



파이브라인스오브코드

(다섯 줄 제한 규칙으로 시작하는 체계적이고 효과적인 리팩터링 수련법)


2023년 08월 30일

XPER 스터디

LEO 김두진

doojin88@gmail.com

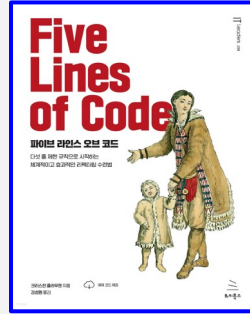
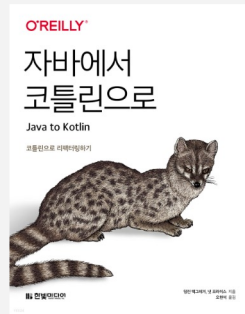
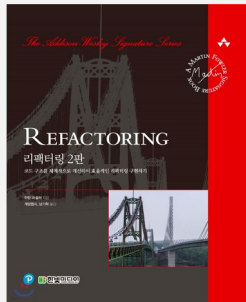
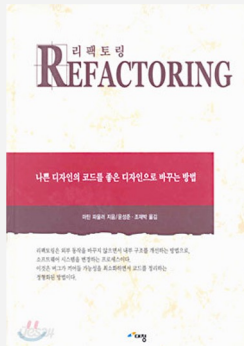
시작



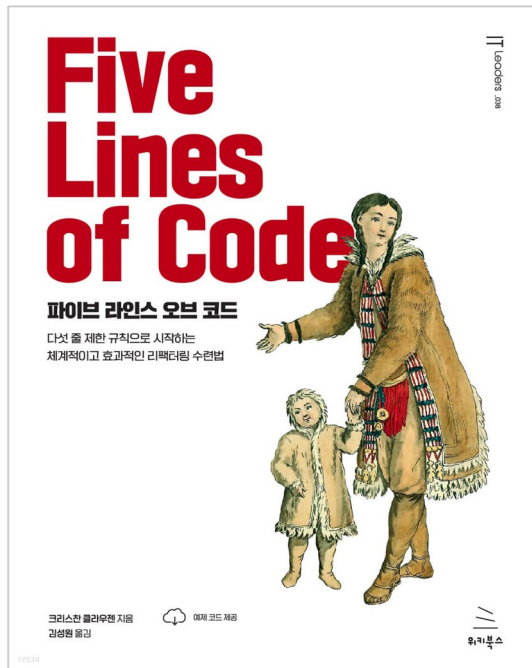
- ✓ 올해 리팩터링 스터디 하고파

어떤 책을

✓ 리팩터링이 좋은 것은 알겠는데 -> 리팩터링 쉽게 접근하기



도서 : 파이브 라인스 오브 코드



소독공제 | 위키북스 IT Leaders 시리즈-038

파이브 라인스 오브 코드 다섯 줄 제한 규칙으로 시작하는 체계적이고 효과적인 리팩터링 수련법

크리스찬 클라우젠 저/김성원 역 | 위키북스 2023년 01월 19일

첫번째 구매 리뷰를 남겨주세요. | 판매지수 6,879 [?](#) [베스트](#) IT 모바일 70위 | IT 모바일 top100 13주

- 1장: 리팩터링 리팩터링하기
- 2장: 리팩터링 깊게 들여다보기
- 3장: 긴 코드 조각내기
- 4장: 타입 코드 처리하기
- 5장: 유사한 코드 융합하기
- 6장: 데이터 보호
- 7장: 컴파일러와의 협업
- 8장: 주석 자제하기
- 9장: 코드 삭제의 미학
- 10장: 코드 추가에 대한 두려움 떨쳐내기
- 11장: 코드 구조 따르기
- 12장: 최적화 및 일반화 회피
- 13장: 나쁜 코드를 식별 가능하게 만들기
- 14장: 마무리

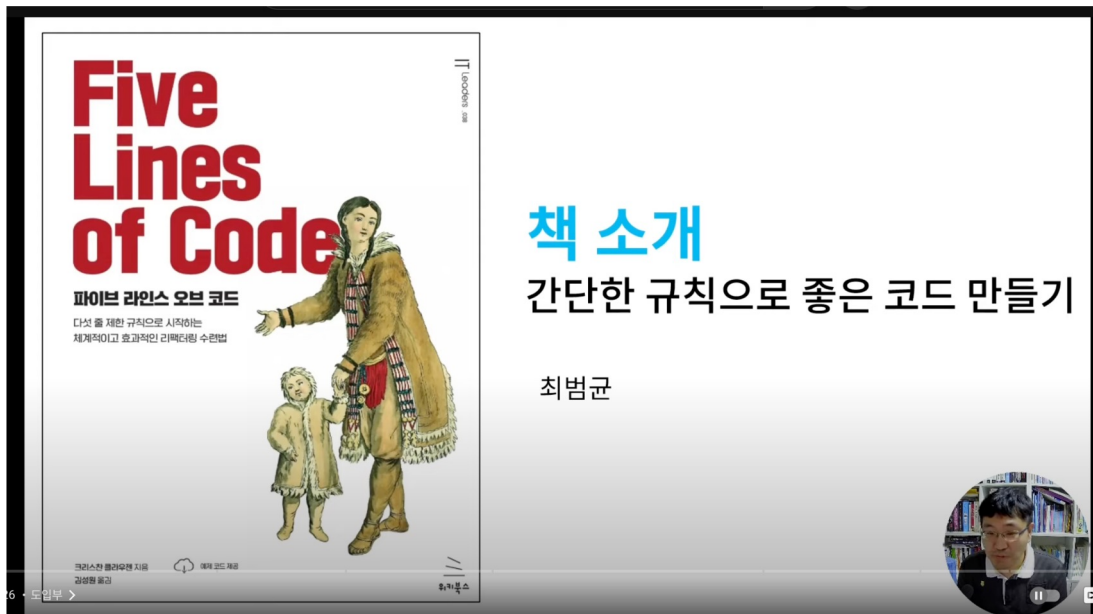
책소개

구체적인 규칙에 초점을 맞춰 모든 메서드를 5줄 이하로 줄이는 리팩터링을 가르쳐 준다!

기존 코드를 개선하는 것(리팩터링)은 프로그래머가 맞닥뜨리는 가장 일반적인 작업 중 하나다. 『파이브 라인스 오브 코드』는 코드 스멜과 같은 감각적인 판단에 의존하지 않고, 적용할 수 있는 명확하고 실행 가능한 리팩터링 규칙을 알려준다. 구체적인 원칙을 따르다 보면 리팩터링과 코드 스멜을 익힐 수 있다는 저자의 전문적인 관점에 따라 언제 코드를 리팩터링해야 하는지, 어떤 문제에 어떤 패턴을 적용할지, 재작업이 필요한 코드에는 어떤 특성이 있는지 배우게 된다.

프로그래머가 다루는 모든 코드 베이스에는 찾아서 수정해야 하는 실수와 효율적인지 않은 부분이 있다. 올바른 방법으로 리팩터링하면 코드가 세련되고 읽기 쉽고 유지 관리가 쉬워진다. 이 책에서는 모든 메서드를 5줄 이하로 구현하는 리팩터링에 대한 독창적인 접근 방식을 배운다. 이 책은 모든 기술 수준의 개발자가 읽을 수 있으며, 예제는 자바 및 C#과 동일한 스타일로 읽기 쉬운 타입스크립트를 사용한다.

최범균님 소개 동영상



- ✓ 스터디 책으로 고른 이유
 - 브라우저에서 실행되는 프로그램
 - 타입 스크립트 사용
 - 리팩토링을 위한 단순한 규칙 제시
- ✓ 다섯 줄로 만들기 위한 기본 패턴
- ✓ 단일 책임을 위한 두 규칙
- ✓ ...
- ✓ ...

소개 - 파이브 라인스 오브 코드

지은이 - 크리스찬 클라우젠



Christian Clausen

CEO, Founder

cc@mistware.eu

Christian Clausen은 기술 애자일 코치로 일하며 팀에게 코드를 올바르게 리팩터링하는 방법을 가르치고 있습니다. 이전에는 자동화된 리팩토링 도구인 Coccinelle 시맨틱 패치 프로젝트의 소프트웨어 엔지니어로 일했습니다. 컴퓨터 공학 석사 학위를 받았으며 대학에서 소프트웨어 품질에 대해 5년 동안 강의한 경험이 있습니다. Manning에서 출간한 'Five Lines of Code'의 저자이기도 합니다. GOTO 오르후스 2022에서 최고 평점을 받은 연사. 그의 저서 'Five Lines of Code'의 사인본을 받기 위해 줄을 서 있는 사람들.

Contact

www.linkedin.com/in/thedrlambda (LinkedIn)
twitter.com/themaxipaxi (Other)
medium.com/@drlambda (Blog)
github.com/maxipaxi (Other)

Top Skills

Java
Compilers
C

Languages

Danish
English

Certifications

Professional Scrum Master I

Publications

Five Lines of Code – A Fresh Look at Refactoring
Reducing Lookups for Invariant Checking
A characterization of Moessner's sieve

Christian Clausen

CEO and Founder of Mistware | Author of Five Lines of Code
Århus, Middle Jutland, Denmark

Summary

Check out my book at Manning: <https://www.manning.com/books/five-lines-of-code>

Experience

Mistware
Founder
May 2021 - Present (2 years 4 months)

Eficode
Technical Agile Coach
November 2019 - August 2022 (2 years 10 months)
Århus

Delegate
3 years 2 months
Senior Consultant
January 2019 - November 2019 (11 months)
Århus

Consultant
February 2018 - December 2018 (11 months)
Århus Area, Denmark

Associate Consultant (External)
October 2016 - January 2018 (1 year 4 months)
Skive

[Five Lines of Code • Christian Clausen & Kevlin Henney • GOTO 2023](#)

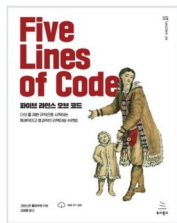
[Five Lines of Codes • Christian Clausen • GOTO 2022](#)

윝긴이 - 김성원

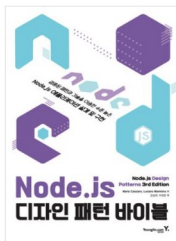
대학원에서 정보보안을 전공했으며, 2000년부터 안랩 등 여러 회사에 근무하면서 다양한 언어로 엔터프라이즈용 응용 프로그램의 설계와 개발에 참여해 왔다. 최근에는 음성인식, 영상인식, NLP 기술에 관심을 가지고 있다.

《새로 쓰는 자바 웹 프로그래밍》(영진닷컴, 2002), 《쉽게 풀어 쓴 자바 데이터베이스 프로그래밍》(영진닷컴, 2003)을 집필했고, 《iPhone 게임 개발자 레퍼런스》(영진닷컴, 2010), 《코딩인터뷰 퀘스천》(영진닷컴, 2014), 《Node.js 디자인 패턴 바이블》(영진닷컴, 2021), 《코딩 테스트로 시작하는 파이썬 프로그래밍》(영진닷컴, 2022) 등 평소 궁금해하는 분야와 관련된 번역을 즐긴다.

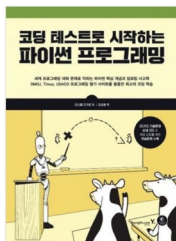
주요 작품



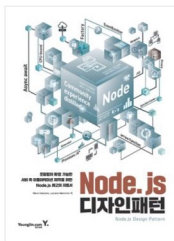
파이브 라인스 오브 코드
위키북스



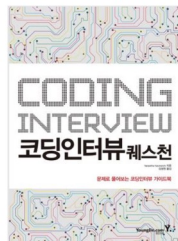
Node.js 디자인 패턴 바이블
영진닷컴



코딩 테스트로 시작하는 파이썬 프로그래밍
영진닷컴



Node.js 디자인 패턴
영진닷컴



코딩 인터뷰 퀘스천
영진닷컴


규칙 소개 - 이 책에서 소개된 규칙 요약

- 다섯 줄 제한(R3.1.1): 메서드는 사용하는 데이터 구조를 탐색하는 데 필요한 것보다 더 많은 코드로 구성해서는 안 됩니다.
- 호출 또는 전달, 한 가지만 할 것(R3.3.1): 함수 내에서는 객체에 있는 메서드를 호출하거나 객체를 인자로 전달할 수 있지만 둘을 섞어 사용해서는 안 됩니다.
- if 문은 함수의 시작에만 배치(R3.5.1): if 문이 있는 경우 해당 if 문은 함수의 첫 번째 항목이어야 합니다.
- if 문에서 else를 사용하지 말 것(R4.1.1): 프로그램에서 이해하지 못하는 타입(형)인지를 검사하지 않는 한 if 문에서 else를 사용하지 마십시오.
- switch를 사용하지 말 것(R4.2.4): default 케이스가 없고 모든 case에 반환 값이 있는 경우가 아니라면 switch를 사용하지 마십시오.
- 인터페이스에서만 상속받을 것(R4.3.2): 상속은 클래스나 추상 클래스가 아닌 오직 인터페이스를 통해서만 받습니다.
- 순수 조건 사용(R5.3.2): 조건문의 조건식에서는 변수에 값을 할당하거나 예외를 발생시키거나 I/O와 상호작용해서는 안 됩니다.
- 구현체가 하나뿐인 인터페이스를 만들지 말 것(R5.4.3): 구현체가 하나뿐인 인터페이스를 사용하지 마십시오.
- getter와 setter를 사용하지 말 것(R6.1.1): 부울(Boolean)이 아닌 필드에 setter나 getter를 사용하지 마십시오.
- 공통 접사를 사용하지 말 것(R6.2.1): 코드에는 공통 접두사나 접미사가 있는 메서드나 변수가 없어야 합니다.

리팩터링 패턴 - 이 책에서 소개된 리팩터링 패턴 요약

- 메서드 추출(P3.2.1): 한 메서드의 일부를 가져와 고유한 메서드로 추출합니다.
- 클래스로 타입 코드 대체(P4.1.3): 열거형을 인터페이스로 변환하고 열거형의 값을 클래스로 만듭니다.
- 클래스로의 코드 이관(P4.1.5): 기능을 클래스로 옮기 때문에 클래스로 타입 코드 대체(P4.1.3)는 클래스로의 코드 이관으로 자연스럽게 이어집니다.
- 메서드의 인라인화(P4.1.7): 프로그램의 가독성에 더 이상 도움을 주지 않는 메서드를 제거합니다.
- 메서드 전문화(P4.2.2): 메서드에서 불필요하고 문제가 있는 일반성을 제거합니다.
- 삭제 후 컴파일하기(P4.5.1): 인터페이스와 클래스의 전체 사용 범위를 알고 있는 경우 사용하지 않는 메서드를 인터페이스와 클래스에서 제거합니다.
- 유사 클래스 통합(P5.1.1): 일련의 상수 메서드에서 서로 다른 두 개 이상의 클래스를 통합합니다.
- if 문 결합(P5.2.1): 동일한 본문을 가진 if 문이 이어진 경우 if 문의 분기들을 결합해서 중복을 줄입니다.
- 전략 패턴의 도입(P5.4.2): if 문을 사용한 분기를 클래스를 인스턴화하는 것으로 대체합니다.
- 구현에서 인터페이스 추출(P5.4.4): 클래스에 대한 종속성을 인터페이스로 바꿉니다.
- getter와 setter 제거하기(P6.1.3): 기능을 데이터에 더 가깝게 옮겨 getter와 setter를 제거합니다.
- 데이터 캡슐화(P6.2.3): 변수와 관련된 불변속성을 지역화하고 응집력을 더 명확히 합니다.
- 순서 강제화(P6.4.1): 특정 순서대로 작업을 수행하는 것을 컴파일러가 보장하게 합니다.

14장을 소개하며 마무리



14.1 이 책의 여정을 돌아보며

- ✓ 이 책을 통해 더 많은 사람이 리팩터링에 접근하고 실행할 수 있기를 바랍니다.
- ✓ 코드 스멜, 컴파일러 활용, 기능 토글 등과 같은 복잡한 개념에 대한 진입 장벽을 낮추고 싶었습니다.
- ✓ 규칙, 리팩터링 패턴 및 각 장의 제목을 통해 여러분의 표현법이 풍부해졌기를 바랍니다.