

CONTENTS

	Foreword		xvii
	Preface		xxi
	Acknowledgments		xxvii
	About the Author		xxix
Chapter 1	Craftsmanship		i
PART I	The Disciplines		ii
	Extreme Programming		13
	The Circle of Life		14
	Test-Driven Development		15
	Refactoring		16
	Simple Design		17
	Collaborative Programming		17
	Acceptance Tests		18
Chapter 2	Test-Driven Development		19
	Overview		20
	Software		22

	The Three Laws of TDD	23
	The Fourth Law	34
	The Basics	35
	Simple Examples	36
	Stack	36
	Prime Factors	52
	The Bowling Game	62
	Conclusion	79
Chapter 3	Advanced TDD	81
	Sort 1	82
	Sort 2	87
	Getting Stuck	95
	Arrange, Act, Assert	103
	Enter BDD	104
	Finite State Machines	105
	BDD Again	107
	Test Doubles	108
	Dummy	111
	Stub	115
	Spy	118
	Mock	121
	Fake	124
	The TDD Uncertainty Principle	126
	London versus Chicago	139
	The Certainty Problem	140
	London	141
	Chicago	142
	Synthesis	143
	Architecture	143
	Conclusion	145
Chapter 4	Test Design	147
	Testing Databases	148
	Testing GUIs	150
	GUI Input	153

	Test Patterns	154
	Test-Specific Subclass	155
	Self-Shunt	156
	Humble Object	157
	Test Design	160
	The Fragile Test Problem	160
	The One-to-One Correspondence	161
	Breaking the Correspondence	163
	The Video Store	164
Chapter 11	Specificity versus Generality	183
	Transformation Priority Premise	184
	{} → Nil	186
	Nil → Constant	187
	Unconditional → Selection	188
	Value → List	189
	Statement → Recursion	189
PART III	Selection → Iteration	190
	Value → Mutated Value	190
	Example: Fibonacci	191
	The Transformation Priority Premise	195
	Conclusion	196
Chapter 5	Refactoring	197
	What Is Refactoring?	199
	The Basic Toolkit	200
	Rename	200
	Extract Method	201
	Extract Variable	202
	Extract Field	204
	Rubik's Cube	217
	The Disciplines	217
	Tests	218
	Quick Tests	218
	Break Deep One-to-One Correspondences	218
	Refactor Continuously	219
	Refactor Mercilessly	219

	Keep the Tests Passing!	219
	Leave Yourself an Out	220
	Conclusion	221
	Simple Examples	36
Chapter 6	Simple Design	223
	YAGNI	226
	Covered by Tests	228
	Coverage	230
	An Asymptotic Goal	231
	Design?	232
	But There's More	232
	Maximize Expression	233
	The Underlying Abstraction	235
	Tests: The Other Half of the Problem	236
	Minimize Duplication	237
	Accidental Duplication	238
	Minimize Size	239
	Simple Design	239
Chapter 7	Collaborative Programming	241
Chapter 8	Acceptance Tests	245
	The Discipline	248
	The Continuous Build	249
PART II	The Standards	251
	Your New CTO	252
Chapter 9	Productivity	253
	We Will Never Ship S**T	254
	Inexpensive Adaptability	256
	We Will Always Be Ready	258
	Stable Productivity	259
Chapter 10	Quality	261
	Continuous Improvement	262

	Fearless Competence	263
	Extreme Quality	264
	We Will Not Dump on QA	265
	The QA Disease	266
	QA Will Find Nothing	266
	Test Automation	267
	Automated Testing and User Interfaces	268
	Testing the User Interface	269
Chapter 11	Courage	271
	We Cover for Each Other	272
	Honest Estimates	274
	You Must Say NO	276
	Continuous Aggressive Learning	277
	Mentoring	278
PART III	The Ethics	279
	The First Programmer	280
	Seventy-Five Years	281
	Nerds and Saviors	286
	Role Models and Villains	289
	We Rule the World	290
	Catastrophes	291
	The Oath	293
Chapter 12	Harm	295
	First, Do No Harm	296
	No Harm to Society	297
	Harm to Function	299
	No Harm to Structure	302
	Soft	303
	Tests	305
	Best Work	306
	Making It Right	307
	What Is Good Structure?	308
	Eisenhower's Matrix	310

	Programmers Are Stakeholders	312
	Your Best	314
	Repeatable Proof	316
	Dijkstra	316
Chapter 6	Proving Correctness	317
	Structured Programming	319
	Functional Decomposition	322
	Test-Driven Development	323
	An Asymptotic Goal	321
Chapter 13	Integrity	327
	Small Cycles	328
	The History of Source Code Control	328
	Git	334
	Short Cycles	336
	Continuous Integration	337
	Branches versus Toggles	338
	Continuous Deployment	340
	Continuous Build	341
	Relentless Improvement	342
Chapter 7	Test Coverage	343
	Mutation Testing	344
Chapter 8	Semantic Stability	344
	Cleaning	345
	Creations	346
	Maintain High Productivity	346
	Viscosity	347
	Managing Distractions	349
	Time Management	352
Chapter 14	Teamwork	355
	Work as a Team	356
	Open/Virtual Office	356
	Estimate Honestly and Fairly	358
	Lies	359
Chapter 10	Honesty, Accuracy, Precision	360
	Story 1: Vectors	361

Story 2: pCCU	363
The Lesson	365
Accuracy	365
Precision	367
Aggregation	369
Honesty	370
Respect	372
Never Stop Learning	373

Index	375
-------	-----

I remember meeting Uncle Bob in the spring of 2003, soon after Scrum was introduced to our company and technology teams. As a skeptical, fledgling ScrumMaster, I remember listening to Bob teach us about TDD and a little tool called FitNesse, and I remember thinking to myself, “Why would we ever write test cases that fail first? Doesn’t testing come *after* coding?” I often walked away scratching my head, as did many of my team members, and yet to this day I distinctly remember Bob’s palpable exuberance for code craftsmanship like it was only yesterday. I recall his directness one day as he was looking at our bug backlog and asking us why on earth we would make such poor decisions about software systems that we did not in fact own—“These systems are *company assets*, not your own *personal assets*.” His passion piqued our curiosity, and a year and half later, we had refactored our way to about 80 percent automated test coverage and a clean code base that made pivoting much easier, resulting in much happier customers—and happier teams. We moved lightning fast after that, wielding our definition of *done* like armor to protect us from the always-lurking code goblins; we had learned, in essence, how to protect us from ourselves. Over time, we developed a warmth for Uncle Bob, who came to truly feel like an uncle to us—a warm, determined, and courageous man who would over time help us learn to stand up for ourselves and do what was right. While some kids’ Uncle