Aspect Oriented Programing

Advice -> This is an actual piece of code that is invoked during the program execution by Spring AOP framework.

There are 5 Types of Advice:

1. Before (@Before)

Run advice before the method execution.

1. After (@After)

Run advice after the method execution, regardless of its outcome.

1. After Returning (@AfterReturning)

Run advice after the method execution, only if method completes successfully.

1. After Throwing (@AfterThrowing)

Run advice after the method execution only if method exits by throwing an exception.

1. Around (@Around)

Run advice before and after the advised method is invoked.

JoinPoint

t is the actual place in the application where an action will be taken using Spring AOP framework.

Pointcut

Pointcut is a set of one or more JoinPoint where an advice should be executed.

@Pointcut(value="execution(\* com.tyss.centralizedlogging.\*.\*.\* (..))")

Which means advice should be applied to all the methods with any parameter, all the class, all the packages which starts with com.tyss.centralizedlogging.

@AfterThrowing

1. @AfterThrowing(pointcut="execution(\* com.tyss.centralizedlogging.\*.\*.\* (..))", throwing="exception")

This throwing="exception", should be used when we need the details of exception. This "exception", is the parameter name of type Exception in the method.

Note: First \* is access modifier

1. @AfterThrowing( methodName which has annotation @PointCut)

When exception details are not required, we can use this.

JoinPoint Interface

It provides access to the states available at given JoinPoint like method parameter, return value etc

ProceedingJoinPoint

ProceedingJoinPoint is an extension of the JoinPoint. It has one extra method proceed().

Ex: - In the case of @Around, ProceedingJoinPoint should be used. First before execution of actual method it comes to the advice, in advice we will call proceed() method so that the actual execution of program happens.