

Refactoring

Improving the Design of Existing Code

Second Edition

Martin Fowler

with contributions by **Kent Beck**

/•Addison-Wesley

Boston • Columbus • New York • San Francisco • Amsterdam • Cape Town
Dubai • London • Madrid • Milan • Munich • Paris • Montreal • Toronto • Delhi • Mexico City
Sao Paulo • Sydney • Hong Kong • Seoul • Singapore • Taipei • Tokyo

Contents

Foreword to the First Edition *xi*

Preface *xiii*

Chapter 1: Refactoring: A First Example **1**

The Starting Point 1

Comments on the Starting Program 3

The First Step in Refactoring 5

Decomposing the statement Function 6

Status: Lots of Nested Functions 22

Splitting the Phases of Calculation and Formatting 24

Status: Separated into Two Files (and Phases) 31

Reorganizing the Calculations by Type 34

Status: Creating the Data with the Polymorphic Calculator 41

Final Thoughts 43

Chapter 2: Principles in Refactoring **45**

Defining Refactoring 45

The Two Hats 46

Why Should We Refactor? 47

When Should We Refactor? 50

Problems with Refactoring 55

Refactoring, Architecture, and Yagni 62

Refactoring and the Wider Software Development Process 63

Refactoring and Performance 64

Where Did Refactoring Come From? 67

Automated Refactorings 68

Going Further 70

Chapter 3: Bad Smells in Code	71
Mysterious Name	72
Duplicated Code	72
Long Function	73
Long Parameter List	74
Global Data	74
Mutable Data	75
Divergent Change	76
Shotgun Surgery	76
Feature Envy	77
Data Clumps	78
Primitive Obsession	78
Repeated Switches	79
Loops	79
Lazy Element	80
Speculative Generality	80
Temporary Field	80
Message Chains	81
Middle Man	81
Insider Trading	82
Large Class	82
Alternative Classes with Different Interfaces	83
Data Class	83
Refused Bequest	83
Comments	84
Chapter 4: Building Tests	85
The Value of Self-Testing Code	85
Sample Code to Test	87
A First Test	90
Add Another Test	93
Modifying the Fixture	95
Probing the Boundaries	96
Much More Than This	99
Chapter 5: Introducing the Catalog	101
Format of the Refactorings	101
The Choice of Refactorings	102

Chapter 6: A First Set of Refactorings	105
Extract Function	106
Inline Function	115
Extract Variable	119
Inline Variable	123
Change Function Declaration	124
Encapsulate Variable	132
Rename Variable	137
Introduce Parameter Object	140
Combine Functions into Class	144
Combine Functions into Transform	149
Split Phase	154
Chapter 7: Encapsulation	161
Encapsulate Record	162
Encapsulate Collection	170
Replace Primitive with Object	174
Replace Temp with Query	178
Extract Class	182
Inline Class	186
Hide Delegate	189
Remove Middle Man	192
Substitute Algorithm	195
Chapter 8: Moving Features	197
Move Function	198
Move Field	207
Move Statements into Function	213
Move Statements to Callers	217
Replace Inline Code with Function Call	222
Slide Statements	223
Split Loop	227
Replace Loop with Pipeline	231
Remove Dead Code	237
Chapter 9: Organizing Data	239
Split Variable	240
Rename Field	244
Replace Derived Variable with Query	248

Change Reference to Value	252
Change Value to Reference	256
Chapter 10: Simplifying Conditional Logic	259
Decompose Conditional	260
Consolidate Conditional Expression	263
Replace Nested Conditional with Guard Clauses	266
Replace Conditional with Polymorphism	272
Introduce Special Case	289
Introduce Assertion	302
Chapter 11: Refactoring APIs	305
Separate Query from Modifier	306
Parameterize Function	310
Remove Flag Argument	314
Preserve Whole Object	319
Replace Parameter with Query	324
Replace Query with Parameter	327
Remove Setting Method	331
Replace Constructor with Factory Function	334
Replace Function with Command	337
Replace Command with Function	344
Chapter 12: Deeding with Inheritance	349
Pull Up Method	350
Pull Up Field	353
Pull Up Constructor Body	355
Push Down Method	359
Push Down Field	361
Replace Type Code with Subclasses	362
Remove Subclass	369
Extract Superclass	375
Collapse Hierarchy	380
Replace Subclass with Delegate	381
Replace Superclass with Delegate	399
<i>Bibliography</i>	405
<i>Index</i>	409