

# Inhaltsverzeichnis

<b>Vorwort</b> .....	<b>v</b>
<b>1 Einleitung</b> .....	<b>1</b>
1.1 Ziel dieses Buches .....	1
1.2 Die Geschichte von Go .....	2
1.3 Installation .....	3
1.4 Sicherheit und Patches .....	4
1.5 Editoren für den Go-Werkzeugkasten .....	5
1.6 Der Spielplatz für Gopher .....	5
1.7 Hello World .....	6
1.8 Eine lesbare technische Spezifikation .....	8
1.9 Ausgabe mit dem fmt-Paket .....	9
<b>2 Vorstellung der Syntax</b> .....	<b>13</b>
2.1 Wörter, Funktionen und Typen von Go .....	13
2.2 Variablen .....	15
2.3 Konstanten .....	17
2.4 Pointer .....	20
2.5 Eigene Typen .....	21
2.6 Typumwandlung .....	22
2.7 Zusammengesetzte Strukturen .....	23
2.8 Funktionen .....	26
2.9 Objektorientierung mit Methode .....	34
2.10 Exportiert und nicht exportiert .....	36
2.11 Arrays .....	38
2.12 Slices .....	40
2.13 Das Slice als Pointer .....	47
2.14 Maps .....	49
2.15 if .....	50
2.16 switch .....	51
2.17 for .....	54
2.18 Labels und goto .....	57
2.19 Blank Identifier .....	59
2.20 UTF-8, strings und runes .....	60

<b>3</b>	<b>Projekt: Command Line Interface</b> .....	<b>61</b>
3.1	Einleitung .....	61
3.2	gocat – File-Ausgabe .....	62
3.3	Den md5-Hash erzeugen .....	67
3.4	Dateien und HTTP-Server als Quellen für gomd5 .....	70
<b>4</b>	<b>Go Tooling</b> .....	<b>78</b>
4.1	Schnelle Hilfe mit go help .....	78
4.2	Kompilieren und Installieren .....	80
4.3	Umgebungsvariablen mit go env .....	82
4.4	Ein Programm für jede Gelegenheit – Build Tags .....	84
4.5	Wie Code formatiert wird – gofmt .....	88
4.6	Automatische Imports mit goimports .....	89
4.7	Dokumentation immer dabei – godoc .....	89
<b>5</b>	<b>Projekt: Ein einfacher Webloader</b> .....	<b>94</b>
5.1	Einleitung .....	94
5.2	CLI – unser Interface .....	94
5.3	HTTP-Request erstellen .....	96
5.4	Implementierung des File-Outputs .....	97
5.5	Ausgabe des HTTP-Headers .....	99
5.6	Gültigkeit der übergebenen URL .....	103
<b>6</b>	<b>Eigene Pakete und Module</b> .....	<b>106</b>
6.1	Go-Code lebt in Paketen .....	106
6.2	Paketnamen .....	107
6.3	Die init()-Funktion .....	108
6.4	Semantic Versioning .....	110
6.5	Pakete leben in Modulen .....	111
6.6	Der Workflow, seit es Module gibt .....	114
6.7	Neuer bedeutet nicht immer besser .....	120
6.8	Update unserer Abhängigkeit .....	121
6.9	Neue Major-Version mit Modulen .....	122
<b>7</b>	<b>Projekt: Code generieren</b> .....	<b>125</b>
7.1	Einleitung .....	125
7.2	Ein Tool, um Code zu generieren .....	126
7.3	Template erstellen .....	131
7.4	Anwenden von go generate .....	135
<b>8</b>	<b>Concurrency-Grundlagen</b> .....	<b>137</b>
8.1	Concurrency mit Go .....	137
8.2	Parallelität im echten Leben .....	139
8.3	Goroutinen .....	141

---

8.4	Channels .....	142
8.5	Einen Channel schließen .....	149
8.6	Select .....	153
8.7	Race Conditions und Data Races .....	158
<b>9</b>	<b>Concurrency Patterns .....</b>	<b>162</b>
9.1	Checkliste zu Goroutinen .....	162
9.2	Goroutinen melden, wenn sie fertig sind .....	164
9.3	Beenden von Goroutinen .....	165
9.4	Context .....	167
9.5	Prüfung eines geschlossenen Channels .....	173
9.6	Pipelines .....	173
9.7	Generator .....	175
9.8	Fan-In und Fan-Out .....	176
9.9	Channel of Channels .....	179
9.10	Worker Pool .....	181
9.11	Semaphore mit einem Buffered Channel .....	185
9.12	State Machine .....	186
<b>10</b>	<b>Projekt: Go Concurrency .....</b>	<b>190</b>
10.1	Einleitung .....	190
10.2	Command Line Interface .....	190
10.3	Argumente parsen .....	191
10.4	Befehle ausführen .....	195
10.5	Abbruch mit context .....	198
10.6	Verbesserung des Tools .....	200
<b>11</b>	<b>Testen und Benchmarks .....</b>	<b>203</b>
11.1	Tests in Go .....	203
11.2	Subtests .....	206
11.3	Tabellarische Tests .....	207
11.4	Eigenes Testpaket .....	209
11.5	Testen mit Beispielen .....	211
11.6	Ein ganzes Projekt testen .....	212
11.7	Benchmarks .....	214
11.8	Syntax der Benchmarks .....	215
11.9	Subbenchmarks .....	216
<b>12</b>	<b>Projekt: Image Resizer .....</b>	<b>220</b>
12.1	Einleitung .....	220
12.2	Command Line Interface – Erstellen der Flags .....	221
12.3	Größe erzeugen .....	223
12.4	Bild verkleinern .....	227
12.5	Filename prüfen .....	229

12.6	Funktionen zusammenführen .....	230
12.7	Refactoring in eine zusätzliche Funktion .....	234
12.8	Eigener Fehlertyp .....	236
12.9	Von sequentieller Ausführung zu nebenläufiger Ausführung .	239
<b>13</b>	<b>Interfaces .....</b>	<b>245</b>
13.1	Bessere Abstraktion mit Interfaces .....	245
13.2	Die richtige Interface-Erstellung .....	248
13.3	Interne Abbildung der Interface-Typen .....	251
13.4	Leeres Interface .....	254
13.5	Vom Interface zum konkreten Typ .....	255
13.6	Interface in andere Interfaces einbinden .....	258
13.7	Interfaces in Strukturen einbinden .....	259
13.8	Mocking und Tests mit io.Reader und io.Writer .....	261
<b>14</b>	<b>Projekt: Kopieren mit Reflection .....</b>	<b>266</b>
14.1	Einleitung .....	266
14.2	Reflection in Go .....	267
14.3	Beschreibung des Pakets .....	270
14.4	Testfälle für unser Paket .....	271
14.5	Umsetzung .....	273
14.6	Verwenden von Tags .....	276
<b>15</b>	<b>Fehlerbehandlung .....</b>	<b>278</b>
15.1	Grundlagen .....	278
15.2	Variablen und Konstanten .....	281
15.3	Eigene Fehlertypen .....	283
15.4	Einem Fehler Kontext hinzufügen .....	285
15.5	Keine Panik .....	288
<b>16</b>	<b>Projekt: Ein einfacher Webserver .....</b>	<b>290</b>
16.1	Einleitung .....	290
16.2	Das Modell für unseren Blog .....	290
16.3	Der Webserver und seine Handler .....	294
16.4	Templates erstellen .....	297
16.5	Kommentarfunktion .....	303
16.6	Files ausliefern .....	309
16.7	API bereitstellen .....	310
16.8	Template nur einmal parsen .....	312
16.9	Nebenläufiger Job für den Index .....	314
16.10	Ein paar kleine Verbesserungen .....	316