

# Practical Ruby Projects

## Ideas for the Eclectic Programmer



Topher Cyll

## **Practical Ruby Projects: Ideas for the Eclectic Programmer**

**Copyright © 2008 by Topher Cyll**

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13 (pbk): 978-1-59059-911-2

ISBN-10 (pbk): 1-59059-911-X

ISBN-13 (electronic): 978-1-4302-0470-1

ISBN-10 (electronic): 1-4302-0470-2

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Lead Editors: Chris Mills and Tom Welsh

Technical Reviewer: Ben Matasar

Editorial Board: Steve Anglin, Ewan Buckingham, Tony Campbell, Gary Cornell, Jonathan Gennick,

Jason Gilmore, Kevin Goff, Jonathan Hassell, Matthew Moodie, Joseph Ottinger, Jeffrey Pepper,

Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh

Project Manager: Candace English

Copy Editor: Kim Benbow

Associate Production Director: Kari Brooks-Copony

Production Editor: Laura Esterman

Compositor: Molly Sharp, ContentWorks

Proofreader: Martha Whitt

Indexer: Carol Burbo

Cover Designer: Kurt Krames

Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail [orders-ny@springer-sbm.com](mailto:orders-ny@springer-sbm.com), or visit <http://www.springeronline.com>.

For information on translations, please contact Apress directly at 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705. Phone 510-549-5930, fax 510-549-5939, e-mail [info@apress.com](mailto:info@apress.com), or visit <http://www.apress.com>.

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers at <http://www.apress.com>.

*Dedicated to the Author and the Engineer, for all they taught me.*



# Contents at a Glance

About the Author .....	xv
About the Technical Reviewer .....	xvii
Acknowledgments .....	xix
<b>CHAPTER 1</b> Introduction .....	1
<b>CHAPTER 2</b> Making Music with Ruby .....	7
<b>CHAPTER 3</b> Animating Ruby .....	51
<b>CHAPTER 4</b> Pocket Change: Simulating Coin Systems with Ruby .....	93
<b>CHAPTER 5</b> Turn-Based Strategy in Ruby .....	119
<b>CHAPTER 6</b> RubyCocoa .....	153
<b>CHAPTER 7</b> Genetic Algorithms in Ruby .....	197
<b>CHAPTER 8</b> Implementing Lisp in Ruby .....	223
<b>CHAPTER 9</b> Parsing in Ruby .....	261
<b>INDEX</b> .....	293



# Contents

About the Author .....	xv
About the Technical Reviewer .....	xvii
Acknowledgments .....	xix
■ <b>CHAPTER 1 Introduction</b> .....	1
Why Ruby? .....	1
The Language .....	1
The Community .....	2
Why This Book? .....	2
Getting Set Up .....	3
Source Code in This Book .....	4
Your Projects .....	5
■ <b>CHAPTER 2 Making Music with Ruby</b> .....	7
MIDI: Giving Yourself a Vocabulary .....	7
Talking C and Making Noise .....	9
Sharing Code .....	10
Interfacing with Windows Multimedia .....	12
Interfacing with CoreMIDI .....	16
Interfacing with ALSA .....	19
Building a Metronome .....	22
Keeping Time .....	23
A Working Metronome .....	25
Fixing Your Timer Drift .....	26
Writing the Play Method .....	26
Avoiding Too Many Timers .....	28
Composing .....	29
Notation .....	29
Patterns .....	29
Playing Songs .....	33

Tempo Tap . . . . .	34
Taking Patterns Further . . . . .	35
Saving Your Music . . . . .	36
Live Coding . . . . .	39
Interfaces for Live Coding . . . . .	40
Improvements for Live Coding . . . . .	45
Summary . . . . .	49

<b>■ CHAPTER 3   Animating Ruby . . . . .</b>	<b>51</b>
Scalable Vector Graphics . . . . .	51
SVG Basics . . . . .	52
SVG Shapes . . . . .	52
The Animator . . . . .	55
Rendering the Animation . . . . .	57
Registering and Running Callbacks . . . . .	58
Embedded Ruby Templating . . . . .	60
Rendering the Frames . . . . .	61
Binding Objects . . . . .	62
Wrapping SVG with Objects . . . . .	64
Drawing One Cube . . . . .	65
Drawing Many Cubes . . . . .	67
Domain-Specific Languages . . . . .	67
Implementing GridDrawer . . . . .	69
Metaprogramming . . . . .	71
The Draw Method . . . . .	73
Deferring Execution . . . . .	74
Adding Deferred Execution to GridDrawer . . . . .	76
A Few More Helper Methods . . . . .	77
Your First Animation . . . . .	78
Putting the Animations Together . . . . .	83
ImageMagick . . . . .	83
iMovie . . . . .	83
JPGVideo . . . . .	85
Don't Give Up . . . . .	86
Spicing It Up . . . . .	86
Summary . . . . .	91



<b>CHAPTER 4</b>	<b>Pocket Change: Simulating Coin Systems with Ruby</b>	93
	Going Shopping	93
	How to Make Change	95
	The Greedy Algorithm	95
	Problems with the Greedy Algorithm	96
	Brute Force	96
	Adding the min_by Method	97
	Putting It All Together	98
	Dynamic Programming	99
	The Customer	100
	Memoization	106
	Hash Problems	107
	Paying	109
	The ChangeSimulator	110
	So How Heavy Are Your Pockets?	111
	Replacing a Coin	111
	Adding a Coin	112
	Optimal Coins	113
	Two Coins	114
	Three Coins	114
	Four Coins	114
	Beyond	115
	Wizard Money	116
	In the Literature	117
	Summary	118
<b>CHAPTER 5</b>	<b>Turn-Based Strategy in Ruby</b>	119
	A Strategy	119
	An Implementation	121
	Building the World Around Us	121
	Starting with Terrain	122
	Implementing Maps with Matrices	122
	Cartography 101	124
	Where Does Terrain Come From?	125
	Representing a Map	128

Meeting Your Heroes . . . . .	129
The Universal Skeleton . . . . .	129
Stubbing Out Undefined Classes . . . . .	132
Representing Units . . . . .	133
Making Choices . . . . .	133
Finding Possible Moves . . . . .	135
Choosing Among Actions . . . . .	135
Taking Action . . . . .	136
The Players . . . . .	139
The Artificial Intelligence Doesn't Seem So Intelligent . . . . .	142
Writing a Command-Line Player . . . . .	143
The Game . . . . .	144
Putting It All Together . . . . .	150
Summary . . . . .	152

<b>■ CHAPTER 6    RubyCocoa . . . . .</b>	<b>153</b>
The Very Basics . . . . .	153
Opening a Window . . . . .	154
Learning Objective-C Basics . . . . .	155
Calling Objective-C from Ruby . . . . .	156
Applications and Windows . . . . .	157
Building a Turn-Based Strategy Game . . . . .	158
Building a Player Using Cocoa . . . . .	158
An Odd Way to Do Things . . . . .	161
Understanding Views, Controls, and Cells . . . . .	162
Adding a View . . . . .	163
Displaying Messages . . . . .	166
Creating a Row of NSButtonCells . . . . .	167
The Choice Bar . . . . .	169
Drawing the Map . . . . .	172
Making Choices . . . . .	177
Selecting Units from the Map . . . . .	180
Highlighting Map Locations . . . . .	180
Handling Clicks . . . . .	181

Using Image Tiles .....	184
PlanetCute to the Rescue .....	184
Switching from Colors to Images .....	185
Adding Image-Based Tilesets to DinoCocoaPlayer .....	186
Fixing the Weirdness .....	187
Packaging It Up .....	192
Summary .....	195

## ■ CHAPTER 7 Genetic Algorithms in Ruby .....

Simulating Evolution .....	198
Implementing the Algorithm .....	199
Running the Iterations .....	200
What's Required to Be a Genome? .....	201
Remembering Winning Solutions .....	202
Thinking About Encodings .....	203
Using Integers As Bit Strings .....	203
Playing with Crossover .....	204
Modeling Crossover .....	205
Uniform Crossover .....	206
Point Crossovers .....	207
Using Mutation .....	208
Subclassing Integer .....	208
Subclassing BitInt .....	209
Wrapping BitInt Return Values .....	210
Making Change ... Again! .....	211
Choosing an Encoding .....	212
Running the Simulation .....	214
Looking at the Results .....	215
Adding Further Improvements .....	216
Dealing with Invalid Genomes .....	216
Letting Parents Live On .....	216
Experimenting with Gray Code .....	217
Roulette Selection .....	219
Summary .....	221

<b>CHAPTER 8</b>	<b>Implementing Lisp in Ruby</b>	223
	Learning Lisp	224
	Choosing Your Lisp Data Types	224
	Building Cons Cells	224
	Saving Values in the Environment	226
	Understanding eval and apply	230
	eval	230
	apply	232
	Talking About Special Forms	233
	Finishing eval	233
	Using the Helper Functions Arrayify and Consify	234
	Making It Look Like Lisp	235
	Choosing Your Primitive Functions	236
	Creating an Interpreter Object	238
	But What About Special Forms?	240
	Adding quote	240
	Adding define and set!	241
	Adding Conditional Expressions	241
	Adding lambda	242
	Implementing Macros	247
	Implementing the let Macro	248
	It Just Ain't Lisp Without eval	250
	Adding Lexical Macros	251
	Interoperating with Ruby	253
	Opening a Window to Ruby	254
	Sending Messages	254
	Making Lisp Lambda Work in Ruby	255
	Summary	256
<b>CHAPTER 9</b>	<b>Parsing in Ruby</b>	261
	Parsing with Ruby	262
	Understanding Grammars	262
	Recursive Descent Parsing	263
	RParsec	263

Parsing S-Expressions .....	265
Revisiting S-Expressions .....	265
Parsing Integers .....	265
Unit Test Everything .....	266
Parsing Floats .....	267
Deciding Between Different Number Types .....	268
Parsing Symbols with Regular Expressions .....	268
Parsing Values .....	270
Parsing Lists and Discarding Return Values .....	271
Using the Lazy Combinator .....	272
Parsing Your First S-Expressions to the End of File Marker .....	273
Quoting in Lisp .....	274
Parsing String Literals .....	274
Abstracting String Parsing .....	276
Putting It to Work .....	277
Parsing List Comprehensions .....	278
Making a Plan .....	278
Creating Abstract Syntax Tree Nodes .....	279
Reusing Combinators from the Last Parser .....	280
Parsing the List Comprehension Syntax .....	281
Testing Your Partial Implementation .....	282
Parsing Method Calls with Dot .....	282
Eliminating Left Recursion .....	283
Method Calls in List Comprehensions .....	285
Running the Comprehensions .....	286
Adding Some Convenience .....	288
Abusing Ruby Bindings .....	289
Summary .....	290

■ INDEX .....	293
---------------	-----



# About the Author

■ **TOPHER CYLL** is a software engineer and writer living in Cambridge, Massachusetts. He received his bachelor's degree in computer science from Williams College and works for a small Boston-area startup.

In reverse alphabetical order, he finds programming languages, music, Free Software, education, bioengineering, and beer terribly exciting.

Topher loves Ruby not only for the language itself, but also for the light-hearted and intellectually curious community that surrounds it.





# About the Technical Reviewer



■ **BEN MATASAR** is a developer at Smallthought Systems, where he works on Dabble DB, an online database written from scratch in Squeak Smalltalk. He considers himself lucky because he is able to make a living writing mostly Smalltalk and Ruby. He earned a B.S. in Electrical Engineering and Computer Science from the University of California at Berkeley, and is a political activist in his home state of Oregon. He bounces between Portland, Oregon, and Vancouver, British Columbia.



# Acknowledgments

**T**hanks go to the wonderful Apress team and all my editors.

I'm grateful to Ben Matasar and Adam Bouhengal for brainstorming and listening to my ideas with a critical ear. Thanks to the hackers on the Intel Oregon CPU Architecture Team and to the sharp minds at Adverplex for their support and enthusiasm. Special thanks to the eclectic programmers of the Portland Ruby Brigade for showing me the curious excitement of Ruby.

Additional thanks to my family, friends, and roommates for cutting me a year's worth of slack. I owe you!

