

Joshua Kerievsky

Refactoring to Patterns



ADDISON-WESLEY

An imprint of Pearson Education

München • Boston • San Francisco • Harlow, England
Don Mills, Ontario • Sydney • Mexico City
Madrid • Amsterdam

Inhalt

Vorwort	13
Vorwort von Martin Fowler	15
Einführung	17
Worum geht es in diesem Buch?	17
Was soll dieses Buch bewirken?	17
E= Wer sollte dieses Buch lesen?	18
E= Welchen Hintergrund sollten Sie mitbringen?	18
E= Wie Sie dieses Buch verwenden sollten	19
E= Die Geschichte hinter diesem Buch	20
E= Auf den Schultern von Riesen	21
E= Danksagungen	21
I Warum ich dieses Buch geschrieben habe	25
1.1 Over-Engineering	25
1.2 Patterns als Wundermittel	26
1.3 Under-Engineering	27
1.4 Testgesteuerte Entwicklung und kontinuierliches Refactoring	29
1.5 Refactoring und Patterns	30
1.6 Evolutionäres Design	32
2 Refactoring	33
2.1 Refactoring – Was ist das?	33
2.2 Welche Gründe gibt es für Refactoring?	34
2.3 Viele Augen sehen mehr	35
2.4 Verständlicher Code	36
2.5 Sauberer Code	38
2.6 Kleine Schritte	38
2.7 Designschulden	40

2.8	Eine neue Architektur entwickeln	41
2.9	Zusammengesetzte und testgesteuerte Refactorings	42
2.10	Die Vorteile von zusammengesetzten Refactorings	44
2.11	Werkzeuge für das Refactoring	45
3	Patterns	47
3.1	Was ist ein Pattern?	47
3.2	Patternoholics	48
3.3	Viele Wege führen zum Pattern	50
3.4	Refactoring in verschiedene Richtungen	53
3.5	Führen Patterns zu kompliziertem Code?	56
3.6	Pattern-Wissen ist Macht	57
3.7	Direktes Design mit Patterns	58
4	Der »Geruch des Codes	61
4.1	Redundanter Code	63
4.2	Lange Methoden	64
4.3	Komplexe Bedingungen	65
4.4	Neigung zu elementaren Typen	66
4.5	Öffentlichkeit als Anstoßigkeit	67
4.6	Unkontrollierte Vermehrung von Lösungen	67
4.7	Alternative Klassen mit unterschiedlichen Schnittstellen	68
4.8	Faule Klasse	68
4.9	Große Klasse	68
4.10	Switch-Anweisungen	69
4.11	Kombinatorische Vermehrung	69
4.12	Überflüssige Lösungen	70
5	Der Refactoring-Katalog	71
5.1	Das Format der Refactorings	71
5.2	Die Projekte in diesem Katalog	73
5.2.1	XML-Builder	74
5.2.2	HTML-Parser	74
5.2.3	Hypothekenrechner	75
5.3	Ein Ausgangspunkt	75
5.4	Reihenfolge für das Selbststudium	76
6	Erzeugung	79
6.1	Konstruktoren durch Erzeugungsmethoden ersetzen	80
6.1.1	Motivation	81
6.1.2	Vorgehensweise	83

6.1.3	Beispiel	84
6.1.4	Varianten	90
6.2	Die Erzeugung der Fabrik überlassen	91
6.2.1	Motivation	92
6.2.2	Vorgehensweise	96
6.2.3	Beispiel	97
6.3	Klassen durch Fabrik kapseln	104
6.3.1	Motivation	104
6.3.2	Vorgehensweise	106
6.3.3	Beispiel	107
6.3.4	Varianten	110
6.4	Polymorphe Erzeugung mit Fabrik-Methoden einführen	112
6.4.1	Motivation	112
6.4.2	Vorgehensweise	114
6.4.3	Beispiel	116
6.5	Kompositum durch Erbauer kapseln	120
6.5.1	Vorgehensweise	123
6.5.2	Beispiel	124
6.5.3	Varianten	136
6.6	Singleton inline stellen	139
6.6.1	Motivation	140
6.6.2	Vorgehensweise	142
6.6.3	Beispiel	143

7	Vereinfachung	147
7.1	Methode komponieren	148
7.1.1	Motivation	148
7.1.2	Vorgehensweise	150
7.1.3	Beispiel	151
7.2	Bedingte Logik durch Strategie ersetzen	153
7.2.1	Motivation	154
7.2.2	Vorgehensweise	156
7.2.3	Beispiel	158
7.3	Ausschmückungen einem Dekorierer überlassen	169
7.3.1	Motivation	170
7.3.2	Vorgehensweise	174
7.3.3	Beispiel	176
7.4	Zustandsverändernde Bedingungen durch den Zustand ersetzen	193
7.4.1	Motivation	193
7.4.2	Vorgehensweise	195
7.4.3	Beispiel	196
7.5	Implizite Bäume durch Kompositum ersetzen	205
7.5.1	Motivation	206
7.5.2	Vorgehensweise	208
7.5.3	Beispiel	210

7.6	Bedingte Verteiler durch Befehl ersetzen	218
7.6.1	Motivation	219
7.6.2	Vorgehensweise	220
7.6.3	Beispiel	222
8	Generalisierung	231
8.1	Template-Methode bilden	232
8.1.1	Motivation	232
8.1.2	Vorgehensweise	235
8.1.3	Beispiel	235
8.2	Kompositum extrahieren	240
8.2.1	Motivation	240
8.2.2	Vorgehensweise	242
8.2.3	Beispiel	243
8.3	1/n-Unterscheidungen durch Kompositum ersetzen	250
8.3.1	Motivation	250
8.3.2	Vorgehensweise	253
8.3.3	Beispiel	254
8.4	Hartcodierte Benachrichtigungen durch Beobachter ersetzen	262
8.4.1	Motivation	263
8.4.2	Vorgehensweise	265
8.4.3	Beispiel	266
8.5	Schnittstellen durch Adapter vereinheitlichen	273
8.5.1	Motivation	273
8.5.2	Vorgehensweise	275
8.5.3	Beispiel	277
8.6	Adapter extrahieren	284
8.6.1	Motivation	284
8.6.2	Vorgehensweise	286
8.6.3	Beispiel	287
8.6.4	Varianten	294
8.7	Implizite Sprache durch Interpreter ersetzen	294
8.7.1	Motivation	294
8.7.2	Vorgehensweise	297
8.7.3	Beispiel	299
9	Schutz	311
9.1	Typenschlüssel durch Klasse ersetzen	312
9.1.1	Motivation	313
9.1.2	Vorgehensweise	314
9.1.3	Beispiel	316
9.2	Instanziierung durch Singleton begrenzen	322
9.2.1	Motivation	323
9.2.2	Vorgehensweise	324
9.2.3	Beispiel	325

9.3	Das NULL-Objekt einführen	327
9.3.1	Motivation	327
9.3.2	Vorgehensweise	330
9.3.3	Beispiel	331
10	Akkumulation	337
10.1	Akkumulation einem Sammelparameter überlassen	338
10.1.1	Motivation	338
10.1.2	Vorgehensweise	340
10.1.3	Beispiel	341
10.2	Akkumulation einem Besucher überlassen	345
10.2.1	Motivation	345
10.2.2	Vorgehensweise	350
10.2.3	Beispiel	355
11	Hilfsprogramme	365
11.1	Konstruktoren verketten	365
11.1.1	Motivation	367
11.1.2	Vorgehensweise	367
11.1.3	Beispiel	367
11.2	Schnittstellen vereinheitlichen	369
11.2.1	Motivation	369
11.2.2	Vorgehensweise	371
11.2.3	Beispiel	371
11.3	Parameter extrahieren	371
11.3.1	Motivation	372
11.3.2	Vorgehensweise	372
11.3.3	Beispiel	373
Nachwort		375
Bibliografie		377
Index		379