING DIRECTORIES AND FILES298
 FILES, DIRECTORIES, AND PATHS299
 MANIPULATING DIRECTORIES AND FILES299
 VERBATIM STRING LITERALS300
 Quick Review301
SERIALIZING OBJECTS TO DISK301
 SERIALIZABLE ATTRIBUTE301
 SERIALIZING OBJECTS WITH BINARYFORMATTER302
 SERIALIZING OBJECTS WITH XMLSERIALIZER304
 Quick Review306
WORKING WITH TEXT FILES306
 SOME ISSUES YOU MUST CONSIDER306
 SAVING DOG DATA TO A TEXT FILE306
 Quick Review308
WORKING WITH BINARY DATA308
 Quick Review310
RANDOM ACCESS FILE I/O310
 TOWARDS AN APPROACH TO THE ADAPTER PROJECT311
 START SMALL AND TAKE BABY STEPS.....311
 OTHER PROJECT CONSIDERATIONS312
 LOCKING A RECORD FOR UPDATES AND DELETES.....312
 MONITOR. ENTER()/MONITOR.EXIT() VS. THE LOCK KEYWORD.....313
 TRANSLATING LOW-LEVEL EXCEPTIONS INTO HIGHER-LEVEL EXCEPTION ABSTRACTIONS.....313
 WHERE TO GO FROM HERE313
 COMPLETE RANDOMACCESSFILE LEGACY DATAFILE ADAPTER SOURCE CODE LISTING313

Quick Review	325
Working With Log Files	326
Quick Review	328
Using FileDialogs	328
Quick Review	330
PERSISTING A BINDINGLIST<T> COLLECTION	331
SUMMARY	337
REFERENCES	338
NOTES	338

18 CREATING CUSTOM COLLECTIONS

INTRODUCTION	340
DECIDING WHEN TO CREATE A CUSTOM COLLECTION CLASS	340
EXTENDING AN EXISTING COLLECTION	340
EXTENDING NON-GENERIC ARRAYLIST	341
SUPERSEDED BY GENERICS	342
GAINING MORE CONTROL OVER THE CUSTOM COLLECTION	342
Quick Review	347
CREATING A CUSTOM COLLECTION FROM SCRATCH	347
EVALUATING REDBLACKTREE	356
SELECTING THE APPROPRIATE INTERFACE	356
IMPLEMENTING IENUMERABLE AND IENUMERABLE<T>	356
<i>NAMED ITERATORS</i>	356
SERIALIZATION CONSIDERATIONS	357
<i>THE NEED FOR CUSTOM SERIALIZATION</i>	357
RESPONDING TO COLLECTION CHANGING EVENTS	357
EXTENDED EXAMPLE: THE REDBLACKTREE COLLECTION	357
Quick Review	378
CUSTOM COLLECTION IMPLEMENTATION SUMMARY TABLE	378
SUMMARY	379
REFERENCES	379
NOTES	380

19 SPECIALIZED COLLECTIONS

INTRODUCTION	382
SYSTEM.COLLECTIONS.SPECIALIZED NAMESPACE	382
Quick Review	384
NAMEVALUECOLLECTION	384
Quick Review	385
HYBRIDDICTIONARY	386
Quick Review	386
BITVECTOR32	387
Quick Review	388
COLLECTIONCHANGEDEVENTMANAGER	388
Quick Review	392
SUMMARY	392
REFERENCES	393
NOTES	393

20 Collections In Action

INTRODUCTION	396
WHAT YOU ARE GOING TO BUILD	396
PRELIMINARIES	397
INSTALLING SQL SERVER 2008 EXPRESS EDITION	397
INSTALLING MICROSOFT SQL SERVER MANAGEMENT STUDIO EXPRESS	398
INSTALLING MICROSOFT ENTERPRISE LIBRARY	399
A SIMPLE TEST APPLICATION	400
INTRODUCTION TO RELATIONAL DATABASES AND SQL	403
TERMINOLOGY	403
STRUCTURED QUERY LANGUAGE (SQL)	404
DATA DEFINITION LANGUAGE (DDL)	404
CREATING THE EMPLOYEETRAINING DATABASE	405
CREATING A DATABASE WITH A SCRIPT	405
CREATING TABLES	406
SQL SERVER DATABASE TYPES	407
DATA MANIPULATION LANGUAGE (DML)	408
USING THE INSERT COMMAND	409
USING THE SELECT COMMAND	409
USING THE UPDATE COMMAND	412
USING THE DELETE COMMAND	412
Quick Review	412
COMPLEX SQL QUERIES	413
CREATING A RELATED TABLE WITH A FOREIGN KEY	413
INSERTING TEST DATA INTO THE tbl_EMPLOYEE_TRAINING TABLE	414
SELECTING DATA FROM MULTIPLE TABLES	416
JOIN OPERATIONS	416
TESTING THE CASCADE DELETE CONSTRAINT	417
Quick Review	417
THE SERVER APPLICATION	418
PROJECT FOLDER ORGANIZATION	418
USING MICROSOFT BUILD TO MANAGE AND BUILD THE PROJECT	419
FIRST ITERATION	421
CODING THE EMPLOYEEVO AND EMPLOYEEDAO	422
APPLICATION CONFIGURATION FILE	430
CREATING TEST APPLICATION	431
SECOND ITERATION	433
TESTING THE CODE - SECOND ITERATION	446
REALITY CHECK	454
THIRD ITERATION	454
THE CLIENT APPLICATION	459
THIRD ITERATION (CONTINUED)	459
FOURTH ITERATION	461
FIFTH ITERATION	467
SIXTH ITERATION	472
COMPILING AND RUNNING THE MODIFIED EMPLOYEETRAININGCLIENT PROJECT	483
WHERE TO GO FROM HERE	485
SUMMARY	486
REFERENCES	486
NOTES	486

APPENDICES

Appendix A: NUMERIC STRING FORMATTING
NUMERIC FORMATTING489