12-3. Annotation

1. 어노테이션이란?

- JDK 1.5부터 도입
- 클래스나 메서드 위에 붙여 사용
- 소스코드에 메타코드(추가정보)를 주는 것
- JDK 에서 기본적으로 제공하는 것과 다른 프로그램에서 제공하는것이 있음
    - @Test 는 JUnit 에서 제공
    - 스프링 프레임워크를 사용하여 웹을 구현시 @Service, @Repository 등을 이용하여 쉽게 구현이 가능한데 이는 스프링에서 제공하기 때문
    - Getter 와 Setter 메서드의 자동생성을 위해 lombok 이라는 라이브러리를 많이 사용하는데, 이 라이브러리에서 Getter/Setter 어노테이션 구현을 확인 가능
- 사용자 정의 가능(커스텀 어노테이션)

2. 표준 어노테이션

- 자바에서 기본적으로 제공하는 어노테이션
- 보통 컴파일러가 사용한다
- 메타 어노테이션을 포함

애너테이션	설명
@Override	컴파일러에게 오버라이딩하는 메서드라는 것을 알린다.
@Deprecated	앞으로 사용하지 않을 것을 권장하는 대상에 붙인다.
@SuppressWarnings	컴파일러의 특정 경고메시지가 나타나지 않게 해준다.
@SafeVarargs	지네릭스 타입의 가변인자에 사용한다.(JDK1.7)
@FunctionalInterface	함수형 인터페이스라는 것을 알린다.(JDK1.8)
@Native	native메서드에서 참조되는 상수 앞에 붙인다.(JDK1.8)
@Target*	애너테이션이 적용가능한 대상을 지정하는데 사용한다.
@Documented*	애너테이션 정보가 javadoc으로 작성된 문서에 포함되게 한다.
@Inherited*	애너테이션이 자손 클래스에 상속되도록 한다.
@Retention*	애너테이션이 유지되는 범위를 지정하는데 사용한다.
@Repeatable*	애너테이션을 반복해서 적용할 수 있게 한다.(JDK1.8)

3. 메타 어노테이션

- 어노테이션 정의와 실행을 위한 어노테이션

    - 적용대상이나 유지기간 등을 지정 시 사용
- java.lang.annotation 패키지에 포함되어 있음

@Target

- 어노테이션이 적용가능한 대상을 지정하는데 사용됨

대상 타입	의미
ANNOTATION_TYPE	애너테이션
CONSTRUCTOR	생성자
FIELD	필드(멤버변수, enum상수)
LOCAL_VARIABLE	지역변수
METHOD	메서드
PACKAGE	패키지
PARAMETER	매개변수
TYPE	타입(클래스, 인터페이스, enum)
TYPE_PARAMETER	타입 매개변수(JDK1.8)
TYPE_USE	타입이 사용되는 모든 곳(JDK1.8)

@Retention

- 어노테이션이 유지되는 기간을 나타냄
- 유지정책은 RetensionPolicy(Enum class)에 정의되어있으며 3가지가 존재
    - SOURCE : 컴파일러에 의해 삭제됨 (컴파일러만 사용)
    - CLASS : 기본값, 클래스 파일에 존재하지만 실행시에 사용은 불가하다
    - RUNTIME : 클래스 파일에 존재하며 실행시에 사용 가능하다

4. 어노테이션 Document 에서 확인하기

@Override

- 상위 타입의 메서드 선언을 대체한다는 의미
- @Retention(RetentionPolicy.SOURCE)
    - 컴파일러가 확인 후 바이트 코드에서는 사라지게 됨
- Override 문서 링크

@Documented

- javadoc 및 유사한 도구에 의해 문서화됨을 의미

- @Retention(value=RUNTIME)
  - Retention 은 RUNTIME 으로 여러 도구에서 해석하여 확인 가능
- Documented 문서 링크

@Repository

- DDD(도메인 주도 설계)에 의해 특정 동작을 캡슐화하는 매커니즘으로 정의된 'Repository'임을 의미
- @Target(value=TYPE)
  - 적용가능한 대상은 클래스, 인터페이스, enum
- @Retention(value=RUNTIME)
  - 유지기간은 실행 시에도 유지됨
- Repository 문서 링크

@Getter

- Compile 시 클래스의 멤버변수의 Getter 메서드를 자동 생성
- @Target({ElementType.FIELD, ElementType.TYPE})
  - 적용가능한 대상은 필드와 클래스, 인터페이스, enum
- @Retention(RetentionPolicy.SOURCE)
  - 유지기간은 컴파일 시 까지

```
package lombok;

import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Target({ElementType.FIELD, ElementType.TYPE})
@Retention(RetentionPolicy.SOURCE)
public @interface Getter {
    AccessLevel value() default AccessLevel.PUBLIC;

    Getter.AnyAnnotation[] onMethod() default {};

    boolean lazy() default false;

    /** @deprecated */
    @Deprecated
    @Retention(RetentionPolicy.SOURCE)
    @Target({})
    public @interface AnyAnnotation {
    }
}
```