
Contents

Special Thanks	xxi
Preface	xxiii

Part I The C# Language

1 The Creation of C#	3
C#'s Family Tree	3
C: The Beginning of the Modern Age of Programming	3
The Creation of OOP and C++	4
The Internet and Java Emerge	4
The Creation of C#	5
The Evolution of C#	7
How C# Relates to the .NET Framework	7
What Is the .NET Framework?	8
How the Common Language Runtime Works	8
Managed vs. Unmanaged Code	9
The Common Language Specification	9
2 An Overview of C#	11
Object-Oriented Programming	11
Encapsulation	12
Polymorphism	12
Inheritance	13
A First Simple Program	13
Using csc.exe, the C# Command-Line Compiler	14
Using the Visual Studio IDE	15
The First Sample Program, Line by Line	19
Handling Syntax Errors	22
A Small Variation	22
A Second Simple Program	23
Another Data Type	25
Two Control Statements	26
The if Statement	27
The for Loop	28
Using Code Blocks	29
Semicolons, Positioning, and Indentation	31
The C# Keywords	32
Identifiers	33
The .NET Framework Class Library	34

3	Data Types, Literals, and Variables	35
	Why Data Types Are Important	35
	C#'s Value Types	35
	Integers	36
	Floating-Point Types	38
	The decimal Type	40
	Characters	41
	The bool Type	42
	Some Output Options	43
	Literals	46
	Hexadecimal Literals	47
	Character Escape Sequences	47
	String Literals	48
	A Closer Look at Variables	49
	Initializing a Variable	50
	Dynamic Initialization	50
	Implicitly Typed Variables	51
	The Scope and Lifetime of Variables	52
	Type Conversion and Casting	55
	Automatic Conversions	55
	Casting Incompatible Types	56
	Type Conversion in Expressions	59
	Using Casts in Expressions	60
4	Operators	63
	Arithmetic Operators	63
	Increment and Decrement	64
	Relational and Logical Operators	67
	Short-Circuit Logical Operators	70
	The Assignment Operator	71
	Compound Assignments	72
	The Bitwise Operators	73
	The Bitwise AND, OR, XOR, and NOT Operators	73
	The Shift Operators	79
	Bitwise Compound Assignments	81
	The ? Operator	82
	Spacing and Parentheses	83
	Operator Precedence	84
5	Program Control Statements	85
	The if Statement	85
	Nested ifs	86
	The if-else-if Ladder	87
	The switch Statement	88
	Nested switch Statements	92
	The for Loop	92
	Some Variations on the for Loop	94

The while Loop	99
The do-while Loop	101
The foreach Loop	102
Using break to Exit a Loop	102
Using continue	104
return	105
The goto	105
6 Introducing Classes and Objects	109
Class Fundamentals	109
The General Form of a Class	109
Define a Class	110
How Objects Are Created	114
Reference Variables and Assignment	115
Methods	116
Add a Method to the Building Class	117
Return from a Method	119
Return a Value	120
Use Parameters	122
Add a Parameterized Method to Building	125
Avoiding Unreachable Code	126
Constructors	126
Parameterized Constructors	128
Add a Constructor to the Building Class	129
The new Operator Revisited	130
Using new with Value Types	130
Garbage Collection and Destructors	131
Destructors	131
The this Keyword	133
7 Arrays and Strings	137
Arrays	137
One-Dimensional Arrays	137
Multidimensional Arrays	141
Two-Dimensional Arrays	141
Arrays of Three or More Dimensions	142
Initializing Multidimensional Arrays	143
Jagged Arrays	144
Assigning Array References	146
Using the Length Property	148
Using Length with Jagged Arrays	150
Implicitly Typed Arrays	151
The foreach Loop	152
Strings	156
Constructing Strings	156
Operating on Strings	157
Arrays of Strings	160

	Strings Are Immutable	161
	Strings Can Be Used in switch Statements	162
8	A Closer Look at Methods and Classes	165
	Controlling Access to Class Members	165
	C#'s Access Modifiers	165
	Applying Public and Private Access	167
	Controlling Access: A Case Study	168
	Pass References to Methods	172
	How Arguments Are Passed	174
	Use ref and out Parameters	176
	Use ref	177
	Use out	178
	Use ref and out on References	181
	Use a Variable Number of Arguments	182
	Return Objects	185
	Return an Array	187
	Method Overloading	188
	Overload Constructors	194
	Invoke an Overloaded Constructor Through this	197
	Object Initializers	199
	The Main() Method	200
	Return Values from Main()	200
	Pass Arguments to Main()	200
	Recursion	202
	Understanding static	205
	Static Constructors	210
	Static Classes	211
9	Operator Overloading	213
	Operator Overloading Fundamentals	213
	Overloading Binary Operators	214
	Overloading Unary Operators	216
	Handling Operations on C# Built-in Types	220
	Overloading the Relational Operators	224
	Overloading true and false	226
	Overloading the Logical Operators	229
	A Simple Approach to Overloading the Logical Operators	229
	Enabling the Short-Circuit Operators	231
	Conversion Operators	235
	Operator Overloading Tips and Restrictions	239
	Another Example of Operator Overloading	240
10	Indexers and Properties	245
	Indexers	245
	Creating One-Dimensional Indexers	245
	Indexers Can Be Overloaded	249

Indexers Do Not Require an Underlying Array	251
Multidimensional Indexers	252
Properties	254
Auto-Implemented Properties	259
Use Object Initializers with Properties	260
Property Restrictions	261
Use Access Modifiers with Accessors	261
Using Indexers and Properties	264
11 Inheritance	269
Inheritance Basics	269
Member Access and Inheritance	272
Using Protected Access	275
Constructors and Inheritance	276
Calling Base Class Constructors	278
Inheritance and Name Hiding	282
Using base to Access a Hidden Name	283
Creating a Multilevel Hierarchy	285
When Are Constructors Called?	288
Base Class References and Derived Objects	289
Virtual Methods and Overriding	294
Why Overridden Methods?	297
Applying Virtual Methods	298
Using Abstract Classes	301
Using sealed to Prevent Inheritance	305
The object Class	305
Boxing and Unboxing	307
Is object a Universal Data Type?	309
12 Interfaces, Structures, and Enumerations	311
Interfaces	311
Implementing Interfaces	312
Using Interface References	316
Interface Properties	318
Interface Indexers	320
Interfaces Can Be Inherited	322
Name Hiding with Interface Inheritance	323
Explicit Implementations	323
Choosing Between an Interface and an Abstract Class	326
The .NET Standard Interfaces	326
Structures	326
Why Structures?	330
Enumerations	332
Initialize an Enumeration	333
Specify the Underlying Type of an Enumeration	334
Use Enumerations	334

13	Exception Handling	337
	The System.Exception Class	337
	Exception Handling Fundamentals	338
	Using try and catch	338
	A Simple Exception Example	338
	A Second Exception Example	340
	The Consequences of an Uncaught Exception	341
	Exceptions Let You Handle Errors Gracefully	343
	Using Multiple catch Clauses	344
	Catching All Exceptions	345
	Nesting try Blocks	346
	Throwing an Exception	347
	Rethrowing an Exception	348
	Using finally	349
	A Closer Look at the Exception Class	351
	Commonly Used Exceptions	352
	Deriving Exception Classes	354
	Catching Derived Class Exceptions	358
	Using checked and unchecked	360
14	Using I/O	363
	C#'s I/O Is Built Upon Streams	363
	Byte Streams and Character Streams	363
	The Predefined Streams	363
	The Stream Classes	364
	The Stream Class	364
	The Byte Stream Classes	365
	The Character Stream Wrapper Classes	365
	Binary Streams	367
	Console I/O	367
	Reading Console Input	367
	Using ReadKey()	369
	Writing Console Output	371
	FileStream and Byte-Oriented File I/O	371
	Opening and Closing a File	372
	Reading Bytes from a FileStream	374
	Writing to a File	375
	Using FileStream to Copy a File	376
	Character-Based File I/O	378
	Using StreamWriter	378
	Using a StreamReader	380
	Redirecting the Standard Streams	381
	Reading and Writing Binary Data	383
	BinaryWriter	383
	BinaryReader	384
	Demonstrating Binary I/O	386

Random Access Files	390
Using MemoryStream	392
Using StringReader and StringWriter	394
Converting Numeric Strings to Their Internal Representation	396
15 Delegates, Events, and Lambda Expressions	399
Delegates	399
Delegate Method Group Conversion	402
Using Instance Methods as Delegates	402
Multicasting	404
Covariance and Contravariance	406
System.Delegate	408
Why Delegates	408
Anonymous Functions	408
Anonymous Methods	409
Pass Arguments to an Anonymous Method	410
Return a Value from an Anonymous Method	410
Use Outer Variables with Anonymous Methods	412
Lambda Expressions	413
The Lambda Operator	413
Expression Lambdas	414
Statement Lambdas	416
Events	419
A Multicast Event Example	421
Instance Methods vs. Static Methods as Event Handlers	422
Using Event Accessors	424
Miscellaneous Event Features	429
Use Anonymous Methods and Lambda Expressions with Events	429
.NET Event Guidelines	430
Use EventHandler	432
Applying Events: A Case Study	433
16 Namespaces, the Preprocessor, and Assemblies	437
Namespaces	437
Declaring a Namespace	438
Namespaces Prevent Name Conflicts	440
using	441
A Second Form of using	443
Namespaces Are Additive	444
Namespaces Can Be Nested	446
The Global Namespace	447
Using the :: Namespace Alias Qualifier	447
The Preprocessor	451
#define	452
#if and #endif	452
#else and #elif	454
#undef	455

#error	456
#warning	456
#line	456
#region and #endregion	456
#pragma	457
Assemblies and the internal Access Modifier	457
The internal Access Modifier	458
17 Runtime Type ID, Reflection, and Attributes	459
Runtime Type Identification	459
Testing a Type with is	459
Using as	460
Using typeof	462
Reflection	463
The Reflection Core: System.Type	463
Using Reflection	465
Obtaining Information About Methods	465
Calling Methods Using Reflection	469
Obtaining a Type's Constructors	471
Obtaining Types from Assemblies	475
Fully Automating Type Discovery	481
Attributes	483
Attribute Basics	483
Positional vs. Named Parameters	487
Three Built-in Attributes	491
AttributeUsage	491
The Conditional Attribute	491
The Obsolete Attribute	493
18 Generics	495
What Are Generics?	495
A Simple Generics Example	496
Generic Types Differ Based on Their Type Arguments	499
How Generics Improve Type Safety	499
A Generic Class with Two Type Parameters	502
The General Form of a Generic Class	503
Constrained Types	503
Using a Base Class Constraint	504
Using an Interface Constraint	512
Using the new() Constructor Constraint	516
The Reference Type and Value Type Constraints	517
Using a Constraint to Establish a Relationship Between Two Type Parameters	520
Using Multiple Constraints	521
Creating a Default Value of a Type Parameter	522
Generic Structures	523
Creating a Generic Method	524

	Using Explicit Type Arguments to Call a Generic Method	527
	Using a Constraint with a Generic Method	527
	Generic Delegates	527
	Generic Interfaces	530
	Comparing Instances of a Type Parameter	534
	Generic Class Hierarchies	537
	Using a Generic Base Class	537
	A Generic Derived Class	539
	Overriding Virtual Methods in a Generic Class	540
	Overloading Methods That Use Type Parameters	542
	How Generic Types Are Instantiated	543
	Some Generic Restrictions	544
	Final Thoughts on Generics	544
19	LINQ	545
	What Is LINQ?	545
	LINQ Fundamentals	546
	A Simple Query	546
	A Query Can Be Executed More Than Once	548
	How the Data Types in a Query Relate	549
	The General Form of a Query	550
	Filter Values with where	551
	Sort Results with orderby	552
	A Closer Look at select	556
	Use Nested from Clauses	560
	Group Results with group	561
	Use into to Create a Continuation	563
	Use let to Create a Variable in a Query	565
	Join Two Sequences with join	566
	Anonymous Types	569
	Create a Group Join	571
	The Query Methods	574
	The Basic Query Methods	574
	Create Queries by Using the Query Methods	575
	Query Syntax vs. Query Methods	577
	More Query-Related Extension Methods	577
	Deferred vs. Immediate Query Execution	580
	Expression Trees	581
	Extension Methods	582
20	Unsafe Code, Pointers, Nullable Types, and Miscellaneous Topics	585
	Unsafe Code	585
	Pointer Basics	586
	Using unsafe	587
	Using fixed	588
	Accessing Structure Members Through a Pointer	589

Pointer Arithmetic	589
Pointer Comparisons	591
Pointers and Arrays	591
Pointers and Strings	593
Multiple Indirection	594
Arrays of Pointers	595
stackalloc	596
Creating Fixed-Size Buffers	596
Nullable Types	598
Nullable Basics	598
Nullable Objects in Expressions	600
The ?? Operator	601
Nullable Objects and the Relational and Logical Operators	602
Partial Types	603
Partial Methods	604
Friend Assemblies	605
Miscellaneous Keywords	605
lock	605
readonly	606
const and volatile	607
The using Statement	607
extern	608

Part II Exploring the C# Library

21 Exploring the System Namespace	615
The Members of System	615
The Math Class	617
The .NET Structures Corresponding to the Built-in Value Types	623
The Integer Structures	623
The Floating-Point Structures	626
Decimal	630
Char	634
The Boolean Structure	640
The Array Class	641
Sorting and Searching Arrays	648
Reversing an Array	650
Copying an Array	651
Using a Predicate	652
Using an Action	653
BitConverter	654
Generating Random Numbers with Random	656
Memory Management and the GC Class	657
Object	659
The IComparable and IComparable<T> Interfaces	659
The IEquatable<T> Interface	660

The IConvertible Interface	660
The ICloneable Interface	660
IFormatProvider and IFormattable	662
22 Strings and Formatting	663
Strings in C#	663
The String Class	664
The String Constructors	664
The String Field, Indexer, and Property	665
The String Operators	665
The String Methods	665
Padding and Trimming Strings	681
Inserting, Removing, and Replacing	682
Changing Case	683
Using the Substring() Method	684
The String Extension Methods	684
Formatting	684
Formatting Overview	685
The Numeric Format Specifiers	686
Understanding Argument Numbers	687
Using String.Format() and ToString() to Format Data	688
Using String.Format() to Format Values	688
Using ToString() to Format Data	691
Creating a Custom Numeric Format	692
The Custom Format Placeholder Characters	692
Formatting Date and Time	695
Creating a Custom Date and Time Format	698
Formatting Enumerations	700
23 Multithreaded Programming	703
Multithreading Fundamentals	703
The Thread Class	704
Creating and Starting a Thread	704
Some Simple Improvements	707
Creating Multiple Threads	708
Determining When a Thread Ends	710
Passing an Argument to a Thread	713
The IsBackground Property	715
Thread Priorities	715
Synchronization	717
An Alternative Approach	721
The Monitor Class and lock	723
Thread Communication Using Wait(), Pulse(), and PulseAll()	723
An Example That Uses Wait() and Pulse()	724
Deadlock and Race Conditions	727
UsingMethodImplAttribute	728
Using a Mutex and a Semaphore	730

The Mutex	730
The Semaphore	734
Using Events	737
The Interlocked Class	739
Terminating a Thread	741
An Abort() Alternative	742
Canceling Abort()	743
Suspending and Resuming a Thread	745
Determining a Thread's State	745
Using the Main Thread	746
Multithreading Tips	747
Starting a Separate Task	747
24 Collections, Enumerators, and Iterators	749
Collections Overview	749
The Non-Generic Collections	750
The Non-Generic Interfaces	751
The DictionaryEntry Structure	755
The Non-Generic Collection Classes	755
Storing Bits with BitArray	771
The Specialized Collections	774
The Generic Collections	774
The Generic Interfaces	775
The KeyValuePair<TK, TV> Structure	778
The Generic Collection Classes	779
Storing User-Defined Classes in Collections	799
Implementing IComparable	801
Implementing IComparable for Non-Generic Collections	802
Implementing IComparable<T> for Generic Collections	803
Using an IComparer	805
Using a Non-Generic IComparer	805
Using a Generic IComparer<T>	806
Accessing a Collection via an Enumerator	808
Using an Enumerator	808
Using the IDictionaryEnumerator	809
Implementing IEnumerable and IEnumerator	811
Using Iterators	813
Stopping an Iterator	815
Using Multiple yield Directives	815
Creating a Named Iterator	816
Creating a Generic Iterator	818
Collection Initializers	819
25 Networking Through the Internet Using System.Net	821
The System.Net Members	821
Uniform Resource Identifiers	823

Internet Access Fundamentals	823
WebRequest	824
WebResponse	826
HttpRequest and HttpResponse	826
A Simple First Example	827
Handling Network Errors	830
Exceptions Generated by Create()	830
Exceptions Generated by GetReponse()	830
Exceptions Generated by GetResponseStream()	831
Using Exception Handling	831
The Uri Class	833
Accessing Additional HTTP Response Information	834
Accessing the Header	834
Accessing Cookies	836
Using the LastModified Property	838
MiniCrawler: A Case Study	839
Using WebClient	842
26 Use System.Windows.Forms to Create Form-Based	
Windows Applications	847
A Brief History of Windows Programming	847
Two Ways to Write a Form-Based Windows Application	848
How Windows Interacts with the User	848
Windows Forms	849
The Form Class	849
A Skeletal Form-Based Windows Program	849
Compiling the Windows Skeleton	851
Adding a Button	852
Button Basics	852
Adding a Button to a Form	852
A Simple Button Example	853
Handling Messages	853
An Alternative Implementation	856
Using a Message Box	856
Adding a Menu	859
Creating a Traditional-Style Main Menu	859
Creating a New-Style Menu with MenuStrip	863
A Documentation Comment Quick Reference	867
The XML Comment Tags	867
Compiling Documentation Comments	868
An XML Documentation Example	869
Index	871