210810 코드리뷰2

생성일	@2021년 8월 10일 오후 8:26
성함	나한주
속성	
수업 유형	이론
수정일	@2021년 8월 10일 오후 9:10
작성자	㊊ 현동빈

1. 왜 좋은 프로그램을 개발해야 하는가?
2. 어떻게 개발해야 하는가?
3. 리팩토링 실전
4. 리팩토링 사례

1. 왜 좋은 프로그램을 개발해야 하는가?

▼ 좋은프로그램이란?

- 원래 목적한 기능을 수행하는 프로그램
- 가독성 좋은 프로그램
- 간결하게 작성한 프로그램
- 주석이 잘 정리된 프로그램
- 유지보수가 용이한 프로그램

▼ 좋은 프로그램을 개발하기 위해서는?

- 프로그램을 많이 개발하기
- 다른 사람이 개발한 프로그램을 많이 보기
- 본인이 앞서 작성한 프로그램을 많이 보기
- 수정을 두려워하지 말기
- 복기하기

- 코드리뷰하기
- 리팩토링하기

2. 어떻게 개발해야 하는가?

▼ 형식에 맞추어서 개발

- 코드 형식 따르기

▼ 세로 밀집도

- 빈 행으로 분리
- 빈 행은 새로운 개념을 시작한다는 시각적 단서
- 세로 밀집도
- 수직 거리

▼ 가로 형식

- 한 눈에 전체 소스가 들어오게 작성
- 공백 넣기
- 들여쓰기

▼ 변수 선언

- 최대한 가까운 곳에 선언
- 인스턴스 변수

▼ 종속 함수

- 호출되는 순서대로 배치
- 신문 기사 작석 순서
- 신문기사처럼 프로그램 개발
- 유사성이 높은 코드는 가까이 배치

▼ 소스코드는 얼마나 길어야 적당한가

- 적절한 행 길이 유지 : 일반적으로 큰 파일보다 작은 파일이 이해하기 쉬움

▼ 슈퍼 클래스(만능 클래스)를 만들지 말기

- 모든 처리를 하는 만능 클래스를 만들기 않기

▼ 함수는 작게 만들기

- 작게 만들기
- 그룹으로 함께 묶을 수 있는 코드 조각이 있으면, 코드의 목적이 잘 드러나도록 메소드의 이름을 지어 별도의 메소드로 뽑아냄
- Block은 메소드로 분리하는 단위
- 두 가지 일 x
- 중복 검증을 하지 말고 코드를 단순하게
- 생각의 폭을 줄이기
- 오류코드를 사용하기 보다는 예외를 던지기(throws Exception)
- 호출자를 고려하여 예외를 정의

  - 감싸기 기법 사용하여 의존성 제거
- 되도록이면 null을 반환하거나 전달하기 말기
- 정상 흐름 제어
- 조건문을 메소드로 추출
- Nested 조건문
- 조건을 반대로 하기
- 중복코드 작성 피하기

  - 좋은 코드를 개발하는 기본은 중복을 제거하기

  - Template Method 패턴을 이용하여 쉽게 중복을 제거할 수 있음

▼ 결론

- 개발한 프로그램이 그저 돌아가는 코드만으로는 부족
- 단순히 돌아가는 코드에 만족하는 프로그래머는 전문가 정신이 부족
- 나쁜 코드는 더 심각하게 개발 프로젝트에 영향
- 나쁜 코드도 깨끗한 코드로 개선 가능
- 처음부터 깨끗한 코드 유지

3. 리팩토링 실전

▼ 리팩토링

- 소프트웨어를 보다 쉽게 이해할 수 있고, 적은 비용으로 수정할 수 있도록 겉으로 보이는 동작의 변환
- 일련의 리팩토링을 적요하여 겉으로 보이는 동작의 변환 없이 소프트웨어의 구조를 바꿈
- 목적 : 소프트웨어를 보다 이해하기쉽고 수정하기 쉽도록 만드는 것
- 소프트웨어를 더 쉽게 이해할 수 있도록 바꾸는 것
- 리팩토링의 겉으로 보이는 소프트웨어의 기능을 변경 x
- 리팩토링 후에도 소프트웨어는 여전히 동일한 기능을 가지고 있어야 함

▼ 리팩토링을 하는 이유

- 소프트웨어 디자인 개선
- 버그찾기
- 개발속도 상승

▼ 언제 리팩토링을 하는가?

- 코드 리뷰 할 때
- 기능을 추가할 때
- 버그를 수정해야 할 때

▼ 리팩토링 순서

- 메소드의 재분배
- 이름 변경
- 위치 변경
- 변수 사용 중복 제거
- 포인트 계산 부분 추출
- 임시변수 제거하기
- 다형성으로 바꾸기
- 상속을 이용하여 switch문 제거

4. 리팩토링 사례