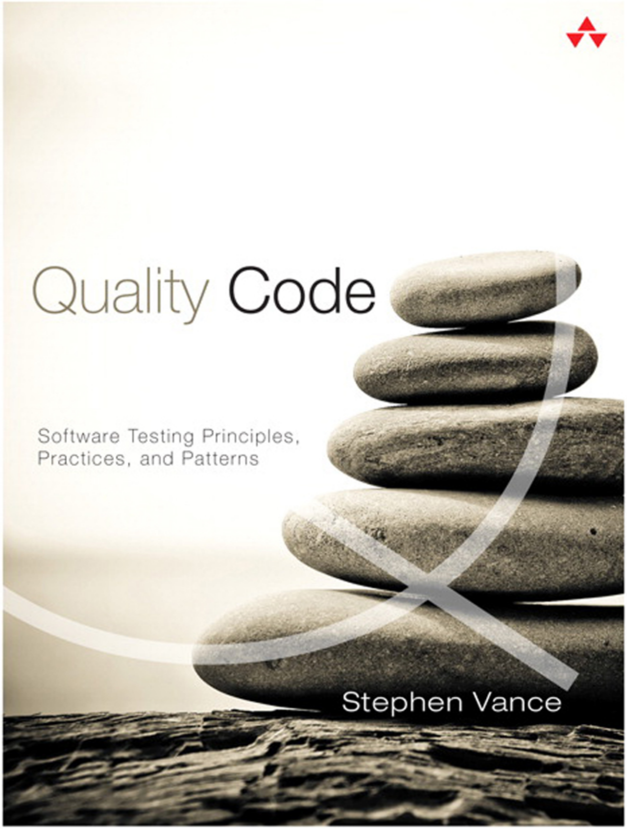
Quality Code

Software Testing Principles, Practices, and Patterns

Stephen Vance



Quality Code

Quality Code: Software Testing Principles, Practices, and Patterns

Table of Contents

Contents

Preface

Acknowledgments

About the Author

Part I: Principles and Practices of Testing

- Chapter 1: Engineering, Craftsmanship, and First-Time Quality
  - Engineering and Craftsmanship
  - The Role of Craftsmanship in First-Time Quality
  - Practices Supporting Software Craftsmanship
  - Unit Testing under Code Checker Constraints
  - Unit Testing for Coverage
- Chapter 2: Intent of Code
  - Where Did I Put That Intent?
  - Separating Intent from Implementation
  - A Simple Example That Makes You Think
- Chapter 3: Where Do I Start?
  - An Approach to Testing
  - The Recipe
- Chapter 4: Design and Testability
  - A Word on Design Paradigms
  - Encapsulation and Observability

Pearson

Table of Contents

Coupling and Testability

Chapter 5: Testing Principles

Craft Your Tests Well

Avoid Test Code in Production

Verify Intent over Implementation

Minimize Coupling

Prefer Minimal, Fresh, Transient Fixtures

Use Available Facilities

Prefer Complete over Partial Verification

Write Small Tests

Separate Your Concerns

Use Unique Values

Keep It Simple: Remove Code

Dont Test the Framework

Sometimes Test the Framework

Part II: Testing and Testability Patterns

Chapter 6: The Basics

Bootstrapping Constructors

Testing Simple Getters and Setters

Share Constants

Locally Redefine

Temporarily Replace

Encapsulate and Override

Adjust Visibility

Verification by Injection

Chapter 7: String Handling

Verification by Containment

Verification by Pattern

Exact Verification by Value

Pearson

Table of Contents

Exact Verification with Formatted Results

Chapter 8: Encapsulation and Override Variations

Data Injection

Encapsulate Loop Conditions

Error Injection

Replace Collaborators

Use Existing No-Op Classes

Chapter 9: Adjusting Visibility

Packaging Tests with Code

Break It Down

Changing Access Levels

Test-Only Interfaces

Naming the Unnamed

Becoming friend-ly

Coerced Access via Reflection

Declarative Scope Changing

Chapter 10: Interlude: Revisiting Intent

Testing the Singleton Pattern

Singleton Intent

The Testing Strategy

Discerning Intent

Chapter 11: Error Condition Verification

Check the Return Value

Verify the Exception Type

Verify the Exception Message

Verify the Exception Payload

Verify the Exception Instance

Thoughts on Exception Design

Chapter 12: Use Existing Seams

Pearson

Table of Contents

Direct Calls

Dependency Injection

Callbacks, Observers, Listeners, and Notifiers

Registries

Factories

Logging and Other Facilities of Last Resort

Chapter 13: Parallelism

A Brief Introduction to Threads and Race Conditions

A Strategy for Race Condition Reproduction

Test the Thread Task Directly

Synchronize through Common Lock

Synchronize through Injection

Use Supervisory Control

Statistical Verification

Debugger APIs

Part III: Worked Examples

Chapter 14: Test-Driven Java

Bootstrapping

First Functionality

Cutting the Cord

Moving to Multiples

Ghost Protocol

Exercising Options

Moving Downstream

Retrospective

Chapter 15: Legacy JavaScript

Getting Started

DOMination

On Toothpaste and Testing

Pearson

Table of Contents

- Scaling Up
- Software Archeology
- Retrospective

Bibliography

Index

Pearson