

Geleitwort	13
Vorwort	15
1 Clean Code	19
Was ist ein Code-Smell?	19
Was ist Refactoring?	20
Was ist ein Rezept?	20
Warum Clean Code?	21
Lesbarkeit, Performance – oder beides?	21
Arten von Software	21
Maschinengenerierter Code	22
Hinweise zu verwendeten Begriffen	22
Entwurfsmuster	22
Paradigmen der Programmiersprachen	23
Objekte versus Klassen	23
Veränderbarkeit	23
2 Festlegung der Axiome	25
Einführung	25
Warum ist es ein Modell?	26
Warum ist es abstrakt?	26
Warum ist es programmierbar?	26
Warum ist es partiell?	27
Warum ist es erklärend?	27
Wieso geht es um Realität?	27
Ableitung der Regeln	28
Das einzig wahre Entwurfsprinzip für Software	28

3	Anämische Modelle	35
	Einführung	35
	Anämische Objekte in Rich Objects konvertieren	36
	Das Wesentliche Ihrer Objekte erkennen	37
	Objekte von Settern befreien	39
	Auf Generatoren verzichten, die anämischen Code produzieren	41
	Automatische Eigenschaften entfernen	42
	DTOs entfernen	44
	Leere Konstruktoren vervollständigen	46
	Getter entfernen	48
	Objektorgie verhindern	50
	Dynamische Eigenschaften entfernen	52
4	Primitive Obsession	55
	Einführung	55
	Small Objects erstellen	56
	Primitive Datentypen in Objekte umwandeln	57
	Assoziative Arrays in Objekte umwandeln	58
	Keine Strings missbrauchen	60
	Zeitstempel in Sequenzierung umwandeln	61
	Teilmengen als Objekte modellieren	62
	String-Validierungen in Objekte umwandeln	63
	Unnötige Eigenschaften entfernen	66
	Datumsintervalle berechnen	68
5	Mutabilität	71
	Einführung	71
	Variablen von var in const ändern	73
	Variablen als veränderlich deklarieren	74
	Veränderungen der Objektessenz verbieten	76
	Veränderliche const-Arrays vermeiden	77
	Lazy Initialization entfernen	78
	Veränderliche Konstanten einfrieren	80
	Seiteneffekte beseitigen	82
	Hoisting verhindern	83
6	Deklarativer Code	85
	Einführung	85
	Geltungsbereich wiederverwendeter Variablen begrenzen	86
	Code durch Aufteilung in Funktionen strukturieren	87
	Versionierte Methoden entfernen	88
	Doppelte Verneinungen entfernen	89
	Falsch zugeordnete Verantwortlichkeiten verschieben	90

Explizite Iterationen ersetzen	92
Entwurfsentscheidungen dokumentieren	93
Magische Zahlen durch Konstanten ersetzen	94
»Was« und »Wie« trennen	95
Reguläre Ausdrücke dokumentieren	96
Yoda-Conditions neu formulieren	97
Scherzhaft benannte Methoden umbenennen	98
Callback Hell vermeiden	98
Gute Fehlermeldungen formulieren	100
Magische Korrekturen vermeiden	102
7 Namensgebung	105
Einführung	105
Abkürzungen ausschreiben	105
Hilfsfunktionen und -klassen umbenennen und aufteilen	107
myObjects umbenennen	110
Ergebnisvariablen umbenennen	111
Variablen umbenennen, die nach Typen benannt sind	112
Zu lange Namen kürzen	114
Abstrakte Namen ändern	115
Rechtschreibfehler korrigieren	116
Klassennamen aus Attributen entfernen	116
Vorangestellte Buchstaben aus Namen von Klassen und Interfaces entfernen	117
Basic-/Do-Funktionen umbenennen	118
Mit Pluralformen benannte Klassen auf Singularform ändern	120
»Collection« aus Namen entfernen	120
Präfix/Suffix »Impl« aus Klassennamen entfernen	121
Argumente je nach Rolle bzw. Aufgabe benennen	122
Redundante Parameternamen ändern	123
Überflüssigen Kontext aus Namen entfernen	124
Benennung als »data« vermeiden	126
8 Kommentare	127
Einführung	127
Kommentierten Code entfernen	127
Veraltete Kommentare entfernen	129
Temporäre logische Steuerungsanweisungen entfernen	130
Getter-Kommentare entfernen	132
Kommentare in Funktionsnamen umwandeln	133
Kommentare innerhalb von Methoden entfernen	134
Kommentare durch Tests ersetzen	136

9	Standards	139
	Einführung	139
	Codestandards befolgen	139
	Einrückungen standardisieren	142
	Schreibweisen vereinheitlichen	143
	Code auf Englisch schreiben	144
	Reihenfolge von Parametern vereinheitlichen	145
	Kleine Mängel beheben	146
10	Komplexität	149
	Einführung	149
	Wiederholten Code entfernen	149
	Einstellungen/Konfigurationen und Funktionsumschalter entfernen	151
	Zustand über Eigenschaften ändern	153
	Übertriebene Raffinesse aus dem Code entfernen	155
	Mehrere Promises parallel auflösen	156
	Lange Kollaborationsketten auflösen	157
	Methode in ein Objekt extrahieren	159
	Array-Konstruktoren überprüfen	161
	Poltergeist-Objekte entfernen	162
11	Aufgeblähter Code	165
	Einführung	165
	Überlange Methoden unterteilen	165
	Überflüssige Argumente reduzieren	167
	Überflüssige Variablen reduzieren	168
	Überflüssige Klammern entfernen	170
	Überflüssige Methoden entfernen	171
	Überzählige Attribute gruppieren	172
	Importlisten kürzen	174
	Funktionen mit mehreren Aufgaben aufteilen	175
	Überladene Interfaces verschlanken	176
12	YAGNI-Prinzip	179
	Einführung	179
	Toten Code entfernen	179
	Code anstelle von Diagrammen verwenden	181
	Refactoring von Klassen mit nur einer Unterklasse	183
	Interfaces entfernen, die nur an einer Stelle genutzt werden	184
	Missbräuchlich verwendete Entwurfsmuster entfernen	185
	Geschäftsspezifische Collections ersetzen	186

13	Fail-Fast-Prinzip	189
	Einführung	189
	Neuzuweisung von Variablen refaktorisieren	189
	Vorbedingungen durchsetzen	191
	Striktere Parameter verwenden	193
	Standardfall bei Switch-Anweisungen entfernen	194
	Beim Iterieren über Collections Änderungen vermeiden	196
	Hash und Gleichheit neu definieren	197
	Refactoring von funktionalen Änderungen trennen	198
14	If-Anweisungen	201
	Einführung	201
	Akzidentelle If-Anweisungen durch Polymorphie ersetzen	202
	Flag-Variablen für Ereignisse deklarativ umbenennen	208
	Boolesche Variablen reifizieren	209
	Switch-/Case-/Elseif-Anweisungen ersetzen	211
	Hartcodierte Bedingungen durch Collections ersetzen	213
	Boolesche Bedingungen in Kurzschluss-Auswertungen umwandeln	214
	Implizites Else in explizites If umwandeln	215
	Verschachtelte Bedingungen refaktorisieren	216
	Short-Circuit-Hacks vermeiden	218
	Verschachtelten Pfeilcode refaktorisieren	219
	Rückgabe boolescher Werte für Bedingungsprüfungen vermeiden	220
	Vergleiche mit booleschen Werten ändern	222
	Ternäre Ausdrücke vereinfachen	223
	Nicht-polymorphe Funktionen in polymorphe umwandeln	225
	Gleichheitsvergleich ändern	226
	Hartcodierte Geschäftsbedingungen reifizieren	227
	Überflüssige boolesche Operatoren entfernen	228
	Verschachtelte ternäre Ausdrücke refaktorisieren	229
15	Nullwerte	231
	Einführung	231
	Nullobjekte erstellen	231
	Optionale Verkettungen entfernen	234
	Optionale Attribute in eine Collection umwandeln	236
	Reale Objekte für Nullwerte verwenden	238
	Darstellung unbekannter Orte ohne Verwendung von null	241
16	Vorzeitige Optimierung	245
	Einführung	245
	IDs für Objekte vermeiden	246
	Vorzeitige Optimierungen entfernen	248

Bitweise vorzeitige Optimierungen entfernen	250
Übergeneralisierung reduzieren	251
Strukturelle Optimierungen ändern	252
»Boat Anchors« beseitigen	253
Caches aus Domänenobjekten extrahieren	255
Auf der Implementierung basierende Callback-Events entfernen	257
Abfragen aus Konstruktoren entfernen	258
Code aus Destrukturen entfernen	259
17 Kopplung	263
Einführung	263
Versteckte Annahmen explizit machen	263
Singletons ersetzen	265
God Objects aufspalten	268
Klassen bei divergenten Änderungen teilen	270
Spezielle als Flags genutzte Werte (wie 9999) in normale Werte umwandeln	271
Shotgun Surgery vermeiden	273
Optionale Argumente entfernen	275
Feature Envy vorbeugen	276
Vermittlerobjekte entfernen	278
Standardargumente ans Ende verschieben	279
Ripple-Effekt vermeiden	281
Zufällige Methoden aus Geschäftsobjekten entfernen	282
Geschäftslogik aus der Benutzeroberfläche entfernen	284
Kopplung an Klassen verringern	287
Datenklumpen beseitigen	289
Unangemessene Intimität auflösen	290
Fungible Objekte konvertieren	292
18 Globals	295
Einführung	295
Globale Funktionen reifizieren	295
Statische Funktionen reifizieren	296
Goto-Anweisungen durch strukturierten Code ersetzen	298
Globale Klassen entfernen	299
Globale Datumserstellung anpassen	301
19 Hierarchien	303
Einführung	303
Tiefe Vererbungshierarchien verflachen	303
Jo-Jo-Hierarchien durchbrechen	306
Subklassifizierung zur Wiederverwendung von Code auflösen	307
»Ist-ein«-Beziehung durch Verhalten ersetzen	309

Verschachtelte Klassen entfernen	311
Isolierte Klassen umbenennen	313
Konkrete Klassen als final deklarieren	314
Klassenvererbung explizit definieren	316
Klassen ohne Verhalten entfernen	317
Keine vorzeitige Klassenbildung vornehmen	318
Geschützte Attribute entfernen	320
Leere Implementierungen vervollständigen	322
20 Testen	325
Einführung	325
Private Methoden testen	326
Beschreibungen zu Assertions hinzufügen	327
assertTrue in spezifischere Assertions konvertieren	329
Mock-Objekte durch echte Objekte ersetzen	330
Generische Assertions verfeinern	332
Unzuverlässige Tests entfernen	333
Vergleiche von Gleitkommazahlen vermeiden	335
Realistische Daten statt Testdaten verwenden	336
Verletzung der Kapselung vermeiden	338
Irrelevante Testinformationen entfernen	340
Keine Pull Requests ohne Testabdeckung zulassen	341
Tests umformulieren, die von Datumswerten abhängen	343
Eine neue Programmiersprache lernen	344
21 Technische Schulden	345
Einführung	345
Produktionsabhängigen Code entfernen	346
Fehlertracker entfernen	347
Warnungen/Strict-Modus nicht ausschalten	349
ToDos und FixMes verhindern und entfernen	350
22 Ausnahmen	353
Einführung	353
Leere Ausnahmeblöcke entfernen	353
Unnötige Ausnahmen entfernen	354
Keine Ausnahmen bei erwarteten Fällen auslösen	356
Verschachtelte Try/Catch-Blöcke vereinfachen	358
Rückgabecodes durch Ausnahmen ersetzen	359
Verschachtelten Pfeilcode refaktorisieren	361
Low-Level-Fehler vor Endbenutzern verstecken	362
Try-Blöcke kurz halten	363

23 Metaprogrammierung	365
Einführung	365
Metaprogrammierung entfernen	365
Anonyme Funktionen reifizieren	369
Auf Präprozessoren verzichten	371
Dynamische Methoden entfernen	372
24 Datentypen	375
Einführung	375
Typprüfungen entfernen	375
Mit truthy-Werten umgehen	377
Gleitkommazahlen in Dezimalzahlen konvertieren	380
25 Sicherheit	383
Einführung	383
Benutzereingaben bereinigen	383
Sequenzielle IDs ändern	385
Paketabhängigkeiten entfernen	386
Problematische reguläre Ausdrücke ersetzen	388
Sichere Deserialisierung von Objekten	389
Glossar	391
Index	405