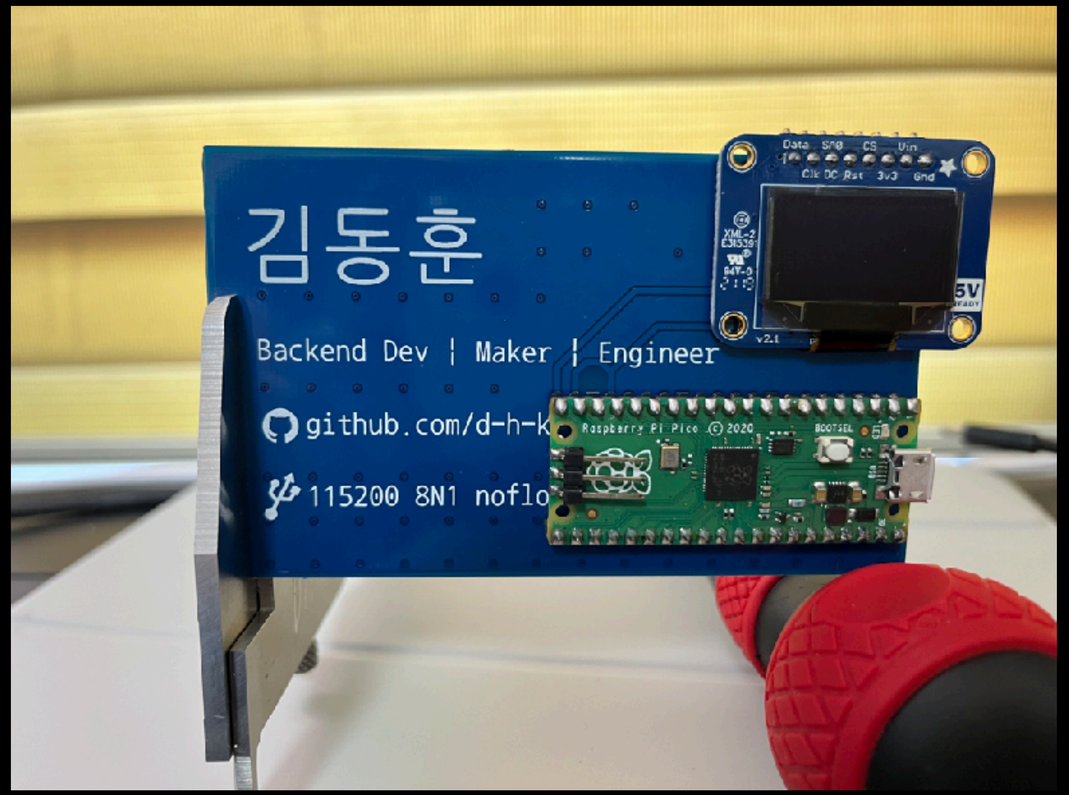
슬기로운 자바 코딩생활

PART1 : 클린코드 by 동훈
PART2 : 테스트코드 by 다빈

2023 . 09 . 02 - FossLightHub 오픈랩데이 발표

저를 소개합니다

- 백엔드 개발을 하고 있습니다
- 스프링/자바&코틀린 개발
  - Go Lang & k8s 관심
  - 도메인이 없어 항상 고민
  - 테크가 저의 도메인입니다
- 많은 실패를 경험하는중, 앞으로도 겪을 용기가 있고 싶은....?? 중꺽마



이 발표의 대상은요

- 클린 코드에 관심이 있으신 분
- 쥬니어 자바 개발자
- 개발자가 되기를 원하시는분
- 나의 코드가 지속가능하지 못하다고 느끼시는 분

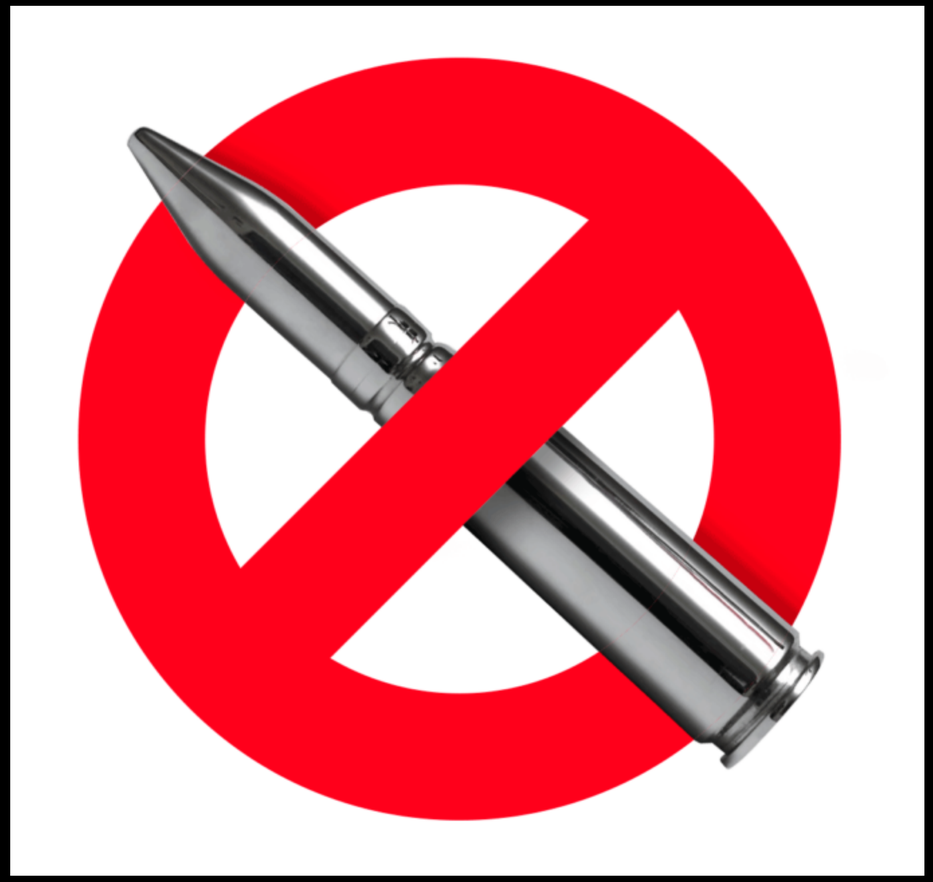
주의사항

모든 상황이 같을수 없음

- 코드 베이스라인
- 도메인

같이 일하는 동료/팀/업계의 컨센셔스

- 프로젝트 일정, 상황



여기 중에서 마음에 들고 적절한것만 취사선택 해주시면 감사하겠습니다!

Part 1 클린코드

Chapter 1 : what

Chapter 2 : why

Chapter 3 : how

What

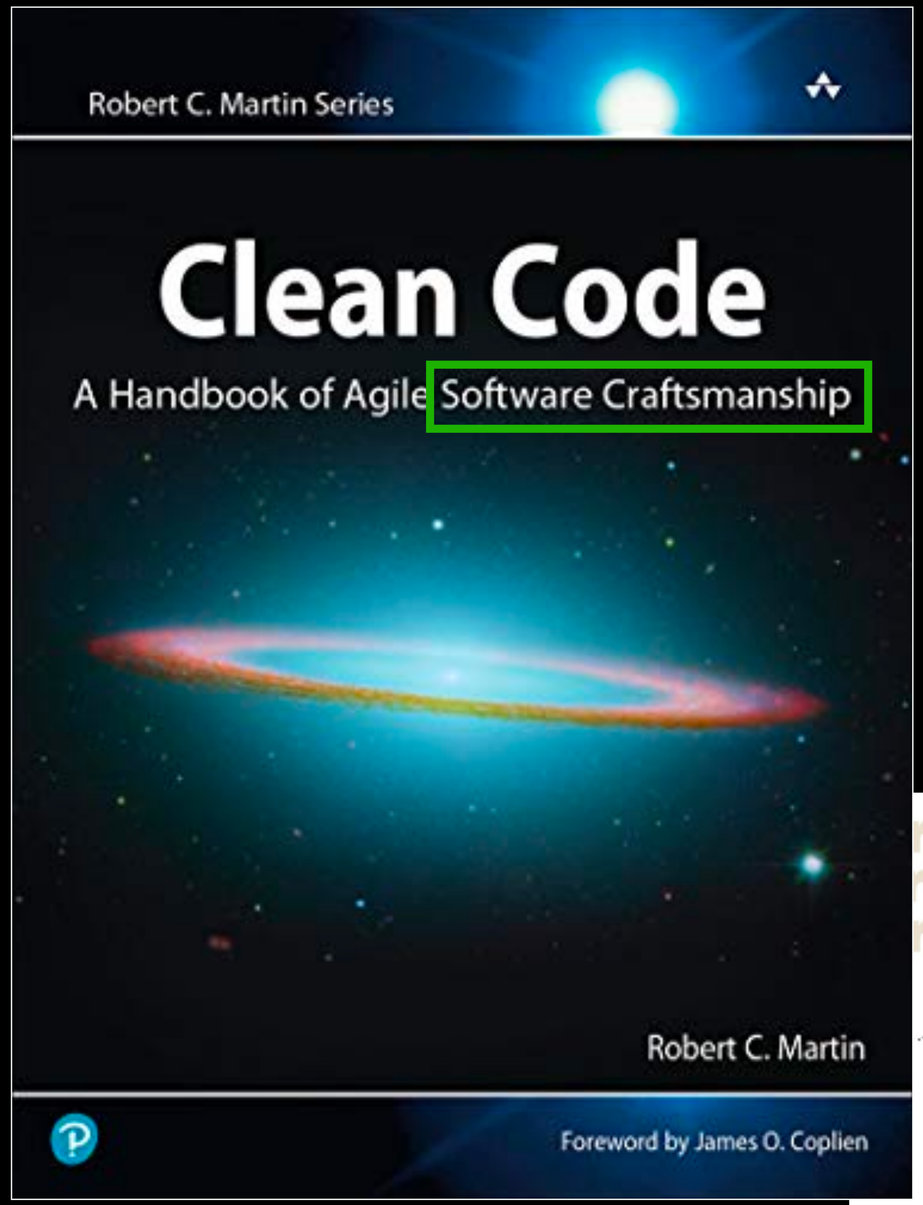
클린코드 ?

What

- 클린코드는 한단어로 정의하면 **craftsmanship** 이라고 생각합니다

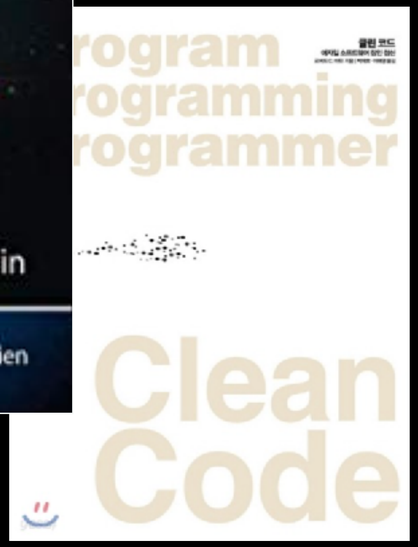
- **장인정신**을 가지고 소프트웨어를 개발하는 방법에 관한 이야기

- 한마디로 **“장인의 길”** - next step



부제목에 정답이 나와있음

원서도 언젠간 읽어보고싶습니다..!



번역서

What
소프트웨어 장인정신

Make it Work, Make it Right

커뮤니티 활동에 열정적인

작게 도전하고 빠르게 실패하는

정직하고 용기있는

기능을 빨리 구현할수 있는 방법

점진적이고 반복적으로 개선하고 숙련되어 목표를 향하는 마음가짐과 행동

끊임없이 변화하는 비지니스 요구사항을 수용할수 있는 유일한 방법

변화에 대응하는것 이상의, 계속해서 가치를 더해가는

주어진 문제를 가장 단순한 방법으로 풀어내는

Q) 소프트웨어 장인정신

에서 가장 먼것, 가까운것 고르기

주어진 일에 최선을 다하는

타인을 돕는데 주저하지 않는

소프트웨어 & IT 산업이 진화하는데 기여하는

끊임없이 훈련하고 자기계발하는

배우고, 가르치고, 공유하는데 열정적인

항상, 누구에게나 무언가를 배울 자세가 준비된

고객이 원하는 무엇이든 달성할 수 있도록 돕는

돌아간다가 이상의, 정교하고 솜씨있게 만들어진 작품을 만드는

주위 동료들이 일을 잘하도록 돕는

품질-비용-시간 (QCT) 에서 Trade-Off 를 결정하는

What

소프트웨어 장인정신

Make it Work, Make it Right

작게 도전하고 빠르게 실패하는

정직하고 용기있는

기능을 빨리 구현할수 있는 방법

점진적이고 반복적으로 개선하고 숙련되어 목표를 향하는

끊임없이 변화하는 비지니스 요구사항을 수용할수 있는 유일한 방법

커뮤니티 활동에 열정적인

주어진 문제를 가장 단순한 방법으로 풀어내는

변화에 대응하는것 이상의, 계속해서 가치를 더해가는

주어진 일에 최선을 다하는

놀랍게도 모두

타인을 돕는데 주저하지 않는

소프트웨어 & IT 산업이 진화하는데 기여하는

끊임없이 훈련하고 자기계발하는

배우고, 가르치고, 공유하는데 열정적인

항상, 누구에게나 무언가를 배울 자세가 준비된

고객이 원하는 무엇이든 달성할 수 있도록 돕는

돌아간다가 이상의, 정교하고 솜씨있게 만들어진 작품을 만드는

주위 동료들이 일을 잘하도록 돕는

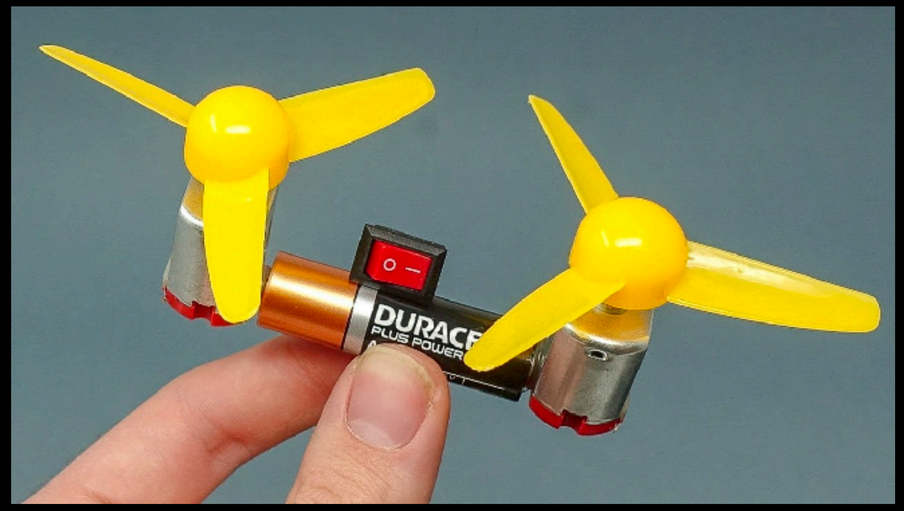
품질-비용-시간 (QCT) 에서 Trade-Off 를 결정하는

Why

클린코드
왜 필요한가?

Why : 클린코드가 필요한 이유

- 돌이켜보면..
  - 학교 코딩과제를 하거나, 토이프로젝트를 하거나, 코테 풀때
  - 클린코드는 그닥 매력적이지 않은 **경우**



아주아주 간단한 토이프로젝트

- 혼자서 한다
- 일회성이다
- 실패해도 부담이 없는

Why : 클린코드가 필요한 이유

- 그렇지 않은 곳 이라면..???
  - 팀단위 프로젝트인 **경우**
  - 수개월 ~ 수년 단위의 긴 호흡을 가지는 경우
  - 실패가 어려운 상황, 허용되지 않는 경우
  - 많은 사람들이 사용하는, 트래픽이 많은 **경우**
  - 영향을 줄수 있는 곳에서 사용되는 **경우**

한마디로

중요하다!

Why : 클린코드가 필요한 이유

이용자 수 많음

문제생기면 뉴스기사 나옴

하루라도 없으면 안됨

오늘도 쓰고 내일도 쓸꺼임

10년 뒤에도 쓸꺼같음

은행 App

정부 App

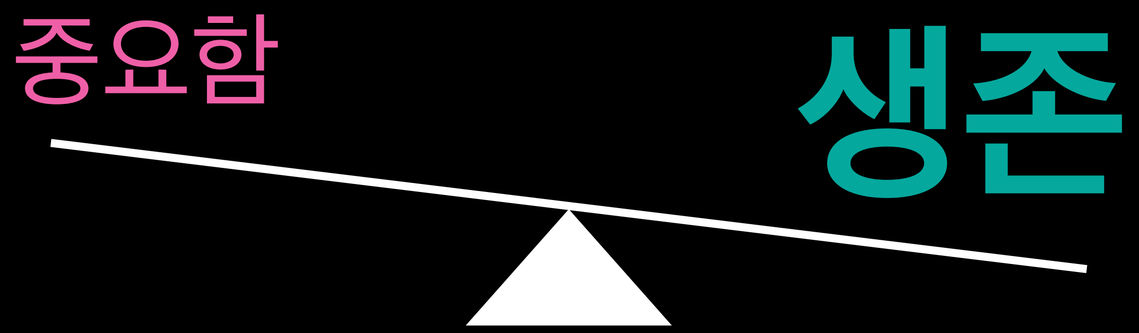
이외 IT 서비스..

중요하다!

여기서는 다루지 못한 중요한 서비스/기업들도 많습니다 !!

Why : 클린코드가 필요한 이유

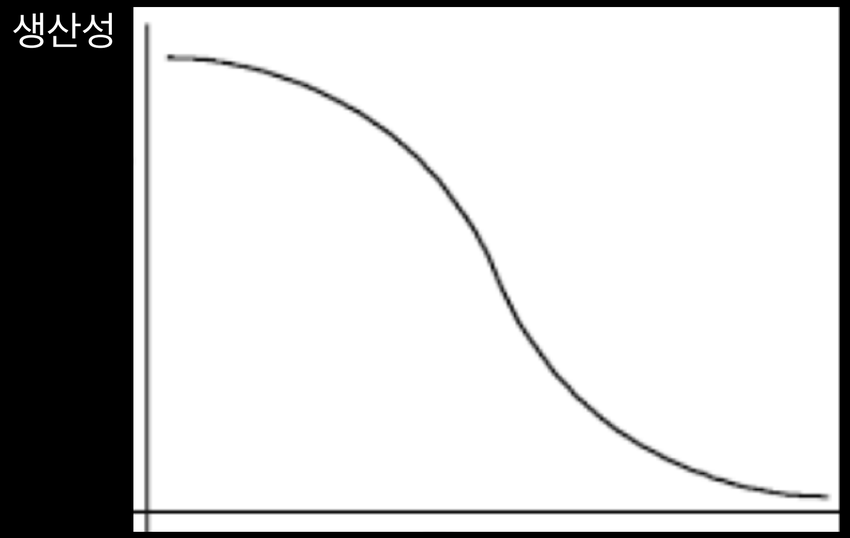
지속가능하게 생존하기 위해서



- “10년” 키워드 > 장시간 유지보수되고 운영할수 있어야 함
- 오랜기간 생존 하기 쉽지 않음 서비스, 조직속 개인, 기업 모두에게 해당

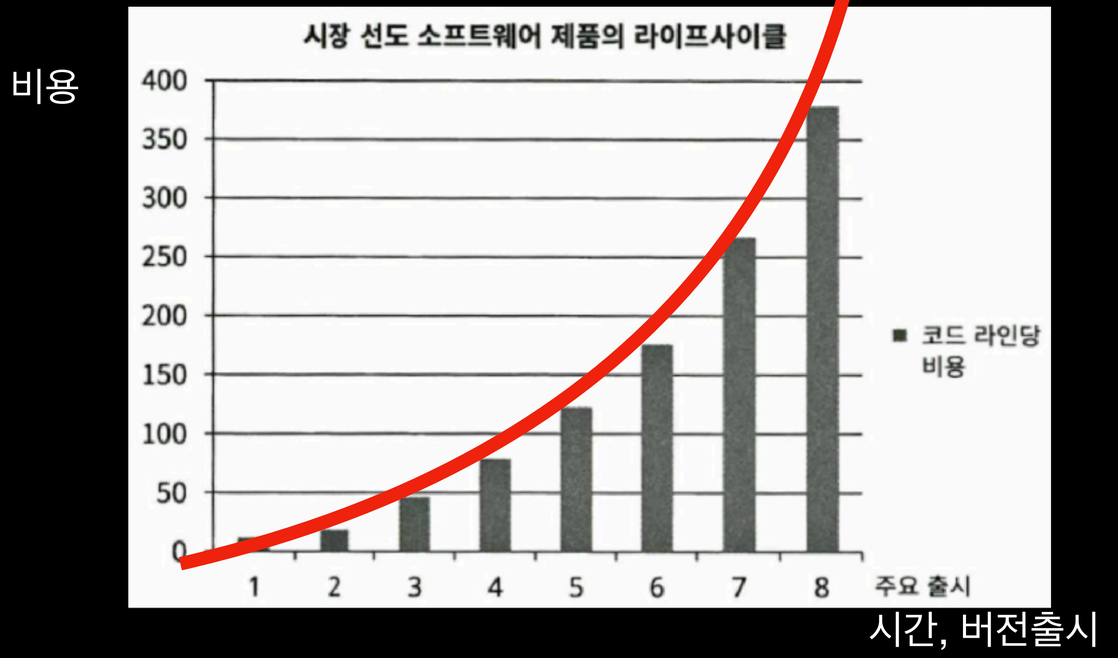
Why : 클린코드가 필요한 이유

지속가능하게 생존하기 위해서



시간이 흐를수록 소프트웨어 생산 효율성이 떨어진다

인건비가 증가하거나
출시가 느려지거나



기능추가가 느려지거나, 비용이 급증한다

Why : 클린코드가 필요한 이유

지속가능하며, 생존하기 위해

- 차세대 프로젝트가 과연 모든것을 해결해줄수 있을까?
  - 모든 차세대가 그런건 아니지만..
  - 그동안 쌓여왔던 모든 요구조건 반영
  - 기존 시스템을 대체할수 있는 안정성

Why : 클린코드가 필요한 이유

지속가능하며, 생존하기 위해

차세대 개발도 살려고 하는거다

클린코드는

기능추가에 오래걸리고

개발속도가 느리다 ?

Why : 클린코드가 필요한 이유

지속가능하며, 생존하기 위해

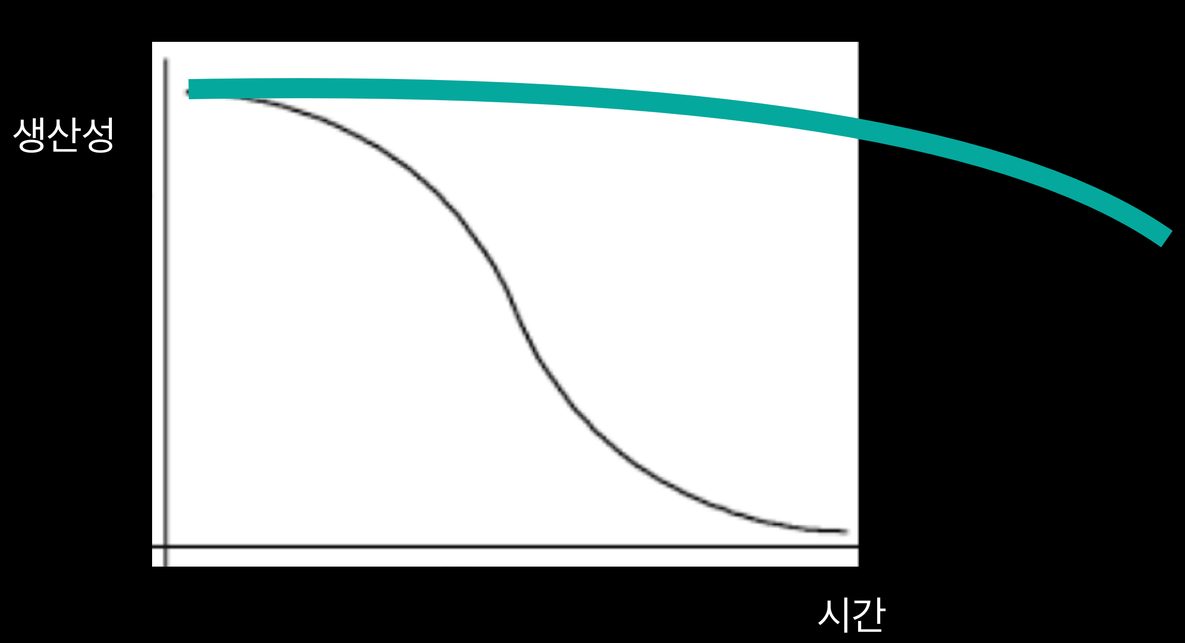
클린코드, 사실과 오해

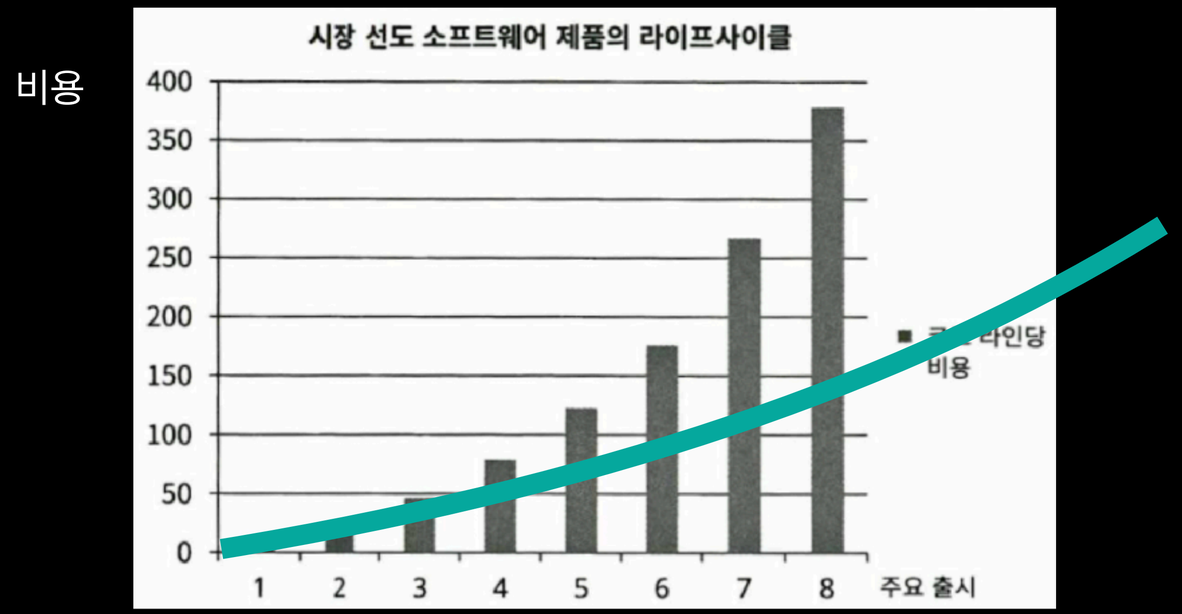
- 클린코드는 원하는 기능을 빨리 구현할수 있는 방법
- 끊임없이 변화하는 비지니스 요구사항을 수용할수 있는 유일한 방법

Why : 클린코드가 필요한 이유

지속가능하게 생존하기 위해서

클린코드가 필요한 이유 첫번째 “생존”





시간, 버전출시

Why : 클린코드가 필요한 이유 2

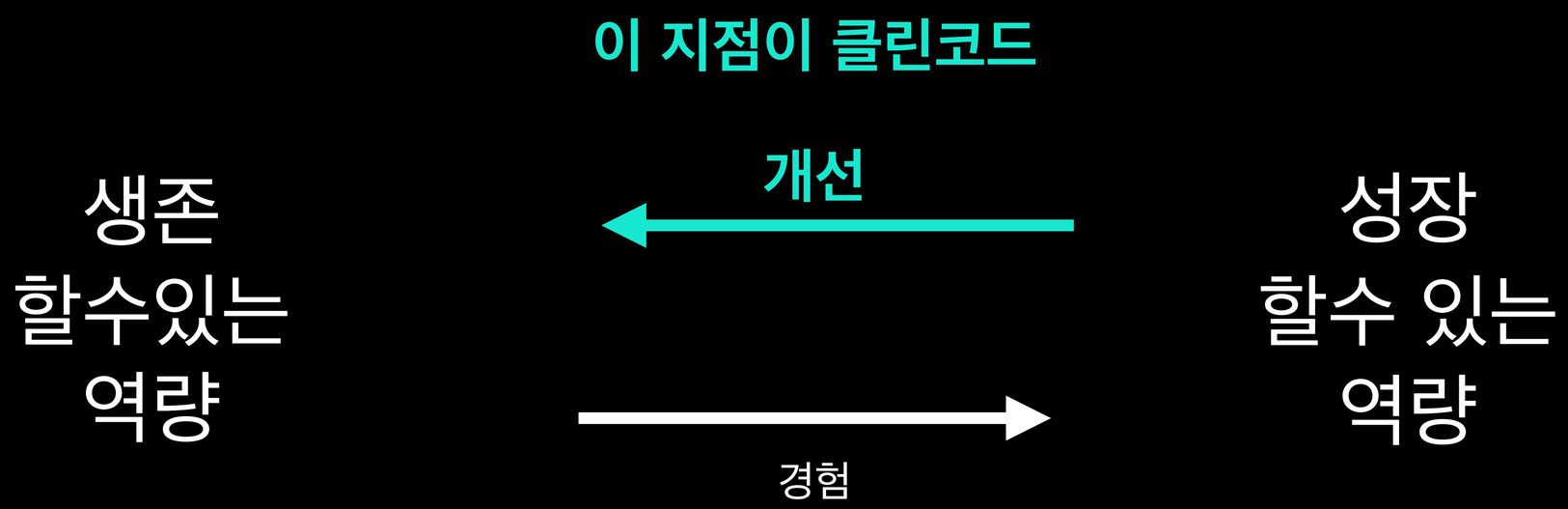
지속가능하며, 생존하기 위해

클린코드는

성장하기

위해서 필요하다

클린코드가 필요한 이유



빨리 가는 유일한 방법은, 언제나 코드를 최대한 깨끗하게 유지하는 습관이다 - 클린코드 7p

클린코드 실천하기

“어렵고 힘들게 성장하자”

이게 진짜진짜 정말정말 어려운 이유

- 혼자서는 절대 할수 없다
- 아군을 만들자
- 컨센셔스를 만들어나가자

빨리 가려면 혼자 가고, 멀리 가려면 함께 가라 - 아프리카 속담

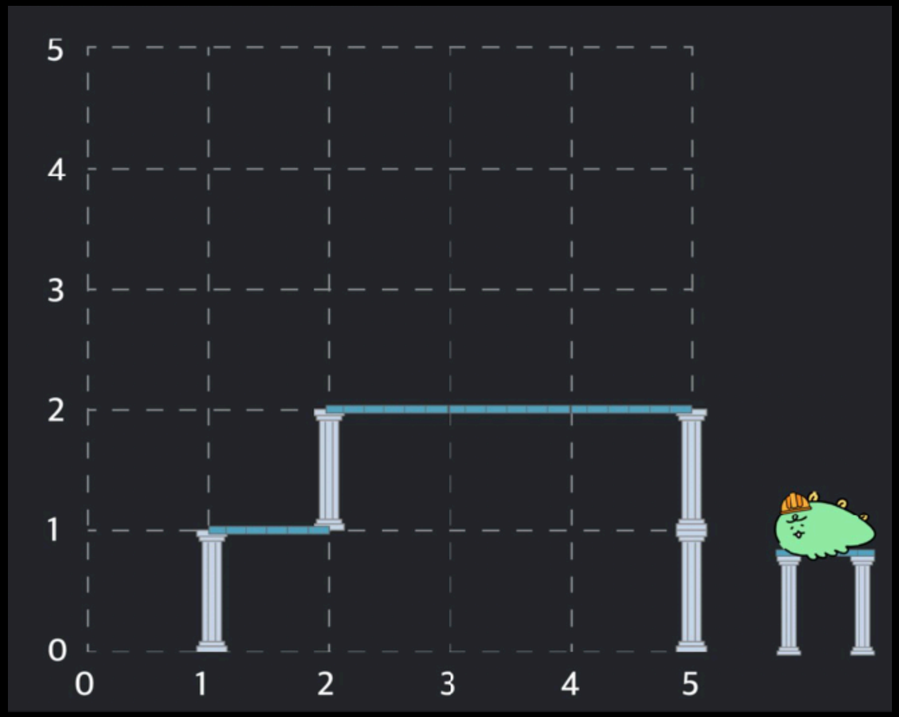
How

클린코드
어떻게 하나?

클린코드 실천하기

하나의 개념에는 하나의 단어만 사용하기

- 하나의 개념에는 하나의 단어만 사용하기!
  - 예시문제 : 2020 KAKAO 기출 - 기둥과 보 설치

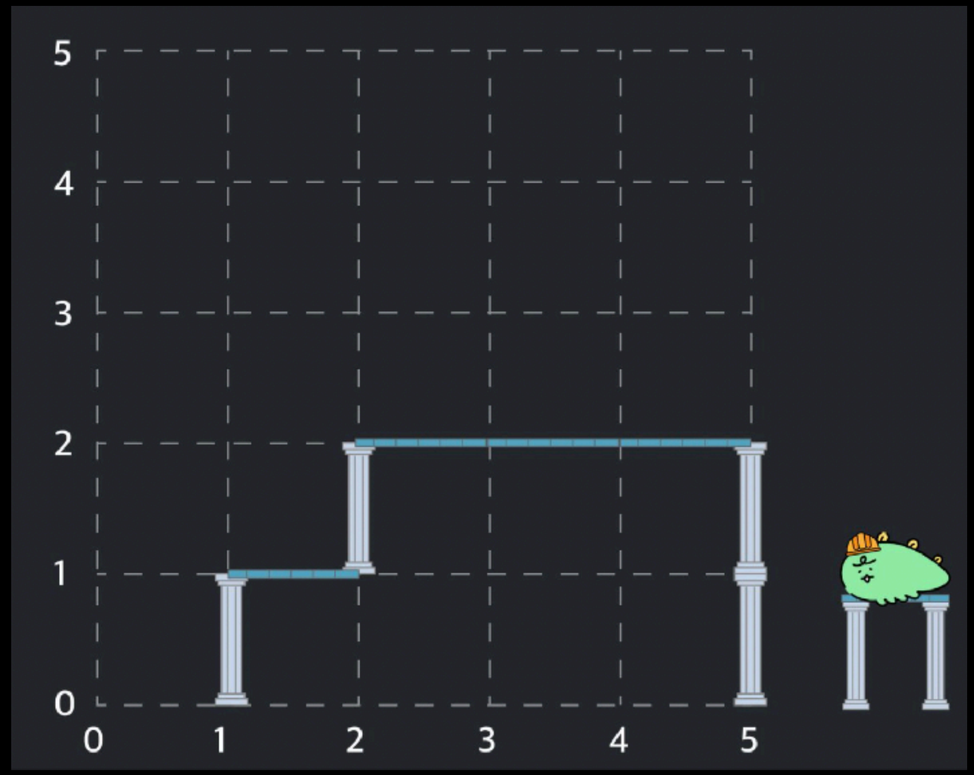


- https://school.programmers.co.kr/learn/courses/30/lessons/60061

클린코드 실천하기

하나의 개념에는 하나의 단어만 사용하기

- Row, column 개념에 대해서 한 단어만 사용하기
  - Row, Column
  - X, Y
  - Vertical, Horizontal
- 다양하게 사용하고 상황에 따라 용어를 결정



- https://school.programmers.co.kr/learn/courses/30/lessons/60061

클린코드 실천하기

하나의 개념에는 하나의 단어만 사용하기

- Row, column 개념에 대해서 한 단어만 사용하기
  - 저는 first, second 를 사용하고 있습니다 (코테 한정)

JDK17 버전을 쓰고있어 record class 사용 권유

```
private class MyPair {
    5 usages
    final int first;
    5 usages
    final int second;

    6 usages
    public MyPair(int first, int second) {
        this.first = first;
        this.second = second;
    }
}
```

현 좌표

```
for (int firstIndex = 0; firstIndex < firstLimit; firstIndex++) {
    for (int secondIndex = 0; secondIndex < secondLimit; secondIndex++) {
        visited[firstIndex][secondIndex] += 1;
    }
}
```

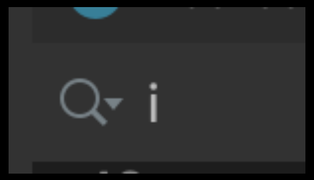
범위 제한 다음 좌표

```
if (rangeAvabilable(firstLimit, secondLimit, nextFirst, nextSecond) &&
    movable(nextFirst, nextSecond, matrix)
) {
    return new MyPair(nextFirst, nextSecond);
}
```

클린코드 실천하기
적절한 이름 붙여주기

- 작명을 잘 해줘야 합니다 : 좋은 이름 만드는 법
  - 적절하지 못한 이름을 쓰면…

```
Integer i = null;
```



딱 하나를 찾아줘…

클린코드 실천하기
적절한 이름 붙여주기

- 작명을 잘 해줘야 합니다 : 좋은 이름 만드는 법
  - 검색하기 쉬운
  - 의도를 명확하게 표현하는
    - 무슨 의도로 선언하고, 만든건지
    - 어떤 데이터를 담는지, 어떤 기능인지
  - 도메인 키워드를 사용하기
    - 도메인 키워드 사전이 있다면 Best !

아하! 아이템의 인덱스를 저장하기 위한 변수로군!

```
Integer indexOfItem = null;
```

클린코드 실천하기

함수 분리하기

```
private MyPair availableNextPosV0(MyPair currentPos, char[][] matrix) {

    int firstLimit = matrix.length;
    int secondLimit = matrix[0].length;
    int nextFirst = diffs.get(direction % 4).first + currentPos.first;
    int nextSecond = diffs.get(direction % 4).second + currentPos.second;

    if (((firstLimit > nextFirst) &&
        (secondLimit > nextSecond) &&
        (0 <= nextFirst) &&
        (0 <= nextSecond)) &&
        (matrix[nextFirst][nextSecond] == '.')
    ) {
        return new MyPair(nextFirst, nextSecond);
    }

    direction++;
    return currentPos;
}
```

Q) 이 코드에서 가장
클린코드와 거리가 먼 부분은 ???

클린코드 실천하기

함수 분리하기

```
private MyPair availableNextPosV0(MyPair currentPos, char[][] matrix) {

    int firstLimit = matrix.length;
    int secondLimit = matrix[0].length;
    int nextFirst = diffs.get(direction % 4).first + currentPos.first;
    int nextSecond = diffs.get(direction % 4).second + currentPos.second;

    if (((firstLimit > nextFirst) &&
        (secondLimit > nextSecond) &&
        (0 <= nextFirst) &&
        (0 <= nextSecond)) &&
        (matrix[nextFirst][nextSecond] == '.')
    ) {
        return new MyPair(nextFirst, nextSecond);
    }

    direction++;
    return currentPos;
}
```

뭘 하는건지 이해하기 어려움

클린코드 실천하기
함수 분리하기

```
private MyPair availableNextPosV0(MyPair currentPos, char[][] matrix) {

    int firstLimit = matrix.length;
    int secondLimit = matrix[0].length;
    int nextFirst = diffs.get(direction % 4).first + currentPos.first;
    int nextSecond = diffs.get(direction % 4).second + currentPos.second;

    if (((firstLimit > nextFirst) &&
        (secondLimit > nextSecond) &&
        (0 <= nextFirst) &&
        (0 <= nextSecond)) &&
        (matrix[nextFirst][nextSecond] == '.')
    ) {
        return new MyPair(nextFirst, nextSecond);
    }

    direction++;
    return currentPos;
}
```

인덱스를 벗어나는지 범위를 검사한다

‘.’ 문자인지를 검사한다

클린코드 실천하기

함수 분리하기

```
private MyPair availableNextPosV0(MyPair currentPos,

    int firstLimit = matrix.length;
    int secondLimit = matrix[0].length;
    int nextFirst = diffs.get(direction % 4).first +
    int nextSecond = diffs.get(direction % 4).second

    if (((firstLimit > nextFirst) &&
        (secondLimit > nextSecond) &&
        (0 <= nextFirst) &&
        (0 <= nextSecond)) &&
        (matrix[nextFirst][nextSecond] == '.')
    ) {
        return new MyPair(ne
    }

    direction++;
    return currentPos;
}
```

```
private static boolean rangeAvailable(
    int limitFirst, int limitSecond,
    int nextFirst, int nextSecond
) {
    return ((limitFirst > nextFirst) &&
        (limitSecond > nextSecond) &&
        (0 <= nextFirst) &&
        (0 <= nextSecond)
    );
}
```

인덱스를 벗어나는지 범위를 검사한다

‘.’ 문자인지를 검사한다

```
private boolean movable(int nextFirst, int nextSecond, char[][] matrix) {
    return matrix[nextFirst][nextSecond] == '.';
}
```

클린코드 실천하기

함수 분리하기

```java
private static boolean rangeAvailable(
    int limitFirst, int limitSecond,
    int nextFirst, int nextSecond
) {
    return ((limitFirst > nextFirst) &&
        (limitSecond > nextSecond) &&
        (0 <= nextFirst) &&
        (0 <= nextSecond)
    );
}
```

범위 체크

```java
if (rangeAvailable(firstLimit, secondLimit, nextFirst, nextSecond) &&
    movable(nextFirst, nextSecond, matrix)
) {
    return new MyPair(nextFirst, nextSecond);
}
```

이동 가능여부 체크

```java
private boolean movable(int nextFirst, int nextSecond, char[][] matrix) {
    return matrix[nextFirst][nextSecond] == '.';
}
```

클린코드 실천하기

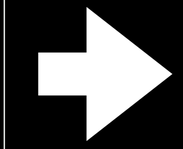
함수 분리하기

```
1 usage
private MyPair availableNextPos(MyPair currentPos,char[][] matrix ) {

    int firstLimit = matrix.length;
    int secondLimit = matrix[0].length;
    int nextFirst = diffs.get(direction % 4).first + currentPos.first;
    int nextSecond = diffs.get(direction % 4).second + currentPos.second;

    if (rangeAvailable(firstLimit, secondLimit, nextFirst, nextSecond) &&
        movable(nextFirst, nextSecond, matrix)
    ) {
        return new MyPair(nextFirst, nextSecond);
    }

    direction++;
    return currentPos;
}
```

클린코드 실천하기
함수 분리하기

- 함수로 적절하게 분리하볏
  - 맥락을 파악하고 흐름을 이해하기가 훨씬 좋습니다

```
if (rangeAvailable(firstLimit, secondLimit, nextFirst, nextSecond) &&
    movable(nextFirst, nextSecond, matrix)
) {
    return new MyPair(nextFirst, nextSecond);
}
```



```
if (((firstLimit > nextFirst) &&
    (secondLimit > nextSecond) &&
    (0 <= nextFirst) &&
    (0 <= nextSecond)) &&
    (matrix[nextFirst][nextSecond] == '.')
) {
    return new MyPair(nextFirst, nextSecond);
}
```

클린코드 실천하기
함수 분리하기

- 함수안에서 오직 두 단계의 들여쓰기만 한다 >> 익숙해지면 한단계
- 함수는 10줄을 넘어가지 않게 설계한다
- 입력인자를 3개 이하로 설계한다
- 한 가지만 해라.
  - 함수는 한 가지를 해야 한다.
  - 그 한 가지를 잘 해야 한다.
  - 그 한 가지만 해야 한다.
- 함수 당 추상화 수준은 하나로
- 서술적인 이름을 사용하라! 이름이 길어도 괜찮다.

클린코드 실천하기

코드 일관성

```
@Component
@RequiredArgsConstructor
public class ItemSagaProcessor {
    private final PlatformTransactionManager platformTransactionManager;
    private final ItemRepository itemRepository;
    private final ItemService service;
    public void run() {
        List<Item> items =  service.findAllV2();
        items.stream().map(item -> item.getQuantity()).filter(i -> i > 100).filter(i -> i < 0).sorted(Comparator.naturalOrder()).findFi
        new TransactionTemplate(platformTransactionManager).execute(
            new TransactionCallback<Object>() {
                public Object doInTransaction(TransactionStatus status) {
                    status.setRollbackOnly();
                    itemRepository.deleteAll();
                    return null;
                }});
        PlatformTransactionManager transactionManager1 = new HibernateTransactionManager();
        PlatformTransactionManager transactionManager2 = new AbstractPlatformTransactionManager() {
            @Nullable
            private SessionFactory sessionFactory;
            @Nullable
            private DataSource dataSource;
            @Override
            protected Object doGetTransaction() throws TransactionException {
                logger.info("Get Transaction Object");
                return null;
            }
            @Override
```

Q) 이 코드에서 가장 클린코드스럽지 못한부분은?

클린코드 실천하기

코드 일관성

```
@Component
@RequiredArgsConstructor
public class ItemSagaProcessor {
    private final PlatformTransactionManager platformTransactionManager;
    private final ItemRepository itemRepository;
    private final ItemService service;
    public void run() {
        List<Item> items =  service.findAllV2();
        items.stream().map(item -> item.getQuantity()).filter(i -> i > 100).filter(i -> i < 0).sorted(Comparator.naturalOrder()).findFi
        new TransactionTemplate(platformTransactionManager).execute(
            new TransactionCallback<Object>() {
                public Object doInTransaction(TransactionStatus status) {
                    status.setRollbackOnly();
                    itemRepository.deleteAll();
                    return null;
                }});
        PlatformTransactionManager transactionManager1 = new HibernateTransactionManager();
        PlatformTransactionManager transactionManager2 = new AbstractPlatformTransactionManager() {
            @Nullable
            private SessionFactory sessionFactory;
            @Nullable
            private DataSource dataSource;
            @Override
            protected Object doGetTransaction() throws TransactionException {
                logger.info("Get Transaction Object");
                return null;
            }
            @Override
```

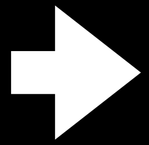
가독성 이전에 코드가 다 보이지도 않는다

밀도가 너무 높다

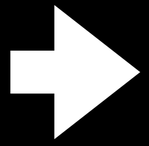
클린코드 실천하기

코드 일관성

```
Integer value = items.stream().map(item -> item.getQuantity()).filter(i -> i < 100).filter(i -> i > 0)
    .sorted(Comparator.naturalOrder()).findFirst().orElseThrow();
```



```
Integer value1 = items.stream()
    .map(item -> item.getQuantity())
    .filter(i -> i < 100)
    .filter(i -> i > 0)
    .sorted(Comparator.naturalOrder())
    .findFirst().orElseThrow();
```



```
Integer value2 = items.stream()
    .map(Item::getQuantity)
    .filter(i -> i > 0)
    .filter(i -> i < 100)
    .min(Comparator.naturalOrder())
    .orElseThrow();
```

클린코드 실천하기

코드 일관성

밀도가 너무 높다

```java
@Component
@RequiredArgsConstructor
public class ItemSagaProcessor {
    private final PlatformTransactionManager platformTransactionManager;
    private final ItemRepository itemRepository;
    private final ItemService service;
    public void run() {
        List<Item> items =  service.findAllV2();
        items.stream().map(item -> item.getQuantity()).filter(i -> i > 100).filter(i -> i < 0).sorted(Comparator.naturalOrder()).findFi
        new TransactionTemplate(platformTransactionManager).execute(
            new TransactionCallback<Object>() {
                public Object doInTransaction(TransactionStatus status) {
                    status.setRollbackOnly();
                    itemRepository.deleteAll();
                    return null;
                }});
        PlatformTransactionManager transactionManager1 = new HibernateTransactionManager();
        PlatformTransactionManager transactionManager2 = new AbstractPlatformTransactionManager() {
            @Nullable
            private SessionFactory sessionFactory;
            @Nullable
            private DataSource dataSource;
            @Override
            protected Object doGetTransaction() throws TransactionException {
                logger.info("Get Transaction Object");
                return null;
            }
            @Override
```

클린코드 실천하기

코드 일관성

공백 몇줄만

추가해줘도

훨씬 보기 편함

```java
@Component
@RequiredArgsConstructor
public class ItemSagaProcessorV2 {

    private final PlatformTransactionManager platformTransactionManager;
    private final ItemRepository itemRepository;
    private final ItemService service;

    public void run() {

        List<Item> items = service.findAllV2();
        Integer value = items.stream()
            .map(Item::getQuantity)
            .filter(i -> i > 0)
            .filter(i -> i < 100)
            .min(Comparator.naturalOrder())
            .orElseThrow();

        new TransactionTemplate(platformTransactionManager).execute(
            new TransactionCallback<Object>() {
                public Object doInTransaction(TransactionStatus status) {
                    status.setRollbackOnly();
                    itemRepository.deleteAll();
                    return null;
                }
            });

        PlatformTransactionManager transactionManager1 = new HibernateTransactionManager();
```

클래스 시작

클래스 인스턴스 변수

메서드 시그니쳐

매서드 내부 변수 선언

뭔가 다른 동작..

클린코드 실천하기

코드 일관성

- 형식 맞추기
  - 상하 밀도를 일정하게 유지한다
  - 좌우 폭을 일정하게 유지한다
- 변수는 사용하는 곳에서 가까운곳에서 선언한다
- 자바 규약을 따라서 순서대로 배치한다
  - 클래스 - 상수 - 인스턴스 변수 - 생성자 - 메서드- 종속메서드
- 인스턴스 변수 간 개념이 유사한 것들끼리 가까이 배치한다

클린코드 실천하기

Java Tips

- print 문 대신 log 사용하기

Synchronized 키워드

```
Prints a String and then terminate the line. This method b
and then println().
Params: x – The String to be printed.
public void println( @Nullable String x) {
    if (getClass() == PrintStream.class) {
        writeln(String.valueOf(x));
    } else {
        synchronized (this) {
            print(x);
            newLine();
        }
    }
}
```

- Optional.get 사용하지 말자! 대신 .orElse() .orThrow() 등을 추천한다

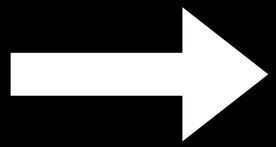
이펙티브 자바 아이템 55

클린코드 실천하기

Java Tips

- for문 보다 foreach 문을 쓰자

```java
public void findAllV1() {
    List<Item> items = itemRepository.findAll();
    for (int i = 0; i < items.size(); i++) {
        Item item = items.get(i);
        item.doSomthing();
    }
}
```



```java
public void findAllV2() {
    List<Item> items = itemRepository.findAll();
    for (Item item : items) {
        item.doSomthing();
    }
}
```

지금까지

- 혹시.. 이런생각 하고 계신가요...??

너무 당연한 이야기인데...?

클린코드 실천하기
정리

클린코드 = 소프트웨어 장인정신

지속가능한 코드로 개발

하나의 개념에는 하나의 단어
적절하게 함수 분리하기
일관성 있는 코드 작성하기
테스트코드 작성하기

고민하고 학습하기

사용하는 프레임워크 배우기
언어 기본기도 충실하게
다른언어도 학습해보기
커뮤니티 활동하기

꾸준한 연습과 실천

토이프로젝트에서
코딩테스트에서
나 혼자서도 시작
내가 작성하는 모든 코드에서

클린코드 실천하기
정리

클린코드 = 소프트웨어 장인정신

지속가능한 코드로 개발

지식

하나의 개념에는 하나의 단어
적절하게 함수 분리하기
일관성 있는 코드 작성하기
테스트코드 작성하기

고민하고 학습하기

태도

사용하는 프레임워크 배우기
언어 기본서도 충실하게
다른언어도 학습해보기
커뮤니티 활동하기

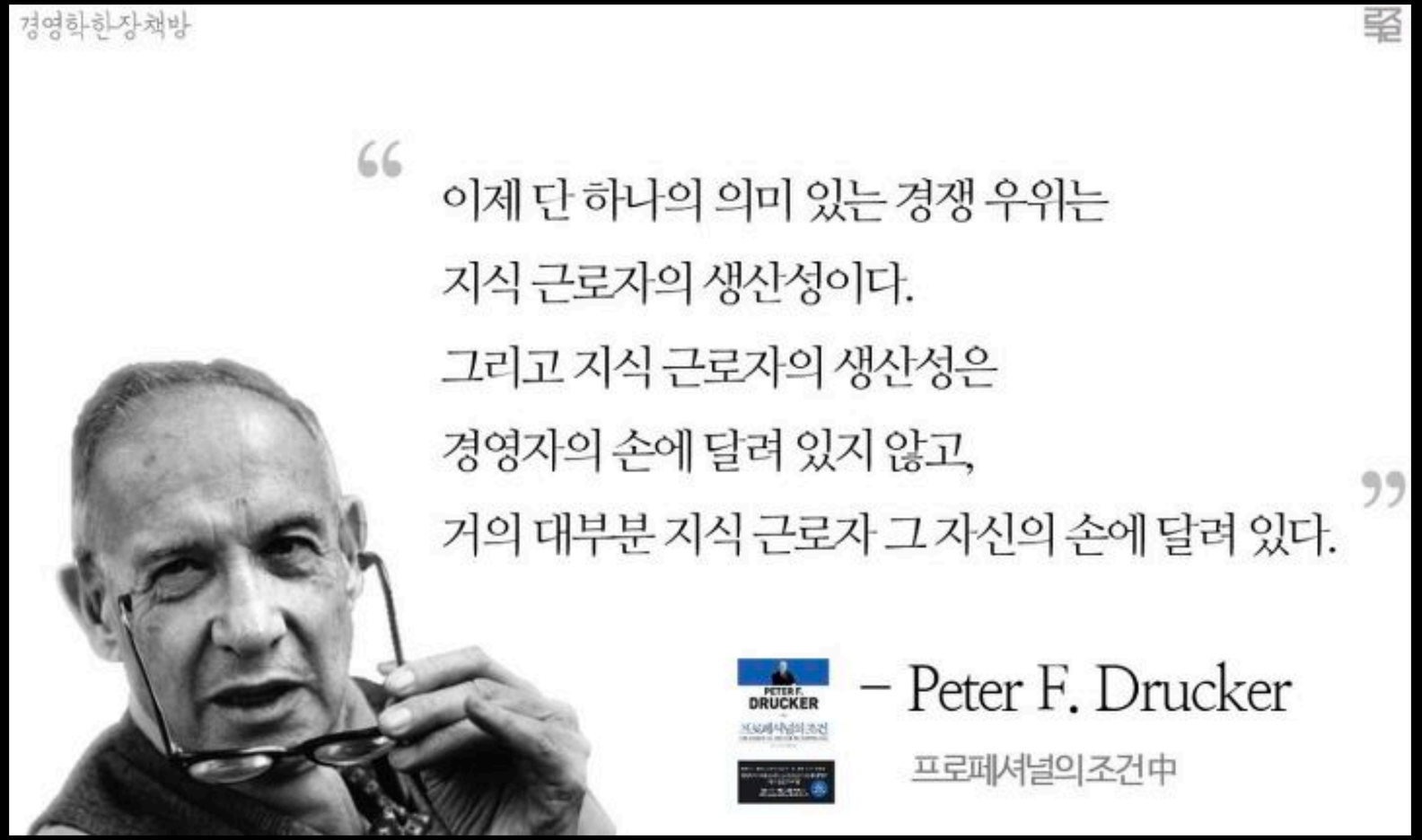
꾸준한 연습과 실천

행동

토이 프로젝트 해보기
코딩테스트 풀어보기
나 혼자서도 시작
내가 작성하는 모든 코드에서

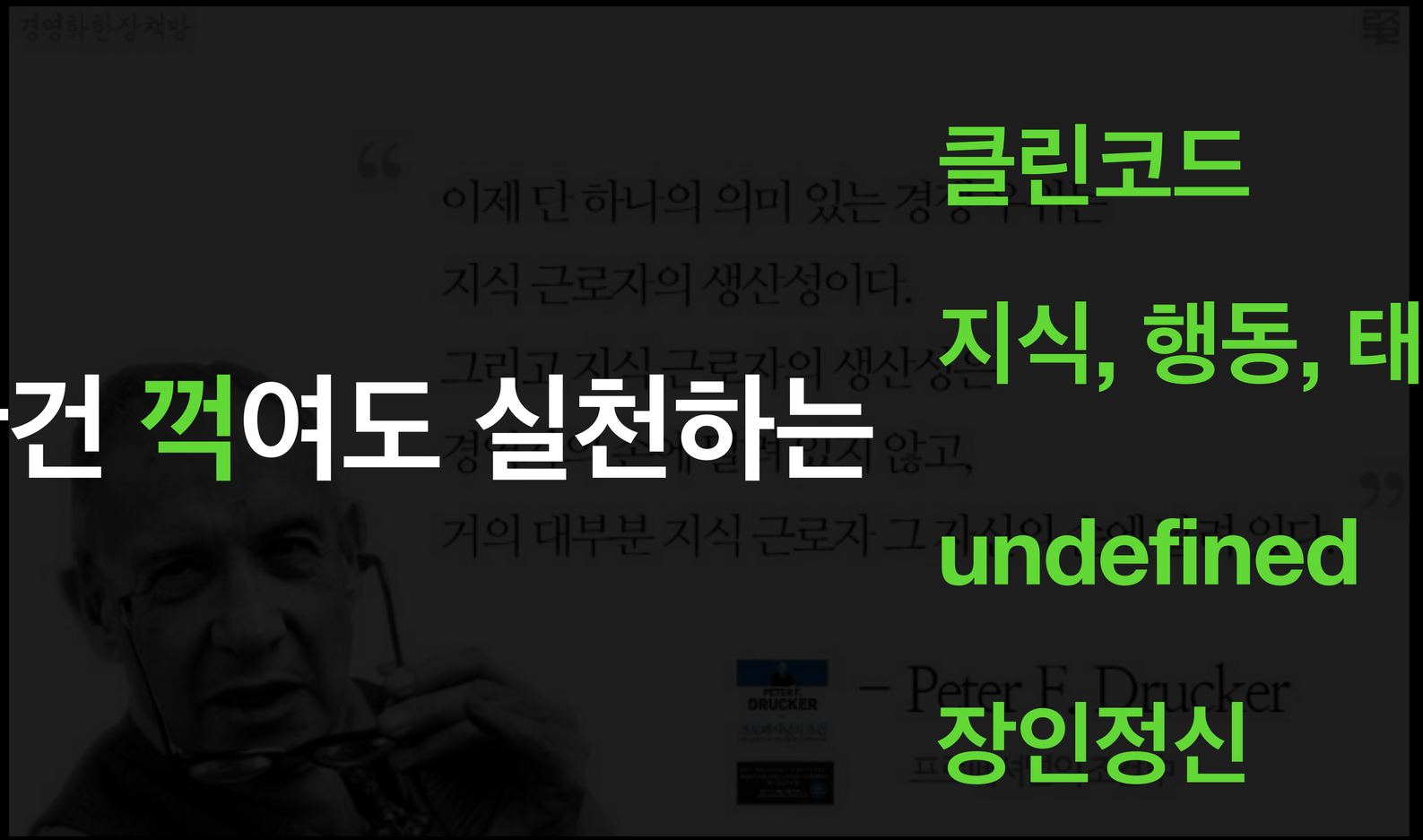
클린코드는 개발자 개개인이 실천해야하는것

- 생존하고, 성장하면서 실천하면서 팀 전체의 생산성이 올라간다



클린코드는 개발자 개개인이 실천해야하는것

- 생존하고, 성장하면서 실천하면서 팀 전체의 생산성이 올라간다



중요한건 꺽여도 실천하는

클린코드

지식, 행동, 태도

undefined

장인정신

중요한건 꺽여도 **실천**하는

undefined

클린코드는 오늘부터 1일

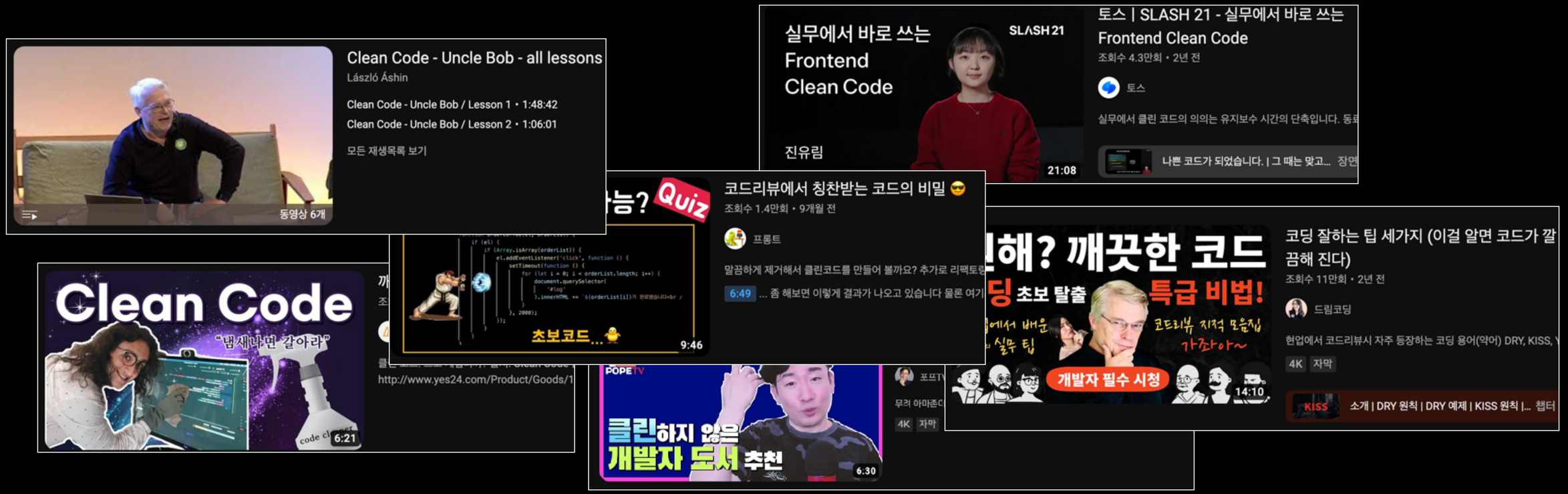
저의 발표는

여기까지

감사합니다

TMI

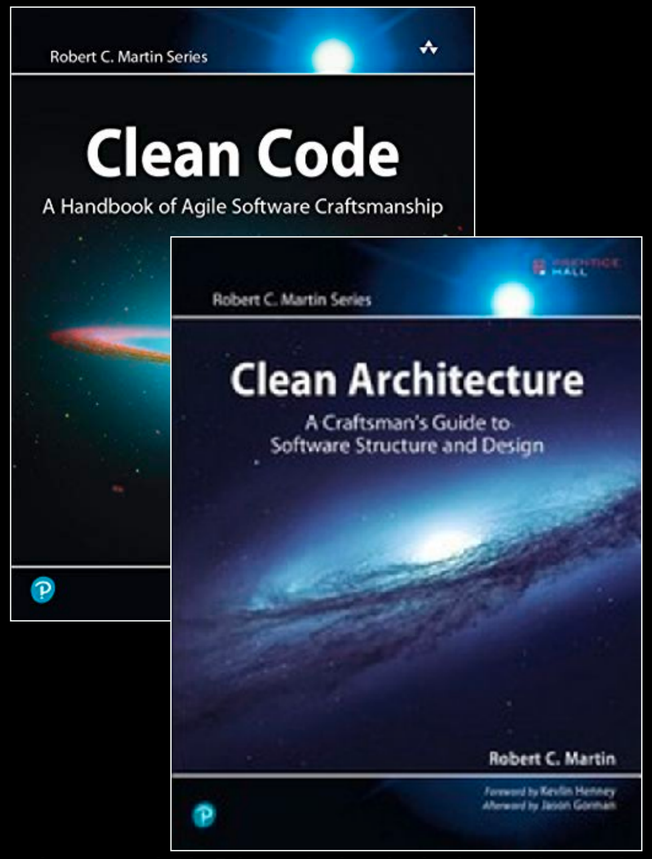
- 클린코드가 무조건 정답! 은 아닐수 있습니다
- 이외에도 클린코드에는 다양한 의견이 존재합니다



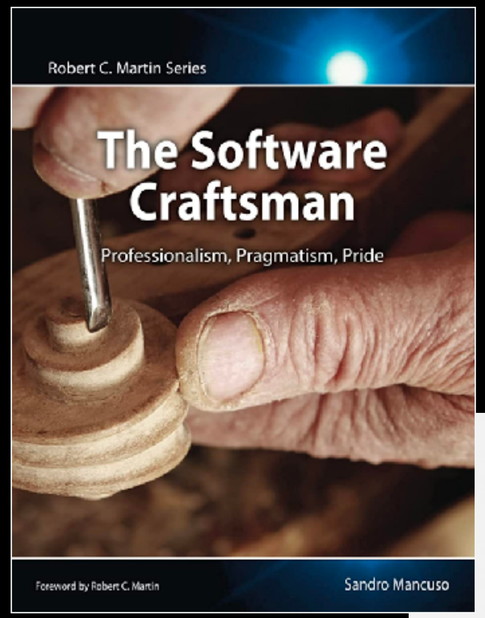
주어진 상황과 문제를 해결하기 위해 다양한 이론이 있고, 더욱 다양한 논의가 있습니다

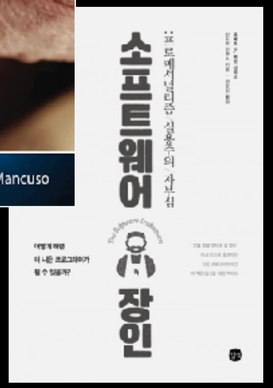
TMI

오늘의 이야기를 있게 해준.... var inputs : Resource[ ]



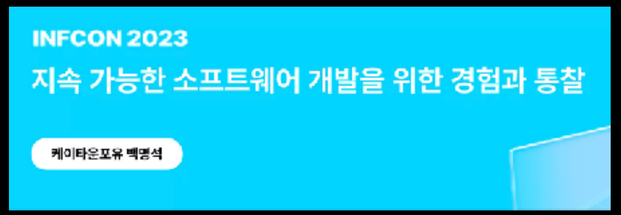
너무 유명한 책들이라 번역서 소개는 생략





번역서

Next step 강의



인프콘 2023 발표 - 백명섭님

계속해서