
Contents

Part 1 Introduction to Object Orientation

1	Introduction to Object Orientation	3
1.1	Introduction	3
1.2	Programming Paradigms	3
1.3	Revolution Versus Evolution	4
1.4	Why Learn a New Programming Paradigm?	5
1.4.1	Software Industry Blues	6
1.4.2	The Advantages Claimed for Object Orientation	7
1.4.3	What Are the Problems and Pitfalls of Object Orientation?	8
1.5	Fundamentals of Object Orientation	11
1.6	The Basic Principles of Object Orientation	12
1.7	Encapsulation	12
1.8	Inheritance	14
1.9	Abstraction	16
1.10	Polymorphism	16
1.10.1	Overloading Operators	17
1.10.2	Overriding Operators	18
1.11	Summary	18
1.12	Further Reading	19
2	Elements of Object Orientation	21
2.1	Introduction	21
2.2	Terminology	21
2.3	Types of Hierarchy	23
2.4	The Move to Object Technology	25
2.5	Summary	26
2.6	Exercises	26
2.7	Further Reading	26

Part 2 Introduction to the C# Language

3	Why Object Orientation?	29
3.1	Introduction	29
3.2	The Procedural Approach	29
3.2.1	A Naked Data Structure	29
3.2.2	Procedures for the Data Structure	30
3.2.3	Packages	30
3.3	Does Object Orientation Do Better?	31
3.3.1	Packages Versus Classes	31
3.3.2	Inheritance	32
3.4	Summary	33
4	Constructing an Object-Oriented System	35
4.1	Introduction	35
4.2	The Application: Windscreen Wipe Simulation	35
4.3	Where Do We Start?	36
4.4	Identifying the Objects	37
4.5	Identifying the Services or Methods	38
4.6	Refining the Objects	39
4.7	Bringing it all Together	39
4.8	Where is the Structure?	42
4.9	Summary	45
4.10	Exercises	45
4.11	Further Reading	45
5	An Introduction to C#	47
5.1	Introduction	47
5.2	Background	47
5.3	What Is C#?	48
5.4	Objects in C#	49
5.5	Commercial Versions of C#	49
5.6	The C# Environment	49
5.7	Comparing C# to Java and C++	50
5.8	C# Keywords	51
5.9	Where to Get More Information	52
6	A Little C#	53
6.1	Introduction	53
6.2	Setting Up the Development Environment	54
6.3	Compiling and Executing C#	54
6.4	Summary	55
7	C# Classes	57
7.1	Introduction	57

7.2	The Basics of the Language	57
7.2.1	Some Terminology	59
7.2.2	The Message-Passing Mechanism	59
7.2.3	The Statement Terminator	59
7.3	Classes	59
7.3.1	Class Definitions	60
7.3.2	Classes and Messages	60
7.3.3	Instances and Instance Variables	61
7.3.4	Classes and Inheritance	61
7.3.5	Instance Creation	63
7.3.6	Constructors	64
7.3.7	Static Constructors Blocks	64
7.3.8	Finalize Methods	65
7.3.9	Supplied Classes	67
7.4	Method Definitions	67
7.4.1	The Comments Section	67
7.4.2	The Local Variables Section	68
7.4.3	The Statements Section	68
7.4.4	The Return Operator	68
7.4.5	An Example Method	69
7.4.6	Static Fields and Methods	69
7.4.7	Constants	72
7.4.8	Read-Only Fields	72
7.4.9	Properties and Indexers	73
8	Structs and Enumerations	75
8.1	Introduction	75
8.2	Structs in C#	75
8.3	Initialization of Structs	77
8.4	Structs and Constructors	77
8.5	Immutable Structs	78
8.6	Enumerations	78
8.7	Enumeration Foundations	79
8.8	Initialization of Enumeration Values	80
8.9	Zero and Enumerations	81
8.10	Bit Flag Enumerations	81
8.11	System Support for Enumerations	82
9	Interfaces	85
9.1	Introduction	85
9.2	Interface Definitions	85
9.3	Interfaces Versus Abstract Classes	86
9.4	Implementing an Interface	86
9.5	Using Interfaces	88
9.6	Multiple Inheritance and Interfaces	89
9.7	Implementing Interfaces and Extending a Superclass	89

9.8	Method Hiding	90
9.9	Interfaces and Structs	92
10	C# Constructs	95
10.1	Introduction	95
10.2	Data Types	95
10.3	Numbers and Numeric Operators	95
10.3.1	Numeric Values	95
10.3.2	Built-In Arithmetic Operators	98
10.3.3	User-Defined Operators	98
10.3.4	Type Operators	99
10.4	Assignments	101
10.5	Compound Assignment	102
10.6	Variables	102
10.6.1	Temporary Variables	102
10.6.2	The <code>this</code> Pseudo-Variable	103
10.6.3	Variable Scope	103
10.6.4	Special Values – <code>true</code> , <code>false</code> and <code>null</code>	104
10.7	Messages and Message Selectors	104
10.7.1	Invoking Methods	104
10.7.2	Precedence	104
10.8	Checked and Unchecked Expressions	105
10.9	Summary	106
11	Characters and Strings	107
11.1	Introduction	107
11.2	Characters	107
11.3	Strings	107
11.4	Converting Objects to Strings	110
11.5	Strings and <code>StringBuilder</code>	110
11.6	Regular Expressions	111
11.7	String Formatting	112
11.8	String Encoding	113
12	An Example C# Class	115
12.1	Introduction	115
12.2	Defining a Class	115
12.2.1	Creating the Class	115
12.3	Defining a Constructors and Methods	116
12.3.1	The <code>Main</code> Method	116
12.3.2	The Constructor	116
12.3.3	The Properties	117
12.3.4	The Birthday Method	118
12.4	Creating an Instance	118

Part 3 C# and Object Orientation

13	Classes, Inheritance and Abstraction	123
13.1	Introduction	123
13.2	Classes Revisited	123
13.2.1	What Are Classes For?	124
13.2.2	Class-Side Methods	125
13.2.3	A Class or an Instance	125
13.3	Inheritance in Classes	127
13.3.1	The Role of a Subclass	127
13.3.2	Capabilities of Classes	128
13.3.3	Overriding Methods	129
13.3.4	Restricting a Subclass (Sealed Classes)	130
13.4	Abstract Classes	130
13.5	Constructors and Their Use	132
13.6	Calling Superclass Constructors	133
13.7	The Main Method	136
14	Encapsulation and Polymorphism	137
14.1	Introduction	137
14.2	Encapsulation	137
14.2.1	Class Modifiers	137
14.2.2	Variable Modifiers	138
14.2.3	Method Modifiers	138
14.3	Namespaces	138
14.3.1	Declaring a Namespace	139
14.3.2	Assemblies	141
14.3.3	Using Namespaces	141
14.3.4	Compiling Using Assemblies	142
14.4	Polymorphism	143
14.4.1	Dynamic or Late Binding	143
14.4.2	Method Selection/Overloading	144
14.4.3	Method Hiding	146
14.4.4	Variable-Length Parameter Lists	147
15	Nested Classes	151
15.1	Introduction	151
15.2	What Are Nested Classes?	151
15.3	How and When Should I Use Nested Classes?	154
15.3.1	As Helper Classes	154
15.3.2	As Event Handlers	154
15.3.3	As a Way of Overcoming Single Inheritance	154
15.3.4	Laying Out a C# Class With Nested Classes	154
15.3.5	Nested Class Guidelines	154

16	Arrays in C#	157
16.1	Introduction	157
16.2	Arrays	157
16.2.1	Arrays of Objects	159
16.2.2	Basic Type Arrays	160
16.2.3	Multi-Dimensional Arrays	160
16.2.4	Jagged/Ragged Arrays	161
16.2.5	Array Conversions	162
16.2.6	System.Array	162
16.2.7	The Main Method	164
16.2.8	Arrays and Methods	165
17	The Collections API	167
17.1	Introduction	167
17.2	Data Structure Classes	167
17.3	What Is in the Collections API?	168
17.4	Collection Interfaces	169
17.5	The core interfaces	170
17.5.1	ICollection	170
17.5.2	IList	171
17.5.3	The IDictionary interface	172
17.6	Comparisons	173
17.7	Abstract Implementations	173
17.7.1	CollectionBase	174
17.7.2	ReadOnlyCollectionBase	175
17.7.3	DictionaryBase	175
17.8	Concrete Implementations	175
17.8.1	ArrayList	176
17.8.2	BitArray	177
17.8.3	The Hashtable class	178
17.8.4	SortedList Collection Class	180
17.8.5	The Queue Class	181
17.8.6	The Stack Class	182
17.9	The System.Collections.Specialized namespace	183
17.10	Enumeration	183
17.11	Iteration over Dictionaries	184
17.12	Summary	185
18	An Object-Oriented Organizer	187
18.1	Introduction	187
18.2	The Organizer Class	187
18.3	The Class Definition	188
18.4	The Updating Protocol	188
18.5	The Accessing Protocol	189
18.6	The Main Method	190

18.7	The Full Listing	191
18.8	Exercise – the Financial Manager Project	193

Part 4 Further C#

19	Control and Iteration	197
19.1	Introduction	197
19.2	Control Structures	197
19.2.1	The If Statement	197
19.2.2	The Conditional Operator	199
19.2.3	The switch Statement	199
19.3	Iteration	201
19.3.1	for Loops	202
19.3.2	while Loops	203
19.3.3	do Loops	204
19.3.4	foreach	204
19.3.5	break and continue	206
19.3.6	The goto Statement	206
19.3.7	An Example of Loops	207
19.4	Recursion	208
19.5	Summary	209
20	Attributes and Versioning	211
20.1	Introduction	211
20.2	Attributes	211
20.2.1	What Are Attributes?	211
20.2.2	Using Attributes	211
20.2.3	Conditional Attributes	212
20.2.4	User-Defined Attributes	212
20.2.5	Attribute Parameters	214
20.3	Versioning	214
21	Delegates	219
21.1	Introduction	219
21.2	Delegates	219
21.3	Delegates in C#	219
21.4	Defining a Delegate	220
21.5	Static and Instance Methods	222
21.6	Multicasting	224
21.7	Delegates as Static Properties	226
21.8	Usage	227
21.9	Summary	227
22	Exception Handling	229
22.1	Introduction	229
22.2	What Is an Exception?	229

22.3	What Is Exception Handling?	231
22.4	Throwing an Exception	232
22.5	Catching an Exception	234
22.6	Defining an Exception	237
22.6.1	Nesting Exceptions	239

Part 5 Graphical User Interfaces

23	Graphical User Interfaces	243
23.1	Introduction	243
23.2	Windows Forms Overview	243
23.3	The Control Class	245
23.4	The Form Class	246
23.5	Building Up a GUI	247
23.6	Using Panels Within Forms	250
23.7	Docking	253
23.8	Anchoring	255
24	Event Handling	257
24.1	Introduction	257
24.2	Event Handling in General	257
24.3	Event Handling in C#	258
24.4	The Inner Workings	259
24.5	An Example	260
25	The JDEdit Application	263
25.1	Introduction	263
25.2	JDEdit	263
25.3	The Structure of JDEdit	264
25.4	The Menu Bar and Menus	266
25.5	The RichTextBox Setup	269
25.6	Monitoring the Application Closure	269
25.7	The Full Listing	271

Part 6 C# Development

26	Streams and Files	277
26.1	Introduction	277
26.2	Files	277
26.3	FileInfo	278
26.4	Directory and DirectoryInfo	280
26.5	Streams	281
26.5.1	What Is a Stream?	281
26.5.2	Using the IO Classes	282
26.6	Summary	288

27	Serialization	289
27.1	Introduction	289
27.2	Binary Serialization	290
27.2.1	Saving Objects	290
27.2.2	Reading Objects	291
27.3	XML Serialization	292
27.3.1	Saving Objects	292
27.3.2	Reading Objects	294
27.4	Controlling Serialization	295
27.5	Custom Serialization	295
27.6	Summary	296
28	Sockets in C#	297
28.1	Introduction	297
28.2	Socket to Socket Communication	297
28.3	Setting up a Connection	297
28.4	An Example Client-Server Application	298
28.4.1	The System Structure	298
28.4.2	Implementing the Server Application	299
28.4.3	Implementing the Client Application	302
29	Data Access	305
29.1	Introduction	305
29.2	OLE DB	305
29.3	Registering a Provider	306
29.4	Opening a Connection	307
29.5	Creating a Table	308
29.6	Obtaining Data From a Database	309
29.6.1	Full Listing	310
29.7	Accessing SQL Server	313
29.8	ODBC	315
29.8.1	What Is ODBC?	315
29.8.2	Using ODBC	316
29.9	Exploiting ADO.NET	318
29.10	Further Reading	319
30	Remoting in .NET	321
30.1	Introduction	321
30.2	Remoting	321
30.2.1	Selecting the Type of Channel	322
30.2.2	Subclassing a Server Class	322
30.2.3	The HelloClient	325
30.2.4	Running the Application	327
30.2.5	What Is Happening?	327
30.3	Remoting Details	329
30.3.1	Remote Objects and Channels	329

30.3.2	Server-Side Registration	329
30.3.3	Default Lifetime	329
30.3.4	Well-Known Objects	330
30.3.5	Client-Activated Objects	330
30.3.6	Programmatic Configuration	331
31	Concurrency	333
31.1	Introduction	333
31.2	Concurrent Processes	333
31.3	Threads	334
31.3.1	Thread States	334
31.3.2	Creating a Thread	334
31.4	The Thread Class	335
31.4.1	Implementing a Thread	338
31.4.2	Asynchronous Method Calls	339
31.4.3	Joining Threads	341
31.4.4	Synchronization	344
31.5	A Producer/Consumer Example	348
32	Using C# in ASP.NET	353
32.1	Introduction	353
32.2	What Is an ASP?	353
32.3	A Very Simple JSP	354
32.4	ASP.NET Tags	357
32.4.1	ASP Directives	357
32.4.2	Code Declaration Blocks	357
32.4.3	Server-Side Object Tag Syntax	357
33	Web-Based User Interfaces	359
33.1	Introduction	359
33.2	Web Forms Overview	359
33.3	The Control Class	361
33.4	The Page Class	362
33.4.1	The System.Web.UI.WebControls Namespace	363
33.4.2	The System.Web.UI.HtmlControls Namespace	363
33.5	Building up a Web Page	363
33.5.1	Basic ASP	364
33.5.2	Using a Page class	366
34	XML and C#	371
34.1	Introduction	371
34.2	XML Introduced	371
34.2.1	What is XML?	371
34.2.2	What Do XML Documents Look Like?	373
34.2.3	XML Vocabularies	374
34.2.4	Working With a DTD	375

34.3	The DOM API	378
34.4	The DOM in C#	379
34.5	Creating an XML Document in C#	380
34.6	Loading an XML Document in C#	382
35	C# Style	387
35.1	Introduction	387
35.2	Code Layout	387
35.3	Variables	388
35.3.1	Naming Variables	388
35.3.2	Using Variables	389
35.4	Classes	391
35.4.1	Naming Classes	391
35.4.2	The Role of a Class	391
35.4.3	Creating New Data Structure Classes	391
35.4.4	Class Comments	392
35.4.5	Using a Class or an Instance	393
35.5	Interfaces	393
35.6	Methods	393
35.6.1	Naming Methods	393
35.6.2	Using Methods	394
35.6.3	Class Methods and Instance Methods	395
35.6.4	Constructors	395
35.6.5	Static Constructors	395
35.6.6	The Destructor Method	396
35.6.7	Programming in Terms of Objects	396
35.6.8	Positioning of Methods	396
35.7	Scoping	396
36	C# Roundup	399
36.1	Introduction	399
36.2	Memory Management	399
36.2.1	Why Have Automatic Memory Management?	399
36.2.2	Memory Management in C#	400
36.2.3	When Is Garbage Collection Performed?	400
36.3	Conversion of Classes	401
36.4	The Main Method	403
36.5	Compiler Directives	404
36.6	Assertions, Debugging and Tracing	405
36.6.1	Debugging and Tracing	405
36.6.2	Assertions	408
36.7	Using Pointers in C#	409
37	The .NET Environment	413
37.1	Introduction	413
37.2	.NET Overview	413

37.3	The .NET Framework	414
37.3.1	Common Language Runtime	414
37.3.2	Framework Class Library	415
37.4	.NET My Services	416
37.5	Language Interoperability	416

Part 7 Object-Oriented Design

38	Object-Oriented Analysis and Design	421
38.1	Introduction	421
38.2	Object-Oriented Design Methods	421
38.3	Object-Oriented Analysis	422
38.4	The Booch Method	422
38.4.1	The Steps in the Booch Method	423
38.4.2	Strengths and Weaknesses	423
38.5	The Object Modeling Technique	423
38.5.1	The Analysis Phase	424
38.5.2	The Design Phase	424
38.5.3	The Implementation Phase	425
38.5.4	Strengths and Weaknesses	425
38.6	The Objectory Method	425
38.6.1	The Requirements Phase	425
38.6.2	The Analysis Phase	426
38.6.3	The Construction Phase	426
38.6.4	Strengths and Weaknesses	426
38.7	The Fusion Method	426
38.8	The Unified Modeling Language	427
38.9	Summary	428
39	The Unified Modeling Language	429
39.1	Introduction	429
39.2	The Meta-Model	430
39.3	The Models	431
39.4	Use Case Diagrams	431
39.5	The Object Model	432
39.5.1	Representing Classes	433
39.5.2	Representing Objects	434
39.5.3	Representing Relationships	435
39.6	Packages	440
39.7	Sequence Diagrams	441
39.8	Collaboration Diagrams	442
39.9	State Machine Diagrams	444
39.9.1	Start Points	444
39.9.2	Events	445
39.9.3	A Set of Transitions	445
39.9.4	A Set of State Variables	446

39.9.5	A Set of States	446
39.9.6	A Set of Exit Points	447
39.10	Deployment Diagrams	447
39.11	Summary	447
40	The Unified Process	449
40.1	Introduction	449
40.2	The Unified Process	449
40.2.1	Overview of the Unified Process	450
40.2.2	Life Cycle Phases	451
40.2.3	Phases, Iterations and Workflows	452
40.2.4	Workflows and Activities	454
40.3	Requirements Workflow	455
40.3.1	Interface Descriptions	455
40.4	Analysis Workflow	455
40.4.1	Analysis Model Classes	456
40.4.2	Constructing the Analysis Model	457
40.5	Design Workflow	460
40.5.1	Identifying Classes	461
40.5.2	Refining the Set of Classes	461
40.5.3	Identifying and Refining Attributes	462
40.5.4	Identifying and Refining Operations	462
40.5.5	Design Use Case Realizations	463
40.5.6	Generating a Sequence Diagram	463
40.5.7	Building a Statechart Diagram	463
40.5.8	Identifying and Refining Associations	464
40.5.9	Identifying Interfaces and Inheritance	464
40.5.10	Remaining Steps	465
40.6	Implementation Workflow	465
40.7	Testing Workflow	465
40.8	Summary	466
	References	467
	Index	471



<http://www.springer.com/978-1-85233-581-6>

Guide to C# and Object Orientation

Hunt, J.

2002, XXVII, 476 p., Hardcover

ISBN: 978-1-85233-581-6