5.AOP

AOP 용어

조인포인트(joinPoint)

advice가 적용될 수 있는 위치

aop가 적용되는 지점

타겟 객체가 구현한 인터페이스의 모든 메서드는 조인포인트가 됨

포인트컷(pointCut)

어드바이스를 적용할 타겟의 메서드를 선별하는 정규 표현식

joinpoint의 부분집합

표현식은 execution으로 시작함

위빙(Weaving)

pointCut에 의해 결정된 타겟의 joinPoint에 advice를 삽입하는 과정

target에 영향을 주지 않으면서 advice를 삽입함

Spring AOP 특징

스프링AOP는 프록시 기반의 AOP 구현체

target 객체에 대한 proxy를 만들어 제공함

스프링 bean에만 AOP 적용 가능

타겟을 감싸는 프록시는 실행시(Runtime)에 생성됨

프록시(Proxy)는 어드바이스를 타겟 객체에 적용하면서 생성되는 객체임

프록시가 호출을 가로챔(Intercept),

전처리 어드바이스 => 타겟 => 후처리 어드바이스

Spring은 자동프록시 생성기를 통해 직접 프록시 객체를 생성한 후 처리해줌

=> JDK 다이나믹 프록시

Proxy Pattern

proxy : 대리자, 대변인

객체를 사용하고자 할 때, 객체를 직접 참조하는 것이 아닌 해당 객체를 감싹 객체를

통해 객체에 접근하는 방식

이 방식을 통해 실제 객체의 기능이 필요한 시점까지 객체의 생성을 미룰 수 있다

장단점

사이즈가 큰 객체가 로딩되기 전에 프록시를 통해 참조 가능

실제 객체의 public, protected 메서드를 숨기고 인터페이스를

통해 노출시킬 수 있음

원래 객체에 대한 사전처리 가능

성능저하

AspectJ

java AOP 소프트웨어, spring AOP에서도 이용

포인트컷 관련 용어

지시자(execution, within, bean)

어드바이스 동작 시점

before

타겟의 메서드가 실행되기 이전 시점에 처리해야하는 어드바이스

After, After-returning, After-throwing

Arround

타겟의 전후 시점 모두 처리할 필요가 있는 어드바이스

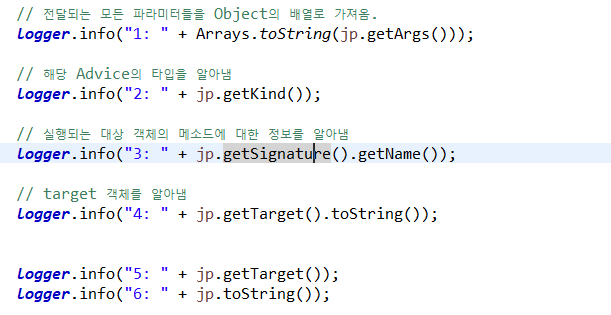
AOP 메서드

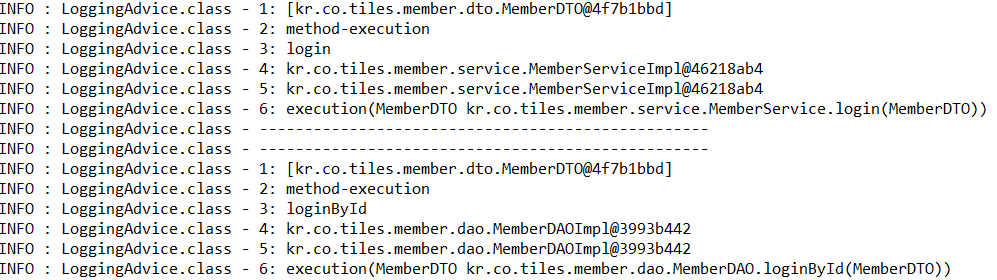
조인포인트

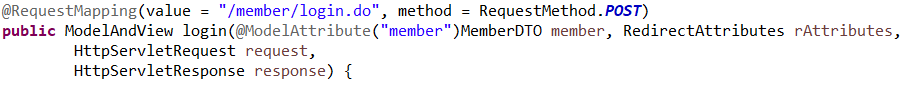
getArgs(): target이 전달받은 argument를 Object 배열로 리턴

getkind() : 클라이언트가 호출한 메소드를 포함하는 클래스 리턴

getSignature() : 어드바이스되는 메서드의 설명 출력함







로그인 과정을 통해 보는 aop 메서드

1. 컨트롤러에서 생성된 memberDTO가 argument로 service와 dad 순서로 전달됨

2. 어떤 종류의 조인포인트인지 알려줌 spring에선 조인포인트는 메서드 뿐이다

3. getSignature().getName()을 통해 실행되는 메서드의 이름을 리턴, Signature는 인터페이스

로 여러 메서드들이 있다.

4, 5타겟 객체를 반환한다.

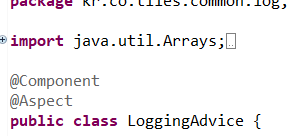
6. 조인포인트의 객체 리턴

Spring AOP 적용 방법



최상위 패키지 클래스(컨트롤러)에 @EnableAspectJAutoProxy 적용

스프링이 자동으로 매서드 호출 전 인터셉트함



어드바이스 클래스에 @Aspect, @Component 어노테이션함

@Component, @Bean 차이

@Bean: 개발자가 직접 제어 불가능한 외부라이브러리

@Component : 직접 만든 class를 bean으로 만듬

포인트컷 표현식

AspectJ에선 포인트컷 표현식은 지시자를 이용하여 작성함

execution : 메서드 실행 조인포인트를 매칭한다. 스프링에서 가장 많이

쓰임

@Before, @After, @Around

@Before("execution(\* kr.co.tiles.\*.service.\*.\*(..)) or "

+ "execution(\* kr.co.tiles.\*.dao.\*.\*(..))")

execution(\* kr.co.tiles.\*.service.\*.\*(..))

\*: 모든 리턴 타입 허용

kr.co.tiles : kr.co.tiles.

\*: 내부의 모든 패키지

service. : service 패키지

\*.: 모든 오브젝트

\*(..): 모든 메서드와 모든 파라미터 허용