interceptor filter aop 차이

호출시점

dispatcherservelt 전후로 구별(servlet 단위) aop proxy패턴 형태로 실행

설정위치

filter web.xml과 interceptor spring-servlet.xml aop 메소드 제작후 각메소드에 지정

구현방식

filter web.xml설정으로만 구현가능 interceptor설정후 메서드 구현도 필요

순서

Filter ->Interceptor -> AOP ->Interceptor ->Filter

init -> doFilter -> preHanlder -> @Before ->method ->@After -> postHanlder -> dofilter

Filter 실행메서드

개념

request 와 response 부분을 정제하는 역활

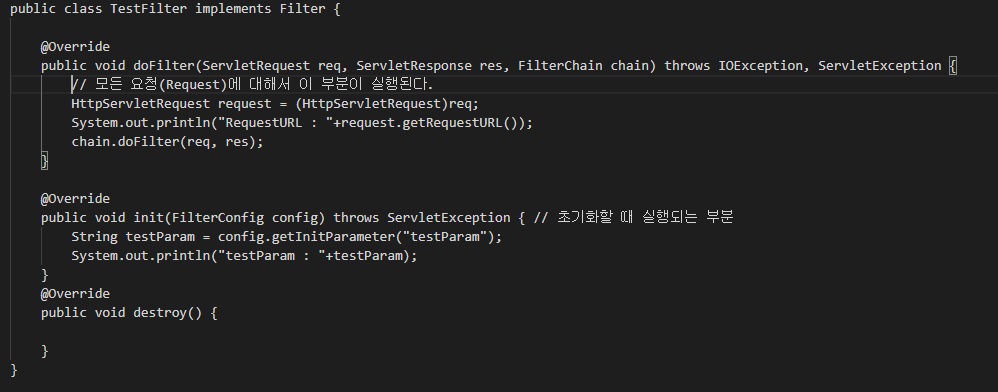
보통 인코딩 변환 처리/XSS방어 등의 요청에 대한 처리 사용

init() - 필터 인스턴스 초기화

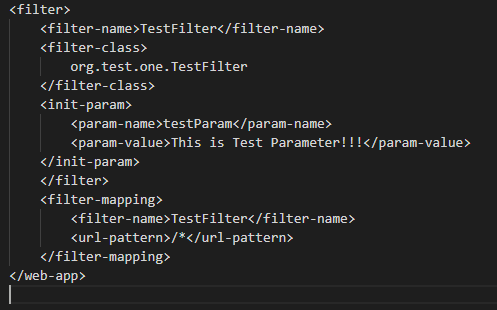
doFilter() - 전/후처리

destroy() - 필터 인스턴스 종료

Filter 예제



web.xml



출처: https://nine01223.tistory.com/238 [스프링연구소(spring-lab)]

Interceptor

스프링 컨텍스트 내부에서 Controller 에 관한 request과 response에 대해 처리

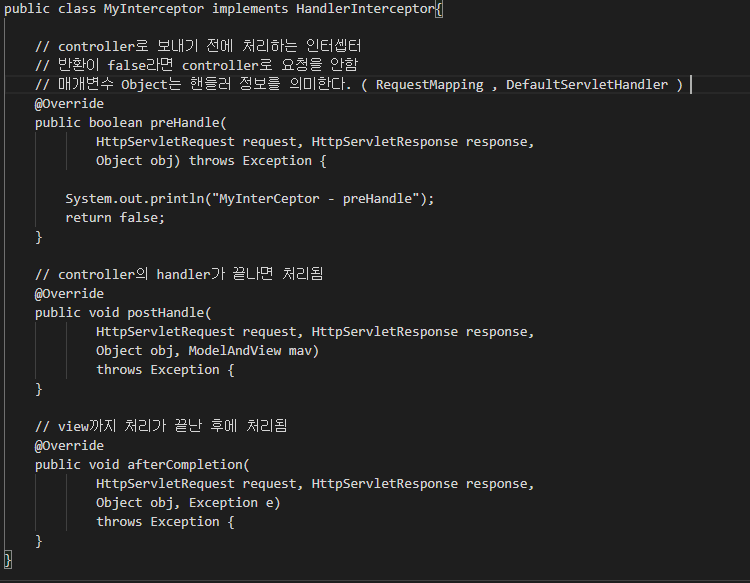
preHandle()- Controller Methods 실행전

postHandle() - Controller Methods 실행후 view페이지 rendering 전

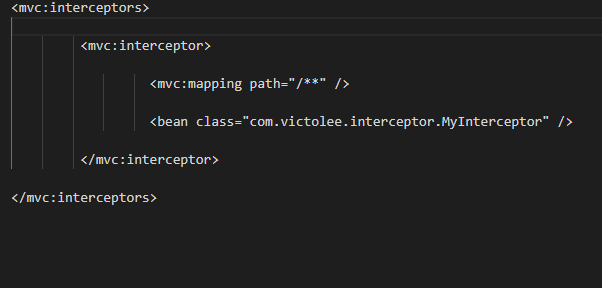
afterCompletion() - view페이지가 rendering 후

intercepter 예제

method



spring-sevlet.xml

출처:https://victorydntmd.tistory.com/176

AOP

구성형태

Aspect - 공통된 부분을 모듈화 한것

Target - Aspect 적용하는곳

Advice - 실적적인 부가기능을 담은 구현체

JointPoint - Adivce 적용될 때 사용

PointCut - JointPoint 상세한 스펙을 정의

사용시기

보통 로깅,트랜젝션,에러처리등 비즈니스로직 부분에서 조금 더 세밀하게 조정할떄 사용

Interceptor와 Filter 와 달리 세밀하게 설정이 가능하다.

AOP는 주소,파라미터,어노테이션등 다양한 방법으로 대상을 지정할 수 있다.

AOP 호출 방법

@before - 지정 method 전

@Aftrer - 지정 method 후

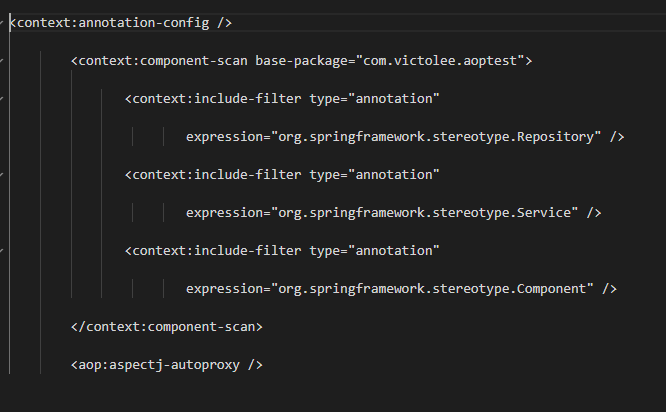
@After - returning 대상 method의 정상적인 실행 후

@After- throwing exception 발생후

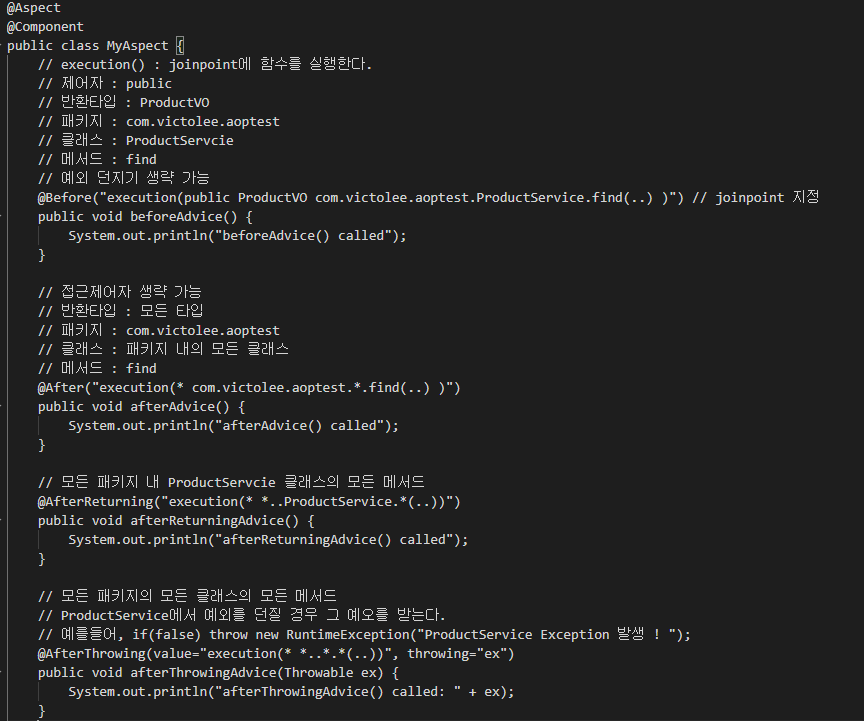
@Around - 지정 method 수행 전/후

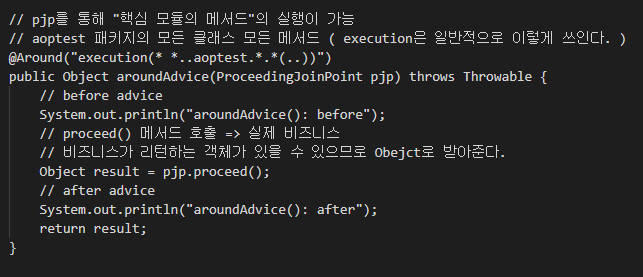
aop 예제

applicationContext.xml



method





출처 https://victorydntmd.tistory.com/178