Inhalt

1 Verhindern Sie den Weltuntergang! 13

1.1 Vorwort ... 13

1.2 Schöne neue Welt ... 14

1.3 Was läuft falsch? ... 15

1.4 Weltuntergang verhindern – aber wie? ... 17

2 Konventionen 21

2.1 Vereinbarungen im Team ... 21

2.1.1 Erlaubte und verbotene Abweichungen ... 22

2.1.2 IDE, Formatierung und Code Style ... 24

2.2 Wenn die Variable »a« sagt (und sonst nichts) ... 26

2.2.1 Eindeutige Bezeichner ... 26

2.2.2 Richtige Sprache ... 29

2.2.3 // Diese Zeile ist kein Kommentar ... 31

2.2.4 Ungarische Notation ... 31

2.2.5 Groß, klein, CamelCase und diese verflixten case-sensitiven Dateisysteme ... 32

2.3 Code-Fokus ... 34

2.3.1 Zuständig laut Formular: God.class ... 34

2.3.2 Spezialisierte Funktionen ... 35

2.4 Checkliste ... 36

3 Willkommen im Team! 37

3.1 Check this out: Subversion ... 38

3.1.1 Überblick ... 38

3.1.2 Subversion-Server ... 38

3.1.3 Subversion-Clients ... 41

3.1.4 Der Trunk ... 44

3.1.5 Branches und Tags ... 46

3.2 Teamwork integriert: Git 49
3.2.1 Überblick und Historie 49
3.2.2 Von Subversion zu Git 50
3.2.3 Bitbucket oder GitHub 54

3.3 »Guckstu!« 56
3.3.1 Vier Augen sehen mehr als zwei 56
3.3.2 Milchmädchenrechnung 57
3.3.3 Was ist ein Code Review? 58
3.3.4 Ad hoc oder Bürokratie 59

3.4 Doppelt hält besser: Pair Programming 61
3.4.1 Pilot und Co-Pilot 61
3.4.2 Grenzen des Pair Programming 62

3.5 Wer macht wann was? 64
3.5.1 Unter dem Wasserfall 65
3.5.2 Kanban 66
3.5.3 Scrum 69

4 Gut, besser, 91,2%: Software-Qualität messen 73

4.1 Muss funktionieren! 74
4.1.1 Elemente einer Anforderung 74
4.1.2 Bewertung von Anforderungen 75
4.1.3 Systematische Störungen 76

4.2 Muss schön sein! 80
4.2.1 Performance 80
4.2.2 Wartbarkeit 82
4.2.3 Benutzbarkeit 86
4.2.4 Sicherheit 87
4.2.5 Skalierbarkeit 93
4.2.6 Portierbarkeit und Kompatibilität 96

4.3 ISO 25010 und andere Buzzword-Sammlungen 99
4.3.1 ISO 25010 99
4.3.2 IEEE 730 und andere 103

5 Jeder ist Architekt 105

5.1 Normalisierte Daten ... 105
5.1.1 Einstellungen in der Datenbank ... 106
5.1.2 Eine Geld-Klasse ... 108
5.1.3 Zu spät! ... 109

5.2 Alles ist ein Objekt, aber welches? ... 114
5.2.1 OOP-Paradigmen ... 115
5.2.2 POJOs und DTOs ... 117

5.3 Entwurfsmuster ... 118
5.3.1 Fabrikmethoden/Fabrik-Klassen (»factory method«/»factory class«) ... 119
5.3.2 Singleton ... 120
5.3.3 Erbauer (»builder«) ... 122
5.3.4 Adapter (»wrapper«) ... 124
5.3.5 Brücke und Proxy (»bridge«/»proxy«) ... 126
5.3.6 Beobachter (»observer«, »listener«, »publisher«, »subscriber«) ... 128
5.3.7 Besucher (»visitor«) ... 129
5.3.8 Iterator (»cursor«) ... 132
5.3.9 Befehl (»command«) ... 133
5.3.10 Zustand (»state«) ... 135

5.4 Was ist eigentlich ein »Item«? ... 138
5.4.1 Hierarchische Datenmodelle ... 139
5.4.2 Dokumentdatenbanken ... 141
5.4.3 Domänenspezifische Sprachen ... 145

5.5 Do- und Don't-Merksatz-Akronyme ... 148
5.5.1 KISS ... 148
5.5.2 POITROAE ... 149
5.5.3 YAGNI ... 150
5.5.4 SMART ... 151
5.5.5 SOLID ... 152
5.5.6 CRUD ... 152

5.6 Neue Räder extra teuer! ... 153
5.6.1 Universal-Bibliotheken ... 154
5.6.2 Spezial-Bibliotheken ... 155
5.6.3 Veraltet oder stabil wie ein Fels? ... 156

5.7 Meins! (Wirklich?) ... 158
5.7.1 GNU General Public License ... 159

5.7.2 Apache-Lizenz 2.0 ... 159
5.7.3 MIT-Lizenz ... 160
5.7.4 BSD-Lizenz ... 160

6 Erst mal testen 163

6.1 Gute und schlechte Unit-Tests ... 164
6.1.1 Einfache Unit-Tests ... 164
6.1.2 Whitebox-Tests ... 169
6.1.3 Ping-Pong ... 173
6.1.4 Testabdeckung ... 175

6.2 Testbar und nicht so gut testbar ... 177
6.2.1 Getrieben von Tests ... 177
6.2.2 Gut testbar ... 178
6.2.3 Nicht so gut testbar ... 179
6.2.4 Unmöglich testbar ... 181

6.3 Umgekehrt wird ein Schuh draus ... 185
6.3.1 Inversion of Control ... 185
6.3.2 Dependency Injection mit Spring Boot ... 186

6.4 Alles einzeln testen ... 190
6.4.1 Unit-Tests mit JMockit ... 190

6.5 Millionen Mausklicks ... 195
6.5.1 UI-Tests mit Selenium ... 195
6.5.2 UI-Tests unter Android ... 198

7 Continuous Integration 203

7.1 Digitaler Bauunternehmer ... 203

7.2 Java-Builds mit Maven ... 205
7.2.1 Dependency Management via Maven Central ... 205
7.2.2 Dependency Scopes ... 209
7.2.3 Applikationspakete bauen ... 212
7.2.4 Empfehlenswerte Maven-Plug-Ins ... 216

7.3 Gradle en vogue ... 219
7.3.1 Gradle vs. Maven ... 219
7.3.2 Hilfreiche Gradle-Plug-Ins ... 222

7.3.3 Gradle und Android Studio ... 224

7.4 Jenkins, stets zu Ihren Diensten! ... 225

7.4.1 Jenkins einrichten ... 225

7.4.2 Ein Jenkins-Projekt konfigurieren ... 226

7.4.3 Jenkins-Plug-Ins für jeden Zweck ... 228

7.5 Nicht nur eine Frage des Stils ... 231

7.5.1 Checkstyle ... 231

7.5.2 FindBugs ... 232

7.6 NuGet für .NET und MS Azure ... 234

7.6.1 Abhängigkeiten verwalten mit NuGet ... 234

7.6.2 Eigene NuGet-Pakete erzeugen ... 236

7.6.3 Entwickeln in der Cloud mit Azure ... 237

8 Dokumentation, Kommentare & Tools 243

8.1 Kommentare sind wie Tooltips ... 243

8.1.1 Notwendige und unnötige Kommentare ... 244

8.1.2 Witzige Kommentare ... 246

8.1.3 Wann und wo? ... 247

8.2 Dokumentiert sich von allein ... 248

8.2.1 Javadoc ... 248

8.2.2 Doxygen ... 251

8.2.3 Visual Studio ... 252

8.2.4 Spezielle Kommentare ... 253

8.3 Teamwork online ... 254

8.3.1 Trac ... 255

8.3.2 Redmine ... 259

8.3.3 JIRA und Confluence ... 263

8.3.4 Team Foundation Server ... 266

9 Betriebssicherheit 269

9.1 »Es ist ein Fehler aufgetreten. Versuchen Sie es noch einmal.« ... 270

9.1.1 Fehlercodes ... 271

9.1.2 Ausnahmen richtig behandeln ... 273

9.1.3 Aussagekräftige Fehlermeldungen ... 278
9.1.4 Systematische Fehlersuche ... 279

9.2 Festplattenweise Protokolle ... 282
9.2.1 Logging-Frameworks ... 283
9.2.2 Log-Levels ... 285
9.2.3 Der langsamste Weg, nichts zu loggen ... 285
9.2.4 Rotation und Konfiguration ... 286
9.2.5 Schnitzeljagd ... 290

9.3 Ungebetene Besucher ... 292
9.3.1 Spurensuche ... 293
9.3.2 Alle Luken dicht ... 294
9.3.3 Starke Kryptografie ... 296
9.3.4 Elliptische Kurven ... 298
9.3.5 Rollen und Rechte ... 301
9.3.6 Code Injection verhindern ... 304
9.3.7 Hacker-Tools ... 305

10 Schrottcode pimpen 307

10.1 Was macht der da? ... 307
10.1.1 Know-how abgreifen ... 308
10.1.2 Code-Bestandsaufnahme ... 310

10.2 Refactoring mit Tools ... 312
10.2.1 Methoden extrahieren ... 312
10.2.2 Klassen extrahieren ... 315
10.2.3 Parameterobjekte ... 317
10.2.4 Interfaces extrahieren ... 320
10.2.5 Weitere Refactoring-Maßnahmen ... 322

10.3 Who sprech Svenska? ... 323
10.3.1 HTML-Templates ... 324
10.3.2 Datenbankschicht abtrennen ... 325

10.4 Endlich: Tests ... 327
10.4.1 Testfälle identifizieren ... 328
10.4.2 Module mocken ... 329
10.4.3 Schrittweise zu höherer Testabdeckung ... 330

11 Trollfütterung 333

11.1 Umsteiger und Ahnungslose im kalten Wasser ... 333
11.1.1 Willkommen im Kotlin-Land! ... 334
11.1.2 Frustration frisst Freude ... 335
11.1.3 Verantwortung delegieren, nicht Aufgaben ... 336

11.2 Früher war alles besser, auch die Betonköpfe ... 336
11.2.1 Ein weitsichtiger Boss ... 336
11.2.2 Früher waren Bücher noch aus Papier ... 337
11.2.3 Sicherheit und Transparenz ... 338

11.3 Das Patchwork-Team ... 338
11.3.1 Chris schießt quer ... 339
11.3.2 Reden ist Gold ... 340
11.3.3 Anerkennung und Kritik ... 341

11.4 Billig im Osten ... 341
11.4.1 Bitte recht freundlich! ... 341
11.4.2 Differenzen ... 343
11.4.3 Integration ... 344

11.5 Der Hase der Produktmanagerin ... 345
11.5.1 Störfaktoren auf dem Schreibtisch ... 345
11.5.2 Hase und Igel ... 346
11.5.3 Toleranz und Grenzen ... 347

11.6 Arbeiten wie die Profis ... 348
11.6.1 Überflieger ... 348
11.6.2 Diagnose: Overperformer ... 349
11.6.3 Keep it simple, Felix! ... 349

11.7 Leuchtendes Beispiel ... 350
11.7.1 Niemand mag Besserwisser ... 350
11.7.2 Diagnose: das engagierte Vorbild ... 351
11.7.3 Was nun? ... 351

12 Parallelwelten 353

12.1 Parallel arbeiten ... 353
12.1.1 Threads und ThreadPools ... 354
12.1.2 Race Conditions ... 356

12.1.3 Synchronisierte Zugriffe 358
12.1.4 Warten macht keinen Spaß 358
12.1.5 Deadlocks 362

12.2 Losgelöst 365
12.2.1 Publisher und Subscriber 366
12.2.2 EventBus im Einsatz 367

12.3 .NET async 369
12.3.1 Das »async«-Sprachelement 370
12.3.2 Locks 371

Anhang

A Quizfragen 373

B Lösungen der Quizfragen 379

Index 383