

Contents at a Glance

Introduction	1	Technique 18: Fixing Breaks with Casts	90
Part I: Streamlining the Means and Mechanics of OOP	5	Technique 19: Using Pointers to Member Functions	96
Technique 1: Protecting Your Data with Encapsulation	7	Technique 20: Defining Default Arguments for Your Functions and Methods	101
Technique 2: Using Abstraction to Extend Functionality	12	Part IV: Classes	107
Technique 3: Customizing a Class with Virtual Functions	19	Technique 21: Creating a Complete Class	109
Technique 4: Inheriting Data and Functionality	23	Technique 22: Using Virtual Inheritance	116
Technique 5: Separating Rules and Data from Code	30	Technique 23: Creating Overloaded Operators	120
Part II: Working with the Pre-Processor	37	Technique 24: Defining Your Own new and delete Handlers	128
Technique 6: Handling Multiple Operating Systems	39	Technique 25: Implementing Properties	136
Technique 7: Mastering the Evils of Asserts	42	Technique 26: Doing Data Validation with Classes	142
Technique 8: Using const Instead of #define	45	Technique 27: Building a Date Class	149
Technique 9: Macros and Why Not to Use Them	48	Technique 28: Overriding Functionality with Virtual Methods	162
Technique 10: Understanding sizeof	52	Technique 29: Using Mix-In Classes	168
Part III: Types	57	Part V: Arrays and Templates	173
Technique 11: Creating Your Own Basic Types	59	Technique 30: Creating a Simple Template Class	175
Technique 12: Creating Your Own Types	63	Technique 31: Extending a Template Class	179
Technique 13: Using Enumerations	70	Technique 32: Creating Templates from Functions and Methods	186
Technique 14: Creating and Using Structures	73	Technique 33: Working with Arrays	192
Technique 15: Understanding Constants	77	Technique 34: Implementing Your Own Array Class	196
Technique 16: Scoping Your Variables	82	Technique 35: Working with Vector Algorithms	200
Technique 17: Using Namespaces	85		

Technique 36: Deleting an Array of Elements	204	Part VIII: Utilities	335
Technique 37: Creating Arrays of Objects	209	Technique 56: Encoding and Decoding Data for the Web	337
Technique 38: Working with Arrays of Object Pointers	213	Technique 57: Encrypting and Decrypting Strings	343
Technique 39: Implementing a Spreadsheet	216	Technique 58: Converting the Case of a String	349
Part VI: Input and Output	223	Technique 59: Implementing a Serialization Interface	354
Technique 40: Using the Standard Streams to Format Data	225	Technique 60: Creating a Generic Buffer Class	360
Technique 41: Reading In and Processing Files	228	Technique 61: Opening a File Using Multiple Paths	366
Technique 42: How to Read Delimited Files	234	Part IX: Debugging C++ Applications	373
Technique 43: Writing Your Objects as XML	240	Technique 62: Building Tracing into Your Applications	375
Technique 44: Removing White Space from Input	246	Technique 63: Creating Debugging Macros and Classes	387
Technique 45: Creating a Configuration File	250	Technique 64: Debugging Overloaded Methods	399
Part VII: Using the Built-In Functionality	263	Part X: The Scary (or Fun!) Stuff	405
Technique 46: Creating an Internationalization Class	265	Technique 65: Optimizing Your Code	407
Technique 47: Hashing Out Translations	279	Technique 66: Documenting the Data Flow	416
Technique 48: Implementing Virtual Files	283	Technique 67: Creating a Simple Locking Mechanism	420
Technique 49: Using Iterators for Your Collections	291	Technique 68: Creating and Using Guardian Classes	425
Technique 50: Overriding the Allocator for a Collection Class	297	Technique 69: Working with Complex Numbers	432
Technique 51: Using the auto_ptr Class to Avoid Memory Leaks	303	Technique 70: Converting Numbers to Words	439
Technique 52: Avoiding Memory Overwrites	307	Technique 71: Reducing the Complexity of Code	447
Technique 53: Throwing, Catching, and Re-throwing Exceptions	312	Index	455
Technique 54: Enforcing Return Codes	323		
Technique 55: Using Wildcards	330		

Table of Contents

Introduction	1	Technique 5: Separating Rules and Data from Code	30
Saving Time with This Book	2	The cDate Class	31
What's Available on the Companion Web Site?	2	Testing the cDate Class	35
Conventions Used in This Book	2		
What's In This Book	3	Part II: Working with the Pre-Processor	37
<i>Part I: Streamlining the Means and Mechanics of OOP</i>	3	Technique 6: Handling Multiple Operating Systems	39
<i>Part II: Working with the Pre-Processor</i>	3	Creating the Header File	39
<i>Part III: Types</i>	3	Testing the Header File	40
<i>Part IV: Classes</i>	3	Technique 7: Mastering the Evils of Asserts	42
<i>Part V: Arrays and Templates</i>	3	The Assert Problem	42
<i>Part VI: Input and Output</i>	4	Fixing the Assert Problem	44
<i>Part VII: Using the Built-in Functionality</i>	4	Technique 8: Using const Instead of #define	45
<i>Part VIII: Utilities</i>	4	Using the const Construct	46
<i>Part IX: Debugging C++ Applications</i>	4	Identifying the Errors	47
<i>Part X: The Scary (or Fun!) Stuff</i>	4	Fixing the Errors	47
Icons Used in This Book	4	Technique 9: Macros and Why Not to Use Them	48
		Initiating a Function with a String Macro — Almost	49
Part I: Streamlining the Means and Mechanics of OOP	5	Fixing What Went Wrong with the Macro	50
Technique 1: Protecting Your Data with Encapsulation	7	Using Macros Appropriately	51
Creating and Implementing an Encapsulated Class	7	Technique 10: Understanding sizeof	52
Making Updates to an Encapsulated Class	10	Using the sizeof Function	52
Technique 2: Using Abstraction to Extend Functionality	12	Evaluating the Results	54
Creating a Mailing-List Application	12	Using sizeof with Pointers	55
Testing the Mailing-List Application	17		
Technique 3: Customizing a Class with Virtual Functions	19	Part III: Types	57
Customizing a Class with Polymorphism	20	Technique 11: Creating Your Own Basic Types	59
Testing the Virtual Function Code	21	Implementing the Range Class	60
Why Do the Destructors Work?	22	Testing the Range Class	62
Technique 4: Inheriting Data and Functionality	23	Technique 12: Creating Your Own Types	63
Implementing a ConfigurationFile Class	24	Creating the Matrix Class	64
Testing the ConfigurationFile Class	27	Matrix Operations	65
Delayed Construction	27		

Multiplying a Matrix by a Scalar Value	66		
Multiplying a Matrix by Scalar Values, Take 2	67		
Testing the Matrix Class	68		
Technique 13: Using Enumerations	70		
Implementing the Enumeration Class	71		
Testing the Enumeration Class	72		
Technique 14: Creating and Using Structures	73		
Implementing Structures	74		
Interpreting the Output	75		
Technique 15: Understanding Constants	77		
Defining Constants	77		
Implementing Constant Variables	78		
Testing the Constant Application	80		
Using the <i>const</i> Keyword	81		
Technique 16: Scoping Your Variables	82		
Illustrating Scope	83		
Interpreting the Output	84		
Technique 17: Using Namespaces	85		
Creating a Namespace Application	86		
Testing the Namespace Application	88		
Technique 18: Fixing Breaks with Casts	90		
Using Casts	91		
Addressing the Compiler Problems	93		
Testing the Changes	94		
Technique 19: Using Pointers to Member Functions	96		
Implementing Member-Function Pointers	97		
Updating Your Code with Member-Function Pointers	99		
Testing the Member Pointer Code	99		
Technique 20: Defining Default Arguments for Your Functions and Methods	101		
Customizing the Functions We Didn't Write	102		
Customizing Functions We Wrote Ourselves	103		
Testing the Default Code	105		
Fixing the Problem	106		
		Part IV: Classes	107
		Technique 21: Creating a Complete Class	109
		Creating a Complete Class Template	110
		Testing the Complete Class	113
		Technique 22: Using Virtual Inheritance	116
		Implementing Virtual Inheritance	118
		Correcting the Code	119
		Technique 23: Creating Overloaded Operators	120
		Rules for Creating Overloaded Operators	121
		Using Conversion Operators	122
		Using Overloaded Operators	122
		Testing the MyString Class	125
		Technique 24: Defining Your Own new and delete Handlers	128
		Rules for Implementing new and delete Handlers	129
		Overloading new and delete Handlers	129
		Testing the Memory Allocation Tracker	133
		Technique 25: Implementing Properties	136
		Implementing Properties	137
		Testing the Property Class	140
		Technique 26: Doing Data Validation with Classes	142
		Implementing Data Validation with Classes	142
		Testing Your SSN Validator Class	146
		Technique 27: Building a Date Class	149
		Creating the Date Class	150
		Implementing the Date Functionality	152
		Testing the Date Class	159
		Some Final Thoughts on the Date Class	161
		Technique 28: Overriding Functionality with Virtual Methods	162
		Creating a Factory Class	163
		Testing the Factory	166
		Enhancing the Manager Class	167
		Technique 29: Using Mix-In Classes	168
		Implementing Mix-In Classes	169
		Compiling and Testing Your Mix-In Class	170

Part V: Arrays and Templates	173	Technique 41: Reading In and Processing Files	228
Technique 30: Creating a Simple Template Class	175	Testing the File-Reading Code	232
Technique 31: Extending a Template Class	179	Creating the Test File	233
Implementing Template Classes in Code	180	Technique 42: How to Read Delimited Files	234
Testing the Template Classes	182	Reading Delimited Files	234
Using Non-class Template Arguments	184	Testing the Code	238
Technique 32: Creating Templates from Functions and Methods	186	Technique 43: Writing Your Objects as XML	240
Implementing Function Templates	186	Creating the XML Writer	241
Creating Method Templates	189	Testing the XML Writer	243
Technique 33: Working with Arrays	192	Technique 44: Removing White Space from Input	246
Using the Vector Class	192	Technique 45: Creating a Configuration File	250
Technique 34: Implementing Your Own Array Class	196	Creating the Configuration-File Class	251
Creating the String Array Class	196	Setting Up Your Test File	260
Technique 35: Working with Vector Algorithms	200	Testing the Configuration-File Class	260
Working with Vector Algorithms	200	Part VII: Using the Built-In Functionality	263
Technique 36: Deleting an Array of Elements	204	Technique 46: Creating an Internationalization Class	265
Examining Allocations of Arrays and Pointers	204	Building the Language Files	266
Technique 37: Creating Arrays of Objects	209	Creating an Input Text File	272
Technique 38: Working with Arrays of Object Pointers	213	Reading the International File	272
Creating an Array of Heterogeneous Objects	213	Testing the String Reader	277
Technique 39: Implementing a Spreadsheet	216	Technique 47: Hashing Out Translations	279
Creating the Column Class	217	Creating a Translator Class	279
Creating the Row Class	218	Testing the Translator Class	281
Creating the Spreadsheet Class	219	Technique 48: Implementing Virtual Files	283
Testing Your Spreadsheet	221	Creating a Virtual File Class	283
Part VI: Input and Output	223	Testing the Virtual File Class	289
Technique 40: Using the Standard Streams to Format Data	225	Improving Your Virtual File Class	290
Working with Streams	225	Technique 49: Using Iterators for Your Collections	291
Technique 41: Reading In and Processing Files	228	Technique 50: Overriding the Allocator for a Collection Class	297
Testing the File-Reading Code	232	Creating a Custom Memory Allocator	298
Creating the Test File	233		
Technique 42: How to Read Delimited Files	234		
Reading Delimited Files	234		
Testing the Code	238		
Technique 43: Writing Your Objects as XML	240		
Creating the XML Writer	241		
Testing the XML Writer	243		
Technique 44: Removing White Space from Input	246		
Technique 45: Creating a Configuration File	250		
Creating the Configuration-File Class	251		
Setting Up Your Test File	260		
Testing the Configuration-File Class	260		

Technique 51: Using the auto_ptr Class to Avoid Memory Leaks	303	Technique 60: Creating a Generic Buffer Class	360
Using the auto_ptr Class	303	Creating the Buffer Class	361
Technique 52: Avoiding Memory Overwrites	307	Testing the Buffer Class	364
Creating a Memory Safe Buffer Class	307	Technique 61: Opening a File Using Multiple Paths	366
Technique 53: Throwing, Catching, and Re-throwing Exceptions	312	Creating the Multiple-Search-Path Class	367
Throwing and Logging Exceptions	312	Testing the Multiple-Search-Path Class	369
Dealing with Unhandled Exceptions	317	Part IX: Debugging C++ Applications	373
Re-throwing Exceptions	319	Technique 62: Building Tracing into Your Applications	375
Technique 54: Enforcing Return Codes	323	Implementing the Flow Trace Class	376
Technique 55: Using Wildcards	330	Testing the Flow Trace System	379
Creating the Wildcard Matching Class	331	Adding in Tracing After the Fact	380
Testing the Wildcard Matching Class	333	Technique 63: Creating Debugging Macros and Classes	387
Part VIII: Utilities	335	The assert Macro	387
Technique 56: Encoding and Decoding Data for the Web	337	Logging	389
Creating the URL Codec Class	338	Testing the Logger Class	390
Testing the URL Codec Class	340	Design by Contract	392
Technique 57: Encrypting and Decrypting Strings	343	Technique 64: Debugging Overloaded Methods	399
Implementing the Rot13 Algorithm	344	Adding Logging to the Application	401
Testing the Rot13 Algorithm	345	Part X: The Scary (or Fun!) Stuff	405
Implementing the XOR Algorithm	346	Technique 65: Optimizing Your Code	407
Testing the XOR Algorithm	347	Making Functions Inline	407
Technique 58: Converting the Case of a String	349	Avoiding Temporary Objects	408
Implementing the transform Function to Convert Strings	350	Passing Objects by Reference	410
Testing the String Conversions	351	Postponing Variable Declarations	412
Technique 59: Implementing a Serialization Interface	354	Choosing Initialization Instead of Assignment	413
Implementing the Serialization Interface	355	Technique 66: Documenting the Data Flow	416
Testing the Serialization Interface	358	Learning How Code Operates	416
		Testing the <i>Properties</i> Class	418

Technique 67: Creating a Simple Locking Mechanism	420
Creating the Locking Mechanism	421
Testing the Locking Mechanism	422
Technique 68: Creating and Using Guardian Classes	425
Creating the File-Guardian Class	426
Testing the File-Guardian Class	430
Technique 69: Working with Complex Numbers	432
Implementing the Complex Class	433
Testing the Complex Number Class	436
Technique 70: Converting Numbers to Words	439
Creating the Conversion Code	440
Testing the Conversion Code	446
Technique 71: Reducing the Complexity of Code	447
A Sample Program	447
Componentizing	449
Restructuring	451
Specialization	452
<i>Index</i>	<i>455</i>

