**11/29 김민식 OJT Day 20**

**Spring FrameWork 심화 교육 Day 02**

**7강. DI 애플리케이션 작성(2)**

[Spring Framework Basic 7강 DI 애플리케이션 작성(2) | T아카데미 - YouTube](https://www.youtube.com/watch?v=dJavXu6DB9s&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=7)

1. Spring-Test를 사용한 DI 테스트 클래스

* 테스트를 지원하는 어노테이션
  + 에러 : 어노테이션을 사용하려면 자바 빌드패스를 사용 해야함
* Spring-Test
  + 에러 : pom.xml에서 scope부분 삭제

**8강. DI 애플리케이션 작성(3)**

[Spring Framework Basic 8강 DI 애플리케이션 작성(3) | T아카데미 - YouTube](https://www.youtube.com/watch?v=lnyi6lQN0nw&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=8)

1. Bean 의존관계 설정 방법

* Setter Injection : <property> 태그
* Constructor Injection : <constructor-arg> 태그

1. 프로퍼티(Property) 값 설정 방법

* 단순 값의 주입 : property 태그의 value 속성을 사용
* 컬렉션(리스트,맵 등) 타입의 값 주입 : list와 value 태그 이용, 맵과 엔트리 태그를 이용

1. 프로퍼티(Property) 파일을 이용한 설정 방법

* 환경에 따라 자주 변경되는 내용의 분리

**9강. DI 애플리케이션 작성(4)**

[Spring Framework Basic 9강 DI 애플리케이션 작성(4) | T아카데미 - YouTube](https://www.youtube.com/watch?v=gs85JaMCUys&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=9)

1. Bean 등록 메타정보 구성 전략

* XML 단독 사용 – 단독 운영환경 적합
* XML과 빈 스캐닝의 혼용 – 개발환경에 적합

1. Bean 등록 및 의존관계 설정 Annotation

* Bean 등록 Annotation
  + @Component : 빈 태그와 동일한 역할을 함
  + @Repository : 파일, 데이터페이스를 가진 클래스
  + @Service : 서비스 레이어, 비즈니스 로직을 가진 클래스
  + @Controller : 프리젠테이션 레이어, 웹 어플리케이션에서 웹 요청과 응답을 처리하는 클래스
* Bean 의존관계 주입 Annotation
  + @Autowired 정밀한 의존관걔 주입이 필요한 경우에 유용
  + @Resource : 프로퍼티, setter 메서드에 적용 가능
  + @Value : 단순한 값을 주입할 때 사용
  + @Qualifier : @Autowired 어노테이션과 같이 사용
* Component Scan을 지원하는 태그

1. 프로퍼티(Property) 파일을 이용한 설정 방법

**10강. 사용자관리 프로젝트**

[Spring Framework Basic 10강 사용자관리 프로젝트 | T아카데미 - YouTube](https://www.youtube.com/watch?v=1onsObyFNGY&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=10)

1. 사용자 관리 프로젝트 아키텍쳐

* 프리젠테이션 계층, 서비스 계층, 데이터액세스 계층 분리
* 도메인 모델 클래스
* 각 계층 사이에서는 인터페이스
  + 프리젠테이션 계층
    - 브라우저상의 웹클라이언트의 요청 및 응답을 처리
    - 도메인 모델을 사용
    - 최종 UI에서 입력한 데이터에 대한 유효성 검증 기능
  + 서비스 계층
    - 비즈니스 로직 처리
    - 트랜잭션 처리
    - 다른 계층들과 통신하기 위한 인터페이스를 제공
  + 데이터 액세스 계층
    - DBMS를 조작하는 데이터 액세스 로직을 객체화
    - DAIO 인터페이스와 @Repository를 이용
  + 도메인 모델 클래스
    - VO(Value Object) DTO(data Transfer Object)
    - 3개의 계층 전체에서 사용
* 테이블 설계

1. 사용자 관리 프로젝트 아키텍쳐

* 클래스의 역할
  + 프리젠테이션 계층
    - UserController 클래스
      * UI 계층과 서비스 계층을 연결하는 역할을 하는 클래스
      * UserController 통해서 UserService를 사용
  + 서비스 계층
    - userService 인터페이스
      * 서비스 계층에 속한 상위 인터페이스
    - UserServiceImpl 클래스
      * UserService 인터페이스를 구현한 클래스
      * 복잡한 업무 로직이 있을 경우 이 클래스에서 업무 로직을 구현
  + 데이터 액세스 계층
    - UserDao 인터페이스
      * 데이터 액세스 계층에 속한 상위 인터페이스
    - UserDaoImplJDBC 클래스
      * 데이터 액세스 로직을 구현
      * SpringJDBC를 사용하는 경우 DataSource를 IoC 컨테이너가 주입
      * MyBatis를 사용하는 경우 SqlSession을 IoC컨테이너가 주입

1. 사용자 관리 프로젝트 아키텍쳐

* UserVO.java
* UserService.java
* UserServiceImpl.java
* UserDao.java
* UserDaoImplJDBC.java

**11강. Spring JDBC 개요**

[Spring Framework Basic 11강 Spring JDBC 개요 | T아카데미 - YouTube](https://www.youtube.com/watch?v=vSkxkBOs_KQ&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=11)

1. 데이터 액세스 공통 개념

* DAO(Data Access Object) 패턴
  + 비즈니스 로직과 데이터 액세스 로직을 분리
* 컨넥션 풀링을 지원하는 DataSource
  + 컨넥션 풀링 : 미리 정해진 개수만큼의 DB 컨넥션을 풀(Pool)에 준비해두고, 애플리케이션이 요청할 때마다 Pool에서 꺼내서 하나씩 할당, 다시 돌려받아서 Pool에 넣는 기법
* DataSource 구현 클래스 종류
  + 테스트환경을 위한 DataSource
    - SimpleDriverDataSource : 가장 단순한 DataSource 구현 클래스
    - SingleConnectionDriverDataSource : 순차적으로 진행되는 통합 테스트
  + 오픈소스 DataSource
    - Apache Commons DBCP : 가장 유명한 오픈소스 DB 커넥션 풀 라이브러리
  + C3p0 JDBC/DataSource Resource Pool

1. Spring JDBC 개요

* JDBC
  + 모든 자바의 데이터 액세스 기술의 근간
  + ORM(Mybatis Hibernate) 도 내부적으로는 DB와의 연동을 위해 JDBC를 이용
  + 로우 레벨 기술로 인식
  + 중복된 코드가 반복적으로 사용
* Spring JDBC
  + 기존 JDBC의 단점을 극복, 간결한 형태의 API 사용법을 제공
  + 반복적으로 해야 하는 많은 작업들을 대신 해준다
* Spring JDBC가 해주는 작업
  + Connection 열기와 닫기
  + Statement 준비와 닫기
  + Statement 실행
  + ResultSet Loop 처리
  + Exception 처리와 반환
  + Transaction 처리
* JdbcTemplate 클래스
  + JDBC의 모든 기능을 최대한 활용
* JdbcTemplate 클래스 생성
  + DataSource를 파라미터로 받아서 아래와 같이 생성
* JdbcTemplate 클래스 update() 메서드
  + INSERT, UPDATE, DELETE와 같은 SQL을 실행할 때 사용
* JdbcTemplate 클래스의 queryForObject() 메서드
  + 여러 개의 칼럼, 반환되는 한 개의 Row
* JdbcTemplate 클래스의 query() 메서드
  + 여러 개의 칼럼, 반환되는 여러 개의 Orw

**12강. Spring JDBC 환경설정**

[Spring Framework Basic 12강 Spring JDBC 환경설정 | T아카데미 - YouTube](https://www.youtube.com/watch?v=EZgwmub6OdY&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=12)

**13강. AOP 개요**

[Spring Framework Basic 13강 AOP 개요 | T아카데미 - YouTube](https://www.youtube.com/watch?v=X4vsFpPg1z4&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=13)

1. AOP 개요와 용어

* 핵심기능과 부가기능
  + 업무(Biz) 로직을 포함하는 기능을 핵심기능
  + 로깅, 보안 등 부가적인 기능을 부가기능
* AOP의 개요
  + 관심사의 분리(기능의 분리)
  + 애스팩트(Aspect)라는 독특한 모듈형태로 만듬
* 애스팩트(Aspect)
  + Advice+PointCut
* AOP 용어
  + 타겟(Target) : 핵심기능을 담고 있는 모듈
  + 어드바이스(Advice) : 타겟에 제공할 부가기능을 담고 있는 모듈
  + 조인 포인트(Join Point) : 어드바이스가 적용될 수 있는 위치
  + 포인트 컷 (Pointcut) : 어드바이스를 적용할 타겟의 메서드를 선별
  + 애스펙트 (Aspect) : 어드바이스 + 포인트컷
  + 어드바이저 (Advisor) : 어드바이스 + 포인트컷, spring Aop에서만 사용
  + 위빙 (Weaving) : 포인트컷에 의해 결정된 타겟의 조인 포인트에 부가기능을 삽입하는 과정

1. Spring AOP의 특징

* Spring은 프록시(Proxy) 기반 AOP를 지원
* 프록시가 호출을 가로챈다
* Spring AOP는 메서드 조인 포인트만 지원
* Spring AOP의 구현 방식
  + Xml 기반의 POJO 클래스를 이용한 AOP 구현
* @Aspect 어노테이션을 이용한 AOP 구현
* **14강. AOP애플리케이션 작성**

[Spring Framework Basic 14강 AOP애플리케이션 작성(1) | T아카데미 - YouTube](https://www.youtube.com/watch?v=x0uuJWhhPgU&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=14)

1. Advice 클래스 작성

* Spring AOP의 구현 방식
  + XML 기반의 POJO 클래스를 이용한 AOP 구현
  + @Aspect 어노테이션을 이용한 AOP 구현
* Advice의 종류
  + Around : Joinpoint 앞과 뒤에서 실행
  + Before : Joinpoint 앞에서 실행
  + After Returning : 정상적으로 종료된 뒤 실행
  + After Throwing : 예외가 던저질 때 실행
* Advice 클래스 정보
  + 클래스명 : PerformanceTraceAdvice.java
  + 클래스 기능 : 타겟 객체의 메서드 실행시간을 계산해서 출력
  + Advice 유형 : Around어드바이스
* JoinPoint 인터페이스
  + AOP가 적용되는 지점
* Advice를 정의하는 태그

1. PointCut 표현식 문법

* PointCut 표현식 문법
  + Execution() 지시자를 사용한 포인트컷 표현식의 문법구조
    - Execution([접근제한자 패턴]타입패턴 [타입패턴.] 이름패턴 (타입패턴|”..”,…)
* PointCut 표현식 예시
  + Execution(\* hello(..))
    - 헬로라는 이름을 가진 메서드를 선정
  + Execution(\* hello())
    - 파라미터 패턴이 ()로 되어 있으니 hello 메서드 중에서 파라미터가 없는 것만 선택

**15강. AOP애플리케이션 작성**

[Spring Framework Basic 15강 AOP애플리케이션 작성(2) | T아카데미 - YouTube](https://www.youtube.com/watch?v=eLJBXb9QO4o&list=PL9mhQYIlKEhfYqQ-UkO2pe2suSx9IoFT2&index=15)

1. Aspect 클래스 선언 및 설정

* Spring AOP의 구현 방식
  + XML 기반의 POJO 클래스를 이용한 AOP 구현
  + @Aspect 어노테이션을 이용한 AOP 구현
* @Aspect 어노테이션
  + Aspect클래스 선언할 때 사용
* Aspect 클래스 정보
  + 클래스명 : LoggingAspect.java
* Aspect 클래스 선언 및 설정
  + 클래스 선언부에 @Aspect 어노테이션을 정의
  + @Component 어노테이션도 함께 정의
  + XML 설정파일에 <aop:aspectj-autoproxy/> 선언

1. Aspect 클래스 구현

* Advice를 정의 하는 어노테이션
  + Before : 메서드가 실행되기 전에 호출
  + After : 메서드가 정상 종료됐을 때와 예외가 발생했을 때 모두 호출
  + Around : 전 후 과정 모두 담을 수 있는 강력한 기능
* Before 어드바이스

질문사항

1. 엑셀파일 활용