# 애스펙트(관점) 지향 프로그래밍

## AOP란 무엇인가?!

- 한 애플리케이션에서 여러부분에 걸쳐 있는 기능을 가리켜 **횡단 관심사(cross-cutting concerns)** 라 한다.

- **횡단 관심사란** 하나의 애플리케이션이

주요관심사에 대해서 보조기능을 모듈화 시킨것이라 볼 수 있다.

- 횡단관심사의 주요 목적은 모듈화에 있다. 횡단관심사를 하나의 어플리케이션의 여러부분에 영향을 주는 기능이라고 정리할 수있는데 이 부분을 하나로 모듈화시켜 관리 할 수 있는 것이 AOP인 것이다.

- 애스펙트 지향 프로그래밍(AOP)은 바로 이러한 횡단관심사의 분리를 위한 것이다.

- 공통적인 기능(횡단관심사)를 재사용 하기 위해선 객체지향에서 상속(inheritance)나 위임(delegation)을 사용하는데 상속은 의존성이 높아지고 위임은 코드의 복잡성이 증가했었다. 하지만 AOP는 이런 부분을 해소시켜준다.

- AOP는 기본적으로 공통기능을 한 곳에 정의하는 것은 동일하다. 하지만 이 기능을 어디에 어떻게 적용할지를 **“선언적 정의”**할 수 있다는 것이 강점이다.

AOP에서는 횡단관심사를 Aspect(애스펙트)라는 특별한 클래스로 모듈화 한다.

AOP가 다른 기법과 차별화되는 장점은 두가지이다. 첫 번째 장점은 여타 서비스모듈이 자신의 주요관심사항에 대한 코드만 포함하고 그 외의 관심사항은 모드 애스펙트로 옮겨지므로 코드가 깔끔해진다. 두번째 장점은 전체코드기반에 흩어져 있는 관심사항이 하나의 장소로 응집된다

### AOP 용어 정리

AOP의 용어의 종류는 어드바이스, 포인트 컷, 조인포인트 세개 이다.

###### 1. 어드바이스 (ADVICE)

- 애스펙트는 자신이 무엇을 해야하는지를 알고 있어야 한다. 애스펙트가 해야할 작업을 **어드바이스** 라고 한다.

- **‘무엇’**을 **‘언제’**할 지 정의한다. Before , After , after-returning , after-throwing , around 등의 수행동작 시점을 선정할 수 있다.

###### 2. 조인포인트(point cut)

- 애플리케이션에서 어드바이스를 적용할 수 있는 곳을 말한다. 이와 같이 어드바이스를 적용할 수 있는 곳을 **“조인포인트”**라 한다. 애플리케이션 조인포인트 지점에 애스펙트를 끼워 넣을 수 있는 지점을 말한다. 이러한 지점에는 메소드 호출 지점 **,** 예외 발생 , 필드 값 수정 등이 있다.

###### 3. 포인트 컷(point cut)

- 포인트 컷은 애스펙트가 어드바이스할 조인포인트의 영역을 좁히는 일을 한다.

- 포인트 컷은 **“어디서”** 할 지를 정의한다.

- 각 포인트 컷은 어드바이스가 위빙(weaving)돼야 하는 하나 이상의 조인포인트를 정의한다.

- 클래스나 메소드명을 직접 사용하여 지정할 수도 잇고 정규식으로 정의할 수도 있다.

###### 4. 애스펙트(Aspect, 관점)

- 애스펙트는 어드바이스와 포인트 컷을 합친(merge) 것이다.

###### 5. 인트로 덕션

- 인트로덕션은 클래스의 코드 변경 없이 메소드나 멤버변수를 추가하는 기능이다.

###### 6. 위빙(weaving)

- 위빙은 타깃 객체(어드바이스 대상)에 애스펙트를 적용해서 새로운 프록시 객체를 생성하는 절차다. 애스펙트는 타깃 객체의 조인포인트로 위빙된다.

위빙은 대상객체의 생애 중 다음과 같은 몇 가지 시점에서 수행될 수 있다.

1. 컴파일 시간(compile time) – 타깃 클래스가 컴파일 될 때 애스펙트가 위빙(프록시 객체 생성)된다.

2. 클래스 로드 시간(classload time) – 클래스가 jvm에 로드 될 때 애스펙트가 위빙된다. (aspectJ 가 필요)

3. 실행 시간(runtime time) – 애플리케이션 실행 중에 애스펙트가 위빙된다.