

97 Things

Every Programmer Should Know

Collective Wisdom from the Experts

Edited by Kevlin Henney

O'REILLY®
Beijing · Cambridge · Farnham · Köln · Sebastopol · Taipei · Tokyo

Contents

Contributions by Category	xv
Preface.	xxiii
Act with Prudence	2
<i>Seb Rose</i>	
Apply Functional Programming Principles	4
<i>Edward Garson</i>	
Ask, “What Would the User Do?” (You Are Not the User)	6
<i>Giles Colborne</i>	
Automate Your Coding Standard.	8
<i>Filip van Laenen</i>	
Beauty Is in Simplicity	10
<i>Jørn Ølmheim</i>	
Before You Refactor	12
<i>Rajith Attapattu</i>	
Beware the Share	14
<i>Udi Dahan</i>	

The Boy Scout Rule	16
<i>Robert C. Martin (Uncle Bob)</i>	
Check Your Code First Before Looking to Blame Others. . .	18
<i>Allan Kelly</i>	
Choose Your Tools with Care	20
<i>Giovanni Asproni</i>	
Code in the Language of the Domain	22
<i>Dan North</i>	
Code Is Design.	24
<i>Ryan Brush</i>	
Code Layout Matters.	26
<i>Steve Freeman</i>	
Code Reviews	28
<i>Mattias Karlsson</i>	
Coding with Reason	30
<i>Yechiel Kimchi</i>	
A Comment on Comments.	32
<i>Cal Evans</i>	
Comment Only What the Code Cannot Say	34
<i>Kevlin Henney</i>	
Continuous Learning.	36
<i>Clint Shank</i>	
Convenience Is Not an -ility	38
<i>Gregor Hohpe</i>	

Deploy Early and Often	40
<i>Steve Berczuk</i>	
Distinguish Business Exceptions from Technical	42
<i>Dan Bergh Johnson</i>	
Do Lots of Deliberate Practice.	44
<i>Jon Jagger</i>	
Domain-Specific Languages	46
<i>Michael Hunger</i>	
Don't Be Afraid to Break Things	48
<i>Mike Lewis</i>	
Don't Be Cute with Your Test Data	50
<i>Rod Begbie</i>	
Don't Ignore That Error!.	52
<i>Pete Goodliffe</i>	
Don't Just Learn the Language, Understand Its Culture . .	54
<i>Anders Norås</i>	
Don't Nail Your Program into the Upright Position.	56
<i>Verity Stob</i>	
Don't Rely on "Magic Happens Here"	58
<i>Alan Griffiths</i>	
Don't Repeat Yourself	60
<i>Steve Smith</i>	
Don't Touch That Code!.	62
<i>Cal Evans</i>	

Encapsulate Behavior, Not Just State	64
<i>Einar Landre</i>	
Floating-Point Numbers Aren't Real	66
<i>Chuck Allison</i>	
Fulfill Your Ambitions with Open Source	68
<i>Richard Monson-Haefel</i>	
The Golden Rule of API Design	70
<i>Michael Feathers</i>	
The Guru Myth.	72
<i>Ryan Brush</i>	
Hard Work Does Not Pay Off	74
<i>Olve Maudal</i>	
How to Use a Bug Tracker	76
<i>Matt Doar</i>	
Improve Code by Removing It.	78
<i>Pete Goodliffe</i>	
Install Me	80
<i>Marcus Baker</i>	
Interprocess Communication Affects Application Response Time	82
<i>Randy Stafford</i>	
Keep the Build Clean.	84
<i>Johannes Brodwall</i>	
Know How to Use Command-Line Tools	86
<i>Carroll Robinson</i>	

Know Well More Than Two Programming Languages. . . .	88
<i>Russel Winder</i>	
Know Your IDE.	90
<i>Heinz Kabutz</i>	
Know Your Limits.	92
<i>Greg Colvin</i>	
Know Your Next Commit	94
<i>Dan Bergh Johnsson</i>	
Large, Interconnected Data Belongs to a Database	96
<i>Diomidis Spinellis</i>	
Learn Foreign Languages.	98
<i>Klaus Marquardt</i>	
Learn to Estimate.	100
<i>Giovanni Asproni</i>	
Learn to Say, "Hello, World"	102
<i>Thomas Guest</i>	
Let Your Project Speak for Itself	104
<i>Daniel Lindner</i>	
The Linker Is Not a Magical Program	106
<i>Walter Bright</i>	
The Longevity of Interim Solutions.	108
<i>Klaus Marquardt</i>	
Make Interfaces Easy to Use Correctly and Hard to Use Incorrectly	110
<i>Scott Meyers</i>	

Make the Invisible More Visible	112
<i>Jon Jagger</i>	
Message Passing Leads to Better Scalability in Parallel Systems	114
<i>Russel Winder</i>	
A Message to the Future	116
<i>Linda Rising</i>	
Missing Opportunities for Polymorphism.	118
<i>Kirk Pepperdine</i>	
News of the Weird: Testers Are Your Friends	120
<i>Burk Hufnagel</i>	
One Binary	122
<i>Steve Freeman</i>	
Only the Code Tells the Truth	124
<i>Peter Sommerlad</i>	
Own (and Refactor) the Build	126
<i>Steve Berczuk</i>	
Pair Program and Feel the Flow.	128
<i>Gudny Hauknes, Kari Røssland, and Ann Katrin Gagnat</i>	
Prefer Domain-Specific Types to Primitive Types	130
<i>Einar Landre</i>	
Prevent Errors	132
<i>Giles Colborne</i>	
The Professional Programmer	134
<i>Robert C. Martin (Uncle Bob)</i>	

Put Everything Under Version Control	136
<i>Diomidis Spinellis</i>	
Put the Mouse Down and Step Away from the Keyboard .	138
<i>Burk Hufnagel</i>	
Read Code	140
<i>Karianne Berg</i>	
Read the Humanities.	142
<i>Keith Braithwaite</i>	
Reinvent the Wheel Often	144
<i>Jason P. Sage</i>	
Resist the Temptation of the Singleton Pattern.	146
<i>Sam Saariste</i>	
The Road to Performance Is Littered with Dirty Code Bombs	148
<i>Kirk Pepperdine</i>	
Simplicity Comes from Reduction	150
<i>Paul W. Homer</i>	
The Single Responsibility Principle.	152
<i>Robert C. Martin (Uncle Bob)</i>	
Start from Yes	154
<i>Alex Miller</i>	
Step Back and Automate, Automate, Automate	156
<i>Cay Horstmann</i>	
Take Advantage of Code Analysis Tools	158
<i>Sarah Mount</i>	

Test for Required Behavior, Not Incidental Behavior.	160
<i>Kevlin Henney</i>	
Test Precisely and Concretely	162
<i>Kevlin Henney</i>	
Test While You Sleep (and over Weekends)	164
<i>Rajith Attapattu</i>	
Testing Is the Engineering Rigor of Software Development	166
<i>Neal Ford</i>	
Thinking in States.	168
<i>Niclas Nilsson</i>	
Two Heads Are Often Better Than One	170
<i>Adrian Wible</i>	
Two Wrongs Can Make a Right (and Are Difficult to Fix) .	172
<i>Allan Kelly</i>	
Ubuntu Coding for Your Friends	174
<i>Aslam Khan</i>	
The Unix Tools Are Your Friends	176
<i>Diomidis Spinellis</i>	
Use the Right Algorithm and Data Structure	178
<i>Jan Christiaan "JC" van Winkel</i>	
Verbose Logging Will Disturb Your Sleep	180
<i>Johannes Brodwall</i>	

WET Dilutes Performance Bottlenecks	182
<i>Kirk Pepperdine</i>	
When Programmers and Testers Collaborate	184
<i>Janet Gregory</i>	
Write Code As If You Had to Support It for the Rest of Your Life.	186
<i>Yuriy Zubarev</i>	
Write Small Functions Using Examples	188
<i>Keith Braithwaite</i>	
Write Tests for People	190
<i>Gerard Meszaros</i>	
You Gotta Care About the Code	192
<i>Pete Goodliffe</i>	
Your Customers Do Not Mean What They Say	194
<i>Nate Jackson</i>	
Contributors	196
Index	221