AOP (Aspect Oriented Programming) 고급

< 1. AOP의 구현 방법 >

AOP를 구현한다는 뜻은 **Advice(부가/공통 기능)**와 **Target(핵심기능)**이 결합되어서 프록시 객체를 만드는 과정이며 **Weaving**(xml 파일에서 공통기능과 핵심기능을 엮어주는 작업)한다는 의미이다.

방법1) XML기반의 POJO 클래스를 이용한 AOP구현(= 위빙)

- Pom.xml에서 의존 설정
- 프록시 클래스 제작(공통기능의 클래스 제작 - Advice 역할 클래스)
- XML 설정 파일에 <aop:config>를 이용해서 Aspect(애스펙트)를 설정

  즉, Advice(어드바이스)와 PointCut(포인트컷)을 설정

방법2) @Aspect 어노테이션을 이용한 AOP구현(= 위빙)

- 의존 설정
- @Aspect를 이용해서 Advice(부가/공통 기능)을 제공하는 Aspect(애스펙트) 클래스를 작성한다.
- XML 설정 파일에 <aop:aspectj-autoproxy/>를 설정

< 2. Advice의 종류 >

1) **<aop:before>**

: 핵심기능 메소드 실행 전에 Advice 실행

2) **<aop:after-returning>**

: 정상적으로 핵심기능 메소드 실행 후에 Advice 실행

3) **<aop:after-throwing>**

: 핵심기능 메소드 실행 중 예외 발생시 Advice 실행

4) **<aop:after>**

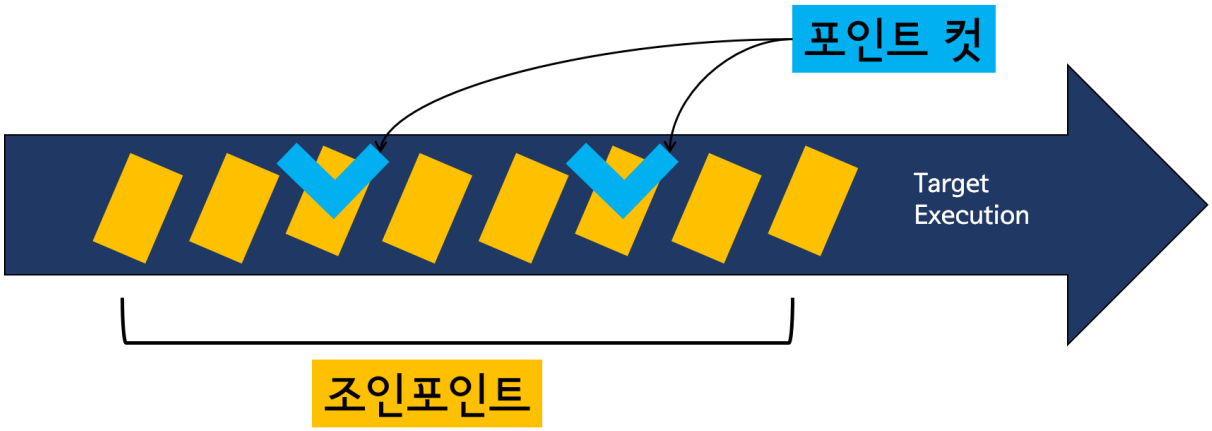
: 핵심기능 메소드 실행 후 예외 발생 여부와 상관없이 Advice 실행

5) **<aop:around>**

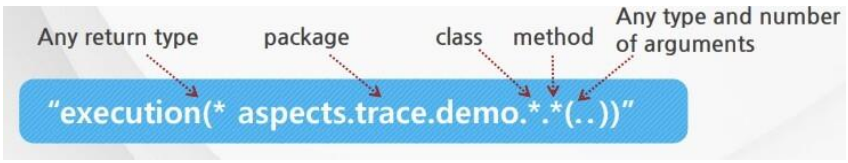
: 핵심기능 메소드 실행 전/후 및 예외 발생시 Advice 실행 (가장 광범위하게 사용)

< 3. PCD = AspectJ pointcut designators 의 종류 >

- 포인트 컷은 특정 조건에 의해 필터링 된 JoinPoint(조인포인트)이다.
- 포인트 컷을 이용 시 Advice 메소드가 적용될 Target 메소드를 정확하게 필터링하는 것이 가능하다.
- PCD는 Target(핵심기능)의 여러 JoinPoint 중에 Advice를 어디에 적용을 시킬 것인지를 AOP에게 알려주는 키워드라고 생각하면 됨.



1) execution : 가장 정교한 PointCut을 만들 수 있음



2) within : 타입패턴 내에 해당하는 모든 것들을 PointCut

3) bean : bean 이름으로 PointCut