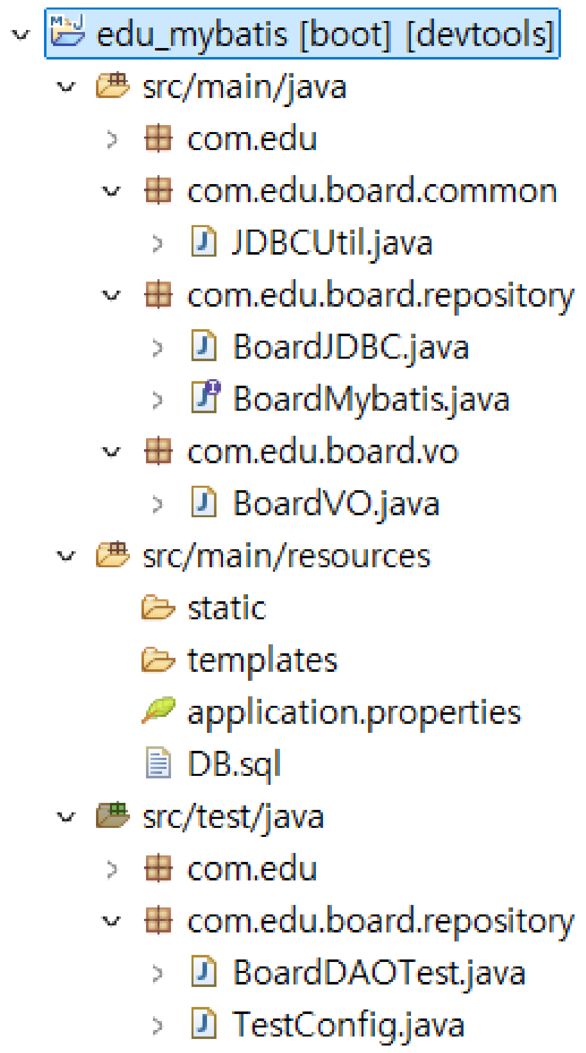
JDBC/MyBatis

Spring Boot

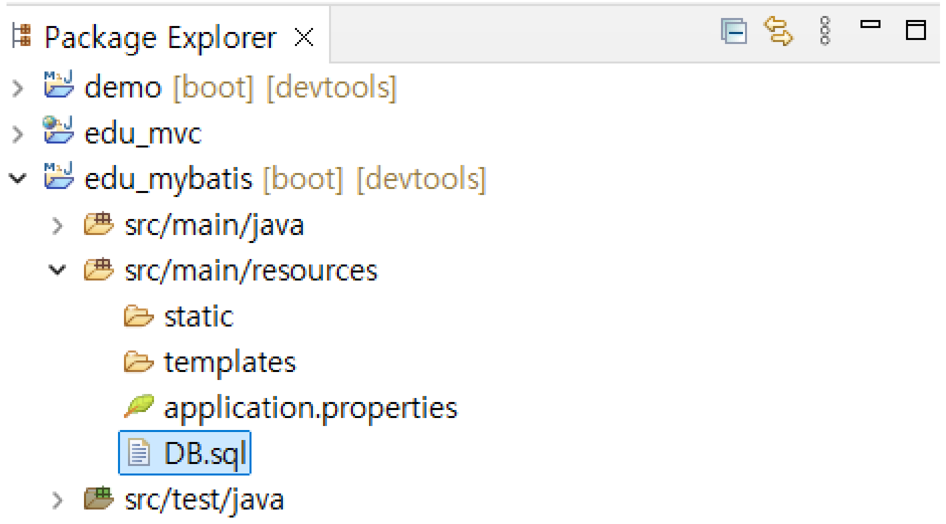
H2/Mybatis

- edu_mybatis.zip



```
edu_mybatis [boot] [devtools]
  src/main/java
    com.edu
    com.edu.board.common
      JDBCUtil.java
    com.edu.board.repository
      BoardJDBC.java
      BoardMybatis.java
    com.edu.board.vo
      BoardVO.java
  src/main/resources
    static
    templates
    application.properties
    DB.sql
  src/test/java
    com.edu
    com.edu.board.repository
      BoardDAOTest.java
      TestConfig.java
```

DB.sql


Package Explorer

```
demo [boot] [devtools]
edu_mvc
edu_mybatis [boot] [devtools]
    src/main/java
    src/main/resources
        static
        templates
        application.properties
        DB.sql
    src/test/java
```

```sql
CREATE TABLE BOARD(
    SEQ LONG NOT NULL AUTO_INCREMENT,
    TITLE VARCHAR2(200),
    WRITER VARCHAR2(20),
    CONTENT VARCHAR2(2000),
    REGDATE TIMESTAMP DEFAULT NOW(),
    CNT NUMBER(5) DEFAULT 0,
    PRIMARY KEY(SEQ)
);

insert into board(title,writer,content) values('질문','홍길동','내용입니다’);

alter table BOARD alter column SEQ  restart with 1 ;
```

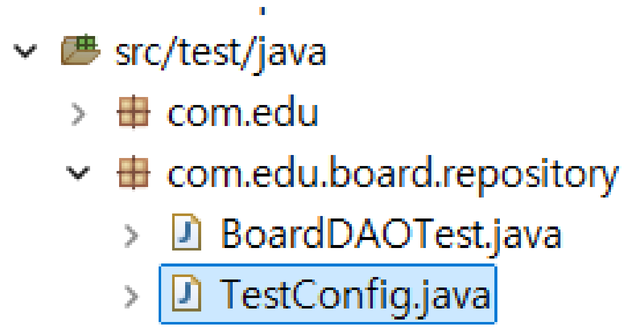
```
package com.edu.board.repository;

~ 생략 ~

@Repository
public class BoardJDBC {

    private Connection conn = null;
    private PreparedStatement stmt = null;
    private ResultSet rs = null;
    private final String BOARD_INSERT = "insert into board(title,writer,content) values(?,?,?)";
    private final String BOARD_UPDATE = "update board set title=?,content=? where seq=?";
    private final String BOARD_DELETE = "delete board where seq=?";
    private final String BOARD_GET = "select * from board where seq=?";
    private final String BOARD_LIST = "select * from board order by seq desc";

    public void insertBoard(BoardVO vo) {
        try {
            conn = JDBCUtil.getConnection();
            stmt = conn.prepareStatement(BOARD_INSERT);
            stmt.setString(1, vo.getTitle());
            stmt.setString(2, vo.getWriter());
            stmt.setString(3, vo.getContent());
            stmt.executeUpdate();
        } catch (Exception e) {
                e.printStackTrace();
        } finally {
                JDBCUtil.close(stmt, conn);
        }
    }
```

```java
public void updateBoard(BoardVO vo) {
    try {
        conn = JDBCUtil.getConnection();
        stmt = conn.prepareStatement(BOARD_UPDATE);
        stmt.setString(1, vo.getTitle());
        stmt.setString(2, vo.getContent());
        stmt.setInt(3, vo.getSeq());
        stmt.executeUpdate();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        JDBCUtil.close(stmt, conn);
    }
}

public void deleteBoard(BoardVO vo) {
    try {
        conn = JDBCUtil.getConnection();
        stmt = conn.prepareStatement(BOARD_DELETE);
        stmt.setInt(1, vo.getSeq());
        stmt.executeUpdate();
    }catch (Exception e) {e.printStackTrace();
    }finally {JDBCUtil.close(stmt, conn); }
}
```

```
public BoardVO getBoard(BoardVO vo) {
    BoardVO board = null;
    try {
        conn = JDBCUtil.getConnection();
        stmt = conn.prepareStatement(BOARD_GET);
        stmt.setInt(1, vo.getSeq());
        rs = stmt.executeQuery();
        if (rs.next()) {
            board = new BoardVO();
            board.setSeq(rs.getInt("SEQ"));
            board.setTitle(rs.getString("TITLE"));
            board.setWriter(rs.getString("WRITER"));
            board.setContent(rs.getString("CONTENT"));
            board.setRegDate(rs.getTimestamp("REGDATE"));
            board.setCnt(rs.getInt("CNT"));
        }
    }catch (Exception e) {e.printStackTrace();
    }finally {JDBCUtil.close(stmt, conn);}
     return board;
}
```

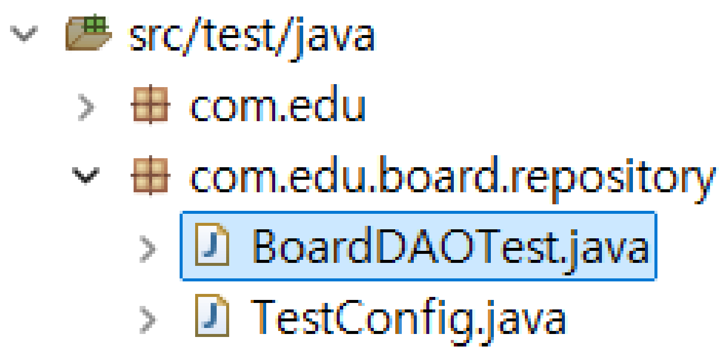
```
public List<BoardVO> getBoardList(BoardVO vo) {
    List<BoardVO> boardList = new ArrayList<BoardVO>();
    try {
        conn = JDBCUtil.getConnection();
        stmt = conn.prepareStatement(BOARD_LIST);
        rs = stmt.executeQuery();
        while (rs.next()) {
            BoardVO board = new BoardVO();
            board.setSeq(rs.getInt("SEQ"));
            board.setTitle(rs.getString("TITLE"));
            board.setWriter(rs.getString("WRITER"));
            board.setContent(rs.getString("CONTENT"));
            board.setRegDate(rs.getTimestamp("REGDATE"));
            board.setCnt(rs.getInt("CNT"));
            boardList.add(board);
        }
    }catch (Exception e) {        e.printStackTrace();
    } finally {JDBCUtil.close(stmt, conn);}
    return boardList;
    }
}
```

DB Test



```
package com.edu.board.repository;

import org.mybatis.spring.annotation.MapperScan;
import org.springframework.boot.test.context.TestConfiguration;
import org.springframework.context.annotation.Bean;

@TestConfiguration
@MapperScan(basePackages = "com.edu.board.repository")
// BoardMybatis 인터페이스가 위치한 패키지를 스캔
public class TestConfig {

    @Bean
    public BoardJDBC boardJDBC() {
        return new BoardJDBC();
    }
}
```

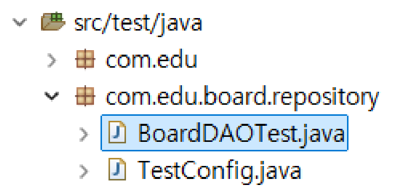
DB Test



```
src/test/java
  com.edu
  com.edu.board.repository
    BoardDAOTest.java
    TestConfig.java
```

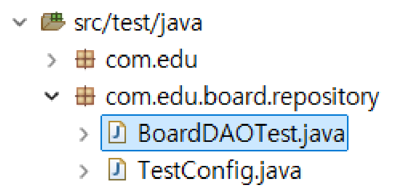
```java
package com.edu.board.repository;

import static org.assertj.core.api.Assertions.assert
import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.DisplayName;
import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.extension.ExtendWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.test.context.ContextConfiguration;
import org.springframework.test.context.junit.jupiter.SpringExtension;

import com.edu.board.vo.BoardVO;

@SpringBootTest
@ContextConfiguration(classes = TestConfig.class)
```

DB Test



```
public class BoardDAOTest {

    @Autowired
    BoardJDBC boardDAO;

    @Test
    @DisplayName("게시글 추출이 정상 동작한다")
    public void getBoardTest() {
        BoardVO vo = BoardVO.builder().seq(1).build();
        BoardVO result = boardDAO.getBoard(vo);
        assertThat(result.getSeq()).isEqualTo(1);
    }

    @Test
    @DisplayName("게시글 등록이 정상 동작한다")
    public void insertBoardTest() {
        BoardVO vo = BoardVO.builder()
                        .title("인사").writer("이순신").content("반갑습니다").build();
        boardDAO.insertBoard(vo);
        BoardVO saveBoard = boardDAO.getBoard(BoardVO.builder().seq(2).build());
        assertThat(saveBoard).isNotNull();
    }
```

DB Test


```
src/test/java
  com.edu
  com.edu.board.repository
    BoardDAOTest.java
    TestConfig.java
```

```
@Test
@DisplayName("게시글 수정이 정상 동작한다")
public void updateBoardTest() {
    BoardVO vo = BoardVO.builder().seq(2).title("hello").content("have a good day").build();
    boardDAO.updateBoard(vo);
    BoardVO updateBoard = boardDAO.getBoard(vo);
    assertEquals(vo.getTitle(), updateBoard.getTitle());
    assertEquals(vo.getContent(), updateBoard.getContent());
}

@Test
@DisplayName("게시글 삭제가 정상 동작한다")
public void deleteBoardTest() {
    BoardVO vo = BoardVO.builder().seq(6).build();
    boardDAO.deleteBoard(vo);
    BoardVO deleteBoard = boardDAO.getBoard(vo);
    assertThat(deleteBoard).isNull();
}

@Test
@DisplayName("게시글 전체 조회가 정상 동작한다")
public void getBoardListTest() {
    assertThat(boardDAO.getBoardList(new BoardVO())).hasSize(1);
}
}
```

pom.xml

```xml
<dependency>
   <groupId>org.mybatis.spring.boot</groupId>
   <artifactId>mybatis-spring-boot-starter</artifactId>
   <version>3.0.3</version>
</dependency>
```

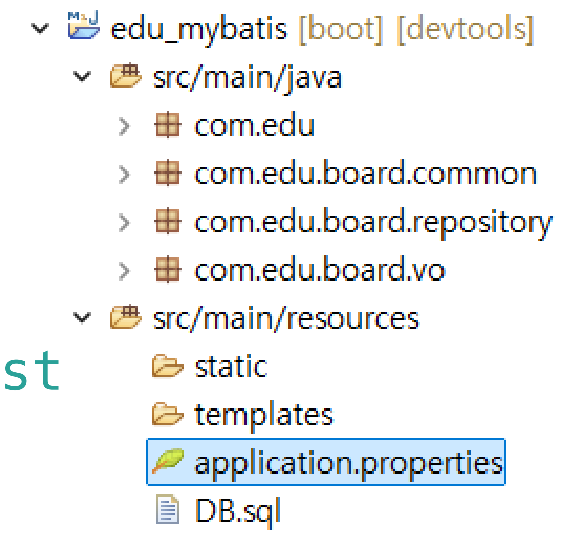
application.properties

- Spring Boot 애플리케이션의 설정을 관리하는 데 사용되는 파일이다. 이 파일을 통해 애플리케이션의 다양한 설정을 외부화할 수 있으며, 환경별로 다른 설정을 적용할 수 있다.

```
# 서버 설정
server.port=8080
server.servlet.context-path=/api

# 데이터베이스 설정
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=password

# JPA 설정
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true

# 로깅 설정
logging.level.org.springframework=INFO
logging.level.com.example=DEBUG
```

application.properties

```
spring.application.name=edu_mybatis


#H2
spring.datasource.url=jdbc:h2:tcp://localhost/~/test
spring.datasource.username=sa
spring.datasource.password=
spring.datasource.driver-class-name=org.h2.Driver
```



```
edu_mybatis [boot] [devtools]
  src/main/java
    com.edu
    com.edu.board.common
    com.edu.board.repository
    com.edu.board.vo
  src/main/resources
    static
    templates
    application.properties
    DB.sql
```

BoardMybatis

```
@Mapper
public interface BoardMybatis {
    @Insert("INSERT INTO board (title, writer, content) VALUES(#{title},#{writer},#{content})")
    public void insertBoard(BoardVO vo);

    @Update("UPDATE board SET title=#{title}, content=#{content} WHERE seq = #{seq}")
    public void updateBoard(BoardVO vo);

    @Delete("DELETE board WHERE seq = #{seq}")
    public void deleteBoard(BoardVO vo);

    @Select("SELECT * FROM board WHERE seq = #{seq}")
    public BoardVO getBoard(BoardVO vo);

    @Select("SELECT * FROM board order by seq desc")
    public List<BoardVO>  getBoardList(BoardVO vo);
}
```

Mybatis Test

```
public class BoardDAOTest {

    @Autowired
    BoardMybatis boardDAO;

    @Test
    @DisplayName("게시글 추출이 정상 동작한다")
    public void getBoardTest() {
        BoardVO vo = BoardVO.builder().seq(1).build();
        BoardVO result = boardDAO.getBoard(vo);
        assertThat(result.getSeq()).isEqualTo(1);
    }

    ~   생략   ~
```