

# REALM OF RACKET

Learn to Program, One Game at a Time!



Forrest Bice, Rose DeMaio, Spencer Florence, Feng-Yun Mimi Lin,  
Scott Lindeman, Nicole Nussbaum, Eric Peterson, Ryan Plessner  
David Van Horn | Conrad Barski, MD  
Matthias Felleisen



# CONTENTS IN DETAIL

## PREFACE

### HELLO WORLD

xv

Why Would I Want to Learn About Racket? . . . . .	xv
Who Should Read This Book? . . . . .	xvi
What Teaching Approach Is Used? . . . . .	xvi
Can I Skip Chapters? . . . . .	xvi
Anything Else I Should Know? . . . . .	xvi

## INTRODUCTION

### OPEN PAREN

1

(.1 What Makes Lisp So Cool and Unusual? . . . . .	2
(.2 Where Did Lisp Come From? . . . . .	2
(.3 What Does Lisp Look Like? . . . . .	5
(.4 Where Does Racket Come From? . . . . .	7
(.5 What Is This Book About? . . . . .	9
HalT—Chapter Checkpoint . . . . .	10

## 1

### GETTING STARTED

19

1.1 Readying Racket . . . . .	19
1.2 Interacting with Racket . . . . .	21
Raise—Chapter Checkpoint . . . . .	23

## 2

### A FIRST RACKET PROGRAM

27

2.1 The Guess My Number Game. . . . .	27
2.2 Defining Variables. . . . .	29
2.3 Basic Racket Etiquette . . . . .	30
2.4 Defining Functions in Racket. . . . .	30
A Function for Guessing . . . . .	31
Functions for Closing In . . . . .	32
The Main Function . . . . .	33
Resume—Chapter Checkpoint . . . . .	34

## 3

### BASICS OF RACKET

35

3.1 Syntax and Semantics. . . . .	35
3.2 The Building Blocks of Racket Syntax. . . . .	36
3.3 The Building Blocks of Racket Semantics. . . . .	38
Booleans . . . . .	38
Symbols . . . . .	39

	Numbers . . . . .	39
	Strings . . . . .	40
3.4	Lists in Racket . . . . .	41
	CONS Cells . . . . .	42
	Functions for CONS Cells . . . . .	42
	Lists and List Functions . . . . .	43
	The CONS Function . . . . .	43
	The LIST Function . . . . .	45
	The FIRST and REST Functions . . . . .	45
	Nested Lists . . . . .	46
3.5	Structures in Racket . . . . .	47
	Structure Basics . . . . .	47
	Nesting Structures . . . . .	49
	Structure Transparency . . . . .	50
	Interrupt—Chapter Checkpoint . . . . .	50

## **4**

### **CONDITIONS AND DECISIONS** **51**

4.1	How to Ask . . . . .	51
4.2	The Conditionals: IF and Beyond . . . . .	56
	One Thing at a Time with IF . . . . .	56
	The Special Form that Does It All: COND . . . . .	58
	A First Taste of Recursion . . . . .	59
4.3	Cool Tricks with Conditionals . . . . .	61
	Using the Stealth Conditionals AND and OR . . . . .	61
	Using Functions that Return More than Just the Truth . . . . .	63
4.4	Equality Predicates, Once More . . . . .	64
4.5	Comparing and Testing . . . . .	68
	Writing a Test . . . . .	68
	What Is Not a Test . . . . .	69
	Testing in the Real World . . . . .	69
	More Testing Facilities . . . . .	70
	Call-with-current-continuation—Chapter Checkpoint . . . . .	70

## **4½**

### **DEFINE DEFINE 'DEFINE** **71**

4½.1	Module-Level Definitions . . . . .	71
	Variable Definitions . . . . .	71
	Function Definitions . . . . .	73
4½.2	Local Definitions . . . . .	73
	Abort—Chapter Checkpoint . . . . .	75

## **5**

### **BIG-BANG** **79**

5.1	Graphical User Interface . . . . .	79
5.2	Landing a UFO . . . . .	80
5.3	Using big-bang: Syntax and Semantics . . . . .	83

5.4	Guessing Goopy . . . . .	85
	The Data . . . . .	85
	The Main Function . . . . .	86
	Key-Events. . . . .	87
	Rendering . . . . .	88
	Time to Stop . . . . .	88
	Exit—Chapter Checkpoint . . . . .	89
	Chapter Challenges . . . . .	89

## 6

### **RECURSION IS EASY 95**

6.1	Robot Snake . . . . .	95
6.2	A Data Representation for the Snake Game . . . . .	96
6.3	The Main Function . . . . .	97
6.4	Clock Ticks . . . . .	98
	Eating and Growing. . . . .	99
	Slithering. . . . .	100
	Rotting Goo . . . . .	102
6.5	Key-Events. . . . .	103
6.6	Rendering . . . . .	105
6.7	End Game . . . . .	107
6.8	Auxiliary Functions . . . . .	108
	Return—Chapter Checkpoint . . . . .	109
	Chapter Challenges . . . . .	109

## 7

### **LAND OF LAMBDA 111**

7.1	Functions as Values . . . . .	111
7.2	Lambda. . . . .	113
7.3	Higher-Order Fun . . . . .	114
7.4	Two More Higher-Order Functions. . . . .	118
7.5	Derive This! . . . . .	121
7.6	apply. . . . .	122
	Break—Chapter Checkpoint . . . . .	123

## 8

### **MUTANT STRUCTS 127**

8.1	Chad's First Battle . . . . .	127
8.2	Orc Battle . . . . .	128
8.3	Setting Up the World, a First Step . . . . .	128
8.4	Action: How Structs Really Work . . . . .	131
8.5	More Actions, Setting Up the World for Good . . . . .	135
8.6	Ready, Set, big-bang . . . . .	136
8.7	Initializing the Orc World . . . . .	140
8.8	Rendering the Orc World . . . . .	142
8.9	The End of the World . . . . .	146
8.10	Actions, A Final Look . . . . .	147

Throw—Chapter Checkpoint . . . . .	151
Chapter Challenges . . . . .	151
<b>9</b>	
<b>THE VALUES OF LOOPS</b>	<b>153</b>
9.1 FOR Loops . . . . .	153
9.2 Multiple Values . . . . .	155
9.3 Back to FOR/FOLD . . . . .	156
9.4 More on Loops . . . . .	157
waitpid—Chapter Checkpoint . . . . .	160
<b>10</b>	
<b>DICE OF DOOM</b>	<b>165</b>
10.1 The Game Tree . . . . .	165
10.2 Dice of Doom, The Game . . . . .	166
10.3 Designing Dice of Doom: Take One . . . . .	166
Filling in the Blanks . . . . .	167
Simplifying the Rules . . . . .	167
End of Game . . . . .	167
Controlling the Game . . . . .	168
10.4 How Game Trees Work . . . . .	168
10.5 Game States and Game Trees for Dice of Doom . . . . .	171
10.6 Roll the Dice . . . . .	174
10.7 Rendering the Dice World . . . . .	176
10.8 Input Handling . . . . .	179
10.9 Creating a Game Tree . . . . .	181
The Game Tree . . . . .	182
Neighbors . . . . .	184
Attacks . . . . .	187
10.10 The End Game . . . . .	189
kill—Chapter Checkpoint . . . . .	190
Chapter Challenges . . . . .	190
<b>11</b>	
<b>POWER TO THE LAZY</b>	<b>193</b>
11.1 Doomsday . . . . .	193
11.2 Lazy Evaluation . . . . .	194
11.3 Memoized Computations . . . . .	196
11.4 Racket Can Be Lazy . . . . .	198
Delay—Chapter Checkpoint . . . . .	198
<b>12</b>	
<b>ARTIFICIAL INTELLIGENCE</b>	<b>203</b>
12.1 An Intelligent Life-form . . . . .	203
12.2 Lazy Games . . . . .	204
12.3 Adding Artificial Intelligence . . . . .	206

stop-when—Chapter Checkpoint . . . . .	208
Chapter Challenges . . . . .	209

## 13

### THE WORLD IS NOT ENOUGH 213

13.1 What Is a Distributed Game? . . . . .	213
13.2 The Data . . . . .	215
Messages . . . . .	215
Previously Fabricated Structures . . . . .	215
Packages . . . . .	216
Bundles . . . . .	216
Mail . . . . .	216
iworld Structures . . . . .	216
13.3 The Network Postal Service . . . . .	217
13.4 Organizing Your Universe . . . . .	218
13.5 Distributed Guess . . . . .	219
The State of the Client and the State of the Server . . . . .	221
The Server . . . . .	222
The Client . . . . .	224
Running the Game . . . . .	225
error—Chapter Checkpoint . . . . .	225
Chapter Challenges . . . . .	226

## 14

### HUNGRY HENRY 231

14.1 King Henry the Hungry . . . . .	231
14.2 Hungry Henry, the Game . . . . .	232
14.3 Two United States . . . . .	232
14.4 Henry's Universe . . . . .	233
Message Data and Structures . . . . .	233
Complex Numbers Are Good Positions . . . . .	236
A Day in the Life of a Server . . . . .	236
A Day in the Life of a Client . . . . .	236
14.5 State of the Union . . . . .	236
State of Henry . . . . .	237
State of the House . . . . .	237
14.6 Main, Take Client . . . . .	238
The Appetizer State . . . . .	239
The Entree State . . . . .	242
14.7 Main, Take Server . . . . .	245
The Join State and Network Events . . . . .	247
The Join State and Tick Events . . . . .	249
The Play State and Network Events . . . . .	251
The Play State and Tick Events . . . . .	253
14.8 See Henry Run . . . . .	257
on-disconnect—Chapter Checkpoint . . . . .	258
Chapter Challenges . . . . .	258

**GOOD-BYE**

**CLOSE PAREN**

**265**

.1 Run Racket Run . . . . . 265  
.2 Racket Is a Programming Language . . . . . 266  
.3 Racket Is a Metaprogramming Language . . . . . 270  
.4 Racket Is a Programming-Language Programming Language . . . . . 274  
So Long . . . . . 281