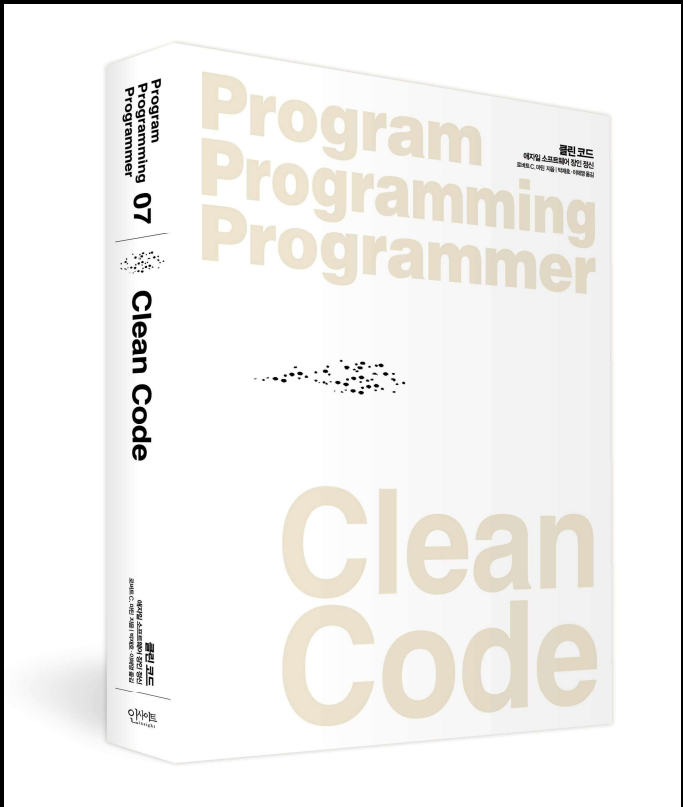
좋은 함수, 나쁜 주석

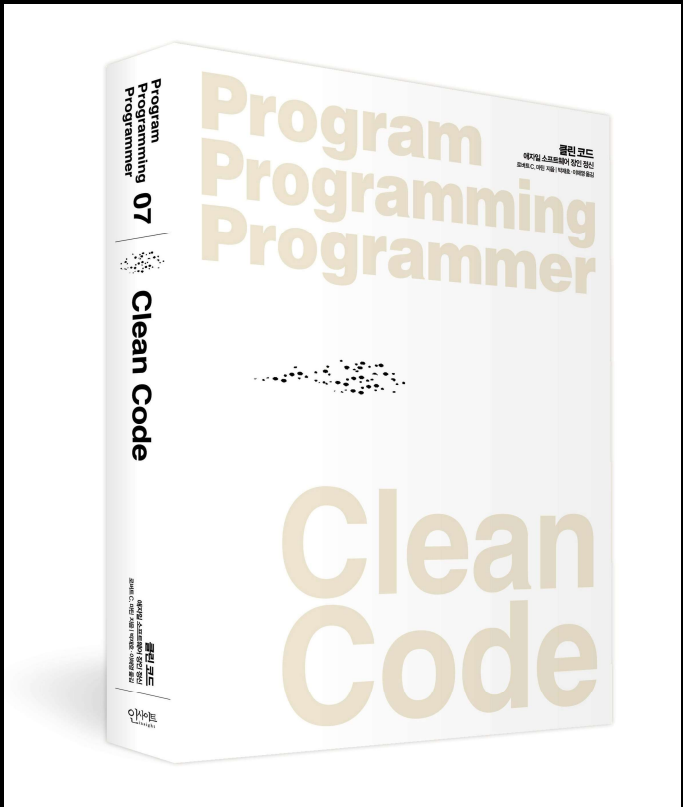
J105 신준수

Clean Code – 좋은 함수



- 작게 만들어라
- 한가지만 해라
- 함수 당 추상화 수준은 하나로
- 서술적인 이름을 사용하라
- 함수의 인수
- 부수 효과를 일으키지 마라
- 명령과 조회를 분리하라
- 오류 코드보다 예외를 허용하라

Clean Code – 좋은 함수

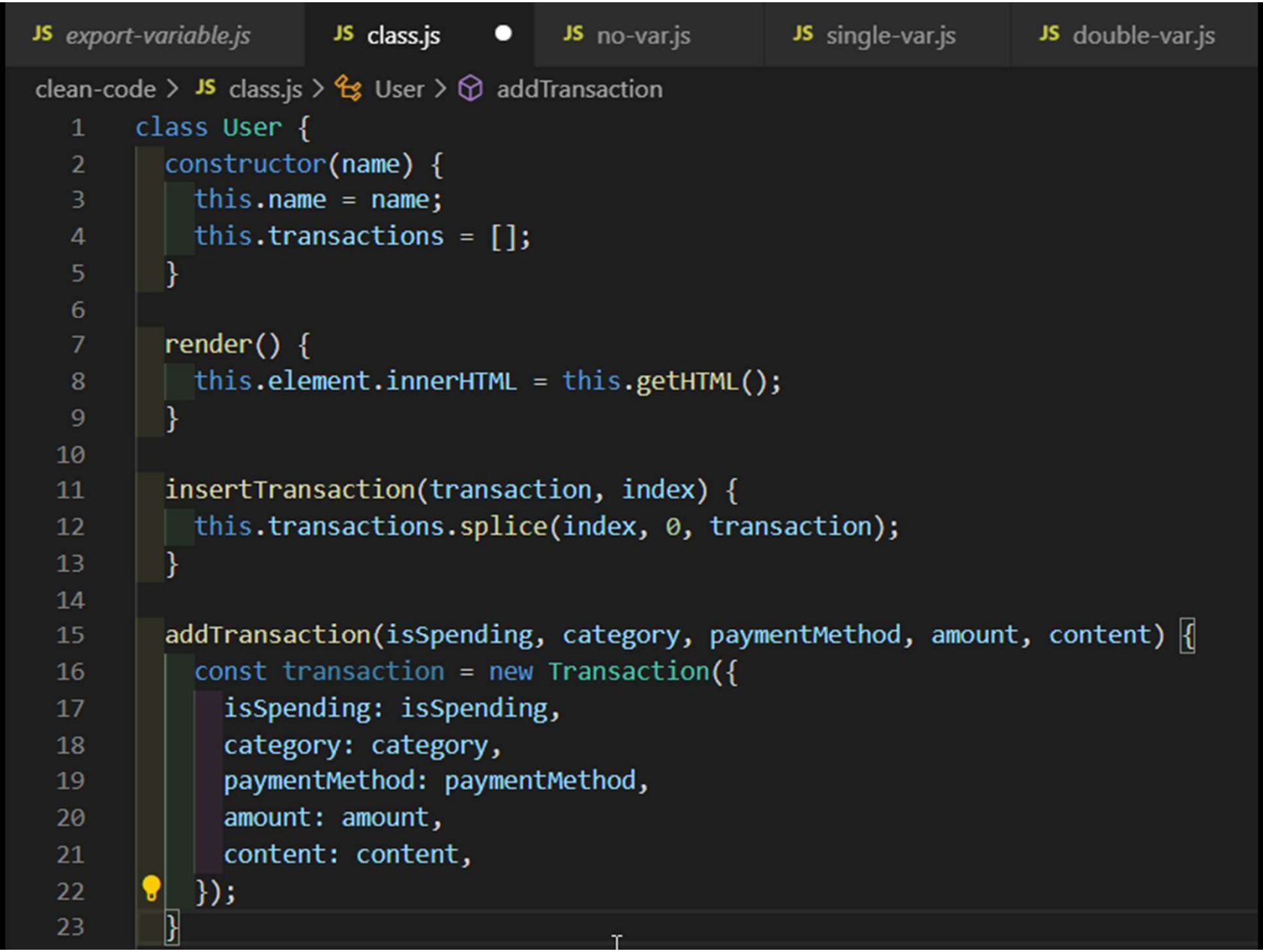


- 작게 만들어라
- 한가지만 해라
- 함수 당 추상화 수준은 하나로
- 서술적인 이름을 사용하라
- **함수의 인수**
- 부수 효과를 일으키지 마라
- **명령과 조회를 분리하라**
- 오류 코드보다 예외를 허용하라

내 코드에서 함수의 인수 개수 평균은?

Slido #35512

https://app.sli.do/event/jerlvrh6


export-variable.js
class.js
no-var.js
single-var.js
double-var.js
clean-code > class.js > User > addTransaction
class User {
constructor(name) {
this.name = name;
this.transactions = [];
}
render() {
this.element.innerHTML = this.getHTML();
}
insertTransaction(transaction, index) {
this.transactions.splice(index, 0, transaction);
}
addTransaction(isSpending, category, paymentMethod, amount, content) {
const transaction = new Transaction({
isSpending: isSpending,
category: category,
paymentMethod: paymentMethod,
amount: amount,
content: content,
});
}

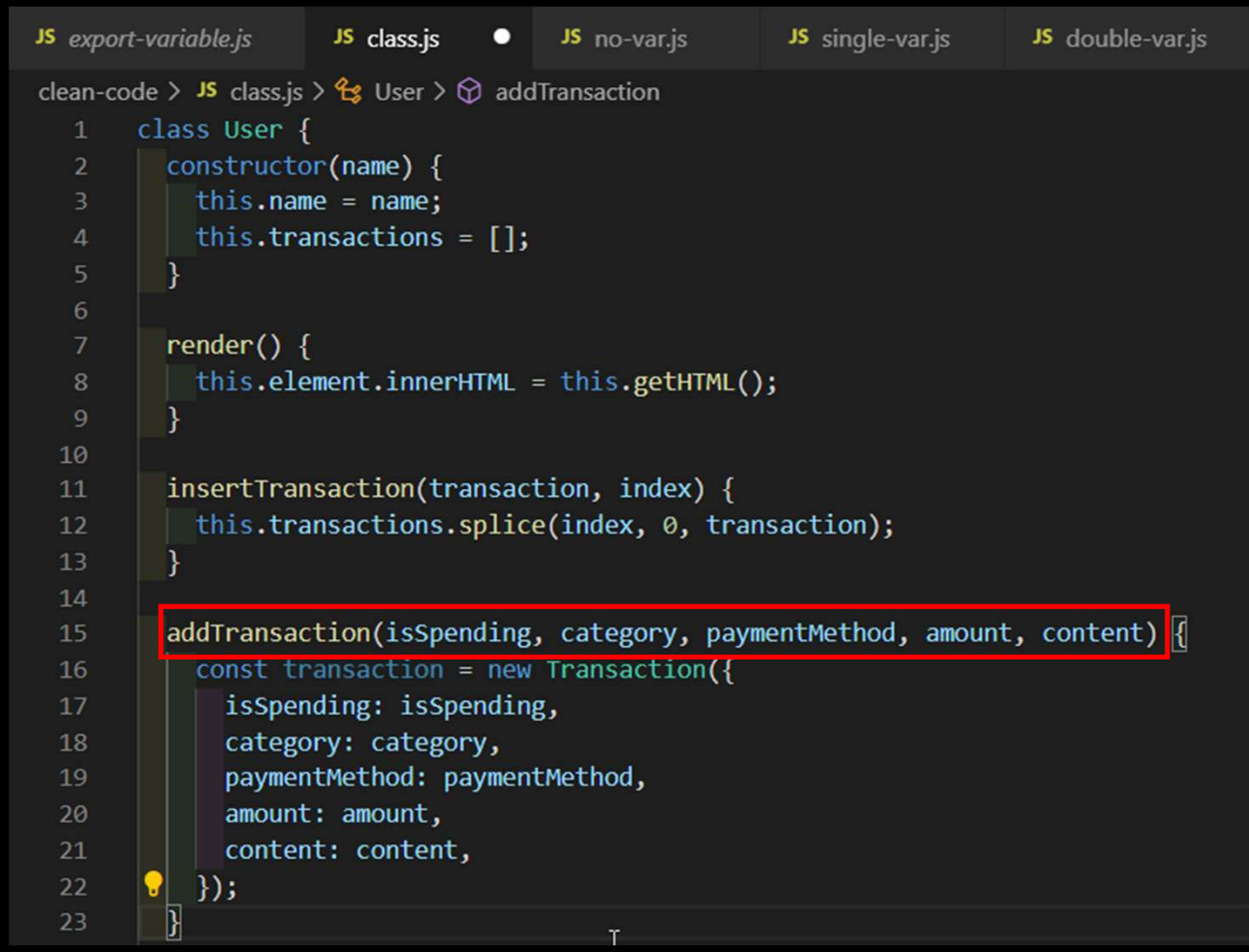
삼항함수 ~

- User.addTransaction(vars...)의 인수 순서를 기억해 보세요!

```
47  // 유저가 새로운 거래내역 form을 입력하였음
48  // User Model에 새로운 transaction을 추가
49
50  user.addTransaction(?, ?, ?, ?, ?);
```

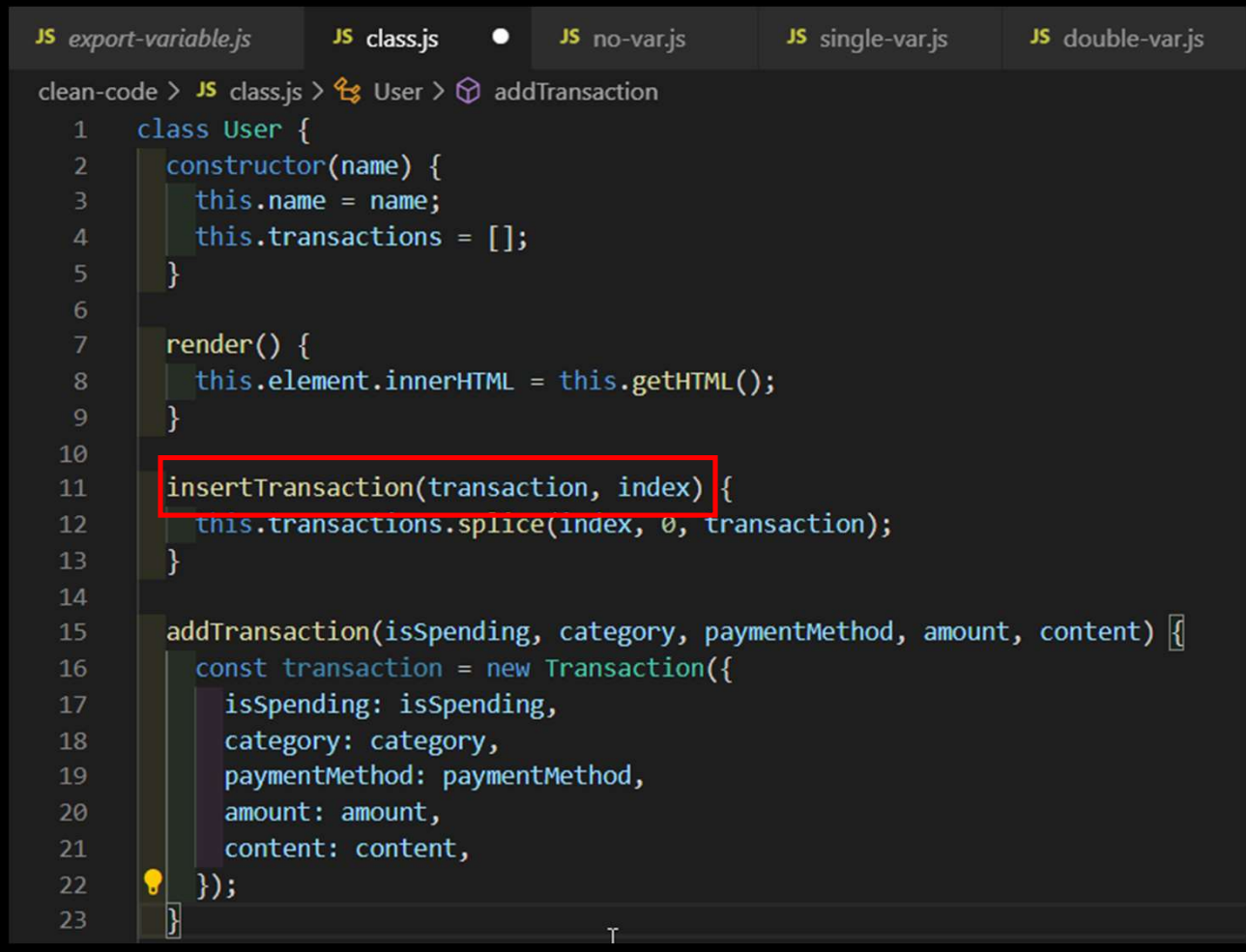
삼항함수 ~

- 함수명 만으로 어떤 인수가 들어가는 지 명확하지 않음
- 인수의 순서를 상식이나 함수명으로 추론하기 어려움
- 다시 돌아가서 소스코드를 보아야 확인가능



이항함수

- 사용자가 인수의 순서를 추론해야 한다
- (Expected, Actual)과 같이 convention이 존재하거나
- insertTransactionToPosition (transaction, index)와 같이 함수 명에서 인수 순서를 명시해줄 수 있다



단항함수

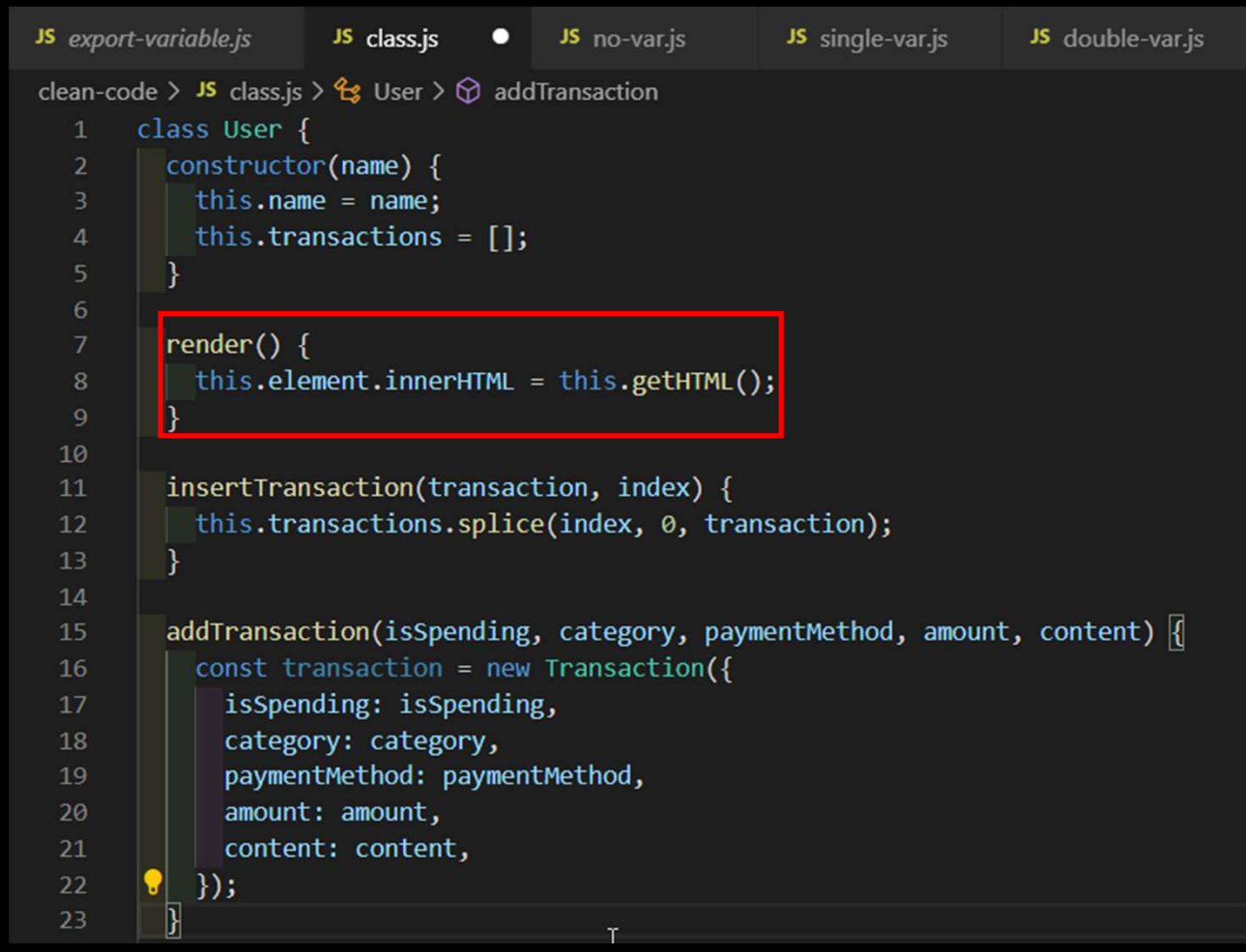
- 인수를 함수명을 통해 명료하게 표현 가능
- 사용하면 좋은 경우
  - 인수를 통해 true/false를 체크
  - User.isMyTx(transaction)
  - 인수를 변환하여 다른 결과를 반환하는 경우

clean-code > class.js > User > addTransaction

```
1   class User {
2     constructor(name) {
3       this.name = name;
4       this.transactions = [];
5     }
6
7     render() {
8       this.element.innerHTML = this.getHTML();
9     }
10
11    insertTransaction(transaction, index) {
12      this.transactions.splice(index, 0, transaction);
13    }
14
15    addTransaction(transaction) {
16      this.transactions.push(transaction);
17    }
```

무항함수

- 사용자가 인수에 대해서 파악할 필요가 없다.
- 호출하는 인스턴스 만을 보고 충분한 것을 알게 해준다.



인수는 적을수록 명료하다

- 무항 함수(클래스 인스턴스)가 베스트
- 단일 인수는 사용자에게 혼란을 주지 않는다
- 이항 함수에서는 인수의 자연적인 순서가 있는지 생각해보자
- 삼항 함수부터는 신중하게 사용을 고려해보자
- (선택)오브젝트 인수로 바꾸는 것을 고려해보자

명령과 조회의 분리

- 수행하거나, 답하거나
- 두개 모두 하려고 하면 다른 개발자를 혼란에 빠뜨린다

```
29    removeTransaction(id) {
30      if(this.transactions.length === 0) return false;
31      else {
32        // find and remove the transaction...
33        return true;
34      }
35    }
36  }
37
38  const user = new User('junsu');
39
40  if(user.removeTransaction(id)) {
41    console.log('removal successful');
42  } else {
43    // do nothing...
44  }
```

명령과 조회의 분리 - Refactored

- 명령하는 함수와 조회하는 함수의 분리
- 함수가 실패할 경우 try... catch... 로 처리

```
38    removeTransaction(id) {
39      const tx = this.transactions.find(id);
40      if(!tx) throw new Error();
41    }
42  }
43
44  try {
45    user.removeTransaction(id);
46  } catch(error) {
47    console.log(error);
48  }
```

주석

```
// 특정 위치에 새로운 거래내역을 삽입한다
insertTransaction(transaction, index) {
  this.transactions.splice(index, 0, transaction);
}

// 사용자에 속하는 새로운 거래내역을 추가한다
addTransaction(transaction) {
  this.transactions.push(transaction);
}
```

이미지 출처: https://www.reddit.com/r/ProgrammerHumor/comments/8w54mx/code_comments_be_like/

주석은 피하는 것이 기본

- 주석을 왜 썼을까?
  - 이해하기 힘들 것이 예상되기 때문에
- 주석으로 나쁜 코드를 가리지 말자
- 주석은 코드보다 관리되기 어렵다
  - 주석 리팩토링 해보신분...?

이미지 출처: https://me.me/i/useful-comments-in-code-comments-in-code-755b63b13b71440bb480614ffb1cf138

써야할 주석은 무엇일까?

- 법적인 주석(라이선스, ...)
- 다른 개발자에게 경고하는 주석
- TODO 주석

clean-code > JS class.js > initTransaction

```
1   // Copyright 2020 Junsu Ltd. All rights reserved
2   // Licensed and Distributable under GNU General Public License 2.0
3
4   // 코드 수행에 10분 이상의 시간이 걸리니 주의할 것
5   const testLotsOfStuff = () => {
6     for(let i = 0; i < 9999999999999999; i++) {
7       test(i);
8     }
9   }
10
11  // TODO: DB Transaction 함수 구현
12  const initTransaction = () => {
13    // 미구현
14  }
```

쓰지 말아야 할 주석

- 의무감에 쓰는 주석
- 코드로 쉽게 파악할 수 있는 내용을 중복하는 주석
- 저자나 수정내역을 표시하는 주석

(Feat. Git)

- 주석처리한 코드
- HTML 주석
- 인접한 내용을 설명하지 않는 주석

Q & A

참고자료

- [Clean Code] by Robert. C. Martin
  - http://www.kyobobook.co.kr/product/detailViewKor.laf?mallGb=KOR&ejkGb=KOR&barcode=9788966260959