디비에 있는 데이터는 커밋을 해줘야 한다

스프링 실행시 순서( Soldesk, Student 클래스 사용 )

스프링이 사용하는 메모리 공간을 만든다 ⇒ 컨텍스트 생성( ApplicationContext라는 객체)

Root-Context.xml 파일에 스프링이 객체를 생성하고 관리하는 설정이 이루어진다

Root-Context.xml에 설정되어 있는 ComponentScan을 사용해서 패키지를 스캔한다

스캔된 패키지에 있는 클래스들 중 스프링이 사용하는 @Component를 찾아 클래스의 인스턴스를 생성한다

Soldesk객체는 Student가 필요하다는 설정을 하고, 스프링은 Student객체의 레퍼런스를 Soldesk객체에 주입한다

root-context.xml을 활용한 TEST ( log4j, RunWith, ContextConfiguration, Data, Select ) 사용

db에 접속하기 위해 HikariConfig 를 빈(Bean) 생성 하고 property로 드라이버,url,id,pw를 등록한다

Datasource,SqlsessionFactoryBean 도 빈(Bean) 생성해준다

src/main/java - mapper패키지에서 인터페이스를 생성한뒤

오라클 db의 기본 dual 테이블에서 현재시간을 가져오기 위해

@Select 어노테이션을 사용해서 select sysdate from dual 으로 현재시간을 출력하는 쿼리문을 작성

해당 어노테이션을 사용하는 메소드 getTime()을 만든다

public interface TMapper {

@Select("SELECT sysdate FROM dual") public String getTime();

}

<mybatis-spring>태그를 통해 root-context.xml 에 mapper 패키지를 스캔해주도록 한다

src/test/java 에서 테스트를 위한 mapper패키지와 test 클래스를 생성한다

생성된 클래스 상단에

@RunWith(SpringJUnit4ClassRunner.class)

@ContextConfiguration("file:src/main/webapp/WEB-INF/spring/root-context.xml")

@Slf4j / log4j

를 작성해서 jUni4를 통해 테스트를 실행하도록 설정하며 root-context.xml 을 경로지정해준다

@Slf4j / log4j 는 로깅유틸리티로서 로그를 생성한다

테스트 클래스 내부에서 @Autowired 로 TMapper 를 자동으로 주입

TMapper 내부에 getTime() 메소드를 테스트하기 위한 메소드를 생성후 @Test 를 부착하며

메소들 내부에서 log4j의 [log.info](http://log.info)() 를 이용 TMapper.getTime() 을 호출한다

@Autowired private TMapper tm;

@Test public void te() {

log.info("getTime1"); log.info(tm.getTime());

}

실행시 Run as JUnit Test 로 실행하면 올바르게 작성시 현재시간이 출력된다

src/main/resources 에서 xml 파일 설정후 테스트하는 방법

db에서 작업한 내용을 가져오기 위한 객체와 그 안의 필드값을 생성한뒤 @Data 를 사용한다

@Data public class BoardVO {

private int bnum;

private String title;

private String content;

private String writer;

private Date redate;

private Date updateDate;

}

인터페이스 mapper 생성과 db안의 값을 가져올 메소드를 만든다

public interface BMapper { public List<BoardVO> getList(); }

src/main/resources 에서 패키지파일을 만들면 폴더처럼 보이지만 실제로는 패키지형식

aa.bb.mapper 패키지 생성 후 xml 파일의 이름은 BMapper.xml 로 생성한다

xml 파일안 의 내용은

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper PUBLIC "-[//mybatis.org//DTD](https://mybatis.org//DTD)Mapper 3.0//EN" "<http://mybatis.org/dtd/mybatis-3-mapper.dtd>"> // 이부분이 있어야 mybatis의 mapper사용이 가능하다

<mapper namespace="aa.bb.mapper.BMapper"> //mapper 인터페이스를 사용하기 위해 경로지정 <select resultType="aa.bb.main.BoardVO" id="getList">

// 결과 값을 받아 올때 타입 지정 과 mapper 의 어떤 함수를 사용할 지 지정해준다 select \* from t\_board order by bnum desc // 사용할 쿼리문 작성 </select>

</mapper>

src/test/java 에서 패키지 생성, 클래스 생성 후

@RunWith(SpringJUnit4ClassRunner.class) @ContextConfiguration("file:src/main/webapp/WEB-INF/spring/root-context.xml") @Log4j

public class TMapperTests {

@Autowired private BMapper bm;

@Test public void bt() {

log.info(bm.getList());

}

}

log4j의 info() 는 object를 반환하고 slf4j의 info()는 String 을 반환하므로

slf4j 사용시에는 toString을 사용해주어야한다