07/12

회의 내용

default , protected 어디에 사용될까 ?

https://wikidocs.net/232

https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=2feelus&logNo=220576845725

private은 같은 클래스 내에서만 접근 가능하고,
default는 같은 폴더(패키지)내에서만 가능하고,
protected는 같은 폴더(패키지)및 그 클래스를 상속(extends)해서 구현하는 경우 접근이 가능하며,
public은 모든 클래스에서 접근이 가능하다.

대부분의 경우에는 상속하는 클래스가 직접 구현할 필요는 없지만, 그래도 특수한 상황에서는 protected로 선언된 메소드는 override해서 처리해야 할 수도 있음을 유저에게 알려주는 것이다. 즉 protected는 굉장히 문서적으로, 클래스 상속을 하는 개발자에게 주의를 주는 방식인 셈이다. (abtraction은 강제적으로 override를 강제하는 구조라고 볼 수 있다)

<예제코드>

```
public class Bird{
   void fly() {
       System.out.println("I am flying")
   }
   protected void moveFast() {
      fly();
   }
}

public class Ostrich extends Bird {
    public static void main(String[] args) {
      Ostrich  ostrich = new Ostrich();
       ostrich.moveFast();
   }
   void run() {
       System.out.println("I am running")
   }
   protected void moveFast() {
      run();
   }
}
```

<Java Study> Final & Static

static : 인스턴스에서 직접 값을 가지고 있지 않고 레퍼런스를 가지고 있음.

final :
호출한 클래스 안에서만 사용하는 접근 제어자 , 가지고 오는 클래스 입장에서 변경할 수 없다.
-> 무결성

<Final 을 spring project 와 연결지어 생각>

: service -> repository
@Autowired
PlaceRepository repo;

-> 외부에서 접근 할 수 없고 , 변경 될 수 없는 클래스 이며 ,
private final

의존성 주입의 이유
-> 다른 클래스의 행동 method 들을 이용하기 위해서

<객체지향의 사실과 오해>

- 우리가 구현하고자 하는 애플리케이션의 목적이 무엇인지를 가장 먼저 파악하는것이 중요할거같다.

토론 주제 (point 93 ~ 94 pg)

1. **같은 행동을 하는데 데이터가 다르다 -> 같은 타입으로 볼 수 있을까? 볼수 있다. (행위가 중요하다.)**

93pg : 객체가 어떤 데이터를 보유하고 있는지는 타입을 결정하는데 아무런 영향도 미치지 않는다.

MeberRepo (interface)

QuerydslMemberRepo
, JpaMemberRepo

2. 좋은 객체지향 설계를 위해서는 행동을 제공하고 , 데이터는 행동 뒤로 감춰야한다.

그 외부는 어디를 말하는 것인가? 왜 감춰야 하는 것인가?

-> 외부는 해당 객체와 협력하는 모든 객체 .

-> 내부 표현 방식과 무관 하게 행동만이 고려 대상이라는 사실은 외부에 내부 데이터를 숨겨야 한다는 뜻이다. (다형성 입장에서 보면 같은 책임을 수행하는 객체로 언제든지 대체 할 수 있다는 상황에서 내부 수행을 다 알기에는 굉장히 힘들다. 객체 들은 해당 책임을 자율적으로 수행할 수 있어야 함으로 외부에서는 협력 하는 객체의 행위만을 호출 하여 협력한다.)

-> 다른 협력하는 객체들에 의해서 데이터가 변경이 되면 -> 객체의 무결성이 무너지기 때문에. (접근 제어자)

3. 왜 타입을 쓸까? 타입을 부여하면 뭔가 더 나아지는가 ?

- 이러한 관점에서 타입은 추상화이다.

앨리스 - 물약을 먹으면 키가 작아져 , 당근을 먹으면 키가 커 , 치킨을 먹으면 몸무게 늘어나 .

-> 상태로 보면 다 다른 타입 .

앨리스 타입 (클래스) -> 앨리스 타입. (내부적으로 상태만 바껴)

:: 상태는 동적으로 변할 수 있지만 타입을 통해 앨리스를 다른 객체와 구별 할 수 있는 식별성을 가질수 있다.

개념은 특정한 객체가 어떤 그룹에 속할지 결정을 한다.
식별성을 통해서 어떤 그룹에 속할 수 있는지 분류 할 수 있다.

이 내용은 클래스를 왜 사용할까? 에 대한 답으로 대체 가능하다.

4.

- 상태 중심 설계가 아닌 행동중심 설계를 하라. 그리고 다이어그램을 그려라.
- 예시를 한번 만들어볼까요? 같이? 65pg.

우리의 스터디 설계 - 과제 ..

사람 요일 주제

5. UML 에 대해서. 정적 다이어그램 과 동적다이어그램 리마인드