

# Beginning Haskell

A Project-Based Approach



Alejandro Serrano Mena

Apress®

# Contents

<b>About the Author .....</b>	<b>xvii</b>
<b>About the Technical Reviewer .....</b>	<b>xix</b>
<b>Acknowledgments .....</b>	<b>xxi</b>
<b>Introduction .....</b>	<b>xxiii</b>
 <b>■Part 1: First Steps.....</b>	 <b>1</b>
<b>■Chapter 1: Going Functional .....</b>	<b>3</b>
Why Haskell?.....	3
Why Pure Functional Programming? .....	4
Why Strong Static Typing?.....	5
The Haskell Ecosystem.....	6
The History of Haskell .....	6
Your Working Environment .....	6
Installing on Windows.....	7
Installing on Mac OS X.....	7
Installing on Linux .....	8
Installing on Linux from Source .....	8
Checking That the Installation is Successful .....	9
Installing EclipseFP.....	9
First steps with GHCi .....	11
The Time Machine Store.....	12
Summary.....	13

<b>Chapter 2: Declaring the Data Model</b>	<b>15</b>
Working with Characters, Numbers, and Lists	15
Characters	15
Numbers	16
Strings	18
Lists	18
Lists Operations	19
Creating a New Project	22
Creating a Project from the Command Line	22
Creating a Project from EclipseFP	23
Understanding Modules	25
Defining Simple Functions	26
Creating a Simple Function	26
Specifying the Function's Type	27
Developing a Robust Example	27
Returning More than One Value	29
Working with Data Types	30
Pattern Matching	33
Simple Patterns	33
Lists and Tuples	37
Guards	38
View Patterns	40
Records	41
Creation and Use	41
The "Default Values" Idiom	43
Summary	45

<b>■Chapter 3: Reusing Code Through Lists.....</b>	<b>47</b>
Parametric Polymorphism .....	47
Functions as Parameters.....	50
Higher-Order Functions .....	50
Anonymous Functions .....	52
Partial Application of a Function.....	54
More on Modules.....	57
Module Imports.....	57
Smart Constructors and Views .....	59
Diving into Lists.....	61
Folds .....	61
Lists and Predicates .....	64
Lists Containing Tuples.....	69
List Comprehensions.....	70
Haskell Origami .....	73
Summary .....	76
<b>■Chapter 4: Using Containers and Type Classes.....</b>	<b>77</b>
Using Packages.....	77
Managing Packages with Cabal and EclipseFP .....	77
Sandboxed Environments.....	82
Containers: Maps, Sets, Trees, Graphs .....	84
Maps.....	84
Sets.....	87
Trees .....	89
Graphs .....	91
Obtaining Help .....	93
Ad-hoc Polymorphism: Type Classes.....	94
Declaring Classes and Instances.....	94
Built-in Type Classes .....	97

<b>Binary Tress for the Minimum Price .....</b>	<b>101</b>
Step 1: Simple Binary Trees.....	101
Step 2: Polymorphic Binary Trees .....	103
Step 3: Binary Trees with Monoidal Cache .....	104
<b>Container-related Type Classes .....</b>	<b>106</b>
Functors.....	106
Foldables .....	108
<b>Summary.....</b>	<b>109</b>
<b>■Chapter 5: Laziness and Infinite Structures .....</b>	<b>111</b>
An Infinite Number of Time Machines .....	111
Lazy Evaluation Model.....	115
Understanding Evaluation in Haskell .....	115
Problems with Laziness.....	119
Pattern Matching and Laziness .....	120
Profiling with GHC .....	122
Strictness Annotations .....	127
Summary.....	129
<b>■Part 2: Data Mining.....</b>	<b>131</b>
<b>■Chapter 6: Knowing Your Clients Using Monads .....</b>	<b>133</b>
Data Mining .....	133
Implementing K-means .....	134
Lenses .....	138
Discovering Monads.....	143
Watching out for Incomplete Data.....	143
Combinators for State.....	145
Dissecting the Combinators.....	148
do Notation .....	150
Monad Laws .....	152

Different Sorts of State.....	153
State and Lenses .....	153
Reader, Writer, and RWS .....	155
Mutable References with ST.....	158
Summary .....	159
<b>■Chapter 7: More Monads: Now for Recommendations .....</b>	<b>161</b>
Returning More Than One Value .....	161
The List Monad .....	162
A New View Over Monads.....	163
Failures and Alternatives.....	164
Association Rules Learning.....	167
Flattening Values into Transactions .....	167
The Apriori Algorithm.....	169
Search Problems .....	171
Paths in a Graph .....	172
The Logic Monad .....	173
Monads and Lists Redux .....	175
Combining Values Under a Monad .....	175
Monad Comprehensions.....	177
Combining Monads.....	179
Monad Transformers.....	180
Monad Classes .....	183
Summary .....	185
<b>■Chapter 8: Working in Several Cores.....</b>	<b>187</b>
Parallelism, Concurrency, Distribution.....	187
The Par Monad .....	188
Futures .....	188
Dataflow Parallelism with IVars.....	190
Parallelizing the Apriori Algorithm .....	192

<b>Software Transactional Memory</b>	<b>194</b>
Concurrent Use of Resources	194
Atomic Transactions	196
Rolling Back Transactions	197
Producer-Consumer Queues	199
<b>Cloud Haskell</b>	<b>200</b>
Looking for Galaxies	200
Looking for Typed Galaxies	204
Extra Features	205
<b>Summary</b>	<b>206</b>
<b>■Part 3: Resource Handling</b>	<b>207</b>
<b>■Chapter 9: Dealing with Files: IO and Conduit</b>	<b>209</b>
Basic Input and Output	209
Randomness	213
Working with Files	215
Reading and Writing	215
Handling Files	217
Error Handling	218
Pure Errors	218
Catching Exceptions	221
Throwing Exceptions	224
Streaming Data with Conduit	225
Problems with Lazy Input/Output	225
Introducing Conduits	226
Accessing Files via Conduit	229
Looking Further than Text Files	231
Basic Networking	231
Binary Serialization	232
Summary	234

<b>■Chapter 10: Building and Parsing Text .....</b>	<b>235</b>
The Five Textual Data Types .....	235
Building as Fast as the Wind .....	239
Parsing with attoparsec .....	241
Introducing New Type Classes .....	246
Applicative .....	247
Functors, Applicatives, and Monads .....	248
Alternative .....	250
Traversable .....	251
Don't Overengineer: Just Use JSON .....	253
Summary .....	258
<b>■Chapter 11: Safe Database Access .....</b>	<b>259</b>
Database Access Landscape .....	259
Abstracting over Several DBMSs .....	260
Introducing Persistent and Esqueleto .....	260
Connection .....	261
Schemas and Migrations .....	262
Describing the Entities .....	263
Creating the Database .....	266
Queries .....	269
Queries by Identifier or Uniqueness .....	269
Selecting Several Entities .....	270
SQL Queries with Esqueleto .....	271
Insertions, Updates, and Deletions .....	274
Summary .....	276
<b>■Chapter 12: Web Applications .....</b>	<b>277</b>
Haskell Web Ecosystem .....	277
Web Frameworks .....	277
Compilation to Javascript .....	278
RESTful Structure .....	279



Backend with Scotty .....	280
Simple Skeleton.....	280
Showing Products from the Database .....	281
Inserting New Products Using Forms .....	285
Frontend with Fay .....	289
Foreign Function Interface.....	290
Fay and jQuery.....	291
Summary.....	293
<b>■Part 4: Domain Specific Languages.....</b>	<b>295</b>
<b>■Chapter 13: Strong Types for Describing Offers .....</b>	<b>297</b>
Domain Specific Languages.....	297
Embedding Your Language in Haskell .....	298
The Offers Language .....	299
Adding Safety to the Expression Language.....	301
Dependent Typing.....	304
Introducing Idris .....	305
Enforcing the Presents Rule in Idris .....	308
Type-level Programming in Haskell.....	309
Two Styles of Programming.....	309
Representing Natural Numbers .....	310
Functional Dependencies .....	311
Categories of Products with FDs .....	311
Enforcing the Presents Rule with FDs .....	314
Type Families .....	317
Enforcing the Presents Rule with TFs.....	317
Categories of Products with TFs.....	319
Data Type Promotion and Singletons.....	322
A Further Refinement to the Presents Rule .....	323
Enforcing the Duration Rule.....	325
Summary.....	330

<b>■Chapter 14: Interpreting Offers with Attributes.....</b>	<b>331</b>
Interpretations and Attribute Grammars.....	331
A Simple Interpretation.....	331
Introducing Attribute Grammars .....	332
Your First Attribute Grammar.....	333
Synthesizing the Result.....	334
Executing the Attribute Grammar .....	335
Integrating UUAGC in Your Package .....	336
Installing UUAGC.....	336
UUAGC and Cabal.....	336
Expressions Interpretation .....	338
Using an Attribute Grammar .....	338
Precomputing Some Values.....	341
A Different (Monadic) View .....	342
Offer Interpretations.....	343
Checking the Presents Rule.....	343
Showing a HTML Description.....	345
Origami Programming Over Any Data Type .....	348
Summary.....	350
 <b>■Part 5: Engineering the Store .....</b>	<b>353</b>
<b>■Chapter 15: Documenting, Testing, and Verifying.....</b>	<b>355</b>
Documenting Binary Trees with Haddock.....	355
Unit Testing with HUnit .....	359
Declaring Tests in Cabal .....	359
Writing Unit Tests.....	360
Randomized Testing with QuickCheck .....	363
Testing List Properties .....	364
Testing Binary Tree Properties.....	365
Formal Verification with Idris .....	367
Summary.....	371

<b>Chapter 16: Architecting Your Application .....</b>	<b>373</b>
Design Patterns and Functional Programming.....	373
Medium-Level Guidelines.....	374
Use Higher-Order Combinators.....	374
Refactor .....	375
Use Type Classes Wisely .....	375
Enforce Invariants via the Type System.....	375
Stay (as) Pure and Polymorphic (as Possible) .....	375
Tools .....	376
Project and Dependency Management.....	376
Code Style.....	376
Documentation .....	376
Test and Verification .....	377
Benchmarking .....	377
Profiling .....	377
Coverage.....	377
Remote Monitoring .....	377
Projects .....	378
Data Mining Library .....	378
Store Network Client .....	379
Administration Interface and Tetris.....	384
Roll Your Own Monad.....	384
Summary .....	387
<b>Appendix A: Looking Further .....</b>	<b>389</b>
Haskell Resources.....	389
Other Functional Languages.....	390
<b>Appendix B: Time Traveling with Haskell .....</b>	<b>391</b>
<b>Index.....</b>	<b>393</b>