**AOP**

관점 지향 프로그래밍(Aspect-Oriented Programming)

속성 지향 프로그래밍(Attribute-Oriented Programming)

**관점 지향 프로그래밍(Aspect-Oriented Programming)**

목표: 관심사의 분리 (핵심기능(로직)과 부가기능(로직)을 분리시키자.)

이것을 가능하게 한 것은? Proxy…

핵심 기능은 종단 관심사 (비즈니스로직)

부가 기능은 횡단 관심사 (호출 로그등 부가적으로 넣는 기능들)

* 핵심 로직은 아니지만 코드에 전반적으로 반복적으로 사용하며, 필요한 로직들.

**ASPECT(부가기능(추상적 개념))**

->추상명사로 횡단 관심사를 의미 ->ex) 로깅, 보안, 트랜잭션 등

**Advice(ASPECT기능을 구현)**

->횡단 관심사를 구현한 객체

**Target(핵심 로직을 구현)**

->핵심 로직을 가지고 있는 객체

**Proxy**

->Target객체 + Advice

프록시를 통해서, 외부에 감추어진 Target객체를 호출한다.

Board의 Mapper Interface를 확인해보면 된다. 그것을 호출 찍어보면 Mapper Proxy가 나오는데 그 Proxy와 같다. BoardController -> BoardSerivce -> BoardServiceImpl -> BoardMapper…

JoinPoint ->Advice의 적용 대상 – 스프링에서는 Target의 메소드

Pointcut ->여러JoinPoint들 중에서 Advice가 적용되는 select 기준(어느 메소드에 적용시킬 것이냐)

-------------------------------실습-------------------------------------------

AOP를 가능하게 해주는 dependencies는 Aspectjrt와 aspectjweaver이다.

새로운 스프링 레거시를 생성 후, pom.xml을 스프링 프로젝트 어딘가에 따와서, aspectjweaver를 넣어주고, 확인. 메이븐 업데이트.

그리고 mybatis-config.xml도 어디선가 따온 후, src/main/resources안에 넣고, 그 xml안에 mapper부분에 <mapper resource = “mappers/TestSqlMapper.xml” />을 작성한다.

그리고 똑같이 src/main/resources밑에 mappers 폴더를 만든 후, 거기에 TestSqlMapper.xml을 만든다.

Root-context.xml에 namespaces에서 context, aop, bean, mybatis 클릭 후 source로 넘어와서

<aop:aspectj-autoproxy/>추가해준다.

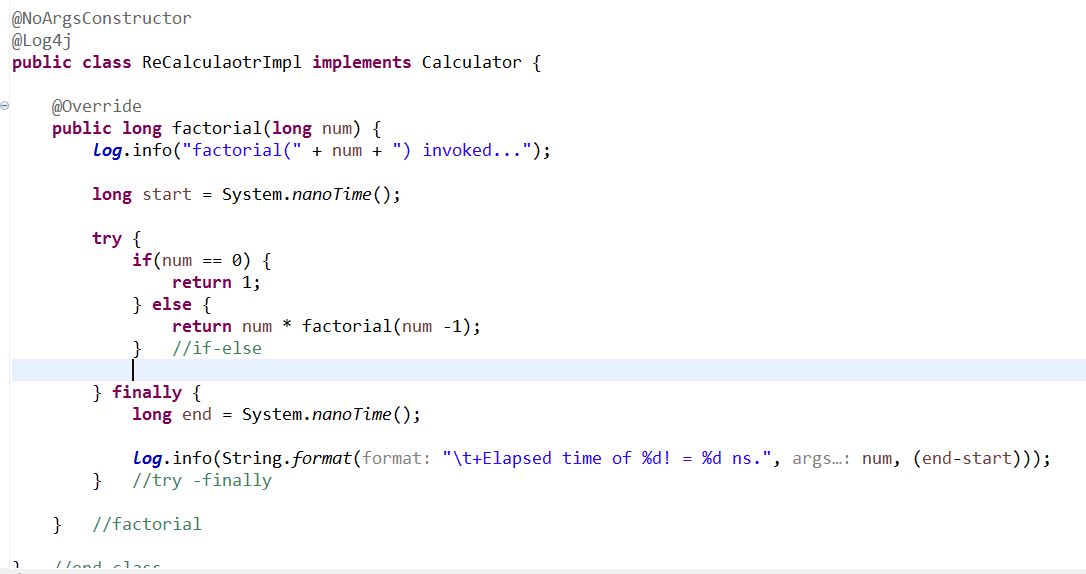
그리고 com.zerock.myapp.aop 패키지를 만들어서, Calculator인터페이스 생성 후,

Public abstract long factorial(long num); 이라는 추상 메소드를 만들어 준다.

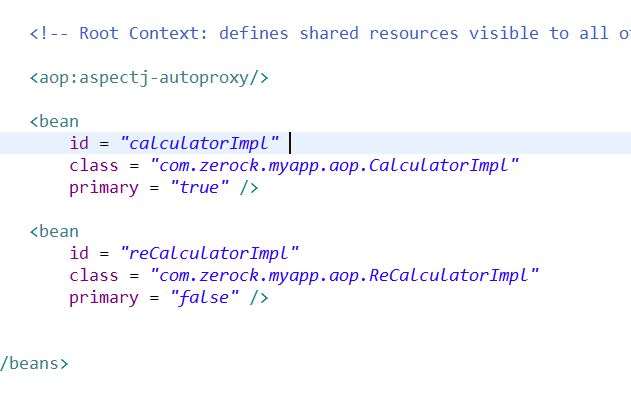
그후 구현 클래스를 만드는데, 그 안의 내용은



이후 또다른 구현 클래스를 만드는데, 그 내용은

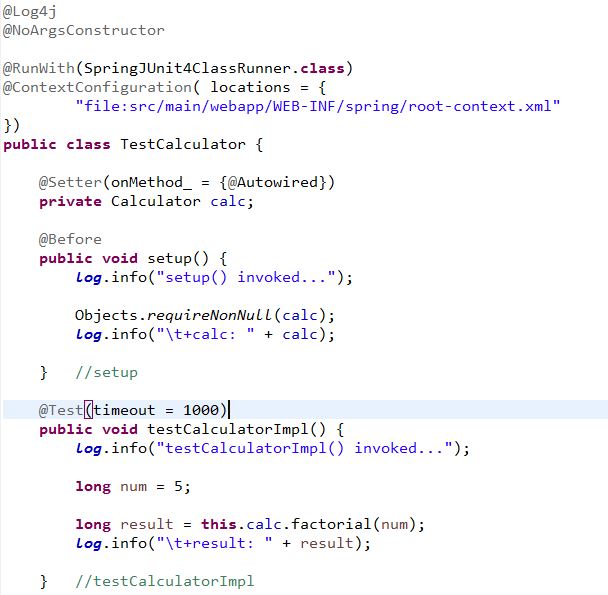


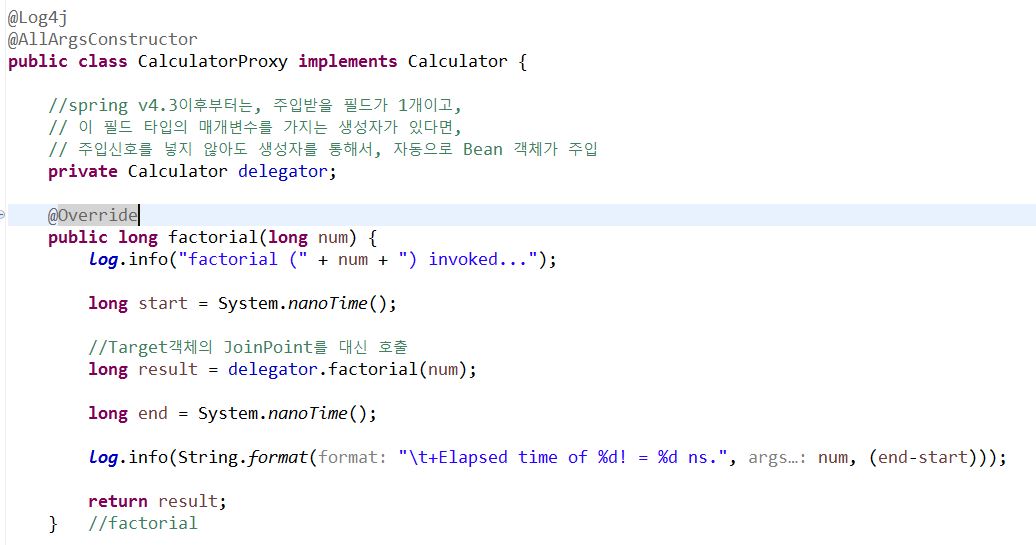
다 작성하고 난 후, 위의 클래스들을 Bean으로 등록하기 위해

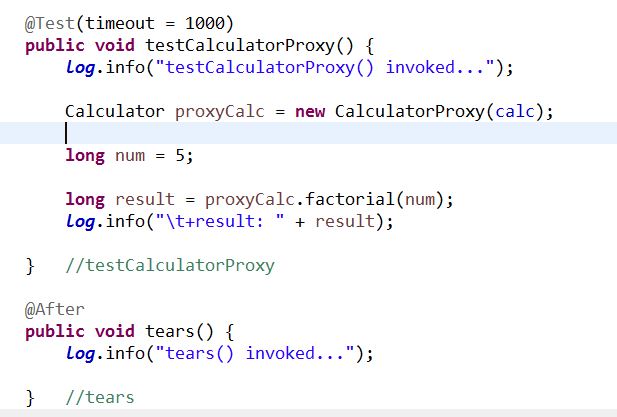


이렇게 만들어주고,

Src/ test/java에 com.zerock.myapp.aop를 하나 만들어서, test클래스를 만들어준다.



이제 Proxy를 사용하기 위해 CalculatorProxy클래스를 만들어주고, 테스트에도 새로운 테스트를 만든다. 



여기서 AOP의 기능은!! 우리가 만약 뭔가를 만들 때 마다 Proxy클래스를 만들면 귀찮으니, 그때 AOP기능이 대신 만들어준다!!! AutoProxy로 바꾸고, Advice로 바꾼다. (어노테이션을 제공 한다.)

AutoProxy는 root-context.xml에 우리가 넣은 것..

root-context.xml 에 <context:annotation-config />추가해준다.

그리고 src/main/java에 com.zerock.myapp.aspect패키지를 하나 만들어준다.

만든 후, ExecutionTimeAspect클래스를 만들어주고, 그 안의 내용은