06 SpringAOP

Pointcut 표현식

- Spring에서 AOP를 적용하는 방식 : XML 설정 파일에 Schema 방식으로 적용시키는 것 Bean 클래스 내에 Annotation을 사용하여 적용시키는 것
- AspectJ 표현식에서 사용되는 와일드 카드 : *, .. , +
- Pointcut 지정 시 사용되는 연산자 : !, &&, ||

SpringAOP 표현식

- 매핑 설정을 위한 가장 기본적인 방법은 "*"를 이용하는 것임
- 클래스 지정 : *, !, + 등을 이용하여 특정 패키지 혹은 특정 클래스, 인터페이스 타입으로 표현되는 클래스를 지정할 수 있음
- Modifier 지정 : Pointcut을 지정하면서 특정 Modifier 설정도 가능함
- 매개변수 지정 : 메소드를 지정하는데 있어, 특정 타입의 매개변수, 특정 개수의 매개변수를 가지는 메소드를 지정할 수 있음
- 생성자 지정 : 객체생성 연산자 "new" 를 이용하여 임의의 생성자를 지정할 수 있음

Schema-based AOP 설정

- <aop:config> : Schema 기반의 AOP 설정 태그 중에서 최상위 태그
- <aop:aspect>
  - Aspect : Pointcut(어디에)과 Advice(어떤 횡단 관점을)를 결합한 개념
  - <aop:config> 태그 내에서 Pointcut 및 Advice 설정을 위해 사용됨
- <aop:pointcut> : Pointcut을 지정하기 위해 사용됨
- <aop:advisor> : 외부에서 정의한 advice를 pointcut과 연결할 목적으로 사용함

Advice 설정

- Advice
  - 각 Joinpoint에 삽입되어 동작할 수 있는 코드
  - Spring에서는 before, after, after-returning, after-throwing, around를 제공함

- Before Advice : Pointcut으로 지정된 메소드 호출 시, 메소드가 실행되기 전에 처리될 내용들을 기술하기 위해 사용됨

- After Returning Advice : Pointcut으로 지정된 메소드가 실행된 후, 리턴되는 시점에 처리할 작업을 추가할 때 사용

- After Throwing Advice : Pointcut으로 지정한 메소드가 실행되는 도중에, Exception이 발생한 경우, 동작할 예외처리 Advice를 지정하기 위해서 사용됨

- After Advice : Pointcut으로 지정한 메소드 호출 후에, 예외가 발생하던 발생하지 않던 무조건 실행될 Advice를 지정할 때 사용

- Around Advice : 하나의 Advice가 메소드 호출 전, 후 모두 수행해야 할 로직을 담아야 할 경우 사용

Advice 매개변수

- Advice 클래스의 메소드 내에서 실제 호출되었던 핵심 관점의 클래스 정보라든가 혹은 메소드 정보가 필요할 때
  → SpringAOP에서는 이를 위해 JoinPoint 클래스가 제공됨

- Before Advice 매개변수 : Parameter 값을 받고 싶으면 메소드의 매개변수로 JoinPoint만 선언하면 이 객체를 통해 핵심 관점과 관련된 여러 가지 정보를 알아낼 수 있음

- After Returning Advice 매개변수 : 리턴 값을 받고 싶으면 XML 설정파일에 선언된 변수 이름으로 매개변수를 선언하면 됨

- After Throwing Advice 매개변수 : 리턴되는 Exception 객체를 받고 싶다면 XML 설정파일에 선언된 변수명의 매개변수를 선언하면 됨

Annotation-based AOP 설정

- Annotation 기반의 AOP 설정을 위해서 Spring이 제공하는 Annotation

Annotation	설명
@Aspect	Aspect를 정의할 때 사용함
@Pointcut	Pointcut을 정의할 때 사용함

- Advice 설정에 사용되는 Annotation

Annotation	설명
@Before	Before Advice를 정의할 때 사용함
@After	After Advice를 정의할 때 사용함
@AfterReturning	After-Returning Advice를 정의할 때 사용함
@AfterThrowing	After-Throwing Advice를 정의할 때 사용함
@Around	Around Advice를 정의할 때 사용함

- Pointcut이 해당 Advice 클래스 내에 선언되어 있지 않고, 외부 클래스에 선언된 것도 이용이 가능함
- Annotation을 이용한 경우도 Advice 메소드 내에서 실제 호출된 Target 클래스의 객체와 호출된 매서드 이름 혹은 파라미터 값 등이 사용될 수 있음
  → 이를 이용하기 위해 제공되는 객체가 Joinpoint