

# CONTENTS IN DETAIL

## INTRODUCTION xv

So, What's Haskell? .....	xv
What You Need to Dive In .....	xvii
Acknowledgments .....	xviii

## 1 1 STARTING OUT

Calling Functions .....	3
Baby's First Functions .....	5
An Intro to Lists .....	7
Concatenation .....	8
Accessing List Elements .....	9
Lists Inside Lists .....	9
Comparing Lists .....	9
More List Operations .....	10
Texas Ranges .....	13
I'm a List Comprehension .....	15
Tuples .....	18
Using Tuples .....	19
Using Pairs .....	20
Finding the Right Triangle .....	21

## 2 23 BELIEVE THE TYPE

Explicit Type Declaration .....	24
Common Haskell Types .....	25
Type Variables .....	26
Type Classes 101 .....	27
The Eq Type Class .....	28
The Ord Type Class .....	28
The Show Type Class .....	29
The Read Type Class .....	29
The Enum Type Class .....	31
The Bounded Type Class .....	31
The Num Type Class .....	32
The Floating Type Class .....	32
The Integral Type Class .....	33
Some Final Notes on Type Classes .....	33

<b>3</b>	<b>SYNTAX IN FUNCTIONS</b>	<b>35</b>
Pattern Matching .....	35	
Pattern Matching with Tuples .....	37	
Pattern Matching with Lists and List Comprehensions .....	38	
As-patterns .....	40	
Guards, Guards! .....	40	
where?! .....	42	
where's Scope .....	44	
Pattern Matching with where .....	44	
Functions in where Blocks .....	45	
let It Be .....	45	
let in List Comprehensions .....	47	
let in GHCi .....	47	
case Expressions .....	48	
 <b>4</b>	 <b>HELLO RECURSION!</b>	 <b>51</b>
Maximum Awesome .....	52	
A Few More Recursive Functions .....	53	
replicate .....	53	
take .....	54	
reverse .....	55	
repeat .....	55	
zip .....	55	
elem .....	56	
Quick, Sort! .....	56	
The Algorithm .....	56	
The Code .....	57	
Thinking Recursively .....	58	
 <b>5</b>	 <b>HIGHER-ORDER FUNCTIONS</b>	 <b>59</b>
Curried Functions .....	59	
Sections .....	62	
Printing Functions .....	63	
Some Higher-Orderism Is in Order .....	63	
Implementing zipWith .....	64	
Implementing flip .....	65	
The Functional Programmer's Toolbox .....	66	
The map Function .....	66	
The filter Function .....	67	
More Examples of map and filter .....	68	
Mapping Functions with Multiple Parameters .....	70	
Lambdas .....	71	

I Fold You So .....	73
Left Folds with foldl .....	74
Right Folds with foldr .....	75
The foldl and foldr1 Functions .....	76
Some Fold Examples .....	76
Another Way to Look at Folds .....	77
Folding Infinite Lists .....	78
Scans .....	79
Function Application with \$ .....	80
Function Composition .....	82
Function Composition with Multiple Parameters .....	83
Point-Free Style .....	84

## 6 **MODULES** **87**

Importing Modules .....	88
Solving Problems with Module Functions .....	90
Counting Words .....	90
Needle in the Haystack .....	91
Caesar Cipher Salad .....	92
On Strict Left Folds .....	94
Let's Find Some Cool Numbers .....	95
Mapping Keys to Values .....	98
Almost As Good: Association Lists .....	98
Enter Data.Map .....	100
Making Our Own Modules .....	104
A Geometry Module .....	104
Hierarchical Modules .....	106

## 7 **MAKING OUR OWN TYPES AND TYPE CLASSES** **109**

Defining a New Data Type .....	109
Shaping Up .....	110
Improving Shape with the Point Data Type .....	112
Exporting Our Shapes in a Module .....	113
Record Syntax .....	114
Type Parameters .....	117
Should We Parameterize Our Car? .....	119
Vector von Doom .....	121
Derived Instances .....	122
Equating People .....	123
Show Me How to Read .....	124
Order in the Court! .....	125
Any Day of the Week .....	126

Type Synonyms .....	127
Making Our Phonebook Prettier .....	128
Parameterizing Type Synonyms .....	129
Go Left, Then Right .....	130
Recursive Data Structures .....	132
Improving Our List .....	133
Let's Plant a Tree .....	135
Type Classes 102 .....	138
Inside the Eq Type Class .....	138
A Traffic Light Data Type .....	139
Subclassing .....	140
Parameterized Types As Instances of Type Classes .....	141
A Yes-No Type Class .....	143
The Functor Type Class .....	146
Maybe As a Functor .....	147
Trees Are Functors, Too .....	148
Either a As a Functor .....	149
Kinds and Some Type-Foo .....	150

## 8

### INPUT AND OUTPUT

153

Separating the Pure from the Impure .....	153
Hello, World! .....	154
Gluing I/O Actions Together .....	156
Using let Inside I/O Actions .....	158
Putting It in Reverse .....	159
Some Useful I/O Functions .....	161
putStr .....	161
putChar .....	162
print .....	162
when .....	163
sequence .....	164
mapM .....	165
forever .....	165
forM .....	166
I/O Action Review .....	167

## 9

### MORE INPUT AND MORE OUTPUT

169

Files and Streams .....	169
Input Redirection .....	170
Getting Strings from Input Streams .....	171
Transforming Input .....	173

Reading and Writing Files .....	175
Using the withFile Function .....	177
It's Bracket Time .....	178
Grab the Handles! .....	179
To-Do Lists .....	180
Deleting Items .....	181
Cleaning Up .....	183
Command-Line Arguments .....	184
More Fun with To-Do Lists .....	185
A Multitasking Task List .....	186
Dealing with Bad Input .....	190
Randomness .....	190
Tossing a Coin .....	193
More Random Functions .....	194
Randomness and I/O .....	195
Bytestrings .....	198
Strict and Lazy Bytestrings .....	199
Copying Files with Bytestrings .....	201

## **10 FUNCTIONALLY SOLVING PROBLEMS 203**

Reverse Polish Notation Calculator .....	203
Calculating RPN Expressions .....	204
Writing an RPN Function .....	205
Adding More Operators .....	207
Heathrow to London .....	208
Calculating the Quickest Path .....	209
Representing the Road System in Haskell .....	211
Writing the Optimal Path Function .....	212
Getting a Road System from the Input .....	215

## **11 APPLICATIVE FUNCTORS 217**

Functors Redux .....	218
I/O Actions As Functors .....	218
Functions As Functors .....	220
Functor Laws .....	223
Law 1 .....	223
Law 2 .....	224
Breaking the Law .....	225
Using Applicative Functors .....	227
Say Hello to Applicative .....	228
Maybe the Applicative Functor .....	229
The Applicative Style .....	230

Lists .....	232
IO Is An Applicative Functor, Too .....	234
Functions As Applicatives .....	235
Zip Lists .....	237
Applicative Laws .....	238
Useful Functions for Applicatives .....	238

## **12 MONOIDS 243**

Wrapping an Existing Type into a New Type .....	243
Using newtype to Make Type Class Instances .....	246
On newtype Laziness .....	247
type vs. newtype vs. data .....	249
About Those Monoids .....	250
The Monoid Type Class .....	252
The Monoid Laws .....	253
Meet Some Monoids .....	253
Lists Are Monoids .....	253
Product and Sum .....	254
Any and All .....	256
The Ordering Monoid .....	257
Maybe the Monoid .....	260
Folding with Monoids .....	262

## **13 A FISTFUL OF MONADS 267**

Upgrading Our Applicative Functors .....	267
Getting Your Feet Wet with Maybe .....	269
The Monad Type Class .....	272
Walk the Line .....	274
Code, Code, Code .....	274
I'll Fly Away .....	276
Banana on a Wire .....	278
do Notation .....	280
Do As I Do .....	282
Pierre Returns .....	282
Pattern Matching and Failure .....	284
The List Monad .....	285
do Notation and List Comprehensions .....	288
MonadPlus and the guard Function .....	288
A Knight's Quest .....	290
Monad Laws .....	292
Left Identity .....	293
Right Identity .....	294
Associativity .....	294

## **14**

### **FOR A FEW MONADS MORE**

## **297**

Writer? I Hardly Knew Her! .....	298
Monoids to the Rescue .....	300
The Writer Type .....	302
Using do Notation with Writer .....	303
Adding Logging to Programs .....	304
Inefficient List Construction .....	306
Using Difference Lists .....	307
Comparing Performance .....	309
Reader? Ugh, Not This Joke Again .....	310
Functions As Monads .....	311
The Reader Monad .....	312
Tasteful Stateful Computations .....	313
Stateful Computations .....	314
Stacks and Stones .....	314
The State Monad .....	316
Getting and Setting State .....	318
Randomness and the State Monad .....	320
Error Error on the Wall .....	321
Some Useful Monadic Functions .....	323
liftM and Friends .....	323
The join Function .....	326
filterM .....	328
foldM .....	331
Making a Safe RPN Calculator .....	332
Composing Monadic Functions .....	335
Making Monads .....	336

## **15**

### **ZIPPERS**

## **343**

Taking a Walk .....	344
A Trail of Breadcrumbs .....	346
Going Back Up .....	348
Manipulating Trees Under Focus .....	350
Going Straight to the Top, Where the Air Is Fresh and Clean! .....	351
Focusing on Lists .....	352
A Very Simple Filesystem .....	353
Making a Zipper for Our Filesystem .....	355
Manipulating a Filesystem .....	357
Watch Your Step .....	358
Thanks for Reading! .....	360

## **INDEX**

## **363**