

Kent Beck

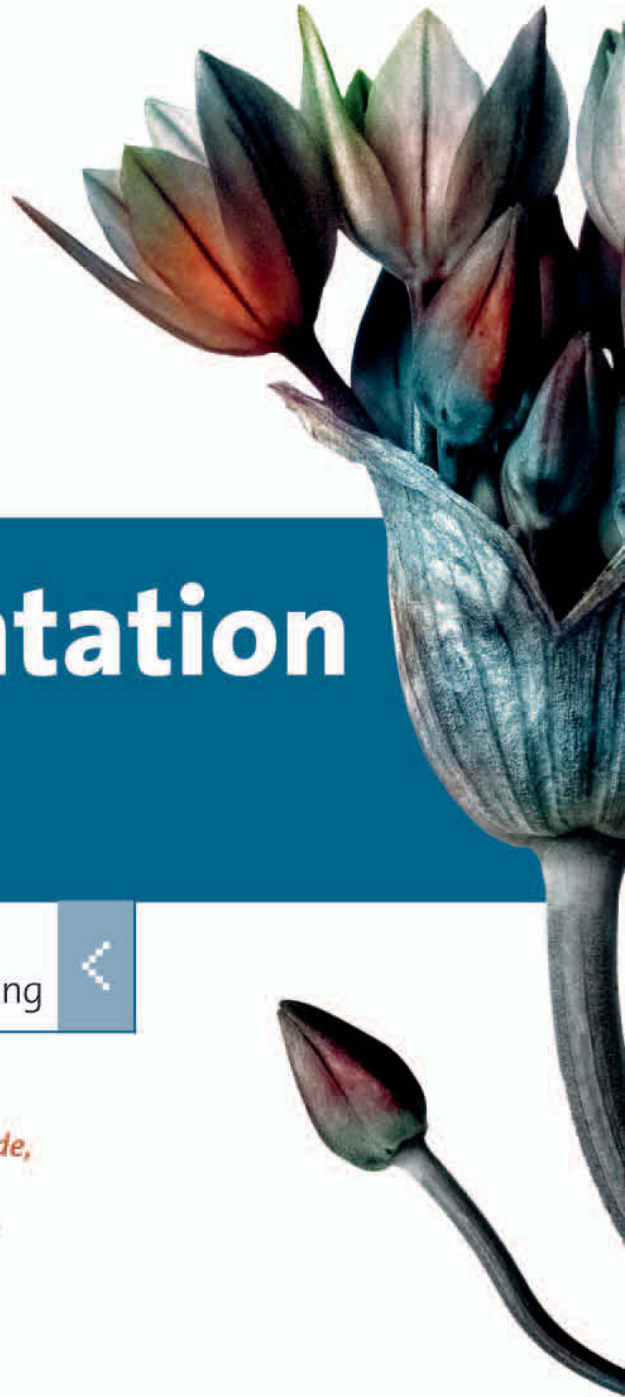
Implementation Patterns

Der Weg zu einfacherer und
kostengünstigerer Programmierung



*»Kent ist ein Meister im Erstellen von Code,
der sich selbst gut dokumentiert, leicht
verständlich ist und sich mit Vergnügen
lesen lässt.«*

Erich Gamma, IBM Distinguished Engineer



Inhalt

Vorwort	9
Danksagung	11
1 Einführung	13
1.1 Tourführer	16
1.2 Und jetzt	17
2 Patterns	19
3 Eine Theorie der Programmierung	23
3.1 Werte	24
3.1.1 Kommunikation	24
3.1.2 Einfachheit	26
3.1.3 Flexibilität	27
3.2 Prinzipien	28
3.2.1 Lokale Konsequenzen	28
3.2.2 Minimale Wiederholung	29
3.2.3 Logik und Daten zusammenhalten	29
3.2.4 Symmetrie	30
3.2.5 Deklarative Ausdrucksform	31
3.2.6 Änderungsgeschwindigkeit	32
3.2.7 Zum Schluss	33
4 Motivation	35
5 Klasse	39
5.1 Klasse	40
5.2 Einfacher Superklassenname	41
5.3 Qualifizierter Subklassenname	42
5.4 Abstraktes Interface	43
5.5 Interface	45
5.6 Abstrakte Klasse	46
5.7 Versioniertes Interface	47
5.8 Wertobjekt	48
5.9 Spezialisierung	51
5.10 Subklasse	52
5.11 Implementierer	54
5.12 Innere Klasse	55
5.13 Instanzspezifisches Verhalten	56
5.14 Bedingungsanweisungen	56

5.15	Delegierung	58
5.16	Pluggable Selektor	60
5.17	Anonyme innere Klasse	62
5.18	Bibliotheksklasse	62
5.19	Zum Schluss	63
6	Zustand	65
6.1	Zustand	66
6.2	Zugriff	68
6.3	Direkter Zugriff	68
6.4	Indirekter Zugriff	69
6.5	Gemeinsamer Zustand	70
6.6	Variabler Zustand	71
6.7	Extrinsischer Zustand	73
6.8	Variable	73
6.9	Lokale Variable	75
6.10	Feld	76
6.11	Parameter	77
6.12	Sammelparameter	80
6.13	Optionalen Parameter	81
6.14	Parameterliste variabler Länge (Var Args)	81
6.15	Parameterobjekt	82
6.16	Konstante	83
6.17	Rollenerklärender Name	83
6.18	Deklariertes Typ	85
6.19	Initialisierung	86
6.20	Frühe Initialisierung	86
6.21	Späte Initialisierung	87
6.22	Zum Schluss	88
7	Verhalten	89
7.1	Control Flow	90
7.2	Main Flow	91
7.3	Message	91
7.4	Choosing Message	92
7.5	Double Dispatch	93
7.6	Decomposing (Sequencing) Message	94
7.7	Reversing Message	94
7.8	Inviting Message	96
7.9	Explaining Message	96
7.9.1	Exceptional Flow	97
7.10	Guard Clause	97
7.11	Exception	99
7.12	Checked Exceptions	100

7.13	Exception Propagation	101
7.14	Zum Schluss	101
8	Methoden	103
8.1	Komponierte Methode	106
8.2	Absichtsorientierter Name	108
8.3	Methodensichtbarkeit	109
8.4	Methodenobjekt	111
8.5	Überschriebene Methode	113
8.6	Überladene Methode	113
8.7	Methodenrückgabetyyp	114
8.8	Methodenkommentar	115
8.9	Hilfsmethode	115
8.10	Debug Print-Methode	116
8.11	Konvertierung	117
8.12	Konvertierungsmethode	118
8.13	Konvertierungskonstruktor	118
8.14	Erstellung	119
8.15	Vollständiger Konstruktor	120
8.16	Factory-Methode	121
8.17	Interne Factory	121
8.18	Collection-Zugriffsmethode	122
8.19	Boolesche Set-Methode	123
8.20	Abfragemethode	124
8.21	Gleichheitsmethode	125
8.22	Get-Methode	126
8.23	Set-Methode	127
8.24	Sichere Kopie	128
8.25	Zum Schluss	129
9	Collections	131
9.1	Metaphern	132
9.2	Relevante Sachverhalte	134
9.3	Interfaces	135
9.3.1	Array	136
9.3.2	Iterable	136
9.3.3	Collection	137
9.3.4	List	137
9.3.5	Set	138
9.3.6	SortedSet	138
9.3.7	Map	139
9.4	Implementierungen	140
9.4.1	Collection	141
9.4.2	List	142

9.4.3	Set	143
9.4.4	Map	144
9.5	Collections	144
9.5.1	Suchen	145
9.5.2	Sortieren	146
9.5.3	Nicht änderbare Collections	146
9.5.4	Collections mit einem Element	147
9.5.5	Leere Collections	147
9.6	Collections erweitern	147
9.7	Zum Schluss	148
10	Frameworks entwickeln	149
10.1	Frameworks ändern ohne Anwendungen zu ändern	150
10.2	Inkompatible Upgrades	151
10.3	Kompatible Änderungen begünstigen	153
10.3.1	Bibliotheksklasse	154
10.3.2	Objekte	154
10.4	Zum Schluss	163
A	Laufzeitmessung	165
A.1	Beispiel	166
A.2	API	167
A.3	Implementierung	168
A.4	MethodTimer	168
A.5	Overhead beseitigen	171
A.6	Tests	171
A.6.1	Collections vergleichen	171
A.6.2	ArrayList und LinkedList vergleichen	174
A.6.3	Sets vergleichen	175
A.6.4	Maps vergleichen	176
A.7	Zum Schluss	177
B	Quellen	179
B.1	Allgemeine Programmierung	179
B.2	Philosophie	181
B.3	Java	182
Index	183