

# Auf einen Blick

---

<b>Über den Autor</b> .....	<b>9</b>
<b>Einleitung</b> .....	<b>23</b>
<b>Teil I: Das Clean-Code-Prinzip</b> .....	<b>29</b>
<b>Kapitel 1:</b> Software ist Code .....	31
<b>Kapitel 2:</b> Dimensionen von Codequalität .....	37
<b>Kapitel 3:</b> Alles unter einen Hut – gute Kompromisse finden .....	45
<b>Kapitel 4:</b> Die Eigenschaften sauberen Codes .....	55
<b>Kapitel 5:</b> In der Praxis: Stolpersteine .....	65
<b>Teil II: An Herausforderungen wachsen</b> .....	<b>75</b>
<b>Kapitel 6:</b> Mehr als Handwerkskunst. ....	77
<b>Kapitel 7:</b> Entwickeln ist (kreative) Wissenschaft. ....	85
<b>Kapitel 8:</b> Modellierungsdilemma und Entscheidungsmüdigkeit .....	95
<b>Kapitel 9:</b> Fallen vermeiden .....	105
<b>Teil III: Sauberen Code schreiben</b> .....	<b>115</b>
<b>Kapitel 10:</b> Namen sind nicht Schall und Rauch .....	117
<b>Kapitel 11:</b> Reine Formfrage – Formatierung .....	129
<b>Kapitel 12:</b> Code zuerst – sind Kommentare nötig? .....	141
<b>Kapitel 13:</b> Kleine Schritte – saubere Methoden .....	153
<b>Kapitel 14:</b> Passend schneiden – Schnittstellen .....	171
<b>Kapitel 15:</b> Objekte und Datensätze unterscheiden .....	191
<b>Kapitel 16:</b> Wege im Dschungel – Regeln .....	207
<b>Kapitel 17:</b> Fehler passieren – Fehlerbehandlung .....	221
<b>Kapitel 18:</b> Ausnahmen regeln – Exceptions .....	233
<b>Kapitel 19:</b> Immer weiter – neue Sprachmittel .....	243
<b>Teil IV: Wege zum Ziel</b> .....	<b>257</b>
<b>Kapitel 20:</b> Miteinander lernen – Code Reviews .....	259
<b>Kapitel 21:</b> Aus Fehlern lernen .....	269
<b>Kapitel 22:</b> Es gibt immer was zu tun – Refactoring .....	277

<b>Teil V: Der Top-Ten-Teil</b> .....	<b>283</b>
<b>Kapitel 23:</b> 10 Fehler, die Sie vermeiden sollten .....	285
<b>Kapitel 24:</b> (Mehr als) 10 nützliche Quellen zum Auffrischen und Vertiefen ...	289
<b>Stichwortverzeichnis</b> .....	<b>295</b>

# Inhaltsverzeichnis

---

<b>Über den Autor</b> .....	<b>9</b>
<b>Einleitung</b> .....	<b>23</b>
Über dieses Buch .....	23
Konventionen in diesem Buch .....	24
Was Sie nicht lesen müssen .....	24
Törichte Annahmen über die Leser .....	25
Wie dieses Buch aufgebaut ist .....	26
Symbole, die in diesem Buch verwendet werden .....	27
Wie es weitergeht .....	28
<b>TEIL I</b>	
<b>DAS CLEAN-CODE-PRINZIP</b> .....	<b>29</b>
<b>Kapitel 1</b>	
<b>Software ist Code</b> .....	<b>31</b>
Erwartungen an Software .....	31
Probleme haben Ursachen .....	32
Allgemeine Ursachen .....	32
Hardwareentwicklung .....	33
Nichts ohne Code .....	34
Das Wichtigste in Kürze .....	35
<b>Kapitel 2</b>	
<b>Dimensionen von Codequalität</b> .....	<b>37</b>
Was bedeutet Qualität? .....	37
Eigenschaften des Qualitätsbegriffs .....	37
Qualitätsindikatoren .....	38
Die Dimensionen von Codequalität .....	39
Korrektheit des Codes .....	39
Lesbarkeit und Wartbarkeit .....	40
Leistungseigenschaften .....	40
Weitere Dimensionen .....	41
Das Qualitätsziel festlegen .....	41
Beispiel: Der euklidische Algorithmus .....	42
Das Wichtigste in Kürze .....	43

<b>Kapitel 3</b>	
<b>Alles unter einen Hut – gute Kompromisse finden</b> .....	<b>45</b>
Warum gute Entscheidungen wichtig sind.....	45
Es kommt drauf an . . . . .	45
Widersprüche überall . . . . .	46
Konflikte akzeptieren. . . . .	47
Entscheidungen systematisch treffen . . . . .	48
Konflikte erkennen. . . . .	48
Alternativen sammeln . . . . .	48
Kriterien finden. . . . .	49
Wahlmöglichkeiten bewerten. . . . .	50
Entscheiden. . . . .	51
Mit Augenmaß . . . . .	51
Das Wichtigste in Kürze . . . . .	52
<b>Kapitel 4</b>	
<b>Die Eigenschaften sauberen Codes</b> .....	<b>55</b>
Des Clean Codes Kern. . . . .	55
Code als Ziel . . . . .	56
Professionalität. . . . .	57
Es geht immer weiter. . . . .	58
Code als Kommunikationsmittel zwischen Menschen. . . . .	58
Lesbarkeit . . . . .	59
Verständlichkeit . . . . .	59
Eleganz. . . . .	60
Gute Wartbarkeit . . . . .	61
Leichter durch Verständlichkeit . . . . .	61
Nicht ohne Test . . . . .	61
Zu guter Letzt . . . . .	62
Das Wichtigste in Kürze . . . . .	63
<b>Kapitel 5</b>	
<b>In der Praxis: Stolpersteine</b> .....	<b>65</b>
Clean Code ist schwer. . . . .	65
Reden wir über die Kosten. . . . .	65
Kurz- und mittelfristige Vorteile. . . . .	66
Langfristige Vorteile. . . . .	66
Bewertung. . . . .	68
Ändern bleibt schwierig . . . . .	68
Manchmal passt es nicht . . . . .	69
Frameworks . . . . .	70
Projektvorgaben. . . . .	70
Starre Abläufe. . . . .	71
Falsche Autoritäten . . . . .	71
Es liegt an Ihnen. . . . .	73
Das Wichtigste in Kürze . . . . .	73

**TEIL II  
AN HERAUSFORDERUNGEN WACHSEN ..... 75**

**Kapitel 6  
Mehr als Handwerkskunst..... 77**

Programmieren ist schwer ..... 77  
 Software professionell entwickeln ..... 79  
 Softwareentwicklung braucht Handwerk..... 80  
 Handwerk allein reicht nicht ..... 82  
 Das Wichtigste in Kürze ..... 83

**Kapitel 7  
Entwickeln ist (kreative) Wissenschaft ..... 85**

Formalisiertes Wissen..... 85  
     Was sind formale Theorien?..... 86  
     Wann braucht es eine (neue) Theorie?..... 87  
 Wie Sie zu einer Theorie kommen?..... 88  
     Mentales Modell als Theorie ..... 88  
     Wenn es so einfach wäre: Viele Hürden ..... 89  
     Und dann auch noch der kleine Rest ..... 90  
 Konsequenzen ..... 91  
     Die Bedeutung des Entwicklers darf nicht unterschätzt werden ..... 91  
     Es werden verschiedene Qualifikationen gebraucht ..... 92  
     Auch die Theorie muss weiterentwickelt werden ..... 93  
 Das Wichtigste in Kürze ..... 94

**Kapitel 8  
Modellierungsdilemma und Entscheidungsmüdigkeit..... 95**

Das Modellierungsdilemma..... 95  
     Was macht ein Modell aus? ..... 95  
     Ein Modell für alle Anforderungen gesucht ..... 96  
     Und wenn es kein umfassendes Modell gibt? ..... 98  
 Entscheiden ermüdet ..... 100  
     Entwickeln heißt entscheiden..... 100  
     Entscheidungskraft optimal nutzen ..... 101  
 Das Wichtigste in Kürze ..... 103

**Kapitel 9  
Fallen vermeiden ..... 105**

Erst mal loslegen ..... 105  
     Agil heißt nicht »kein Konzept« ..... 105  
     Abgrenzung ist alles..... 106  
     Wenn es nicht anders geht..... 107  
 Schön flexibel bleiben..... 108  
     Flexible Programme..... 109  
     Flexibilität bläht auf ..... 109  
     Die Sinnfrage..... 110

## 16 Inhaltsverzeichnis

Modularisierung übertreiben .....	111
Davon verschwindet die Komplexität nicht .....	111
Zerlegung will geübt sein .....	112
Schon wieder: Kosten .....	113
Wachsen im Korsett .....	113
Das Wichtigste in Kürze .....	114

## TEIL III SAUBEREN CODE SCHREIBEN..... 115

### **Kapitel 10 Namen sind nicht Schall und Rauch..... 117**

Benennungen.....	117
Namen versus Bezeichner .....	118
Namen versus Begriffe .....	118
Woher nehmen? .....	120
Lösungsdomäne.....	120
Anwendungsdomäne .....	120
Eigenschaften guter Namen.....	121
Den Sinn vermitteln.....	121
Nicht in die Irre führen .....	121
Sinnvolle Unterschiede .....	122
Verschlüsselungen vermeiden.....	123
Verwendbarkeit .....	124
Klassen und Methoden.....	125
Die Qual der Sprachwahl .....	125
Englisch .....	126
Deutsch .....	126
Keine Empfehlung .....	126
Was zu tun ist.....	127
Das Wichtigste in Kürze .....	127

### **Kapitel 11 Reine Formfrage – Formatierung..... 129**

Das Auge liest mit .....	129
Vertikales Formatieren .....	131
Codelänge.....	131
Vorbild Zeitung.....	131
Vertikale Abstände.....	132
Vertikale Ordnung .....	134
Horizontales Formatieren .....	135
Zeilenlänge .....	135
Horizontale Abstände .....	136
Einrückungen .....	138
Automatische Formatierung .....	139
Vorteile.....	139
Nachteile .....	140
Das Wichtigste in Kürze .....	140

<b>Kapitel 12</b>	
<b>Code zuerst – sind Kommentare nötig? .....</b>	<b>141</b>
Code allein reicht nicht .....	141
Erklärung gesucht .....	141
Das große Missverständnis: Code spricht nur den Computer an. ....	142
Kommentare – hilfreich oder störend? .....	142
Kommentare lügen – oft. ....	143
Sinnvolle Kommentare .....	143
Rechtshinweise .....	143
Unerledigtes .....	143
Klarstellungen und Warnungen .....	144
Algorithmen .....	144
Spezifikationen .....	144
Pragmatisches .....	146
Schlechte Kommentare. ....	147
Nichtssagendes .....	147
Auskommentierter Code. ....	148
Unterschiedliche Sprachen. ....	148
Fehlender Bezug. ....	148
JavaDoc .....	149
Dokumentationen .....	150
Schönheit .....	151
Das Wichtigste in Kürze .....	152
<b>Kapitel 13</b>	
<b>Kleine Schritte – saubere Methoden .....</b>	<b>153</b>
Methoden .....	153
Begriffliche Klärung .....	154
Eigenschaften .....	154
Der Inhalt .....	155
Abstraktion .....	155
Trennung von Bearbeitung und Abfrage .....	155
Testen .....	156
Die Größe .....	156
Eine Aufgabe. ....	156
Zeilenzahl .....	157
Schachtelungsstruktur .....	158
Parameter .....	159
Anzahl .....	160
Stellung .....	162
Parameter vermeiden .....	162
Testen .....	163
Flag-Parameter .....	163
Resultate .....	164
Rückgabewerte .....	164
Rückgabewert null .....	165
Ergebnisparameter .....	166
Rückkehrcodes .....	167

Seiteneffekte .....	167
Auswahlanweisungen .....	168
Alles fließt .....	170
Das Wichtigste in Kürze .....	170

## Kapitel 14

### **Passend schneiden – Schnittstellen ..... 171**

Die Rolle von Schnittstellen .....	171
Mehr als ein Interface .....	171
Isoliert betrachtet .....	172
Im Verbund .....	172
Komponenten .....	173
Interface Segregation .....	174
Schlanke Schnittstellen .....	174
Kohäsion .....	175
Kombination .....	179
Keine Missverständnisse .....	180
Exakte Beschreibung .....	180
Voraussetzungen auführen .....	181
Vollständige Definition .....	182
Tests und Mocks .....	182
Kein Code ohne Fremdcode .....	183
Eine unsichtbare Grenze .....	184
Abhängigkeiten isolieren .....	184
Wie es gehen könnte .....	185
Das Wichtigste in Kürze .....	189

## Kapitel 15

### **Objekte und Datensätze unterscheiden ..... 191**

Was ist ein Objekt? .....	191
Und ein Datensatz? .....	192
Die Praxis .....	193
Die Objekt-Datensatz-Antisymmetrie .....	194
Java und Objektorientierung .....	194
Prozeduraler Code .....	195
Objektorientierter Code .....	197
Schlussfolgerungen .....	199
Das Gesetz von Demeter .....	199
Internes intern halten .....	200
Trotzdem kommunikativ sein .....	200
Das gilt auch umgekehrt .....	202
Aufrufketten .....	203
Fazit .....	205
Das Wichtigste in Kürze .....	205

<b>Kapitel 16</b>	
<b>Wege im Dschungel – Regeln</b> .....	<b>207</b>
Wiederholungen vermeiden .....	207
Die Regel .....	207
Motivation .....	208
Umsetzung .....	209
Schwierigkeiten .....	210
Liefere, was verlangt wird .....	211
Die Regel .....	211
Motivation .....	212
Umsetzung .....	212
Schwierigkeiten .....	213
Jedes für sich .....	213
Die Regel .....	213
Motivation .....	214
Umsetzung .....	214
Schwierigkeiten .....	214
Die SOLID-Regeln .....	215
Single Responsibility Principle – SRP .....	215
Open Closed Principle – OCP .....	216
Liskov Substitution Principle – LSP .....	217
Interface Segregation Principle – ISP .....	217
Dependency Inversion Principle – DIP .....	217
Einfach besser .....	218
Halte es einfach .....	218
Geringste Überraschung .....	219
Fazit .....	219
Das Wichtigste in Kürze .....	219
<b>Kapitel 17</b>	
<b>Fehler passieren – Fehlerbehandlung</b> .....	<b>221</b>
Ausgangslage .....	221
Fehlerarten .....	222
Datenfehler .....	222
Seltene Datenfehler .....	223
Häufige Datenfehler .....	223
Funktionsfehler .....	225
Hardwarefehler .....	226
Semantische Fehler .....	227
Plausibilitätsprüfung .....	227
Wertebereichs-Überschreitungen .....	229
Keine Panik .....	231
Das Wichtigste in Kürze .....	231

**Kapitel 18**  
**Ausnahmen regeln – Exceptions** ..... **233**

- Sinn und Zweck ..... 233
- Checked und Unchecked Exceptions ..... 234
- Kosten ..... 235
- Werfen von Exceptions ..... 235
  - Generische Exceptions verwenden ..... 235
  - Spezielle Exceptions definieren ..... 236
- Fangen von Exceptions ..... 238
  - Funktionsblöcke bestimmen ..... 238
  - Fachliche und technische Exceptions ..... 239
- Verpacken von Exceptions ..... 240
- Loggen von Exceptions ..... 240
- Angemessenheit ..... 241
- Das Wichtigste in Kürze ..... 242

**Kapitel 19**  
**Immer weiter – neue Sprachmittel** ..... **243**

- Wie beurteilen? ..... 243
- Annotationen ..... 245
  - Funktion ..... 246
  - Anwendungsarten ..... 246
  - Risiken minimieren ..... 248
- Lambda-Ausdrücke ..... 248
  - Klippen ..... 248
  - So vielleicht ..... 250
- Streams ..... 251
  - Die Idee ..... 252
  - Anwendung ..... 252
  - Aber Vorsicht ..... 253
- Fazit ..... 254
  - Spezialisierung ..... 254
  - Beschränkung ..... 255
- Das Wichtigste in Kürze ..... 256

**TEIL IV**  
**WEGE ZUM ZIEL** ..... **257**

**Kapitel 20**  
**Miteinander lernen – Code Reviews** ..... **259**

- Zweck ..... 260
- Was nicht geht ..... 260
- Das Potenzial ..... 261
- Durchführung ..... 262
  - Erfolgsvoraussetzungen ..... 262
  - Vorbereitung ..... 263

Review-Rollen .....	265
Review-Werkzeuge und Metriken .....	266
Review-Bewertung .....	267
Codequalität .....	267
Review-Qualität .....	267
Das Wichtigste in Kürze .....	268

## **Kapitel 21**

### **Aus Fehlern lernen ..... 269**

Fehler macht jeder .....	269
Fehler analysieren .....	270
Fehlerursachen ermitteln .....	271
Fehlerarten .....	272
Priorisierung .....	272
Denken Sie an .....	273
Erkenntnisse nutzen .....	275
Code Reviews nutzen .....	275
Ergebnisse dokumentieren .....	275
Wiederholen: Erkenntnisse erneut erörtern .....	275
Das Wichtigste in Kürze .....	276

## **Kapitel 22**

### **Es gibt immer was zu tun – Refactoring ..... 277**

Die Idee .....	277
Die Praxis .....	278
Vorbereitung .....	278
Schritt für Schritt .....	280
Im Großen und im Kleinen .....	280
Ein Beispiel .....	281
Das Wichtigste in Kürze .....	282

## **TEIL V**

### **DER TOP-TEN-TEIL ..... 283**

## **Kapitel 23**

### **10 Fehler, die Sie vermeiden sollten..... 285**

Buch in Schrank stellen .....	285
Nicht sofort anfangen .....	285
Aufgeben .....	286
Nicht streiten .....	286
Schematisch anwenden .....	286
Kompromisse verweigern .....	286
Unrealistische Terminzusagen .....	287
Überheblichkeit .....	287
Denken, fertig zu sein .....	287
Alles tierisch ernst nehmen .....	287

<b>Kapitel 24</b>	
<b>(Mehr als) 10 nützliche Quellen zum Auffrischen und Vertiefen</b>	<b>289</b>
Clean Code – das Buch und der Blog	289
Clean Code Developer	290
Software Craftsmanship	290
Java Code Conventions	290
97 Dinge, die jeder Programmierer wissen sollte	290
The Pragmatic Bookshelf	291
Prinzipien der Softwaretechnik	291
Refactoring	291
Code Reviews	291
Codeanalyse	292
Verzögerungskosten	292
Project Oberon	292
<b>Stichwortverzeichnis</b>	<b>295</b>