

2024년 상반기 K-디지털 트레이닝

비즈니스 계층

[KB] IT's Your Life



DTO, Data Transfer Object

- o VO, Value Object
 - 데이터베이스 테이블의 모양을 그대로 반영한 객체
 - Dao에서 사용

o DTO

- Dao를 제외한 나머지 계층(Service, Controller, View 등)에서 사용
- 테이블에 없지만 비즈니스 로직에 필요한 추가 정보도 포함 가능

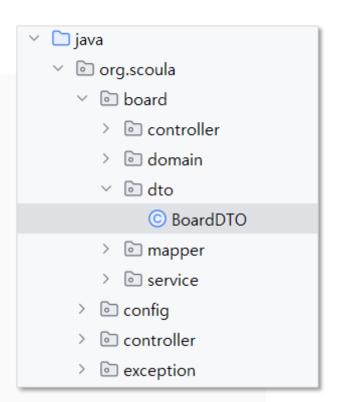
o VO와 DTO간 상호 변환 기능 필요

- VO → DTO
- DTO → VO

비즈니스 계층의 설정

☑ BoardDTO.java

```
package org.scoula.board.dto;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
@Data
@NoArgsConstructor
@AllArgsConstructor
@Builder
public class BoardDTO {
    private Long no;
    private String title;
    private String content;
    private String writer;
    private Date regDate;
    private Date updateDate;
```



☑ BoardDTO.java

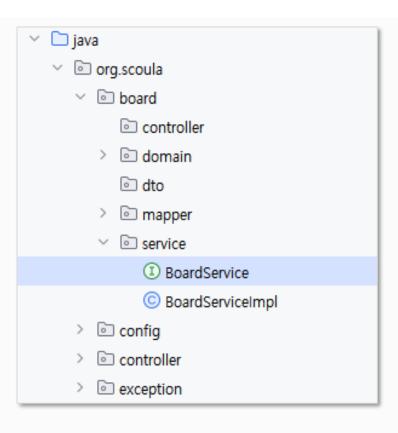
```
// V0 → DTO 변환
public static BoardDTO of(BoardVO vo) {
    return vo == null ? null : BoardDTO.builder()
            .no(vo.getNo())
            .title(vo.getTitle())
            .content(vo.getContent())
            .writer(vo.getWriter())
            .regDate(vo.getRegDate())
            .updateDate(vo.getUpdateDate())
            .build();
// DTO → VO 변환
public BoardV0 toVo() {
    return BoardVO.builder()
            .no(no)
            .title(title)
            .content(content)
            .writer(writer)
            .regDate(regDate)
            .updateDate(updateDate)
            .build();
```

RootConfig.java

```
package org.scoula.config;
import javax.sql.DataSource;
import org.apache.ibatis.session.SqlSessionFactory;
import org.mybatis.spring.SqlSessionFactoryBean;
import org.mybatis.spring.annotation.MapperScan;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import com.zaxxer.hikari.HikariConfig;
import com.zaxxer.hikari.HikariDataSource;
@Configuration
@ComponentScan(basePackages={ "org.scoula.board.service" })
@MapperScan(basePackages = {"org.scoula.board.mapper"})
public class RootConfig {
```

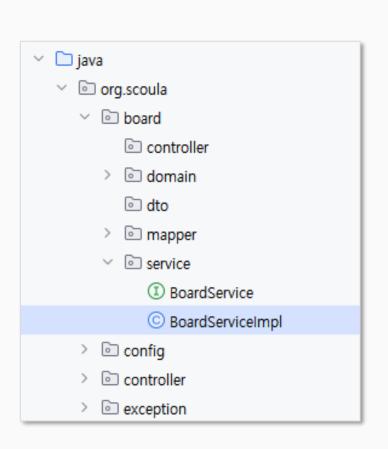
BoardService.java

```
package org.scoula.board.service;
import org.scoula.board.dto.BoarDTO;
import java.util.List;
import java.util.Optional;
public interface BoardService {
    public List<BoardDTO> getList();
    public BoardDTO get(Long no);
    public void create(BoardDTO board);
    public boolean update(BoardDTO board);
    public boolean delete(Long no);
```



1 비즈니스 계층의 설정

```
package org.scoula.board.service;
import java.util.List;
import org.springframework.stereotype.Service;
import org.scoula.board.domain.BoardVO;
import org.scoula.board.mapper.BoardMapper;
import lombok.AllArgsConstructor;
import lombok.extern.log4j.Log4j;
               final 멤버를 인자로 가지는
@Log4j
                     생성자 추가
@Service
@RequiredArgsConstructor
public class BoardServiceImpl implements BoardService {
   final private BoardMapper mapper;
               생성자가 1개인 경우
              생성자 주입으로 초기화
```

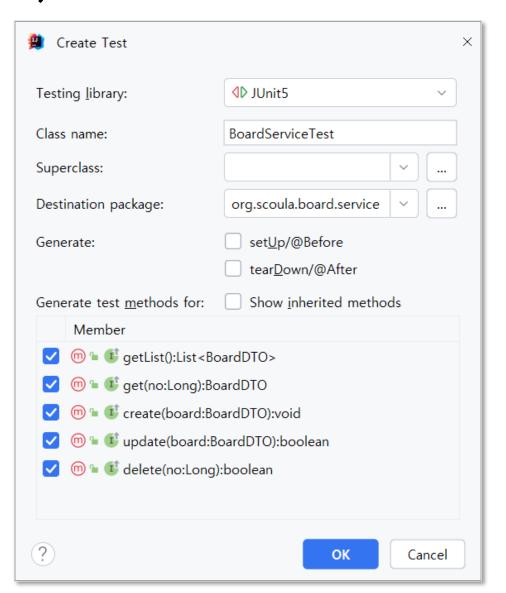


비즈니스 계층의 설정

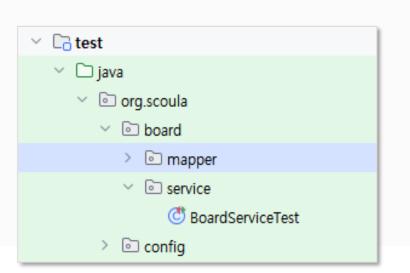
```
@Override
public List<BoardDTO> getList() { ... }
@Override
public BoardDTO get(Long no) { ... }
@Override
public void create(BoardDTO board) { ... }
@Override
public boolean update(BoardDTO board) { ... }
@Override
public boolean delete(Long no) { ... }
```

비즈니스 계층의 구현과 테스트

test ∷ BoardServiceTest.java




```
@ExtendWith(SpringExtension.class)
@ContextConfiguration(classes = {RootConfig.class} )
@Log4j
public class BoardServiceTest {
  @Autowired
  private BoardService service;
```



```
@Override
public List<BoardDTO> getList() {
 log.info("getList....");
     return mapper.getList().stream() // BoardV0의 스트림
            .map(BoardDT0::of) // BoardDT0의 스트림
            .toList();
                      // List<BoardDTO> 변환
```

비즈니스 계층의 구현과 테스트

test :: BoardServiceTest.java

```
@Test
public void getList() {
   for(BoardDTO board: service.getList()) {
     log.info(board);
   }
}
```

INFO: org.scoula.board.service.BoardServiceImpl - getList.........

INFO: jdbc.sqlonly - select * from tbl_board

INFO: org.scoula.board.service.BoardServiceImplTest - BoardDTO(no=1, title=테스트 제목1, content=테스트 내용1, writer=user00, regDate=Tue Jul 02 13:37:07 KST 2024, updateDate=Tue Jul 02 13:37:07 KST 2024)

INFO: org.scoula.board.service.BoardServiceImplTest - BoardDTO(no=2, title=테스트 제목2, content=테스트 내용2, writer=user00, regDate=Tue Jul 02 13:37:07 KST 2024, updateDate=Tue Jul 02 13:37:07 KST 2024)

INFO: org.scoula.board.service.BoardServiceImplTest - BoardDTO(no=3, title=테스트 제목3, content=테스트 내용3, writer=user00, regDate=Tue Jul 02 13:37:07 KST 2024, updateDate=Tue Jul 02 13:37:07 KST 2024)

INFO: org.scoula.board.service.BoardServiceImplTest - BoardDTO(no=4, title=테스트 제목4, content=테스트 내용4, writer=user00, regDate=Tue Jul 02 13:37:07 KST 2024, updateDate=Tue Jul 02 13:37:07 KST 2024)

INFO: org.scoula.board.service.BoardServiceImplTest - BoardDTO(no=5, title=수정된 제목, content=수정된 내용, writer=user00, regDate=Tue Jul 02 13:37:07 KST 2024, updateDate=Tue Jul 02 13:58:14 KST 2024)

INFO : org.scoula.board.service.BoardServiceImplTest - BoardDTO(no=6, title=새로 작성하는 글, content=새로 작성하는 내용, writer=user0, regDate=Tue Jul 02 13:56:51 KST 2024, updateDate=Tue Jul 02 13:56:51 KST 2024)

> Task :test

///////

test :: BoardServiceTest.java

```
@Test
void get() {
    log.info(service.get(1L));
```

```
INFO: org.scoula.board.service.BoardServiceImpl - get.....1
INFO : jdbc.sqlonly - select * from tbl_board where no = 1
```

INFO: org.scoula.board.service.BoardServiceImplTest - BoardDTO(no=1, title=테스트 제목1, content=테스트 내용1, writer=user00, regDate=Tue Jul 02 13:37:07 KST 2024, updateDate=Tue Jul 02 13:37:07 KST 2024)

> Task :test

```
@Override
public void create(BoardDTO board) {
  log.info("create....." + board);
   BoardV0 vo = board.toVo();
   mapper.create(vo);
   board.setNo(vo.getNo());
```

test :: BoardServiceTest.java

```
@Test
public void create() {
  BoardDTO board = new BoardDTO();
  board.setTitle("새로 작성하는 글");
  board.setContent("새로 작성하는 내용");
  board.setWriter("user1");
  service.create(board);
  log.info("생성된 게시물의 번호: " + board.getNo());
```

INFO : org.scoula.board.service.BoardServiceImpl - create.....BoardDTO(no=null, title=새로 작성하는 글, content=새로 작성하는 내용, writer=user1, regDate=null, updateDate=null)

INFO: jdbc.sqlonly - insert into tbl_board (title, content, writer) values ('새로 작성하는 글', '새로 작성하는 내용', 'user1') INFO: org.scoula.board.service.BoardServiceTest - 생성된 게시물의 번호: 10

```
@Override
public boolean update(BoardDTO board) {
  log.info("update....." + board);
  return mapper.update(board.toVo()) == 1;
```



```
@Test
public void update() {
   BoardDTO board = service.get(1L);
   board.setTitle("제목 수정합니다.");
   log.info("update RESULT: " + service.update(board));
```

```
INFO: org.scoula.board.service.BoardServiceImpl - update.....BoardDTO(no=1, title=제목 수정합니다., content=테스트 내용1,
writer=user00, regDate=Tue Jul 02 13:37:07 KST 2024, updateDate=Tue Jul 02 13:37:07 KST 2024)
INFO : jdbc.sqlonly - update tbl_board set title = '제목 수정합니다.', content = '테스트 내용1', writer = 'user00', update_date
= now() where no = 1
```

INFO: org.scoula.board.service.BoardServiceImplTest - update RESULT: true > Task :test

```
@Override
public boolean delete(Long no) {
  log.info("delete...." + no);
  return mapper.delete(no) == 1;
```

2 비즈니스 계층의 구현과 테스트


```
@Test public void delete() {

// 게시물 번호의 존재 여부를 확인하고 테스트할 것 log.info("delete RESULT: " + service.delete(2L));
}
```

```
INFO : org.scoula.board.service.BoardServiceImpl - delete....2
INFO : jdbc.sqlonly - delete from tbl_board where no = 2
INFO : org.scoula.board.service.BoardServiceImplTest - delete RESULT: true > Task :test
```