07.26 스터디

<JDBC, HIBERNATE, ORM>

우리가 사실 사용하고 있는건 HIBERNATE 구현체인데 왜 다들 JPA 라고 할까?

-> 객체지향사실과 오해에서도 나와있듯이 인터페이스와 구현체는 철저히 구분되어야 한다.
어떻게 구현했는지는 사용자 입장에서 철저히 숨기고 JPA 가 제공하는 기능만을 사용하도록 하기 위해서 .

ORM의 단점?

1. 러닝커브
2. 잘못 구현된 경우 속도 저하 및 일관성이 무너질 수도 있기 때문에 충분한 이해가 필수적이다.

<객체지향의 사실과 오해 >

149pg

메시지를 수신한 객체가 런타임에 메서드를 선택할 수 있다는 사실은 다른 프로그래밍 언어와 객체지향 언어를 구분짓는 핵심적인 특징중 하나이다. 프로시저 호출에 대한 실행코드를 컴파일 시간에 결정하는 절차적인 언어와 확연히 구분되는 특징이다.

-> 절차지향 언어에 대표적인 C언어 환경에서는 컴파일 시간에 프로시저를 결정한다. 그에 반해 객체지향 언어 기반에서는 런타임 시간에 메서드를 선택 하여 실행할 수 있다. 그에 대한 예시로 다형성을 구현한 오버로딩, 오버라이딩 이 있다.