Bad Smell

클래스명

- 클래스명에 굳이 MovieVideo, DocumentaryVideo, SportsVideo 라고 써야할까?(leekiwon)

접근 제한자

- getter 메써드가 있음에도 불구하고 굳이 Video 클래스의 속성을 protected 로 선언할 필요가 있을까?(leekiwon, kanisuka)

thin object

- Customer 와 Video 클래스가 단순히 getter 와 setter 역할만을 하고 있다.(keesun, sungyoon)

method 의 복잡도와 중복코드**(**비디오 요금을 구하는 로직 구현부분**)**

- 1 단계 : Video 의 getRentalCharge 의 복잡도가 너무 높다.(dazzilove) => 로직이 하나에 모여 있으나 복잡도가 상당히 높다.
- 2 단계 : RentalInfo 의 totalRentalFee()의 복잡도가 높다.(keesun) => 단 로직이 하나의 method 에 모여있으나 if 절이 발생하고 있음. 변경되는 부분은 Video 클래스
- 3 단계 : if 절은 제거할 수 있으나 중복 코드 발생, 테스트 코드 중복
- Documentary 와 Movie 클래스에서 구현한 getCharge() 메써드에 중복코드가 발생하고 있다.(kanisuka)
- Video 하위 클래스의 calcRentalFee()에서 중복 코드 발생하고 있다.(leekiwon)
- 4 단계 : 중복 코드를 제거하고 if 절을 제거해라.

Over Design

- Rental 클래스의 기능을 Video 로 이동하는 것이 어떨까? 다음 메써드를 보면 대부분의 값이 Video 에서 추출되고 있다. 로직을 구현하기 위한 데이터가 Video 에 있다면 해당 method 도 Video 클래스에 있는 것이 좋지 않을까?(nije)

```
private double getRentalDiscountPrice() {

                    return mVideo.getPrice() * mVideo.getDiscountCondition()

                                          + mVideo.getPrice() *
mVideo.getDiscountPrice() * (mnRentalDayCount - mVideo.getDiscountCondition());

          }
```

클래스간의 관계와 역할 구분이 모호함**.**

- Customer 와 Video 사이의 관계를 만들고 역할을 가지도록 해야함.

총평

- TDD 를 하면서 좀 더 OOP 적으로 설계하고 개발하는 습관을 가지면서 TDD 의 장점을 이해하도록 노력할 필요가 있을 것으로 생각한다.
- TDD 의 최종 목적지는 설계를 하지 않은 상태에서 좀 더 좋은 설계를 만들어 가고자 함이다. 우리는 Java 라는 언어를 사용하고 있기 때문에 OOP 에도 집중하면 좋겠다.

각 개발자별 리뷰 내용

dazzilove

- Video 의 getRentalCharge 의 복잡도가 너무 높다. 만약 새로운 유형의 Video 가 추가된다면..각 유형별로 일정한 패턴이 있다. 이 패턴을 한 곳으로 모으고 달라지는 부분만 변경할 수 있도록 추가적인 리팩토링을 하는 것이 좋겠다.
- Video 클래스가 상속 구조를 가질 때의 장점은?
- Customer 와 Video 사이에 Map 역할을 할 수 있는 별도의 클래스를 두는 것이 좋지 않을까?
- 왜 Map 을 두어야할까? Customer 와 Video 의 관계..
- todoList.txt 를 보면 todo list 가 너무 상세하다. 예를 들어 대여기간에 따른 요금 계산에서 너무 많은 경우의 수를 고려한 것으로 판단된다. 테스트 수는 일정 수준으로 유지하는 것이 좋겠다. 또한 todo list 와 Test 클래스가 일치하지 않는다.
- 하나의 method 에 대하여 테스트 수가 많아질 때의 문제점은?

kanisuka

- Documentary 와 Movie 클래스에서 구현한 getCharge() 메써드에 중복코드가 발생하고 있다.
- Sports 의 getCharge() 메써드에는 중복코드가 없을까?
- 이 중복 코드를 제거했을 때의 효과는?
- getCharge() 메써드의 중복으로 인해 DocumentaryTest, MovieTest, SportsTest 클래스에 많은 중복 코드가 발생하고 있다. 나쁜 냄새..
- Video 클래스의 속성은 protected 이면서 setter Method 를 가진다. Why? private 으로 바꾸면 안될까?
- rentalPeriod 가 Video 클래스의 속성이 맞는가? Customer 와 Video 사이에 연결고리를 가지는 것이 어떨까?
- CustomerTest 가 너무 상세한 것은 아닐까? 필요한 기능만 테스트할 수 있도록 줄인다면..?
- testTotalRentalInfo()의 Video assert 부분을 Video Object 와 Video Object 를 assertEquals 하는게 좋겠다.
- Object 와 Object 의 assertEquals 를 가능하도록 하려면?

keesun

- RentalInfo 의 totalRentalFee()의 복잡도가 높다. 복잡도를 줄일 수 있도록 리팩토링 한다면..?
- if/else 를 사용하지 않고 이 기능을 구현하려면..?
- Customer 에 비디오 rent 메써드를 두고 대여한 비디오 목록을 관리하도록 한다면..?
- Customer 와 Video(하위 클래스 포함) 클래스는 단순히 setter, getter 만을 가진다. 이 클래스들에게 더 많은 역할을 위임하면 좋겠다.
- 현재는 RentalInfo 에 너무 많은 기능이 집중되어 있다. 각 역할에 맞도록 분담하는 것이 어떨까?

nije

- 테스트 용이성을 위하여 Customer 에서 RentalList 클래스의 setter 메써드를 가지는 것이 좋지 않을까? 현재 소스 상태에서..
- 현 재와 같은 방식으로 Customer 에서 RentalList mRentalList = new RentalList()와 같이 RentalList 를 직접 생성하고 있다면 RentalList 클래스의 rental() 메써드를 Customer 로 이동하는 것이 좋지 않을까? 개인적으로 생각으로는 후자가 좋을 것으로 판단된다. Why?
- 만약 RentalList 의 rental() 메써드를 Customer 로 이동한다면 RentalList 의 다른 메써드도 이동할 필요가 있을 것으로 판단된다.
- Rental 클래스의 기능을 Video 로 이동하는 것이 어떨까? 다음 메써드를 보면 대부분의 값이 Video 에서 추출되고 있다. 로직을 구현하기 위한 데이터가 Video 에 있다면 해당 method 도 Video 클래스에 있는 것이 좋지 않을까?

```
private double getRentalDiscountPrice() {

                    return mVideo.getPrice() * mVideo.getDiscountCondition()

                                          + mVideo.getPrice() * mVideo.getDiscountPrice() *
(mnRentalDayCount - mVideo.getDiscountCondition());

          }
```

leekiwon

- 클래스명에 굳이 MovieVideo, DocumentaryVideo, SportsVideo 라고 써야할까?

- getter 메써드가 있음에도 불구하고 굳이 Video 클래스의 속성을 protected 로 선언할 필요가 있을까?
- Video 하위 클래스의 calcRentalFee()에서 중복 코드 발생하고 있다. => 각 테스트 클래스 또한 중복 코드 발생하고 있다.

sungyoon

- ❒ 웹 애플리케이션을 개발자들의 가장 전형적인 개발 방법을 보여주고 있다. VideoShop 에 모든 비지니스 로직 집중되어 있다.
- ❒ 나머지 클래스들은 단순히 setter 와 getter 를 가지는 형태로 구현되어 있음.