

Video Shop TDD

Code Review



채수원
doortts.tistory.com

감명 깊었던 ToDo List

Kanisuka님

테스트 케이스를 항목별 상세화가 잘 되어 있음. TXT파일임에도 가독성 높음

TODO List

- 고객(Customer)은 이름을 가지고 있다.
 - *- 이름을 넣고, 잘 들어갔는지 비교
- 고객은 한번에 여러개의 비디오를 대여할 수 있으나 각각의 대여(Rental)기간은 다를 수 있다.
 - *- 대여 안 하면, 총 대여 비디오 갯수는 0
 - *- 대여를 하면, 총 대여 갯수만큼 숫자가 나온다.
 - *- 렌탈을 안 하면, 대여 기간은 0
 - *- 렌탈을 하면, 대여 기간은 1일 이상의 숫자가 들어간다.
 - *- 특정 이름의 비디오를 검색했을 때, 해당 사용자가 가지고 있지 않은 경우, 리턴되는 List는 item이 0개이어야 한다.
 - *- 똑같은 이름의 비디오를 여러 개 대여를 하면, 이름으로 검색했을 때 대여한 숫자만큼 나온다.
 - *- 대여를 했는데 대여 기간이 0보다 작거나 같으면 false를 리턴한다.
- 비디오(Video)는 영화,스포츠,다큐멘타리의 세종류가 있다.
 - *- 영화 Type 체크
 - *- 스포츠 Type 체크
 - *- 다큐멘타리 Type 체크
- 각각의 비디오는 독립적인 일일 대여요금을 가지고 있다.
- 영화는 대여기간이 2일 이상되면 3일째 부터는 대여요금이 1/2로 할인된다.
 - *- 영화의 대여기간이 1일인 요금 == default charge
 - *- 영화의 대여기간이 2일인 요금 == default charge * 2
 - *- 영화의 대여기간이 3일 이상인 요금
 - *- $((1일\ 요금 * 2) + (영화\ 전체\ 대여일 - 2) * (1일\ 요금 / 2))$
 - *- 영화 대여기간 3일 요금 == $(1일\ 요금 * 2) + 1 * (1일\ 요금 / 2)$
 - *- 영화 대여기간 4일 요금 == $(1일\ 요금 * 2) + 2 * (1일\ 요금 / 2)$

감명 깊었던 ToDo List

Dazzlove님

매우 정교한 테스트 케이스를 설계함, 어디서든 Tester, QA로 스카우트 해야 할 듯

4.각각의 비디오는 독립적인 일일 대여요금을 가지고 있다.

4.1.각각의 비디오는 자신의 정보를 가지고 있다.

4.1.1.각 비디오의 정보를 가져오기 (O)

4.2.각 비디오의 일일 대여요금을 설정

4.2.1.대여요금을 설정하고, 설정된 대여요금을 확인 (O)

5.영화는 대여기간이 2일 이상되면 3일째 부터는 대여요금이 1/2로 할인된다.

- 대여요금이 나누어 떨어지지 않을 경우...

1555를 2로 나누려하면 나누어 떨어지지 않으므로.. 2로 나누어야 하는 일자에는 과금시 버림처리하여 과금하고
초과 2일, 4일과 같이.. 2로 나눌필요가 없을 경우에는.. 나누지않고 2일에 1555를 적용하여 과금처리한다.

5.1.대여기간이 1일 일때의 요금 계산 테스트

- 비디오 종류가 영화이며, 일일 요금이 1000원인 경우 = $1000 * 1 = 1000$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $1500 * 1 = 1500$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $1555 * 1 = 1555$ (O)

5.2.대여기간이 2일 일때의 요금 계산 테스트

- 비디오 종류가 영화이며, 일일 요금이 1000원인 경우 = $1000 * 2 = 2000$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $1500 * 2 = 3000$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $1555 * 2 = 3110$ (O)

5.3.대여기간이 3일 일때의 요금 계산 테스트

- 비디오 종류가 영화이며, 일일 요금이 1000원인 경우 = $(1000 * 2) + (1000/2 * 1) = 2500$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $(1500 * 2) + (1500/2 * 1) = 3750$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $(1555 * 2) + (1555/2 * 1) = 3887$ (O)

- 대여요금이 나누어 떨어지지 않을경우에는 버림 처리 한다.

5.4.대여기간이 4일 일때의 요금 계산 테스트

- 비디오 종류가 영화이며, 일일 요금이 1000원인 경우 = $(1000 * 2) + (1000/2 * (2\%2)) + (1000 * 2/2) = 3000$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $(1500 * 2) + (1500/2 * (2\%2)) + (1500 * 2/2) = 4500$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $(1555 * 2) + (1555/2 * (2\%2)) + (1555 * 2/2) = 4665$ (O)

5.5.대여기간이 5일 일때의 요금 계산 테스트

- 비디오 종류가 영화이며, 일일 요금이 1000원인 경우 = $(1000 * 2) + (1000/2 * (3\%2)) + (1000 * 3/2) = 3500$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $(1500 * 2) + (1500/2 * (3\%2)) + (1500 * 3/2) = 5250$ (O)

- 비디오 종류가 영화이며, 일일 요금이 1500원인 경우 = $(1555 * 2) + (1555/2 * (3\%2)) + (1555 * 3/2) = 5442$ (O)

소스는 당신 것이 아니다 (It's not yours!! Focus Naming!!)

```
public class Customer {  
  
    private String msName;  
    public void setName(String psName) {  
        this.msName = psName;  
    }  
}
```

소스는 당신 것이 아니다 (It's not yours!! What is the logic composed? #1)

```
public class MovieVideo extends Video {  
  
    @Override  
    void initialize() {  
  
        if (period >= 3) {  
            price = ((price / 2) / 100) * 100;  
        }  
    }  
}
```

소스는 당신 것이 아니다 (It's not yours!! What is the logic composed? #2)

```
@Override  
  
public int calcRentalFee(int rentalPeriod) {  
    int rentalFeeTotal = 0;  
    if (rentalPeriod <= discountPeriod) {  
        rentalFeeTotal = (int) this.defaultRentalFee * rentalPeriod;  
    } else {  
        rentalFeeTotal = (int) this.defaultRentalFee * discountPeriod;  
        rentalFeeTotal += this.defaultRentalFee * discountPercentage  
            * (rentalPeriod - discountPeriod);  
    }  
    return rentalFeeTotal;  
}
```

소스는 당신 것이 아니다 (It's not yours!! What is the logic composed? #3)

```
@Override
public int getCharge() {
    int result = 0;
    if (rentalPeriod > 2) {
        result = (defaultCharge * 2)
            + (rentalPeriod - 2) * (defaultCharge / 2);
    } else {
        result = defaultCharge * rentalPeriod;
    }
    return result;
}
```

소스는 당신 것이 아니다 (It's not yours!! What is the logic composed? #4)

```
public int getPoint(Customer customer) {  
    for (int i = 0; i < customers.size(); i++) {  
        Customer c = customers.get(i);  
        if (customer.equals(c)) {  
            return c.getPoint();  
        }  
    }  
    return NO_DATA;  
}
```


소스는 당신 것이 아니다 (It's not yours!! What is the logic composed? #4)

```
public int getPoint(Customer customer) {  
    if( isRegisteredCustomer(customer) ){  
        return findCustomer(customer).getPoint();  
    } else {  
        return NO_DATA;  
    }  
}
```

테스트 케이스는 무엇을 테스트 하는 것인지 알 수 있어야 한다.
(Out of order! So, what?)

```
@Test
public void testMovieVideo() {

    Video matrix = new MovieVideo("Matrix", 4, 1000);
    assertEquals("Matrix", matrix.getName());
    assertEquals(500, matrix.getPrice());
}

public void testMovieVideoRound() {
...
}
```

Object Null Check 누락

```
@Test
public void testCustomerName() {
    customer = new Customer();
    customer.setName("권상우");
    assertEquals("권상우", customer.getName());
}
```

* 객체 생성 구성요소 중 일부의 정보를 객체 생성시 반드시 포함하도록 강제화 할 것인지 고민해야 한다.

→ 객체 주요 정보 요소(Primary Information Element)

Ex>

고객: 이름

계좌: 계좌번호

당신이 무엇을 상상하든 그 이상을 하고 있다. (I'm cheating on you!!)

```
public boolean AddToRentalVideoList(Video video) {  
    RentalVideoList.add(video);  
    totalVideoCount++;  
    totalFee += video.RentalFee;  
    return true;  
}
```

Naked

```
public class Video {  
  
    public enum VideoGenre { Movie, Sports, Documentary }  
    public int fee = 0;  
    public Object Genre = null;  
    public String videoName;  
  
    public int RentalFee = 0;  
    public int RentalPeriod = 0;  
}
```

오래된 미신 (Old Fashioned Myth)

```
@Override
public String toString() {
    StringBuilder builder = new StringBuilder();
    builder.append(" ");
    builder.append(title);
    builder.append(" ");
    builder.append(fee);
    return builder.toString();
}
```



```
@Override
public String toString() {
    return " " + title + " " + fee;
}
```

임시변수를 이용한 역할극 (Role Playing Game with Temp vars. #1)

```
int term = rentalVideos.get(video);  
int daysForSale  
    = video.getNumOfDaysForSaleApplying();  
int fee = video.getFee();  
  
if(term < daysForSale) {  
    totalRentalFee += (fee * term);  
} else {  
    totalRentalFee += (fee * (daysForSale - 1));  
    totalRentalFee += (fee * video.getSaleRate() *  
(term - daysForSale + 1));  
}
```

임시변수를 이용한 역할극 (Role Playing Game with Temp vars. #2)

```
public void testGetVideoInfos() {  
    String inputVideoType  
        = VideoType.VIDEO_TYPE_MOVIE;  
    String inputVideoName = "Twilight";  
    int inputVideoRentalFee = 2000;  
  
    Video video = new MovieVideo(inputVideoName,  
inputVideoRentalFee);  
    String outputVideoName = video.getName();  
    String outputVideoType = video.getType();  
    int outputVideoRentalFee  
        = video.getDefaultRentalFee();  
  
    assertEquals(inputVideoName, outputVideoName);  
    assertEquals(inputVideoType, outputVideoType);  
    assertEquals(inputVideoRentalFee,  
outputVideoRentalFee);  
}
```


임시변수를 이용한 역할극 (Role Playing Game with Temp vars. #2)

```
public void testGetVideoInfos() {  
  
    Video video = new MovieVideo("Twilight", 2000);  
    assertEquals("Twilight", video.getName());  
    assertEquals(VideoType.VIDEO_TYPE_MOVIE,  
video.getType());  
    assertEquals(2000, video.getDefaultRentalFee());  
}
```

대체 뭐가 문젠데? (What's wrong with you?)

```
public double totalRentalFee() {  
    if(rentalVideos.size() == 0)  
        return 0.0;  
    double totalRentalFee = 0.0;  
    for(Video video : rentalVideos.keySet()){  
        ...  
    }  
}
```



```
public double totalRentalFee() {  
    if( notExistsRentedVideo() )  
        return 0.0;  
    double totalRentalFee = 0.0;  
    for(Video video : rentalVideos.keySet()){  
        ...  
    }  
}
```

테스트 항목과 테스트의 불일치 (NT, new technology? NO! Not Testable!!)

```
@Test
public void 고객은이름속성이있다() {
    Customer person = new Customer("person");
    //person.setName("person");
    assertNotNull(person.getName());
}
```

```
@Test
public void 비디오종류는3가지() {
    new Video("movie", "");
    new Video("sports", "");
    new Video("documentary", "");
}
```

쓸 거면 확실히, 아니면 간단히 (Clearly or simply, you choose!)

```
@Override  
public Double getSaleRate() {  
    return 1.0/2.0;  
}
```

변경될 내용이 아니라면 아니라고 말한다. (I'm a Immutable!)

```
public class VideoTest {  
    Video movie;  
    Video sports;  
    Video documentary;  
  
    @Before  
    public void setUp() {  
        movie = new Movie("300", 300);  
        sports = new Sports("EPL", 600);  
        documentary = new Documentary("그 섬에 가고싶다",  
1200);  
    }  
}
```

변경될 내용이 아니라면 아니라고 말한다. (I'm a Immutable!)

```
public class VideoTest {  
    final Video movie = new Movie("300", 300);;  
    final Video sports = new Sports("EPL", 600);;  
    final Video documentary  
        = new Documentary("그 섬에 가고싶다", 1200);;  
  
    @Before  
    public void setUp() {  
    }  
}
```

소스는 당신 것이 아니다. (How about your condition?)

```
final int DISCOUNT_CONDITION = 2;

@Override
public int getDiscountCondition() {
    return DISCOUNT_CONDITION;
}
```

소스는 당신 것이 아니다 (Use brace #1)

```
public int getRentalPrice() {  
    if ( mVideo.enableDiscount() )  
        if ( isDiscountRental() )  
            return (int) (getRentalDiscountPrice());  
  
    return getRentalNonDiscountPrice();  
}
```

```
public int getRentalPrice() {  
    if ( mVideo.enableDiscount() ){  
        if ( isDiscountRental() ){  
            return (int) (getRentalDiscountPrice());  
        }  
    }  
    return getRentalNonDiscountPrice();  
}
```


테스트의 목적을 명확히 (Who.are.you?)

```
@Test
public void testMovieVideo() {

    Video matrix = new MovieVideo("Matrix", 4, 1000);
    assertEquals("Matrix", matrix.getName());
    assertEquals(500, matrix.getPrice());
}

@Test
public void testMovieVideoRound() {

    Video matrix = new MovieVideo("Matrix", 4, 900);
    assertEquals("Matrix", matrix.getName());
    assertEquals(400, matrix.getPrice());
}
```

레벨 불균형 (API vs Model #1)

```
@Test
public void testRentalInfo() {
...
...
    List<Video> list = new ArrayList<Video>();
    list.add(matrix);
    list.add(nba);
    list.add(earth);

    List<Video> info = shop.getRentalInfo(thewmat);
    assertEquals(list.toArray(),
info.toArray())
```

Java API (=List)

Logic(=model)

레벨 불균형 (API vs Model #2)

```
public void testRentVideosByCustomer() {  
    ...  
    ...  
    List<RentalVideo> videoList  
        = new ArrayList<RentalVideo>();  
    videoList.add(rv);  
    videoList.add(rv1);  
  
    RentalInfo rental  
        = new RentalInfo(customer, videoList);  
    ...  
    ...  
}
```

Model API (=List)

레벨 불균형 (API vs Model #3)

```
@Test public void rent_finished() {  
    Calendar rentalDateTime  
        = Calendar.getInstance();  
    rentalDateTime.add(Calendar.DAY_OF_MONTH, -3);  
  
    Customer customer = new Customer("javajigi");  
    RentalVideo rentalVideo  
        = customer.rent(  
            new Movie("블랙", 1000), rentalDateTime, 2);  
    assertFalse(rentalVideo.isRent());  
}
```

Model

API (=Calendar)

비디오 가게가 고객과 비디오를 함께 빌렸다.

(Hmmm.. In my opinion, clerk's name must be "Video Shop")

```
Public class VideoShop {  
...  
...  
    public void rental(Customer customer, Video video) {  
  
        for (int i = 0; i < customers.size(); i++) {  
            Customer c = customers.get(i);  
            if (customer.equals(c)) {  
                customers.remove(i);  
                c.setRentVideo(video);  
                c.addPoint(point(video));  
                customers.add(c);  
                return;  
            }  
        }  
        customer.setRentVideo(video);  
        customer.addPoint(point(video));  
        customers.add(customer);  
    }  
}
```

경우에 따라서는 테스트 일 수도 있다.
(I have a falcon eye.)

```
public void testRentVideosByCustomer() {  
    ...  
    ...  
    int index = 0;  
    for (RentalVideo video : rental.getRentalVideos()) {  
        assertEquals(  
            video, rental.getRentalVideos().get(index));  
        index++;  
    }  
    ...  
}
```

소스는 당신 것이 아니다 (It's not yours!! What is the logic composed? #5)

```
public void testRentVideoByBonusPoints() {  
    ...  
    ...  
    if (rental.getCustomer().getName().equals(customer.getName()))  
    {  
        for (RentalVideo video : rental.getRentalVideos()) {  
            if (video.getVideo().getType().equals(  
                VideoType.VIDEO_TYPE_MOVIE))  
                assertEquals(1, video.getVideo().getBonusPoints());  
            else if (video.getVideo().getType().equals(  
                VideoType.VIDEO_TYPE_DOCUMENTARY))  
                assertEquals(1, video.getVideo().getBonusPoints());  
            else if (video.getVideo().getType().equals(  
                VideoType.VIDEO_TYPE_SPORTS))  
                assertEquals(2, video.getVideo().getBonusPoints());  
        }  
    }  
    ...  
    ...  
}
```

상속은 했지만, 상속 받은 것이 별로 없다.
(extended but nothing to extend)

```
public class DocumentaryVideo extends Video {
    private final int discountPeriod = 3;
    private final double discountPercentage = 0.25;
    private final int defaultBonusPoints = 1;

    @Override
    public int calcRentalFee(int rentalPeriod) {
        int rentalFeeTotal = 0;
        if (rentalPeriod <= discountPeriod) {
            rentalFeeTotal = (int) this.defaultRentalFee * rentalPeriod;
        } else {
            rentalFeeTotal = (int) this.defaultRentalFee * discountPeriod;
            rentalFeeTotal += this.defaultRentalFee * discountPercentage
                             * (rentalPeriod - discountPeriod);
        }
        return rentalFeeTotal;
    }
}
```

유산으로 받은 상자를 열어보니, '열심히 살아라'라는 편지 한 장 있더라.

소스는 당신 것이 아니다 (It's not yours!! Focus Naming #2)

```
public class Movie extends Video {  
    ...  
    ...  
    public Double getSaleRate() {  
        ...  
    }  
  
    public int getNumOfDaysForSaleApplying() {  
        ...  
    }  
}
```

테스트 에서도 최대한 SRP를 지킨다
(Always keep in mind, SRP)

```
if(c.rent(v, 1)) {  
    Assert.assertEquals(1, v.getRentalPeriod());  
} else {  
    fail();  
}
```

*SRP: Single Responsibility Principle

절친한 부자관계 (Coupling from somewhere over the rainbow)

```
public class Sports extends Video {  
    static public int SPORTS_POINT = 2;  
  
    public Sports(int defaultCharge) {  
        super(defaultCharge);  
        type = VideoType.SPORTS;  
    }  
}
```

type 은 어디쯤에?

만일 코드에 순서가 필요하다면 그것도 테스트 한다
(A parts on the conveyor belt can be stolen)

```
MockVideo video
    = new MockVideo(VideoType.MOVIE, "블랙", fee);
video.setDiscountRule(discountRule);
video.setStartDiscountDay(startDiscountDay);

assertEquals(2500, video.getRentalFee(rentalPeriod));
```

조금 더 고민해 볼만한 내용(More things to think #1)

```
private List<Video> rentalVideoList  
    = new ArrayList<Video>();  
...  
getRentalVideoList() 같은 Method 없음
```



```
private ArrayList<Video> rentalVideoList  
    = new ArrayList<Video>();
```

조금 더 고민해 볼만한 내용(More things to think #2)

```
public class Sports extends Video {  
    private static final int START_DISCOUNT_DAY  
        = Integer.MAX_VALUE;  
    ...  
    ...  
}
```

조금 더 고민해 볼만한 내용(More things to think #3)

```
public RentalVideo from(Calendar rentalDateTime) {  
    this.rentalDateTime = rentalDateTime;  
    return this;  
}  
  
public RentalVideo during(int rentalPeriod) {  
    this.period = rentalPeriod;  
    return this;  
}  
...  
...  
    RentalVideo rentalVideo = new RentalVideo(this,  
video).from(rentalDateTime).during(period);  
  
rentalVideo.from()  
rentalVideo.during()
```

조금 더 고민해 볼만한 내용(More things to think #4)

```
private long sessionRentalFee;  
private int sessionPoint;  
private List<RentalVideo> rentalVideos  
    = new ArrayList<RentalVideo>( );  
  
private void addSessionPoint(int point) {  
    this.sessionPoint += point;  
}  
  
private void addSessionRentalFee(RentalVideo  
rentalVideo) {  
    this.sessionRentalFee  
        += rentalVideo.getRentalFee( );  
}
```

상태 저장을 사용할 것인가?
필요할 때 구할 것인가?

조금 더 고민해 볼만한 내용(More things to think #5)

Lazy Loading

```
private void prepareList() {  
    if( rentalList == null ) {  
        rentalList = new ArrayList<VideoRental>( );  
    }  
}
```



소스는 당신 것이 아니다
(Source isn't yours!!)

소스는 분석하는 것이 아니라 읽는 것
(Source has to be easily readable)

테스트 대상을 명확히 표현한다
(Clearly express what test you want to)

업무 규칙의 경계값을 구분하는 것은 테스트의 기본
(0,1, under, exact, over, very big)

Google talk 등록은 →

doortts@gmail.com

- 주저하지 말고 등록하세요.
도움 드릴 수 있는 내용이 있으면
최대한 도와 드립니다. :)

