11장. 의식적으로 지름길 사용하기

지름길

지름길을 방지하기 위해 먼저 지름길 자체를 파악하라

- 지름길은 깨진 창문 같다.
- 품질이 떨어진 코드에서 더 낮은 품질의 코드를 추가하기 쉽다.
- 코딩 규칙을 많이 어긴 코드에서 다른 규칙을 어기기 쉽다.
- 지름길을 많이 사용한 코드에서 다른 지름길을 추가하기 쉽다.

깨끗한 상태로 시작할 책임

그니까 가능한 한 지름길을 쓰지 말자 ?!

지름길이 더 좋을 때

- 중요도가 낮다.
- 프로토타이핑 작업이다.
- 기타 경제적인 이유 ...

지름길 사용법

- 의도된 지름길은 세심하게 기록할 것

- 모두에게 인지시킬 것

유스케이스 간 모델 공유하기

유스케이스들이 기능적으로 묶여 있다면 입출력 모델을 공유해도 좋다

- 특정 세부 사항을 변경할 경우 두 유스케이스 모두에 영향을 주고 싶을 때
- 서로 미치는 영향 없이 독립적으로 진화해야 하는 경우 이는 지름길이 된다.
- 독립적으로 진화해야 한다면 처음부터 분리해서 시작해야 한다.

도메인 엔티티를 입출력 모델로 사용하기

엔티티를 인커밍 포트의 입출력 모델로 사용하고 싶다면?

- 인커밍 포트는 도메인 엔티티에 의존성을 가지고 있다.
- 엔티티는 변경할 또 다른 이유가 생겼다.
- 갈수록 도메인 로직이 늘어나기 때문에 독립적인 전용 입력 모델이 필요하다.

인커밍 포트 건너뛰기

인커밍 포트는 의존성 역전에 필수적이지는 않다.

- 인커밍 포트를 제거함으로써 인커밍 어댑터 - 애플리케이션 계층 사이의 추상화 계층을 줄일 수 있다.
- 애플리케이션 중심에 접근하는 진입점이 사라졌다.
- 특정 유스케이스를 구현하기 위해 어떤 서비스 메서드를 호출해야 하는가?
- 애플리케이션의 동작에 대해 더 잘 알아야 한다.
- 진입점의 식별을 위해 전용 인커밍 포트를 유지하자.

애플리케이션 서비스 건너뛰기

간단한 유스케이스는 서비스가 필요없지 않을까?

- 인커밍 어댑터와 아웃고잉 어댑터 간 모델을 공유하게 된다.
- 즉 도메인 모델을 입력 모델로 사용하게 된다.
- 유스케이스가 사라진다.
- 유스케이스가 복잡해지면 도메인 로직을 아웃고잉 어댑터에 추가하게 된다.
- 도메인 로직이 흩어지고 유지보수가 어려워진다.

결론

- ~~다 하지 마라~~
- 언제 복잡해지는지 합의한다면 지름길을 쓸 수도 있겠다.
- 단순 CRUD가 평생 간다면 유지보수 비용을 위해 지름길을 유지하는게 더 낫다.
- 왜 지름길을 선택했는가를 꼼꼼히 기록하자.