Legacy Code Testing

오재훈 (ojh420@gmail.com)

Legacy Code 정의하기

“Code Without Test”

(Michael Feathers)

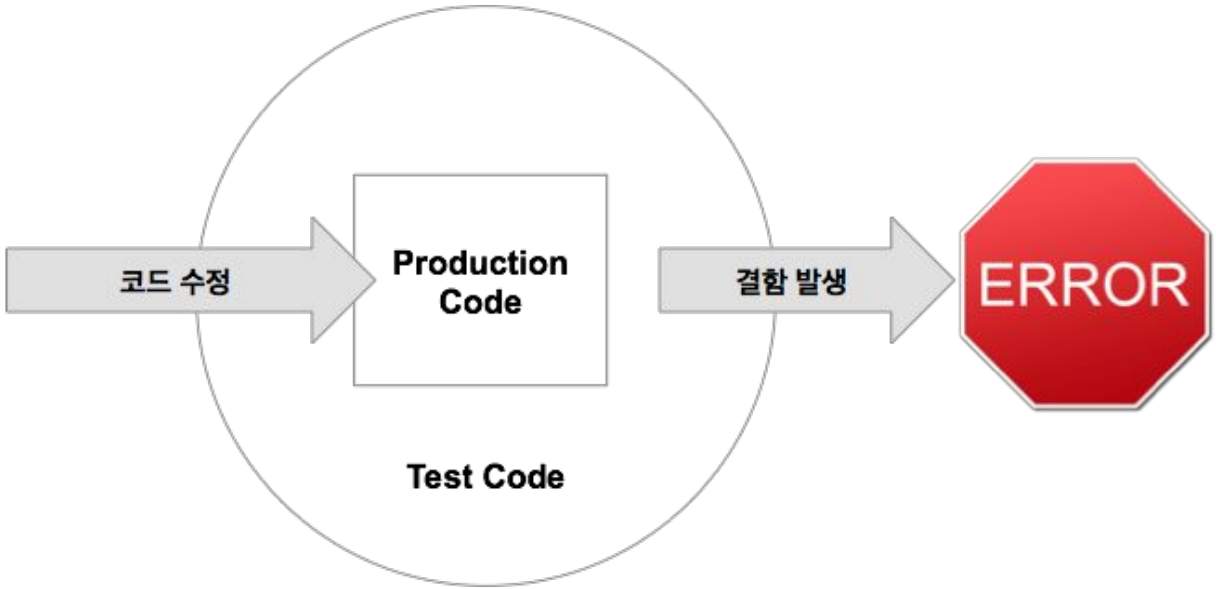
People are writing legacy code right now, **maybe on your project.**

(http://www.objectmentor.com/resources/articles/WorkingEffectivelyWithLegacyCode.pdf)

Working Effectively with Legacy Code

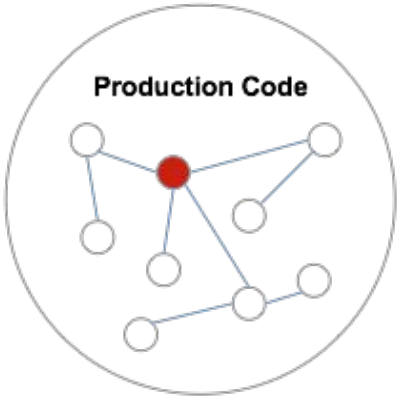
Michael Feathers

코드 수정



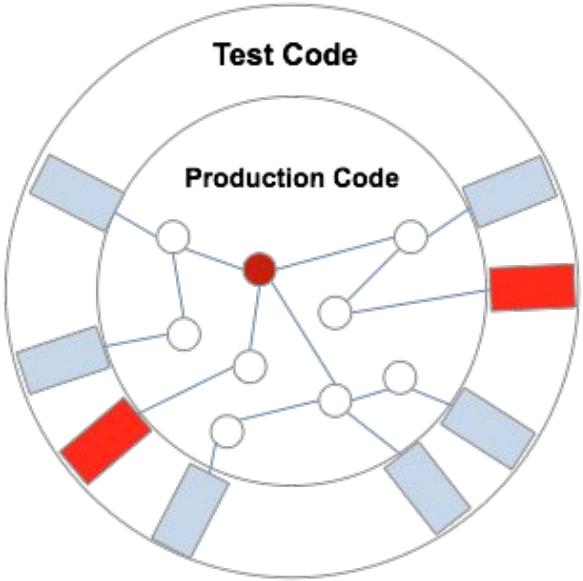
Test Code

Production Code Only



- 잠재적인 결함 내포
- 결함 진동 가능성
- 배포시 사후 처리 비용
- 개발자들의 자신감 결여

Production Code + Test Code



- Production Code 에 대한 보호막
- 결함 조기 발견
- 심리적 안정감
- 개발자들이 코드 수정에 자신감을 가짐

리팩토링이란?

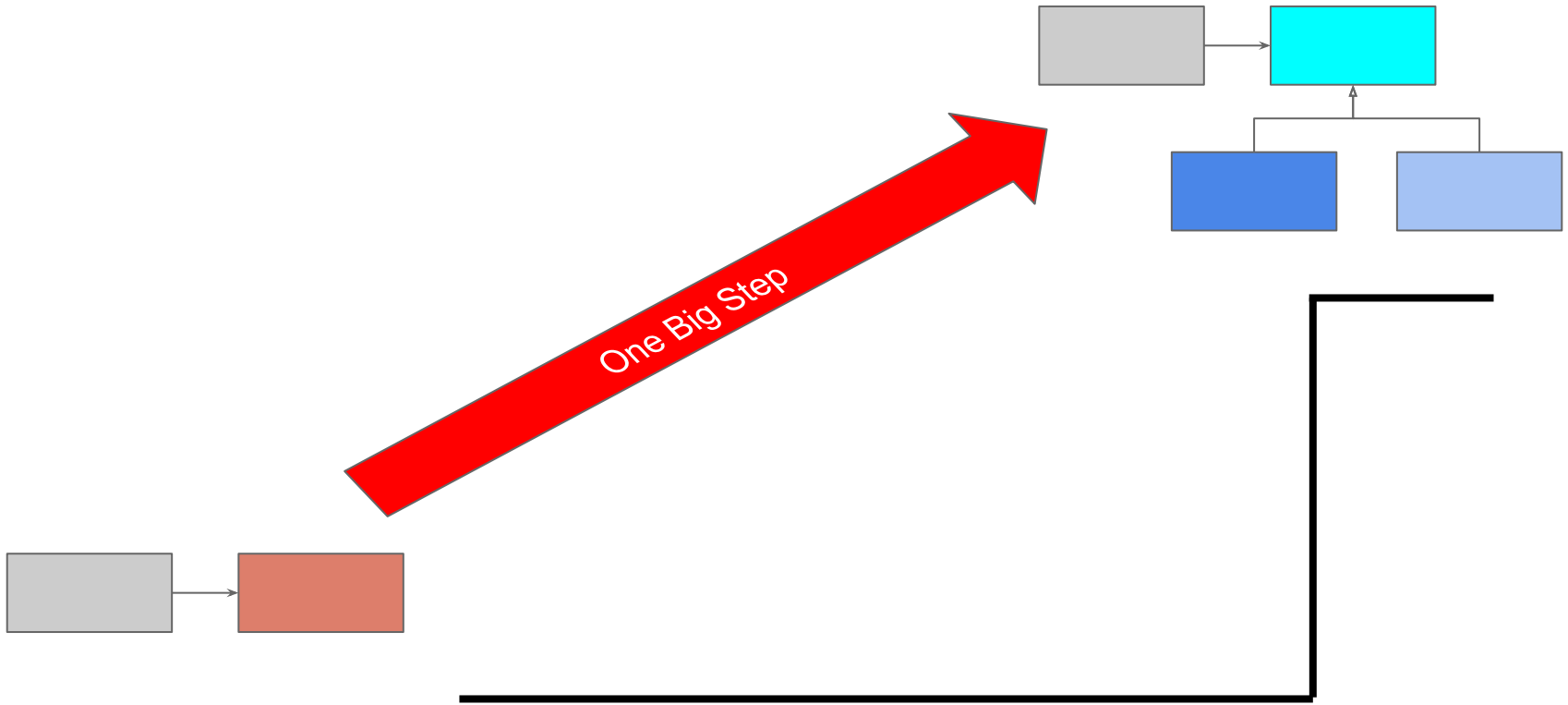
1.명사형

- 소프트웨어를 더 쉽게 이해할 수 있고,
- 적은 비용으로 수정할 수 있도록
- 겉으로 보이는 **동작의 변화 없이 내부 구조를 변경하는 것**

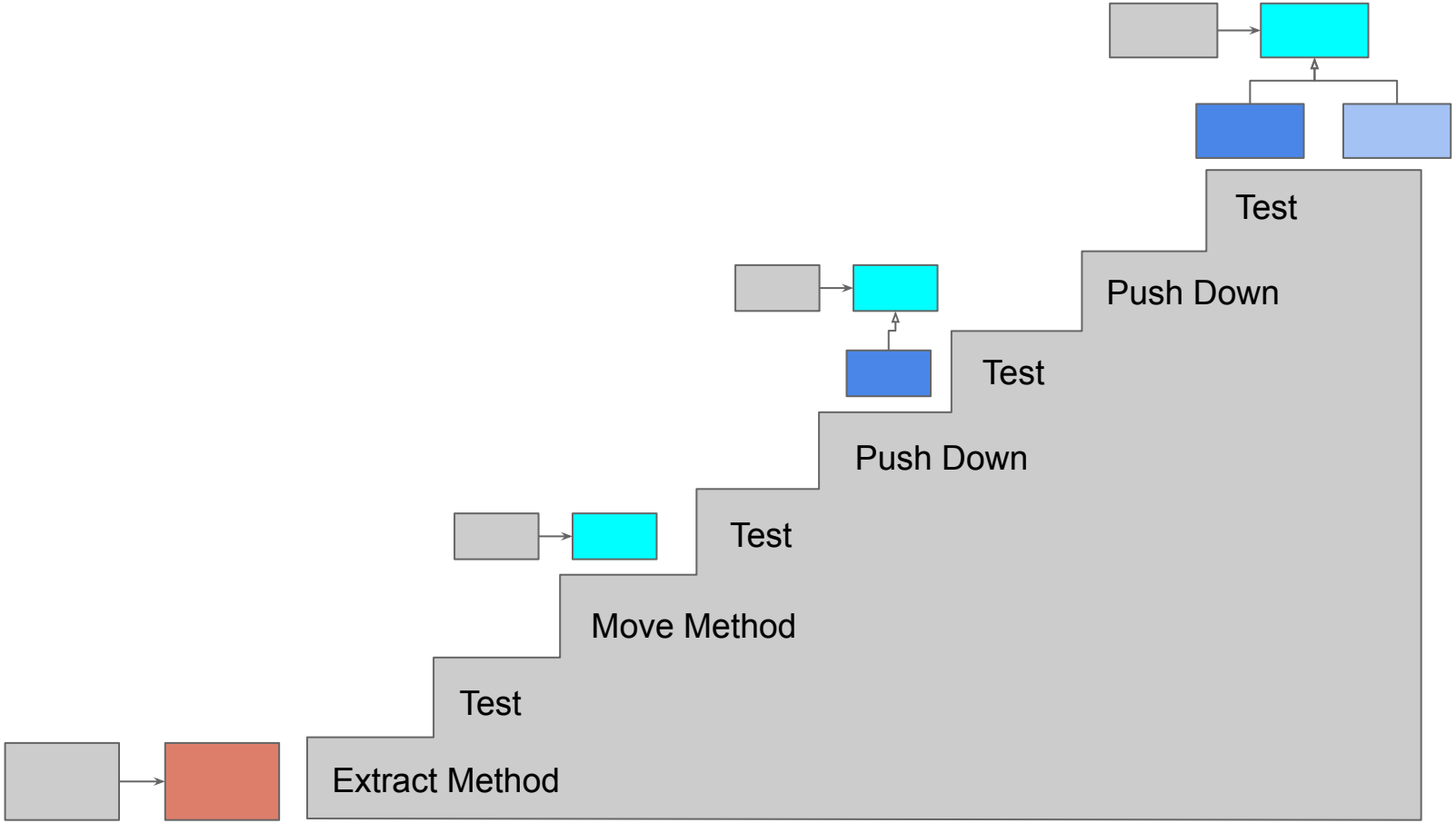
2.동사형(Refactor)

- 일련의 리팩토링을 적용하여 겉으로 보이는 동작의 변화없이
- 소프트웨어의 구조를 바꾸다.

레거시 코드 변경 방법 - One Big Step



레거시 코드 변경 방법 - Baby Step



Test 수준 결정하기

- Black Box Testing
- White Box Testing (Characterization Test)
- Gray Box Testing (Integration Test)

레거시 코드 단위 테스트 작성하기

1. 생성자 테스트 작성하기
2. 테스트 커버리지를 측정한다.
3. 테스트 되지 않은 코드를 커버할 수 있는 새로운 테스트를 작성한다. (Characterization Test)
4. 테스트 커버리지가 100%에 가까워질 때까지 테스트 코드를 작성한다.

무엇을 테스트할 것인가?

- 핵심 비지니스 로직
- 프리젠테이션은 배제

Testability (테스트 용이성)

- 테스트하기 쉬운 코드
  - 순수 함수
  - 사이드 이펙트가 자기 객체에 제한된 메소드
  - 다른 객체에 사이드 이펙트가 발생하는 메소드
  - 파일을 변경하는 메소드
- Seam
  - Legacy Code 의 Testability 를 높이기 위해 변경해야 할 Point

테스트 코드를 얼마만큼 작성해야 할까?

- 자신감
- Code Coverage 활용