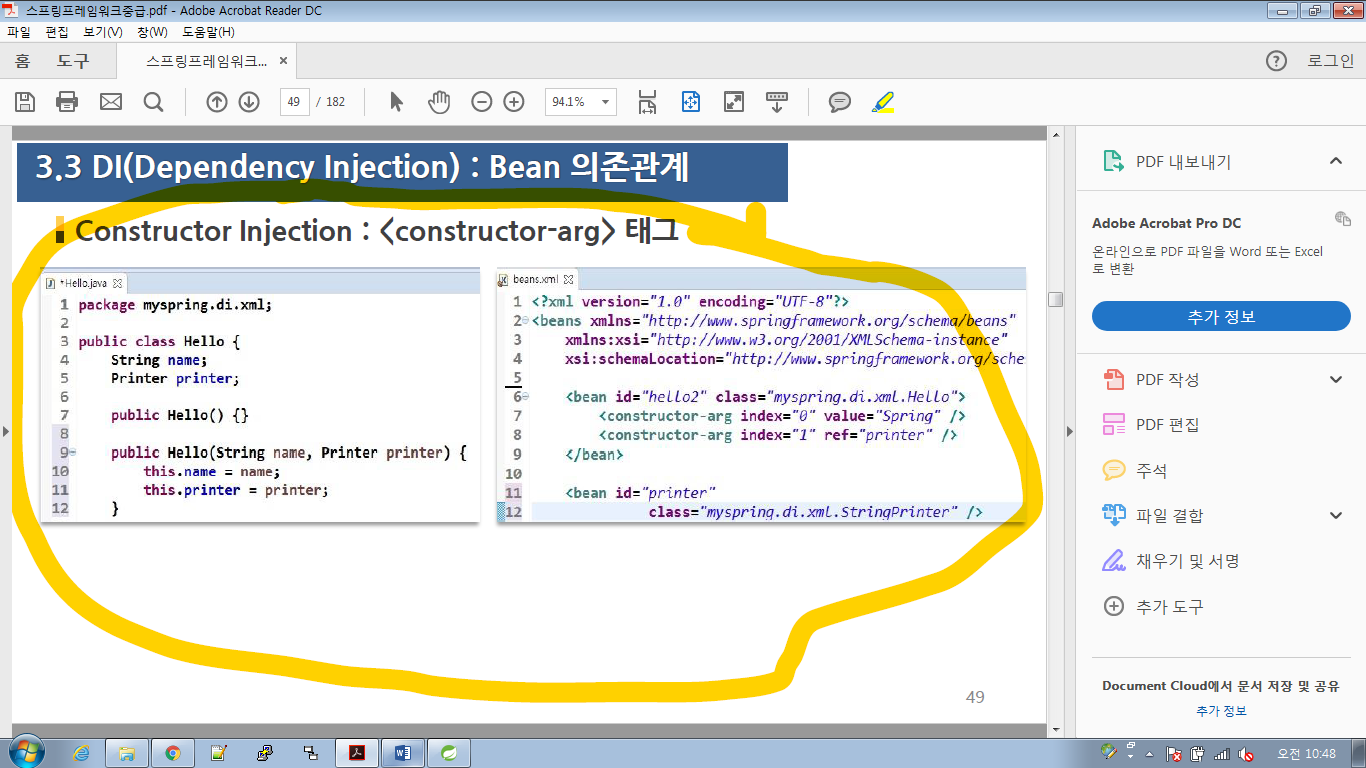
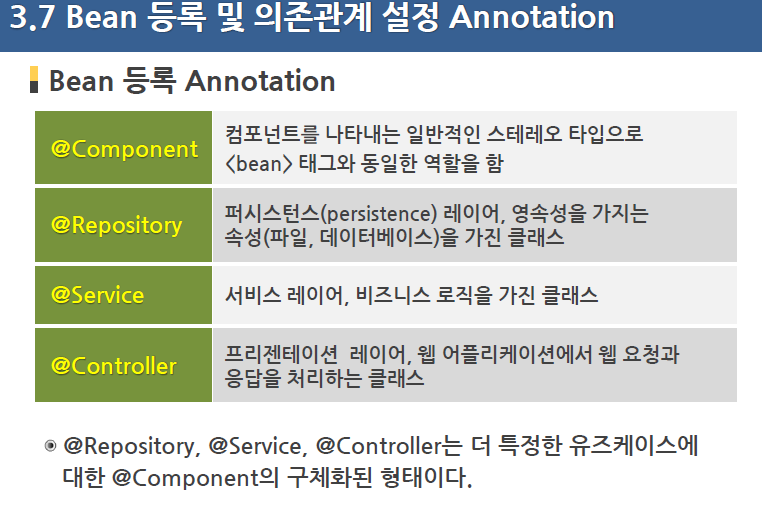
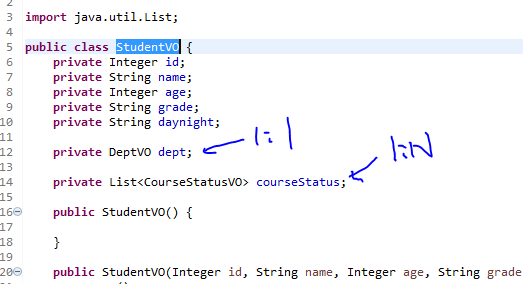
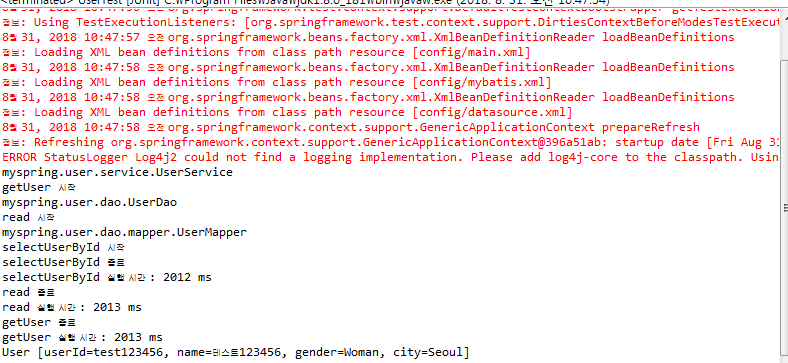
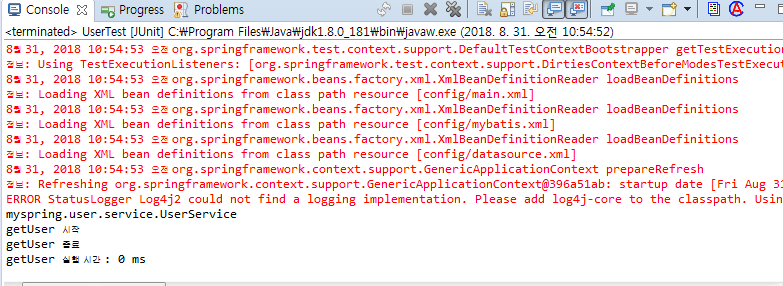
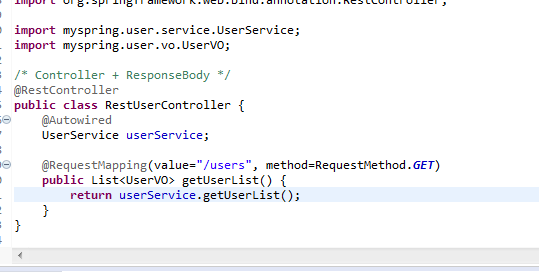
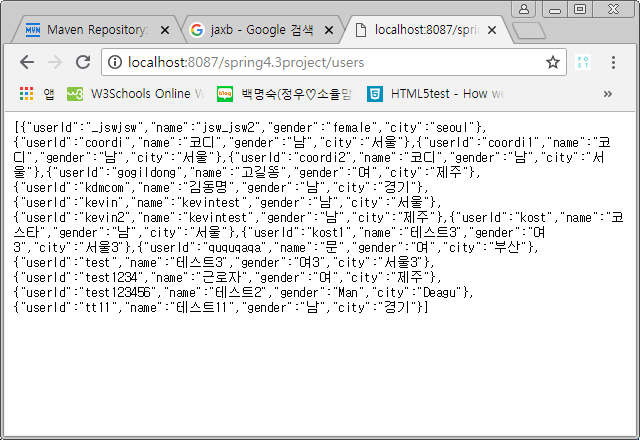
1. 환경 준비
   1. Blog.naver.com/vega2k
   2. Jdk 1.8
   3. STS 64bit
   4. Tomcat 8.5.33
   5. Maven
   6. Java Configuration
      1. Java3부터 지원
      2. Xml로 관리하던 것을 java config로 설정할 수 있도록 한다.
   7. AOP, Transaction, REST API를 중점으로 다룸
   8. 스프링부트
      1. <http://spring.io/projects> > spring boot
      2. Spaing batch : 스케쥴러 기능은 없음. 배치 프로그램 작성할 수있도록 해주는 프로젝트
   9. 레퍼런스
      1. 4.3.18 API : <https://docs.spring.io/spring/docs/4.3.18.RELEASE/javadoc-api/>
2. 스프링프레임워크
   1. IoC는 제어의흐름이 역순이다. Xml로 먼저 설정 후 다음 단계는 java 나아가서 sprispringboot로 나아간다.
   2. DL : WAS에 의존적인 설정. DI : WAS와 상관없이 독립적으로 돌아간다.
   3. DI
      1. 전략1
         1. xml설정이 많아지는 단점이 발생. -> 그래서 어떻게 ? Xml과 어노테이션을 섞어서 사용하는 방법이 나옴(2.5부터)
      2. 전략2
         1. 현재 TRS는 혼용해서 사용하고 있음. 컨트롤러는 어노테이션, Dao는 Xml정의 등… 일부 bean객체 확인 필요.
         2. 같은 Interface를 바라보는 bean객체는 Component(리소스명), Qualifier(리소스명) 을 사용하면 됨.
      3. 전략3
         1. 설정할 때, Java만 사용
         2. @Configuration 을 통해서 설정 Java 명시
         3. Java 설정을 사용하면 객체를 직접 생성해줘야 한다. Xml방식은 개발자가 직업 생성하지 않았음
3. 실습환경
   1. scope이란 tag는 제거
   2. Spring context : 4.3.18
   3. Junit : 4
   4. Spring test : context 버전과 맞춰줘야 함
      1. Junit을 wrapping 함 . Test case 작성
   5. MyBatis
      1. 
      2. 
      3. Pk를 표현할때는 id로 가져온다. ㄴ
      4. 연관이 있는 객체는 참조형태로만 가지고 오는 게 테이블이 변경되었을 때, 서로 영향을 최소한으로 줄 수 있는 방법이다.
   6. AOP(관점 지향 프로그래밍)
      1. Spring Aop
      2. Aspectj
      3. Aspectj weaver 라이브러리 활용
      4. Aspectj 포인트컷 표현식
         1. .. : 하위 디렉토리
         2. . : 파일
      5. 실습 결과
         1. proceed처리
            1. 
         2. Proceed 안하면..
            1. 
      6. JackSon라이브러리
         1. Java -> Json 형태, Json -> Java 형태로 변환시켜준다.
         2. 
         3. 
4. 기타
   1. Css 참고 : https://cssreference.io