

Inhaltsverzeichnis

	Vorwort	15
	Einführung	21
	Danksagungen	25
	Über den Autor	29
	Auf dem Titelbild	31
	Unverzichtbare Einführung	33
I	Professionalität	39
1.1	Seien Sie vorsichtig, wonach Ihnen verlangt	39
1.2	Verantwortung übernehmen	40
1.3	Erstens: Richte keinen Schaden an	42
1.3.1	Beschädige nicht die Funktion	42
1.3.2	Beschädige nicht die Struktur	45
1.4	Arbeitsethik	47
1.4.1	Sie sollten sich in Ihrem Bereich auskennen	48
1.4.2	Lebenslanges Lernen	49
1.4.3	Praxis	50
1.4.4	Teamwork	51
1.4.5	Mentorenarbeit	51
1.4.6	Sie sollten sich in Ihrem Arbeitsgebiet auskennen	51
1.4.7	Identifizieren Sie sich mit Ihrem Arbeitgeber bzw. Kunden	52
1.4.8	Bescheidenheit	52
1.5	Bibliografie	52
2	Nein sagen	53
2.1	Feindliche Rollen	55
2.1.1	Was ist mit dem Warum?	58
2.2	Hoher Einsatz	58
2.3	Ein »Teamplayer« sein	60
2.3.1	Versuchen	62
2.3.2	Passive Aggression	64

2.4	Die Kosten eines Ja	65
2.5	Code unmöglich	72
3	Ja sagen	75
3.1	Verbindliche Sprache	76
3.1.1	So erkennt man mangelnde Selbstverpflichtung	77
3.1.2	Wie echte Selbstverpflichtung klingt	78
3.1.3	Zusammenfassung	80
3.2	Lernen, wie man »Ja« sagt	81
3.2.1	Die Kehrseite von »Ich versuch's mal«	81
3.2.2	Der Disziplin verpflichtet	82
3.3	Schlussfolgerung	84
4	Programmieren	85
4.1	Bereit sein	86
4.1.1	Code um drei Uhr früh	87
4.1.2	Sorgencode	88
4.2	Der Flow-Zustand	89
4.2.1	Musik	90
4.2.2	Unterbrechungen	91
4.3	Schreibblockaden	92
4.3.1	Kreativer Input	92
4.4	Debugging	93
4.4.1	Zeit zum Debuggen	96
4.5	Die eigene Energie einteilen	96
4.5.1	Wann man den Stift weglegen muss	97
4.5.2	Die Heimfahrt	97
4.5.3	Die Dusche	97
4.6	In Verzug sein	98
4.6.1	Hoffnung	98
4.6.2	Sich beeilen	98
4.6.3	Überstunden	99
4.6.4	Unlautere Ablieferung	99
4.6.5	Definieren Sie »fertig und erledigt«	100
4.7	Hilfe	100
4.7.1	Anderen helfen	101
4.7.2	Hilfe annehmen	101
4.7.3	Mentorenarbeit	102
4.8	Bibliografie	102

5	Test Driven Development	103
5.1	The Jury is in	104
5.2	Die drei Gesetze des TDD	105
5.2.1	Die Litanei der Vorteile	105
5.2.2	Die professionelle Option	108
5.3	Was TDD nicht ist	109
5.4	Bibliografie	109
6	Praktizieren und Üben	111
6.1	Etwas Hintergrund übers Üben	111
6.1.1	22 Nullen	112
6.1.2	Durchlaufzeiten	113
6.2	Das Coding Dojo	114
6.2.1	Kata	115
6.2.2	Waza	116
6.2.3	Randori	117
6.3	Die eigene Erfahrung ausbauen	117
6.3.1	Open Source	118
6.3.2	Ethisch handeln	118
6.4	Schlussfolgerung	118
6.5	Bibliografie	118
7	Akzeptanztests	119
7.1	Anforderungen kommunizieren	119
7.1.1	Verfrühte Präzisierung	121
7.2	Akzeptanztests	124
7.2.1	Die »Definition of Done«	124
7.2.2	Kommunikation	127
7.2.3	Automatisierung	127
7.2.4	Zusätzliche Arbeit	128
7.2.5	Wer schreibt die Akzeptanztests und wann?	128
7.2.6	Die Rolle des Entwicklers	129
7.2.7	Verhandlungen über die Tests und passive Aggression	130
7.2.8	Akzeptanz- und Unit-Tests	132
7.2.9	GUIs und andere Komplikationen	132
7.2.10	Andauernde Integration	134
7.3	Schlussfolgerung	134

8	Teststrategien	135
8.1	Für die Qualitätssicherung sollte nichts übrig bleiben	135
8.1.1	Die Qualitätssicherung gehört zum Team	135
8.2	Die Pyramide der Testautomatisierung	136
8.2.1	Unit-Tests	136
8.2.2	Komponententests	137
8.2.3	Integrationstests	138
8.2.4	Systemtests	139
8.2.5	Manuelle explorative Tests	139
8.3	Schlussfolgerung	140
8.4	Bibliografie	140
9	Zeitmanagement	141
9.1	Meetings	142
9.1.1	Absagen	142
9.1.2	Sich ausklinken	143
9.1.3	Tagesordnung und Ziel	143
9.1.4	Stand-up-Meetings	144
9.1.5	Planungstreffen zur Iteration	144
9.1.6	Retrospektive und Demo der Iteration	145
9.1.7	Auseinandersetzungen und Meinungsverschiedenheiten ...	145
9.2	Fokus-Manna	146
9.2.1	Schlaf	147
9.2.2	Koffein	147
9.2.3	Die Akkus aufladen	147
9.2.4	Muskelfokus	147
9.2.5	Input vs. Output	148
9.3	Zeitfenster und Tomaten	148
9.4	Vermeidung	149
9.4.1	Umkehrung der Prioritäten	149
9.5	Sackgassen	150
9.6	Morast, Moore, Sümpfe und andere Schlamassel	150
9.7	Schlussfolgerung	151
10	Aufwandsschätzungen	153
10.1	Was eine Aufwandsschätzung ist	155
10.1.1	Ein Commitment	155
10.1.2	Eine Aufwandsschätzung	155
10.1.3	Implizierte Commitments	157

10.2	PERT.....	158
10.3	Aufgaben schätzen	161
	10.3.1 Wideband Delphi	161
10.4	Das Gesetz der großen Zahlen.....	163
10.5	Schlussfolgerung	164
10.6	Bibliografie.....	164
11	Außerer Druck.....	165
11.1	Druck vermeiden.....	167
	11.1.1 Commitments.....	167
	11.1.2 Sauber arbeiten.....	167
	11.1.3 Verhalten in der Krise	168
11.2	Umgang mit Druck	168
	11.2.1 Keine Panik.....	168
	11.2.2 Kommunizieren Sie	169
	11.2.3 Verlassen Sie sich auf Ihre Disziplinen	169
	11.2.4 Hilfe holen	169
11.3	Schlussfolgerung	170
12	Teamwork	171
12.1	Programmierer kontra Menschen	172
	12.1.1 Programmierer kontra Arbeitgeber	173
	12.1.2 Programmierer kontra Programmierer	175
12.2	Kleinhirne	177
12.3	Schlussfolgerung	178
13	Teams und Projekte.....	179
13.1	Harmoniert es?	179
	13.1.1 Das zusammengeschweißte Team	179
	13.1.2 Aber wie managt man so etwas?	181
	13.1.3 Das Dilemma des Product Owner.....	181
13.2	Schlussfolgerung	182
13.3	Bibliografie.....	182
14	Mentoring, Lehrzeiten und die Handwerkskunst.....	183
14.1	Der Grad des Versagens	183
14.2	Mentoring.....	184
	14.2.1 Digi-Comp I – Mein erster Computer	184
	14.2.2 Die ECP-18 in der Highschool.....	185

14.2.3	Unkonventionelles Mentoring.	188
14.2.4	Schicksalsschläge	189
14.3	Die Lehrzeit	189
14.3.1	Die Lehrzeit bei der Software	191
14.3.2	Die Realität	192
14.4	Die Handwerkskunst.	193
14.4.1	Menschen überzeugen.	193
14.5	Schlussfolgerung	193
A	Werkzeuge und Hilfsmittel.	195
A.1	Tools	196
A.2	Quellcodekontrolle.	197
A.2.1	Ein »Enterprise«-System der Quellcodekontrolle.	197
A.2.2	Pessimistisches kontra optimistisches Locking	197
A.2.3	CVS/SVN.	198
A.2.4	git	198
A.3	IDE/Editor	201
A.3.1	vi.	201
A.3.2	Emacs.	201
A.3.3	Eclipse/IntelliJ	201
A.3.4	TextMate	202
A.4	Issue-Tracking-Systeme.	202
A.4.1	Bug-Zähler	203
A.5	Continuous Build.	203
A.6	Tools für Unit-Tests.	204
A.7	Tools für Komponententests.	205
A.7.1	Die »Definition of Done«.	205
A.7.2	FitNesse	205
A.7.3	Andere Tools	206
A.8	Tools für Integrationstests	206
A.9	UML/MDA	207
A.9.1	Die Details	207
A.9.2	Keine Hoffnung, keine Änderung.	209
A.10	Schlussfolgerung	209
	Stichwortverzeichnis	210