6장 객체와 자료구조

* 변수를 비공개로 정의하는 이유가 있다. 남들이 변수에 의존하지 않게 만들고 싶어하기 때문 따라서 이번장에서는 어떻게 수많은 프로그래머가 조회 함수와 설정 함수를 당연하게 공개해 비공개 변수를 외부에 노출하는 것일까?
* 자료 추상화
  + 변수나 함수를 구현해야할 떄 객체가 포함하는 자료를 표현할 가장 좋은 방법을 심각하게 고민해야 한다.
  + 구현을 감추려면 추상화가 필요,
* 자료 객체 비대칭
  + 객체는 추상화 뒤로 자료를 숨긴 채 자료를 다루는 함수만 공개한다.
  + 자료는 자료를 그대로 공개하며 별다른 함수는 제공하지 않는다.
  + 객체지향코드에서 어려운 변경은 절차적인 코드에서 쉬우며, 절차적인 코드에서 어려운 변경은 객체지향코드에서 쉽다.
* 복잡한 시스템을 짜다 보면 새로운 함수가 아니라 새로운 자료 타입이 필요한 경우가 생긴다. 이때는 클래스와 객체지향 기법이 가장 적합하다. 반면 새로운 자료타입이 아니라 새로운 함수가 필요한 경우도 생기는데 이때 절차적인 코드와 자료구조가 더 적합하다.
* 디미터 법칙
  + 잘 알려진 휴리스틱(발견볍)이다. 자신이 조작하는 객체의 속사정을 몰라야 한다는 법칙이다.
  + 객체는 자료를 숨기고 함수를 공개한다. 즉 객체는 조회함수로 내부 구조를 공개하면 안된다는 것.
    - 디미터 법칙은 클래스 c의 메서드 f는 다음과 같은 객체의 메서드만 호출해야 한다
      * 클래스c
      * F가 생성한 객체
      * f인수로 넘어온 객체
      * c 인스턴스 변수에 저장된 객체
  + 위 객체에서 허용된 메서드가 반환하는 객체의 메서드는 호출하면 안된다. 다시말해 낯선 사람은 경계하고 친구랑만 놀라는 의미
  + 기차충돌
    - 객채의 하위객체의 하위객체의 하위객체의 매소드를 선언함으로 원하는 데이터 값을 가져오는 방식
    - 고로 각 객체별로 나눠서 선언하는 것이 좋다. 언제 해당객체를 또 불러오려고 주렁주렁 작성하면 안되니까.
  + 잡종구조
  + 구조체 감추기
    - 객체의 내부구조를 감춰야 하는데 임시디렉터리의 절대 경로는 어떻게 얻어야 할까?
    - 예시
      * Ctxt.getAbsolutePathOfScratchDirectoryOption();
      * Ctxt.getScratchDirectoryOption().getAbsolutePath();
      * 첫번째 방법은 ctxt 객체에 공개해야 하는 메서드가 너무 많아진다.
      * 두번째 방법은 getScratchDirectoryOption()이 객체가 아니라 자료 구조를 반환한다고 가정한다.
* 자료 전달 객체
  + 자료 구조체의 전형적인 형테는 공개변수가 있고 함수가 없는 클래스. 어떤 자료구조체를 때로는 자료전달 객체라 한다(DTO)
  + 이 자료전달 객체는 데이터베이스에 저장된 가공되지 않은 정보를 애플리케이션 코드에서 사용할 객체로 변환하는 일련의 단계에서 가장 처음으로 사용하는 구조체
  + 활성 레코드
    - DTO의 특수한 형태이다
    - 공개변수가 있거나 비공개 변수에 조회 설정함수가 있는 자료구조지만 대개 SAVE나 FIND와 같은 탐색 함수도 제공한다.
    - 불행하게도 활성레코드에 비즈니스 규칙 메소드를 추가해 이런 자료구조를 객체로 취급하는 개발자가 흔하다.
* 결론
  + 객체는 동작을 공개하고 자료를 숨긴다.
  + 그래서 기존 동작을 변경하지 않으면서 새 객체 타입을 추가하기는 쉬운 반면, 기존 객체에 새 동작을 추가하기는 어렵다.