# Table of Contents

**Preface. . . . . . . . . . xvii**

**1. Setting Up Your Go Environment. . . . . . . . . . 1**

Installing the Go Tools 1

  Troubleshooting Your Go Installation 2

  Go Tooling 2

Your First Go Program 3

  Making a Go Module 3

  go build 4

  go fmt 5

  go vet 7

Choose Your Tools 8

  Visual Studio Code 8

  GoLand 9

  The Go Playground 10

Makefiles 12

The Go Compatibility Promise 13

Staying Up-to-Date 14

Exercises 14

Wrapping Up 15

**2. Predeclared Types and Declarations. . . . . . . . . . 17**

The Predeclared Types 17

  The Zero Value 17

  Literals 18

  Booleans 19

  Numeric Types 20

  A Taste of Strings and Runes 25

Explicit Type Conversion 26
Literals Are Untyped 27
var Versus := 28
Using const 30
Typed and Untyped Constants 32
Unused Variables 32
Naming Variables and Constants 33
Exercises 35
Wrapping Up 36

**3. Composite Types. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37**
Arrays—Too Rigid to Use Directly 37
Slices 39
len 41
append 41
Capacity 42
make 43
Emptying a Slice 44
Declaring Your Slice 45
Slicing Slices 46
copy 49
Converting Arrays to Slices 50
Converting Slices to Arrays 51
Strings and Runes and Bytes 52
Maps 56
Reading and Writing a Map 57
The comma ok Idiom 58
Deleting from Maps 59
Emptying a Map 59
Comparing Maps 59
Using Maps as Sets 60
Structs 61
Anonymous Structs 63
Comparing and Converting Structs 64
Exercises 65
Wrapping Up 66

**4. Blocks, Shadows, and Control Structures. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 67**
Blocks 67
Shadowing Variables 68
if 71
for, Four Ways 72

The Complete for Statement 72
The Condition-Only for Statement 73
The Infinite for Statement 74
break and continue 75
The for-range Statement 76
Labeling Your for Statements 82
Choosing the Right for Statement 83
switch 84
Blank Switches 87
Choosing Between if and switch 89
goto—Yes, goto 89
Exercises 92
Wrapping Up 92

**5. Functions. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 93**
Declaring and Calling Functions 93
Simulating Named and Optional Parameters 94
Variadic Input Parameters and Slices 95
Multiple Return Values 96
Multiple Return Values Are Multiple Values 97
Ignoring Returned Values 97
Named Return Values 98
Blank Returns—Never Use These! 99
Functions Are Values 100
Function Type Declarations 103
Anonymous Functions 103
Closures 105
Passing Functions as Parameters 107
Returning Functions from Functions 108
defer 109
Go Is Call by Value 114
Exercises 116
Wrapping Up 117

**6. Pointers. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 119**
A Quick Pointer Primer 119
Don’t Fear the Pointers 123
Pointers Indicate Mutable Parameters 125
Pointers Are a Last Resort 129
Pointer Passing Performance 130
The Zero Value Versus No Value 131
The Difference Between Maps and Slices 131

Slices as Buffers 135
Reducing the Garbage Collector's Workload 136
Tuning the Garbage Collector 139
Exercises 141
Wrapping Up 141

**7. Types, Methods, and Interfaces. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 143**
Types in Go 143
Methods 144
Pointer Receivers and Value Receivers 145
Code Your Methods for nil Instances 148
Methods Are Functions Too 149
Functions Versus Methods 150
Type Declarations Aren't Inheritance 150
Types Are Executable Documentation 151
iota Is for Enumerations—Sometimes 152
Use Embedding for Composition 154
Embedding Is Not Inheritance 156
A Quick Lesson on Interfaces 157
Interfaces Are Type-Safe Duck Typing 158
Embedding and Interfaces 162
Accept Interfaces, Return Structs 162
Interfaces and nil 164
Interfaces Are Comparable 165
The Empty Interface Says Nothing 166
Type Assertions and Type Switches 167
Use Type Assertions and Type Switches Sparingly 170
Function Types Are a Bridge to Interfaces 173
Implicit Interfaces Make Dependency Injection Easier 174
Wire 178
Go Isn't Particularly Object-Oriented (and That's Great) 178
Exercises 178
Wrapping Up 179

**8. Generics. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 181**
Generics Reduce Repetitive Code and Increase Type Safety 181
Introducing Generics in Go 184
Generic Functions Abstract Algorithms 187
Generics and Interfaces 188
Use Type Terms to Specify Operators 190
Type Inference and Generics 193
Type Elements Limit Constants 193

Combining Generic Functions with Generic Data Structures 194
More on comparable 196
Things That Are Left Out 198
Idiomatic Go and Generics 199
Adding Generics to the Standard Library 201
Future Features Unlocked 201
Exercises 201
Wrapping Up 202

**9. Errors. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 203**
How to Handle Errors: The Basics 203
Use Strings for Simple Errors 205
Sentinel Errors 205
Errors Are Values 208
Wrapping Errors 210
Wrapping Multiple Errors 212
Is and As 214
Wrapping Errors with defer 217
panic and recover 218
Getting a Stack Trace from an Error 220
Exercises 221
Wrapping Up 221

**10. Modules, Packages, and Imports. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 223**
Repositories, Modules, and Packages 223
Using go.mod 224
  Use the go Directive to Manage Go Build Versions 225
  The require Directive 226
Building Packages 227
  Importing and Exporting 227
  Creating and Accessing a Package 227
  Naming Packages 230
  Overriding a Package's Name 230
  Documenting Your Code with Go Doc Comments 231
  Using the internal Package 234
  Avoiding Circular Dependencies 235
  Organizing Your Module 236
  Gracefully Renaming and Reorganizing Your API 238
  Avoiding the init Function if Possible 239
Working with Modules 240
  Importing Third-Party Code 240
  Working with Versions 245

Minimal Version Selection 247
Updating to Compatible Versions 248
Updating to Incompatible Versions 248
Vendoring 250
Using pkg.go.dev 251
Publishing Your Module 251
Versioning Your Module 252
Overriding Dependencies 254
Retracting a Version of Your Module 255
Using Workspaces to Modify Modules Simultaneously 255
Module Proxy Servers 259
Specifying a Proxy Server 259
Using Private Repositories 260
Additional Details 260
Exercises 261
Wrapping Up 261

**11. Go Tooling. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 263**
Using go run to Try Out Small Programs 263
Adding Third-Party Tools with go install 264
Improving Import Formatting with goimports 266
Using Code-Quality Scanners 267
staticcheck 268
revive 269
golangci-lint 270
Using govulncheck to Scan for Vulnerable Dependencies 272
Embedding Content into Your Program 274
Embedding Hidden Files 277
Using go generate 278
Working with go generate and Makefiles 281
Reading the Build Info Inside a Go Binary 281
Building Go Binaries for Other Platforms 283
Using Build Tags 284
Testing Versions of Go 285
Using go help to Learn More About Go Tooling 286
Exercises 286
Wrapping Up 286

**12. Concurrency in Go. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 287**
When to Use Concurrency 287
Goroutines 289
Channels 291

Reading, Writing, and Buffering 291
Using for-range and Channels 292
Closing a Channel 292
Understanding How Channels Behave 293
select 294
Concurrency Practices and Patterns 297
Keep Your APIs Concurrency-Free 297
Goroutines, for Loops, and Varying Variables 298
Always Clean Up Your Goroutines 299
Use the Context to Terminate Goroutines 300
Know When to Use Buffered and Unbuffered Channels 301
Implement Backpressure 302
Turn Off a case in a select 304
Time Out Code 304
Use WaitGroups 306
Run Code Exactly Once 308
Put Your Concurrent Tools Together 309
When to Use Mutexes Instead of Channels 313
Atomics—You Probably Don't Need These 317
Where to Learn More About Concurrency 317
Exercises 317
Wrapping Up 318

**13. The Standard Library. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 319**
io and Friends 319
time 324
Monotonic Time 326
Timers and Timeouts 327
encoding/json 327
Using Struct Tags to Add Metadata 327
Unmarshaling and Marshaling 329
JSON, Readers, and Writers 330
Encoding and Decoding JSON Streams 331
Custom JSON Parsing 332
net/http 336
The Client 336
The Server 337
ResponseController 342
Structured Logging 344
Exercises 347
Wrapping Up 347

**14. The Context. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 349**
What Is the Context? 349
Values 352
Cancellation 358
Contexts with Deadlines 363
Context Cancellation in Your Own Code 367
Exercises 368
Wrapping Up 369

**15. Writing Tests. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 371**
Understanding the Basics of Testing 371
Reporting Test Failures 373
Setting Up and Tearing Down 373
Testing with Environment Variables 376
Storing Sample Test Data 376
Caching Test Results 377
Testing Your Public API 377
Using go-cmp to Compare Test Results 378
Running Table Tests 380
Running Tests Concurrently 382
Checking Your Code Coverage 384
Fuzzing 386
Using Benchmarks 393
Using Stubs in Go 397
Using httptest 402
Using Integration Tests and Build Tags 405
Finding Concurrency Problems with the Data Race Detector 406
Exercises 408
Wrapping Up 408

**16. Here Be Dragons: Reflect, Unsafe, and Cgo. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 409**
Reflection Lets You Work with Types at Runtime 410
Types, Kinds, and Values 411
Make New Values 415
Use Reflection to Check If an Interface's Value Is nil 417
Use Reflection to Write a Data Marshaler 417
Build Functions with Reflection to Automate Repetitive Tasks 422
You Can Build Structs with Reflection, but Don't 423
Reflection Can't Make Methods 424
Use Reflection Only if It's Worthwhile 424
unsafe Is Unsafe 425
Using Sizeof and Offsetof 426

Using unsafe to Convert External Binary Data 428
Accessing Unexported Fields 432
Using unsafe Tools 433
Cgo Is for Integration, Not Performance 433
Exercises 438
Wrapping Up 438

**Index. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 441**