AOP

# Declaring Pointcuts

# Pointcuts

In Spring AOP (Aspect-Oriented Programming), pointcut expressions define the join points **where** advice (cross-cutting concerns like logging, security, etc.) should be applied. These expressions use AspectJ syntax and can be defined using annotations or XML configurations in Spring Boot.

Official doc: here

# exeuction kind pointcut

This is the most widely used pointcut expression. It allows you to target method executions in your classes.

```
@Before("execution(* com.codingshuttle.aopApp.services.*.*(..))")
public void beforeMethod() {
    // Advice code
}
```

# within kind pointcut

Use within() when you want to limit the advice to a particular class or package, without focusing on specific methods. This pointcut applies to any join point within the com.codingshuttle.aopApp.services package, including methods, fields, and constructors.

```
@Before("within(com.codingshuttle.aopApp.services.*)")
public void logWithinServicePackage() {
    // Advice code
}
```

# annotation kind pointcut

Use @annotation() when you want to apply advice to methods annotated with a particular annotation. You can also create your custom annotations for this pointcut.

```
@Before("@annotation(org.springframework.transaction.annotation.Transactional)"
public void logTransactionalMethods() {
    // Advice code
}
```

# Pointcut declaration

The pointcut is defined separately using @Pointcut and then referenced by the advice annotations.

```
@Pointcut("execution(* com.example.service.*.*(..))")
public void serviceMethods() {}
```

```
// Advice using the pointcut
@Before("serviceMethods()")
public void logBeforeMethod(JoinPoint joinPoint) {
    System.out.println("Logging before method: " + joinPoint.getSignat
}
```