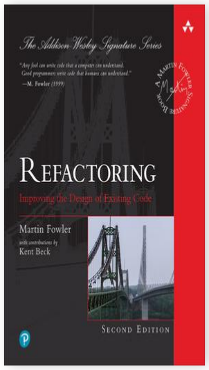
Sign In START FREE TRIAL



Refactoring: Improving the Design of Existing Code

8 reviews

by Martin Fowler
Publisher: Addison-Wesley Professional
Release Date: November 2018
ISBN: 9780134757681

View table of contents

START READING

Book Description

Fully Revised and Updated–Includes New Refactorings and Code Examples

- *“Any fool can write code that a computer can understand. Good programmers write code that humans can understand.”*
  —M. Fowler (1999)

For more than twenty years, experienced programmers worldwide have relied on Martin Fowler’s *Refactoring* to improve the design of existing code and to enhance software maintainability, as well as to make existing code easier to understand.

This eagerly awaited new edition has been fully updated to reflect crucial changes in the programming landscape. ***Refactoring, Second Edition,*** features an updated catalog of refactorings and includes JavaScript code examples, as well as new functional examples that demonstrate refactoring without classes.

Like the original, this edition explains what refactoring is; why you should refactor; how to recognize code that needs refactoring; and how to actually do it successfully, no matter what language you use.

- Recognize “bad smells” in code that signal opportunities to refactor
- Explore the refactorings, each with explanations, motivation, mechanics, and simple examples
- Build solid tests for your refactorings
- Recognize tradeoffs and obstacles to refactoring

Includes free access to the canonical web edition, with even more refactoring resources. (See inside the book for details about how to access the web edition.)

Table of Contents

Cover Page
About This E-Book
Front End Paper
Title Page
Copyright Page
Dedication Page
Contents
Foreword to the First Edition
Preface
    What Is Refactoring?
    What’s in This Book?
    Who Should Read This Book?
    Building on a Foundation Laid by Others
    Acknowledgments

Sign In START FREE TRIAL

Comments on the Starting Program

The First Step in Refactoring

Decomposing the statement Function

Status: Lots of Nested Functions

Splitting the Phases of Calculation and Formatting

Status: Separated into Two Files (and Phases)

Reorganizing the Calculations by Type

Status: Creating the Data with the Polymorphic Calculator

Final Thoughts

Chapter 2 Principles in Refactoring

Defining Refactoring

The Two Hats

Why Should We Refactor?

When Should We Refactor?

Problems with Refactoring

Refactoring, Architecture, and Yagni

Refactoring and the Wider Software Development Process

Refactoring and Performance

Where Did Refactoring Come From?

Automated Refactorings

Going Further

Chapter 3 Bad Smells in Code

Mysterious Name

Duplicated Code

Long Function

Long Parameter List

Global Data

Sign In START FREE TRIAL

Shotgun Surgery

Feature Envy

Data Clumps

Primitive Obsession

Repeated Switches

Loops

Lazy Element

Speculative Generality

Temporary Field

Message Chains

Middle Man

Insider Trading

Large Class

Alternative Classes with Different Interfaces

Data Class

Refused Bequest

Comments

Chapter 4 Building Tests

The Value of Self-Testing Code

Sample Code to Test

A First Test

Add Another Test

Modifying the Fixture

Probing the Boundaries

Much More Than This

Chapter 5 Introducing the Catalog

Format of the Refactorings

Sign In START FREE TRIAL

Extract Function

Inline Function

Extract Variable

Inline Variable

Change Function Declaration

Encapsulate Variable

Rename Variable

Introduce Parameter Object

Combine Functions into Class

Combine Functions into Transform

Split Phase

Chapter 7 Encapsulation

Encapsulate Record

Encapsulate Collection

Replace Primitive with Object

Replace Temp with Query

Extract Class

Inline Class

Hide Delegate

Remove Middle Man

Substitute Algorithm

Chapter 8 Moving Features

Move Function

Move Field

Move Statements into Function

Move Statements to Callers

Replace Inline Code with Function Call

Sign In START FREE TRIAL

Replace Loop with Pipeline

Remove Dead Code

Chapter 9 Organizing Data

Split Variable

Rename Field

Replace Derived Variable with Query

Change Reference to Value

Change Value to Reference

Chapter 10 Simplifying Conditional Logic

Decompose Conditional

Consolidate Conditional Expression

Replace Nested Conditional with Guard Clauses

Replace Conditional with Polymorphism

Introduce Special Case

Introduce Assertion

Chapter 11 Refactoring APIs

Separate Query from Modifier

Parameterize Function

Remove Flag Argument

Preserve Whole Object

Replace Parameter with Query

Replace Query with Parameter

Remove Setting Method

Replace Constructor with Factory Function

Replace Function with Command

Replace Command with Function

Chapter 12 Dealing with Inheritance

Sign In START FREE TRIAL

Pull Up Constructor Body

Push Down Method

Push Down Field

Replace Type Code with Subclasses

Remove Subclass

Extract Superclass

Collapse Hierarchy

Replace Subclass with Delegate

Replace Superclass with Delegate

Bibliography

Index

Back End Paper

Code Snippets

Explore

Tour

Pricing

Enterprise

Government

Education

Queue App

Sign In START FREE TRIAL

Contact

Careers

Press Resources

Support

Twitter

GitHub

Facebook

LinkedIn

Terms of Service

Membership Agreement

Privacy Policy

Copyright © 2019 Safari Books Online.