

Join the discussion @ [p2p.wrox.com](http://p2p.wrox.com)



Wrox Programmer to Programmer™



# Functional Programming in C#

*Classic Programming Techniques for Modern Projects*

Oliver Sturm



# CONTENTS

## INTRODUCTION

xiii

### PART I: INTRODUCTION TO FUNCTIONAL PROGRAMMING

#### CHAPTER 1: A LOOK AT FUNCTIONAL PROGRAMMING HISTORY 3

What Is Functional Programming?	3
Functional Languages	5
The Relationship to Object Oriented Programming	7
Summary	8

#### CHAPTER 2: PUTTING FUNCTIONAL PROGRAMMING INTO A MODERN CONTEXT 9

Managing Side Effects	10
Agile Programming Methodologies	11
Declarative Programming	11
Functional Programming Is a Mindset	12
Is Functional Programming in C# a Good Idea?	13
Summary	13

### PART II: C# FOUNDATIONS OF FUNCTIONAL PROGRAMMING

#### CHAPTER 3: FUNCTIONS, DELEGATES, AND LAMBDA EXPRESSIONS 17

Functions and Methods	17
Reusing Functions	19
Anonymous Functions and Lambda Expressions	23
Extension Methods	26
Referential Transparency	28
Summary	30

#### CHAPTER 4: FLEXIBLE TYPING WITH GENERICS 31

Generic Functions	32
Generic Classes	34
Constraining Types	35
Other Generic Types	37

Covariance and Contravariance	38
Summary	41
<b>CHAPTER 5: LAZY LISTING WITH ITERATORS</b>	<b>43</b>
The Meaning of Laziness	43
Enumerating Things with .NET	44
Implementing Iterator Functions	47
Returning IEnumerator	50
Chaining Iterators	51
Summary	53
<b>CHAPTER 6: ENCAPSULATING DATA IN CLOSURES</b>	<b>55</b>
Constructing Functions Dynamically	55
The Problem with Scope	56
How Closures Work	57
Summary	60
<b>CHAPTER 7: CODE IS DATA</b>	<b>61</b>
Expression Trees in .NET	63
Analyzing Expressions	64
Generating Expressions	68
.NET 4.0 Specifics	72
Summary	74
<b>PART III: IMPLEMENTING WELL-KNOWN FUNCTIONAL TECHNIQUES IN C#</b>	
<b>CHAPTER 8: CURRYING AND PARTIAL APPLICATION</b>	<b>77</b>
Decoupling Parameters	77
Manual Currying	78
Automatic Currying	80
Calling Curried Functions	82
The Class Context	82
What FCSlib Contains	85
Calling Parts of Functions	86
Why Parameter Order Matters	88
Summary	89

<b>CHAPTER 9: LAZY EVALUATION</b>	<b>91</b>
What's Good about Being Lazy?	92
Passing Functions	93
Explicit Lazy Evaluation	95
Comparing the Lazy Evaluation Techniques	98
Usability	98
Efficiency	98
How Lazy Can You Be?	99
Summary	100
<b>CHAPTER 10: CACHING TECHNIQUES</b>	<b>101</b>
The Need to Remember	101
Precomputation	102
Memoization	107
Deep Memoization	110
Considerations on Memoization	114
Summary	115
<b>CHAPTER 11: CALLING YOURSELF</b>	<b>117</b>
Recursion in C#	117
Tail Recursion	119
Accumulator Passing Style	121
Continuation Passing Style	122
Indirect Recursion	126
Summary	129
<b>CHAPTER 12: STANDARD HIGHER ORDER FUNCTIONS</b>	<b>131</b>
Applying Operations: Map	132
Using Criteria: Filter	132
Accumulating: Fold	133
Map, Filter, and Fold in LINQ	138
Standard Higher Order Functions	140
Summary	140

<b>CHAPTER 13: SEQUENCES</b>	<b>141</b>
Understanding List Comprehensions	141
A Functional Approach to Iterators	142
Ranges	143
Restrictions	146
Summary	147
<b>CHAPTER 14: CONSTRUCTING FUNCTIONS FROM FUNCTIONS</b>	<b>149</b>
Composing Functions	149
Advanced Partial Application	152
Combining Approaches	155
Summary	158
<b>CHAPTER 15: OPTIONAL VALUES</b>	<b>159</b>
The Meaning of Nothing	159
Implementing Option(al) Values	160
Summary	165
<b>CHAPTER 16: KEEPING DATA FROM CHANGING</b>	<b>167</b>
Change Is Good — Not!	167
False Assumptions	168
Being Static Is Good	169
A Matter of Depth	170
Cloning	171
Automatic Cloning	173
Implementing Immutable Container Data Structures	177
Linked List	177
Queue	183
Unbalanced Binary Tree	185
Red/Black Tree	187
Alternatives to Persistent Data Types	190
Summary	191
<b>CHAPTER 17: MONADS</b>	<b>193</b>
What's in a Typeclass?	194
What's in a Monad?	197
Why Do a Whole Abstraction?	198

A Second Monad: Logging	201
Syntactic Sugar	203
Binding with SelectMany?	204
Summary	205

## **PART IV: PUTTING FUNCTIONAL PROGRAMMING INTO ACTION**

<b>CHAPTER 18: INTEGRATING FUNCTIONAL PROGRAMMING APPROACHES</b>	<b>209</b>
<b>Refactoring</b>	<b>210</b>
List Filtering with a Windows Forms UI	211
Calculating Mandelbrot Fractals	216
<b>Writing New Code</b>	<b>224</b>
Use Static Methods	224
Prefer Anonymous Methods Over Named Ones	226
Prefer Higher Order Functions over Manual Algorithm Implementation	227
Prefer Immutable Data	228
Watch Behavior Implementation in Classes	229
<b>Finding Likely Candidates for Functional Programming</b>	<b>229</b>
Shades of Grey	230
Using What's There	231
<b>Summary</b>	<b>232</b>
<b>CHAPTER 19: THE MAPREDUCE PATTERN</b>	<b>233</b>
<b>Implementing MapReduce</b>	<b>234</b>
<b>Abstracting the Problem</b>	<b>237</b>
<b>Summary</b>	<b>240</b>
<b>CHAPTER 20: APPLIED FUNCTIONAL MODULARIZATION</b>	<b>241</b>
<b>Executing SQL Code from an Application</b>	<b>241</b>
<b>Rewriting the Function with Partial Application and Precomputation in Mind</b>	<b>243</b>
<b>Summary</b>	<b>245</b>

<b>CHAPTER 21: EXISTING PROJECTS USING FUNCTIONAL TECHNIQUES</b>	<b>247</b>
<hr/>	
The .NET Framework	247
LINQ	249
LINQ to Objects	249
LINQ to a Query Backend	253
Parallelization	255
Google MapReduce and Its Implementations	257
NUnit	258
Summary	260
<b>INDEX</b>	<b>261</b>