

CONTENTS IN DETAIL

ACKNOWLEDGMENTS

xvii

INTRODUCTION	1
What Makes Lisp So Cool and Unusual?	2
If Lisp Is So Great, Why Don't More People Use It?	3
Where Did Lisp Come From?	4
Where Does Lisp Get Its Power?	10

SECTION I: LISP IS POWER

1	
GETTING STARTED WITH LISP	15
Lisp Dialects	15
A Tale of Two Lisps	16
Up-and-Coming Lisps	17
Lisp Dialects Used for Scripting	17
ANSI Common Lisp	17
Getting Started with CLISP	18
Installing CLISP	18
Starting Up CLISP	19
What You've Learned	19
2	
CREATING YOUR FIRST LISP PROGRAM	21
The Guess-My-Number Game	21
Defining Global Variables in Lisp	23
Defining the small and big Variables	23
An Alternative Global Variable Definition Function	23
Basic Lisp Etiquette	24
Defining Global Functions in Lisp	25
Defining the guess-my-number Function	25
Defining the smaller and bigger Functions	27
Defining the start-over Function	28
Defining Local Variables in Lisp	28
Defining Local Functions in Lisp	29
What You've Learned	30

3	EXPLORING THE SYNTAX OF LISP CODE	31
	Syntax and Semantics	31
	The Building Blocks of Lisp Syntax	32
	Symbols	33
	Numbers	34
	Strings	35
	How Lisp Distinguishes Between Code and Data	35
	Code Mode	36
	Data Mode	37
	Lists in Lisp	37
	Cons Cells	38
	List Functions	38
	Nested Lists	41
	What You've Learned	45

SECTION II: LISP IS SYMMETRY

4	MAKING DECISIONS WITH CONDITIONS	49
	The Symmetry of nil and ()	49
	Empty Equals False	50
	The Four Disguises of ()	51
	The Conditionals: if and Beyond	52
	One Thing at a Time with if	52
	Going Beyond if: The when and unless Alternatives	55
	The Command That Does It All: cond	56
	Branching with case	57
	Cool Tricks with Conditions	58
	Using the Stealth Conditionals and and or	58
	Using Functions That Return More than Just the Truth	60
	Comparing Stuff: eq, equal, and More	62
	What You've Learned	65

5	BUILDING A TEXT GAME ENGINE	67
	The Wizard's Adventure Game	68
	Our Game World	68
	Basic Requirements	69
	Describing the Scenery with an Association List	70
	Describing the Location	71
	Describing the Paths	72
	How Quasiquoting Works	73
	Describing Multiple Paths at Once	73
	Describing Objects at a Specific Location	77
	Listing Visible Objects	77
	Describing Visible Objects	78

Describing It All	79
Walking Around in Our World	81
Picking Up Objects	82
Checking Our Inventory	83
What You've Learned	84

6 INTERACTING WITH THE WORLD: READING AND PRINTING IN LISP 85

Printing and Reading Text	86
Printing to the Screen	86
Saying Hello to the User	87
Starting with print and read	88
Reading and Printing Stuff the Way Humans Like It	90
The Symmetry Between Code and Data in Lisp	91
Adding a Custom Interface to Our Game Engine	92
Setting Up a Custom REPL	93
Writing a Custom read Function	94
Writing a game-eval Function	96
Writing a game-print Function	96
Trying Out Our Fancy New Game Interface	99
The Dangers of read and eval	101
What You've Learned	101

6.5 LAMBDA: A FUNCTION SO IMPORTANT IT DESERVES ITS OWN CHAPTER 103

What lambda Does	103
Why lambda Is So Important	105
What You've Learned	106

7 GOING BEYOND BASIC LISTS 107

Exotic Lists	107
Dotted Lists	108
Pairs	109
Circular Lists	110
Association Lists	111
Coping with Complicated Data	113
Visualizing Tree-like Data	113
Visualizing Graphs	114
Creating a Graph	114
Generating the DOT Information	115
Turning the DOT File into a Picture	120
Creating a Picture of Our Graph	123
Creating Undirected Graphs	124
What You've Learned	127

8

THIS AIN'T YOUR DADDY'S WUMPUS

129

The Grand Theft Wumpus Game	131
Defining the Edges of Congestion City	135
Generating Random Edges	135
Looping with the loop Command	136
Preventing Islands	137
Building the Final Edges for Congestion City	139
Building the Nodes for Congestion City	142
Initializing a New Game of Grand Theft Wumpus	144
Drawing a Map of Our City	145
Drawing a City from Partial Knowledge	146
Walking Around Town	148
Let's Hunt Some Wumpus!	149
What You've Learned	152

9

ADVANCED DATATYPES AND GENERIC PROGRAMMING 153

Arrays	153
Working with Arrays	154
Using a Generic Setter	154
Arrays vs. Lists	156
Hash Tables	157
Working with Hash Tables	157
Returning Multiple Values	159
Hash Table Performance	160
A Faster Grand Theft Wumpus Using Hash Tables	161
Common Lisp Structures	163
Working with Structures	163
When to Use Structures	165
Handling Data in a Generic Way	166
Working with Sequences	166
Creating Your Own Generic Functions with Type Predicates	170
The Orc Battle Game	172
Global Variables for the Player and Monsters	173
Main Game Functions	174
Player Management Functions	175
Helper Functions for Player Attacks	177
Monster Management Functions	178
The Monsters	179
To Battle!	187
What You've Learned	189

SECTION III: LISP IS HACKING

loop and format: The Seedy Underbelly of Lisp	193
---	-----

10	LOOPING WITH THE LOOP COMMAND	195
The loop Macro		195
Some loop Tricks		196
Everything You Ever Wanted to Know About loop		202
Using loop to Evolve!		202
Growing Plants in Our World		204
Creating Animals		205
Simulating a Day in Our World		212
Drawing Our World		212
Creating a User Interface		213
Let's Watch Some Evolution!		214
Explaining the Evolution		218
What You've Learned		219
11	PRINTING TEXT WITH THE FORMAT FUNCTION	221
Anatomy of the format Function		221
The Destination Parameter		222
The Control String Parameter		222
Value Parameters		223
Control Sequences for Printing Lisp Values		223
Control Sequences for Formatting Numbers		225
Control Sequences for Formatting Integers		225
Control Sequences for Formatting Floating-Point Numbers		226
Printing Multiple Lines of Output		226
Justifying Output		228
Iterating Through Lists Using Control Sequences		231
A Crazy Formatting Trick for Creating Pretty Tables of Data		232
Attack of the Robots!		233
What You've Learned		235
12	WORKING WITH STREAMS	237
Types of Streams		238
Streams by Type of Resource		238
Streams by Direction		238
Working with Files		242
Working with Sockets		244
Socket Addresses		245
Socket Connections		246
Sending a Message over a Socket		246
Tidying Up After Ourselves		248
String Streams: The Oddball Type		249
Sending Streams to Functions		249
Working with Long Strings		250
Reading and Debugging		250
What You've Learned		251

13	LET'S CREATE A WEB SERVER!	253
Error Handling in Common Lisp		253
Signaling a Condition		254
Creating Custom Conditions		254
Intercepting Conditions		255
Protecting Resources Against Unexpected Conditions		255
Writing a Web Server from Scratch		256
How a Web Server Works		256
Request Parameters		258
Parsing the Request Header		261
Testing get-header with a String Stream		262
Parsing the Request Body		263
Our Grand Finale: The serve Function!		263
Building a Dynamic Website		265
Testing the Request Handler		265
Launching the Website		266
What You've Learned		267

FUNCTIONAL PROGRAMMING IS BEAUTIFUL **269**

SECTION IV: LISP IS SCIENCE

14	RAMPING LISP UP A NOTCH WITH FUNCTIONAL PROGRAMMING	291
What Is Functional Programming?		292
Anatomy of a Program Written in the Functional Style		295
Higher-Order Programming		298
Code Composition with Imperative Code		298
Using the Functional Style		299
Higher-Order Programming to the Rescue		300
Why Functional Programming Is Crazy		300
Why Functional Programming Is Fantastic		301
Functional Programming Reduces Bugs		301
Functional Programs Are More Compact		301
Functional Code Is More Elegant		302
What You've Learned		302

15	DICE OF DOOM, A GAME WRITTEN IN THE FUNCTIONAL STYLE	303
The Rules of Dice of Doom		304
A Sample Game of Dice of Doom		304

Implementing Dice of Doom, Version 1	306
Defining Some Global Variables	306
Representing the Game Board	307
Decoupling Dice of Doom's Rules from the Rest of the Game	309
Generating a Game Tree	311
Calculating Passing Moves	312
Calculating Attacking Moves	313
Finding the Neighbors	314
Attacking	315
Reinforcements	316
Trying Out Our New game-tree Function	317
Playing Dice of Doom Against Another Human	318
Creating an Intelligent Computer Opponent	321
The Minimax Algorithm	323
Turning Minimax into Actual Code	323
Creating a Game Loop with an AI Player	324
Playing Our First Human vs. Computer Game	325
Making Dice of Doom Faster	326
Closures	326
Memoization	328
Tail Call Optimization	331
A Sample Game on the 3-by-3 Board	334
What You've Learned	336

16

THE MAGIC OF LISP MACROS 339

A Simple Lisp Macro	340
Macro Expansion	341
How Macros Are Transformed	342
Using the Simple Macro	345
More Complex Macros	345
A Macro for Splitting Lists	346
Avoiding Repeated Execution in Macros	347
Avoiding Variable Capture	348
A Recursion Macro	350
Macros: Dangers and Alternatives	352
What You've Learned	353

17

DOMAIN-SPECIFIC LANGUAGES 355

What Is a Domain?	355
Writing SVG Files	356
Creating XML and HTML with the tag Macro	357
Creating SVG-Specific Macros and Functions	361
Building a More Complicated SVG Example	362
Creating Custom Game Commands for Wizard's Adventure Game	365
Creating New Game Commands by Hand	366
Let's Try the Completed Wizard's Adventure Game!	371
What You've Learned	373

18		
LAZY PROGRAMMING		375
Adding Lazy Evaluation to Lisp		376
Creating the lazy and force Commands		378
Creating a Lazy Lists Library		380
Converting Between Regular Lists and Lazy Lists		381
Mapping and Searching Across Lazy Lists		383
Dice of Doom, Version 2		384
Making Our AI Work on Larger Game Boards		387
Trimming the Game Tree		387
Applying Heuristics		389
Winning by a Lot vs. Winning by a Little		389
Alpha Beta Pruning		393
What You've Learned		400

19		
CREATING A GRAPHICAL, WEB-BASED VERSION OF DICE OF DOOM		401
Drawing the Game Board Using the SVG Format		402
Drawing a Die		403
Drawing a Tile		405
Drawing the Board		406
Building the Web Server Interface		408
Writing Our Web Request Handler		408
Limitations of Our Game Web Server		409
Initializing a New Game		410
Announcing a Winner		410
Handling the Human Player		410
Handling the Computer Player		412
Drawing the SVG Game Board from Within the HTML		412
Playing Version 3 of Dice of Doom		413
What You've Learned		415

20		
MAKING DICE OF DOOM MORE FUN		417
Increasing the Number of Players		417
Rolling the Dice		418
Building Chance Nodes		419
Doing the Actual Dice Rolling		420
Calling the Dice Rolling Code from Our Game Engine		420
Updating the AI		422
Improving the Dice of Doom Reinforcement Rules		423
Conclusion		425

EPILOGUE		429
-----------------	--	------------

INDEX		465
--------------	--	------------