

Contents at a Glance

<i>Introduction</i>	1
<i>Part I: Creating Your First C# Programs</i>	9
Chapter 1: Creating Your First C# Windows Program	11
Chapter 2: Creating Your First C# Console Application	29
<i>Part II: Basic C# Programming</i>	37
Chapter 3: Living with Variability — Declaring Value-Type Variables	39
Chapter 4: Smooth Operators	57
Chapter 5: Controlling Program Flow	71
<i>Part III: Object-Based Programming</i>	99
Chapter 6: Collecting Data — The Class and the Array	101
Chapter 7: Putting on Some High-Class Functions	127
Chapter 8: Class Methods	163
Chapter 9: Stringing in the Key of C#	187
<i>Part IV: Object-Oriented Programming</i>	211
Chapter 10: Object-Oriented Programming — What's It All About?	213
Chapter 11: Holding a Class Responsible	221
Chapter 12: Inheritance — Is That All I Get?	251
Chapter 13: Poly-what-ism?	273
<i>Part V: Beyond Basic Classes</i>	301
Chapter 14: When a Class Isn't a Class — The Interface and the Structure	303
Chapter 15: Asking Your Pharmacist about Generics	333
<i>Part VI: The Part of Tens</i>	365
Chapter 16: The 10 Most Common Build Errors (And How to Fix Them)	367
Chapter 17: The 10 Most Significant Differences between C# and C++	379
<i>Appendix: About the CD</i>	385

<i>Bonus Chapters on the CD-ROM!</i>	<i>CD</i>
Bonus Chapter 1: Some Exceptional Exceptions	CD1
Bonus Chapter 2: Handling Files and Libraries in C#	CD27
Bonus Chapter 3: Stepping through Collections	CD55
Bonus Chapter 4: Using the Visual Studio Interface	CD99
Bonus Chapter 5: C# on the Cheap	CD139
<i>Index</i>	<i>391</i>
<i>End-User License Agreement</i>	<i>Back of Book</i>

Table of Contents

<i>Introduction</i>	1
What's New in C# 2.0	2
About This Book	3
What You Need to Use the Book	3
How to Use This Book	4
How This Book Is Organized	4
Part I: Creating Your First C# Programs	4
Part II: Basic C# Programming	4
Part III: Object-Based Programming	5
Part IV: Object-Oriented Programming	5
Part V: Beyond Basic Classes	5
Part VI: The Part of Tens	5
About the CD-ROM	6
Icons Used in This Book	6
Conventions Used in This Book	7
Where to Go from Here	7
<i>Part I: Creating Your First C# Programs</i>	9
Chapter 1: Creating Your First C# Windows Program	11
Getting a Handle on Computer Languages, C#, and .NET	11
What's a program?	12
What's C#?	12
What's .NET?	13
What is Visual Studio 2005? What about Visual C#?	14
Creating a Windows Application with C#	15
Creating the template	15
Building and running your first Windows Forms program	18
Painting pretty pictures	20
Make it do something, Daddy	25
Trying out the final product	27
Visual Basic 6.0 programmers, beware!	28
Chapter 2: Creating Your First C# Console Application	29
Creating a Console Application Template	29
Creating the source program	30
Taking it out for a test drive	31
Creating Your First Real Console App	32
Reviewing the Console Application Template	33
The program framework	33
Comments	34
The meat of the program	34



Part II: Basic C# Programming37

**Chapter 3: Living with Variability —
Declaring Value-Type Variables39**

- Declaring a Variable40
- What's an int?40
 - Rules for declaring variables42
 - Variations on a theme — different types of int42
- Representing Fractions43
- Handling Floating Point Variables44
 - Declaring a floating point variable45
 - Converting some more temperatures46
 - Examining some limitations of floating point variables46
- Using the Decimal Type — A Combination of Integers and Floats48
 - Declaring a decimal48
 - Comparing decimals, integers, and floating point types49
- Examining the bool Type — Is It Logical?49
- Checking Out Character Types50
 - Char variable type50
 - Special char types50
 - The string type51
- What's a Value-Type?52
- Comparing string and char53
- Declaring Numeric Constants54
- Changing Types — The Cast55

Chapter 4: Smooth Operators57

- Performing Arithmetic57
 - Simple operators57
 - Operating orders58
 - The assignment operator60
 - The increment operator61
- Performing Logical Comparisons — Is That Logical?62
 - Comparing floating point numbers:
 - Is your float bigger than mine?63
 - Compounding the confusion with
 - compound logical operations64
- Finding the Perfect Date — Matching Expression Types66
 - Calculating the type of an operation67
 - Assigning types68
- The Ternary Operator — I Wish It Were a Bird and Would Fly Away69

Chapter 5: Controlling Program Flow71

- Controlling Program Flow72
 - Introducing the if statement73
 - Examining the else statement75
 - Avoiding even the else76
 - Embedded if statements77

Looping Commands	80
Introducing the while loop	80
Using the do...while loop	84
Breaking up is easy to do	84
Looping until you get it right	86
Focusing on scope rules	89
Understanding the Most Common Control: The for Loop	90
An example	91
Why do you need another loop?	91
Nested Loops	92
The switch Control	96
The Lowly goto Statement	98

***Part III: Object-Based Programming* 99**

Chapter 6: Collecting Data — The Class and the Array 101

Showing Some Class	102
Defining a class	102
What's the object?	103
Accessing the members of an object	104
Can you give me references?	107
Classes that contain classes are the happiest classes in the world	108
Generating static in class members	110
Defining const data members	111
The C# Array	111
The argument for the array	112
The fixed-value array	112
The variable-length array	114
Lining Up Arrays of Objects	118
A Flow Control Made foreach Array	120
Sorting through Arrays of Objects	122

Chapter 7: Putting on Some High-Class Functions 127

Defining and Using a Function	127
An Example Function for Your Files	129
Having Arguments with Functions	135
Passing an argument to a function	136
Passing multiple arguments to functions	136
Matching argument definitions with usage	138
Overloading a function does not mean giving it too much to do	139
Implementing default arguments	140
Passing value-type arguments	142
Returning Values after Christmas	147
Returning a value via return postage	147
Returning a value using pass by reference	148
When do I return and when do I out?	149
Defining a function with no value	152

The Main() Deal — Passing Arguments to a Program	153
Passing arguments from a DOS prompt	155
Passing arguments from a window	157
Passing arguments from Visual Studio 2005	159
Chapter 8: Class Methods	163
Passing an Object to a Function	163
Defining Object Functions and Methods	165
Defining a static member function	165
Defining a method	167
Expanding a method's full name	168
Accessing the Current Object	169
What is the this keyword?	171
When is this explicit?	172
What happens when I don't have this?	174
Getting Help from Visual Studio — Auto-Complete	176
Getting help on built-in functions from the System Library	177
Getting help with your own functions and methods	179
Adding to the help	180
Generating XML documentation	185
Chapter 9: Stringing in the Key of C#	187
Performing Common Operations on a String	188
The union is indivisible, and so are strings	188
Equality for all strings: The Compare() method	189
Would you like your compares with or without case?	193
What if I want to switch case?	193
Reading character input	194
Parsing numeric input	196
Handling a series of numbers	198
Controlling Output Manually	200
Using the Trim() and Pad() methods	201
Using the Concatenate function	203
Let's Split() that concatenate program	205
Controlling String.Format()	206
Part IV: Object-Oriented Programming	211
Chapter 10: Object-Oriented Programming —	
What's It All About?	213
Object-Oriented Concept #1 — Abstraction	213
Preparing functional nachos	214
Preparing object-oriented nachos	215
Object-Oriented Concept #2 — Classification	215
Why Classify?	216
Object-Oriented Concept #3 — Usable Interfaces	217
Object-Oriented Concept #4 — Access Control	218
How Does C# Support Object-Oriented Concepts?	219

Chapter 11: Holding a Class Responsible	221
Restricting Access to Class Members	221
A public example of public BankAccount	222
Jumping ahead — other levels of security	224
Why Worry about Access Control?	225
Accessor methods	226
Access control to the rescue — an example	227
So what?	230
Defining class properties	231
Getting Your Objects Off to a Good Start — Constructors	233
The C#-Provided Constructor	233
The Default Constructor	235
Constructing something	236
Executing the constructor from the debugger	238
Initializing an object directly — the default constructor	241
Seeing that construction stuff with initializers	242
Overloading the Constructor (Is That Like Overtaxing a Carpenter?)	243
Avoiding Duplication among Constructors	245
Being Object Stingy	249
Chapter 12: Inheritance — Is That All I Get?	251
Inheriting a Class	252
Why Do You Need Inheritance?	253
A More Involved Example — Inheriting from a BankAccount Class	254
IS_A versus HAS_A — I'm So Confused	257
The IS_A relationship	257
Gaining access to BankAccount through containment	258
The HAS_A relationship	259
When to IS_A and When to HAS_A?	260
Other Features That Support Inheritance	261
Changing class	261
Invalid casts at run time	262
Avoiding invalid conversions using the <code>is</code> and <code>as</code> keywords	263
Inheritance and the Constructor	265
Invoking the default base class constructor	265
Passing arguments to the base class constructor — <code>mama sing base</code>	266
The Updated BankAccount Class	269
The Destructor	271
Chapter 13: Poly-what-ism?	273
Overloading an Inherited Method	274
It's a simple case of function overloading	274
Different class, different method	275
Peek-a-boo — hiding a base class method	275
Calling back to base	280

Polymorphism	282
What's wrong with using the declared type every time?	283
Using "is" to access a hidden method polymorphically	285
Declaring a method virtual	286
C# During Its Abstract Period	288
Class factoring	288
I'm left with nothing but a concept — the abstract class	293
How do you use an abstract class?	294
Creating an abstract object — not!	296
Restarting a Class Hierarchy	296
Sealing a Class	300

***Part V: Beyond Basic Classes*301**

Chapter 14: When a Class Isn't a Class — The Interface and the Structure303

What Is CAN_BE_USED_AS?	303
What Is an Interface?	305
Can I Get a Short Example?	306
Can I See a Program That CAN_BE_USED_AS an Example?	307
Creating your own interface at home in your spare time	308
Predefined interfaces	309
Putting it all together	311
Inheriting an Interface	316
Facing an Abstract Interface	316
The C# Structure Has No Class	319
The C# structure	320
The structure constructor	322
The wily methods of a structure	323
Putting a struct through its paces in an example	323
“Oh, the Value and the Reference Can Be Friends . . .” —	
Unifying the Type System	327
Predefined structure types	327
So, how do common structures unify the type system?	
An example	328
Boxing and unboxing value types	330

Chapter 15: Asking Your Pharmacist about Generics333

Getting to Know Nongeneric Collections	334
Inventorying nongeneric collections	334
Using nongeneric collections	335
Writing a New Prescription: Generics	336
Generics are type-safe	336
Generics are efficient	337
Using Generic Collections	338
Figuring out <T>	338
Using List<T>	338

Classy Generics: Writing Your Own	340
Shipping packages at OOPs	341
Queuing at OOPs: PriorityQueue	341
Unwrapping the package	345
Touring Main()	347
Writing generic code the easy way	348
Saving PriorityQueue for last	349
Tending to unfinished business	351
Generically Methodical	353
Generic methods in nongeneric classes	355
Generic methods in generic classes	356
You may need to constrain a generic method, too	356
Up Against the (Generic) Interface	357
Nongeneric vs. generic interfaces	357
Using a (nongeneric) Simple Factory class	358
Building a generic factory	359

***Part VI: The Part of Tens*365**

**Chapter 16: The 10 Most Common Build Errors
(And How to Fix Them)367**

The name 'memberName' does not exist in the class or namespace 'className'	368
Cannot implicitly convert type 'x' into 'y'	369
'className.memberName' is inaccessible due to its protection level	371
Use of unassigned local variable 'n'	372
Unable to copy the file 'programName.exe' to 'programName.exe'. The process cannot	373
'subclassName.methodName' hides inherited member 'baseclassName.methodName'. Use the new keyword if hiding was intended	374
'subclassName' : cannot inherit from sealed class 'baseclassName'	375
'className' does not implement interface member 'methodName'	376
'methodName' : not all code paths return a value	376
} expected	377

**Chapter 17: The 10 Most Significant Differences
between C# and C++379**

No Global Data or Functions	380
All Objects Are Allocated Off the Heap	380
Pointer Variables Are All but Disallowed	381
C# Generics Are Like C++ Templates — or Are They?	381
I'll Never Include a File Again	382
Don't Construct — Initialize	382

Define Your Variable Types Well	383
No Multiple Inheriting	383
Projecting a Good Interface	383
Unified Type System	384

Appendix: About the CD **385**

System Requirements	385
Using the CD	386
What You'll Find on the CD	387
The C# programs	387
Five bonus chapters	388
NUnit	389
SharpDevelop	389
TextPad	389
Troubleshooting	389

Bonus Chapters on the CD-ROM! **CD**

Bonus Chapter 1: Some Exceptional Exceptions **CD1**

Handling an Error the Old-Fashioned Way — (Re)Turn It	CD1
Returning an error indication	CD4
I'm here to report, that seems fine to me	CD7
Using an Exceptional Error-Reporting Mechanism	CD9
Can I Get an Example?	CD10
Creating Your Own Exception Class	CD13
Assigning Multiple Catch Blocks	CD15
Letting some throws slip through your fingers	CD17
Rethrowing an object	CD20
Thinking through how you should respond to an exception	CD21
Overriding the Exception Class	CD22

Bonus Chapter 2: Handling Files and Libraries in C# **CD27**

Dividing a Single Program into Multiple Source Files	CD28
Dividing a Single Program into Multiple Assemblies	CD29
Collecting Source Files into Namespaces	CD30
Declaring a namespace	CD31
Seeing the importance of namespaces	CD32
Accessing classes in the same namespace	
with fully qualified names	CD34
Using a namespace	CD35
How about using a fully qualified example?	CD36
Collecting Classes into Class Libraries	CD39
Creating a class library project	CD39
Creating classes for the library	CD40
Creating a "driver" project	CD41

Collecting Data into Files	CD43
Using StreamWriter	CD45
Improving your reading speed and comprehension through StreamReader	CD50

Bonus Chapter 3: Stepping through CollectionsCD55

Iterating through a Directory of Files	CD55
Writing Your Own Collection Class: The Linked List	CD62
An example linked-list container	CD63
Why bother with a linked list?	CD73
Iterating foreach Collections: Iterators	CD73
Accessing a collection: The general problem	CD74
Letting C# access data foreach container	CD76
Accessing Collections the Array Way: Indexers	CD77
Indexer format	CD78
Example indexer program	CD78
Looping around the Iterator Block	CD82
Iterating days of the month: A first example	CD87
What's a collection, really?	CD89
Iterator syntax gives up so easily	CD90
Iterator blocks of all shapes and sizes	CD92
Where to put your iterator	CD95

Bonus Chapter 4: Using the Visual Studio InterfaceCD99

Customizing the Window Layout	CD100
Examining the window display states	CD100
Hiding a window	CD103
Rearranging windows	CD103
Stacking windows	CD105
More cool nifties, 'er, "productivity tools"	CD106
Stirring the Solution Explorer	CD107
Simplifying life with projects and solutions	CD107
Displaying the project	CD108
Multisourcing your way to success: Adding a class	CD111
Completing the example classes	CD112
Converting classes into a program	CD115
Considering What Code Should Look Like	CD116
Getting Help — Quickly!	CD120
F1 Help	CD120
Index Help	CD122
Search Help	CD124
More Help goodies	CD125
"Auto list members" Help	CD126
"De"-Debugging Windows	CD127
Your program has bugs: It's time to call the exterminator!	CD128
Learning the single-step dance	CD130
Let me break my point	CD134
Operator, trace that call stack!	CD137
"It's soup"	CD138

Bonus Chapter 5: C# on the CheapCD139

Working Without a Net — But Not a .NET	CD140
Grabbing the free ingredients	CD141
Going around the C# development cycle	CD142
Doing C# with SharpDevelop	CD142
Examining SharpDevelop	CD143
Comparing SharpDevelop features with Visual Studio	CD144
Getting help	CD145
Configuring SharpDevelop	CD146
Adding a tool to launch the debugger	CD146
Running the debugger from SharpDevelop	CD147
Missing debugger stuff	CD150
Doing C# with TextPad	CD150
Creating a C# .CS document class	CD153
Adding a tool of your own: Build C# Debug	CD155
Configuring a tool to do a Release build	CD157
Explaining the configuration options for the Debug and Release tools	CD158
Dealing with compiler errors	CD162
Configuring the rest of the tools	CD162
Testing It with NUnit	CD165
Running NUnit	CD166
Testing? I have to do testing?	CD166
Writing NUnit tests	CD168
Debugging bugs in your test code	CD175
Writing Windows Forms Code without a Form Designer	CD177
It's just code	CD177
Doing it the designer's way	CD178
Understanding the new partial classes	CD179
Doing it your own way	CD180
Making Sure Your Users Can Run Your C# Programs	CD180
A Poor Coder's Visual Studio	CD180

Index.....391***End-User License Agreement.....Back of Book***