



박경태  
[comsi.java@gmail.com](mailto:comsi.java@gmail.com)

# 고급 자바 프로그래밍 : STS를 이용한 Spring 프로그래밍

# Mybatis 프레임워크 시작하기

# BoardDAO 비교하기 – JDBC vs. MyBatis

```
public class BoardDAO {  
    // JDBC 관련 변수  
    private Connection conn = null;  
    private PreparedStatement pstmt = null;  
    private ResultSet rset = null;  
  
    private final String BOARD_LIST = "select * from board order by seq desc";  
  
    // 글 목록 조회  
    public List<BoardVO> getBoardList(BoardVO vo) {  
        System.out.println("==> JDBC로 getBoardList() 기능 처리");  
        List<BoardVO> boardList = new ArrayList<BoardVO>();  
        try {  
            conn = JDBCUtil.getConnection();  
            pstmt = conn.prepareStatement(BOARD_LIST);  
            rset = pstmt.executeQuery();  
            while (rset.next()) {  
                BoardVO board = new BoardVO();  
                board.setSeq(rset.getInt("SEQ"));  
                board.setTitle(rset.getString("TITLE"));  
                board.setWriter(rset.getString("WRITER"));  
                board.setContent(rset.getString("CONTENT"));  
                board.setRegDate(rset.getDate("REGDATE"));  
                board.setCnt(rset.getInt("CNT"));  
                boardList.add(board);  
            }  
        } catch (Exception e) {  
            e.printStackTrace();  
        } finally {  
            JDBCUtil.close(rset, pstmt, conn);  
        }  
        return boardList;  
    }  
}
```

JDBC

```
public class BoardDAO {  
    public List<BoardVO> getBoardList(BoardVO vo) {  
        SqlSession mybatis = SqlSessionFactoryBean.getSqlSessionInstance();  
        return mybatis.selectList("BoardDAO.getBoardList", vo);  
    }  
}
```

# MyBatis

## 1.1 Mybatis 프레임워크 2가지 특징

- 한 줄의 자바 코드로 DB 연동을 처리한다는 것

- XML 파일에 저장된 SQL 명령어를 대신 실행하고 실행 결과를 VO 자바 객체에 자동으로 매핑시켜 줌. → 데이터맵퍼(Data Mapper)라고 함

- SQL 명령어를 자바 코드에서 분리하여 XML파일에 따로 관리한다는 것

- 만약, SQL 명령어가 DAO 같은 자바 클래스에 저장되면 SQL 명령어만 수정하는 상황에서도 자바 클래스를 다시 컴파일해야 한다.
- SQL 명령어들을 한 곳에 모아서 관리하기도 쉽지 않다.
- 결국, SQL 명령어에 대한 통합 관리를 위해서라도 자바 소스에서 SQL을 분리하는 게 매우 중요함.

# XML파일 분리의 예

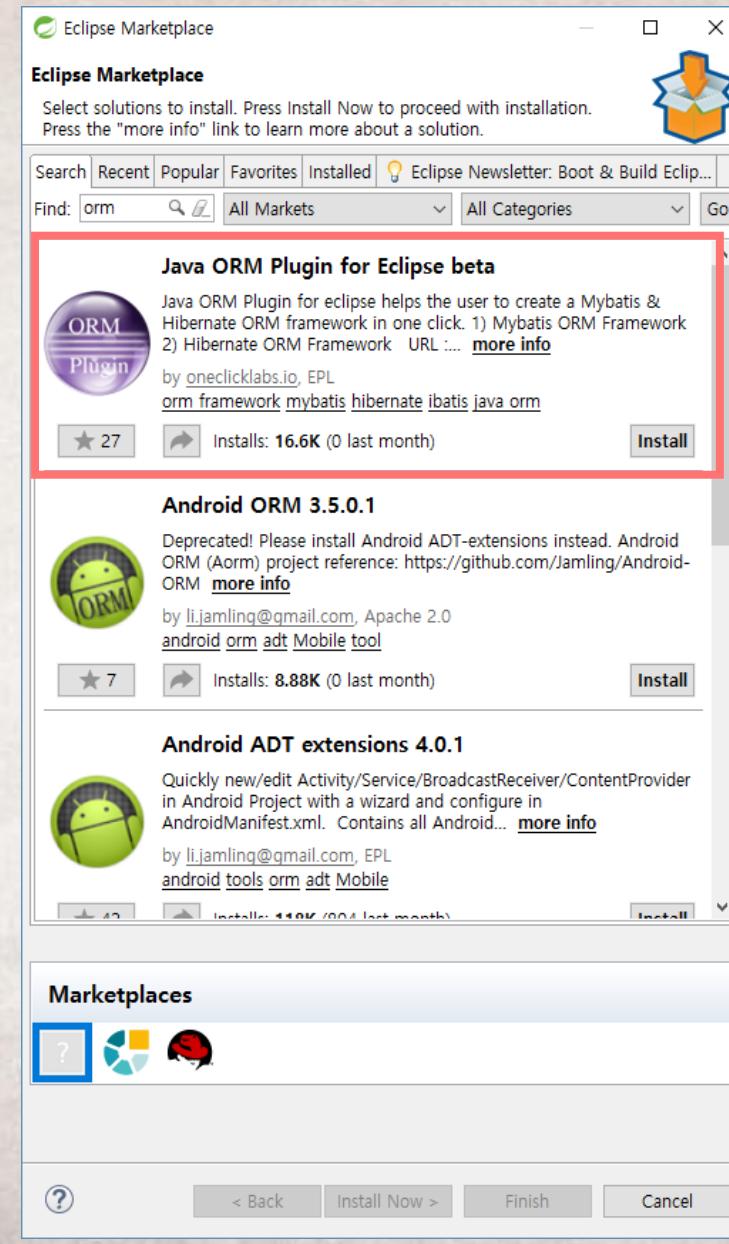
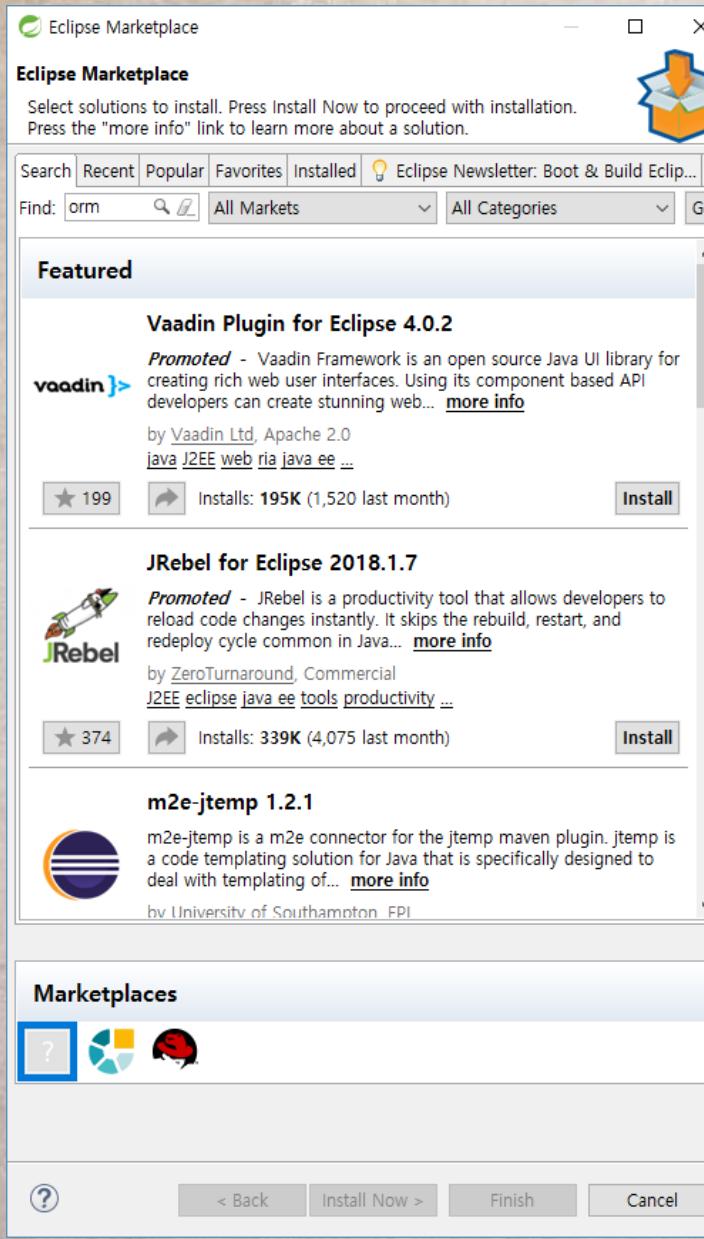
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="kr.ac.inje.comsi.board.BoardVO">

  <select id="getBoardList" resultType="kr.ac.inje.comsi.board.BoardVO" >
    select * from board
  </select>

</mapper>
```

# 1.2 ORM 플러그인 설치 → 제대로 다운로드가 안 됨ㅠㅠ

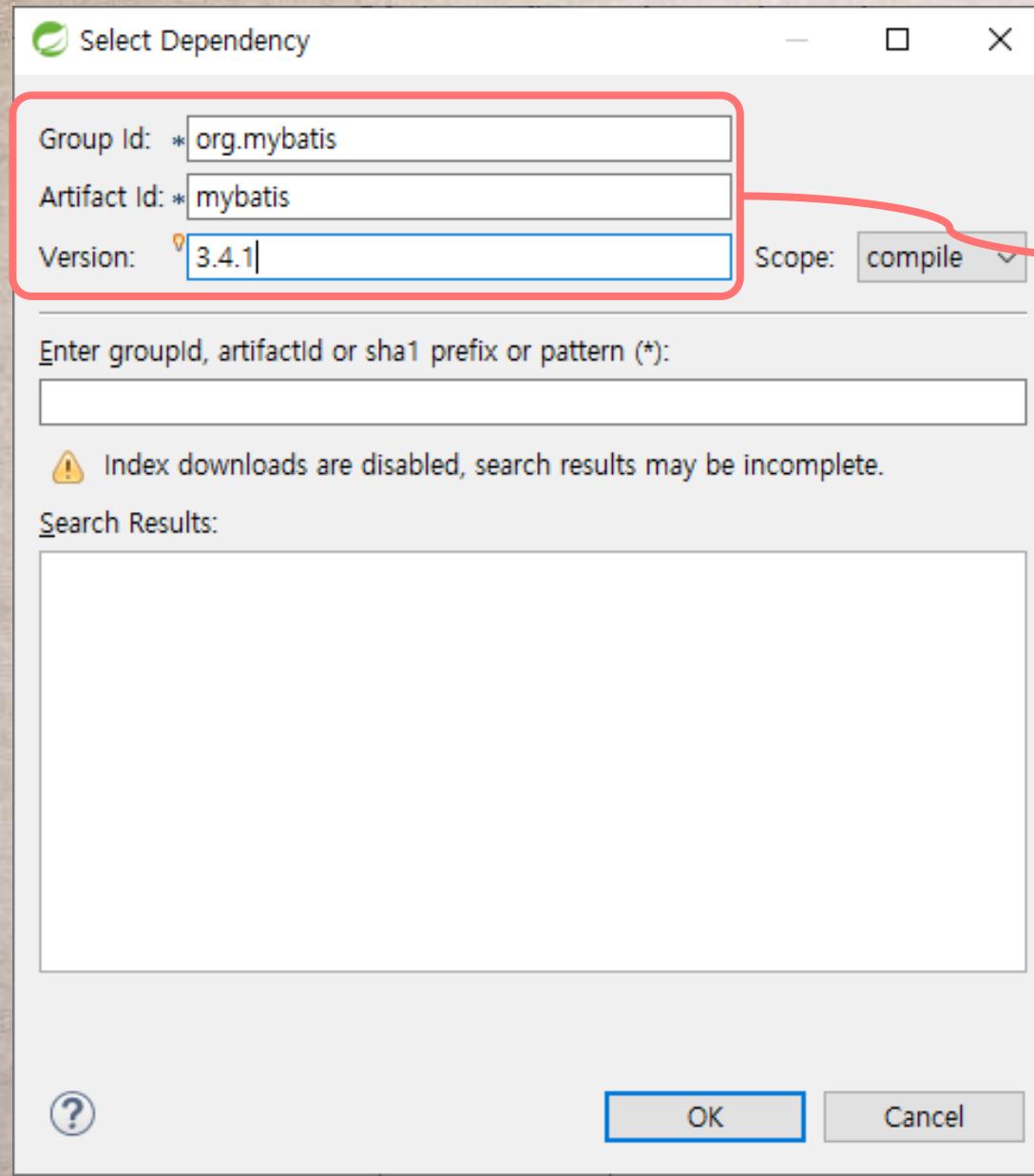


# 1.3 라이브러리 추가 – Maven Dependency(pom.xml 파일)

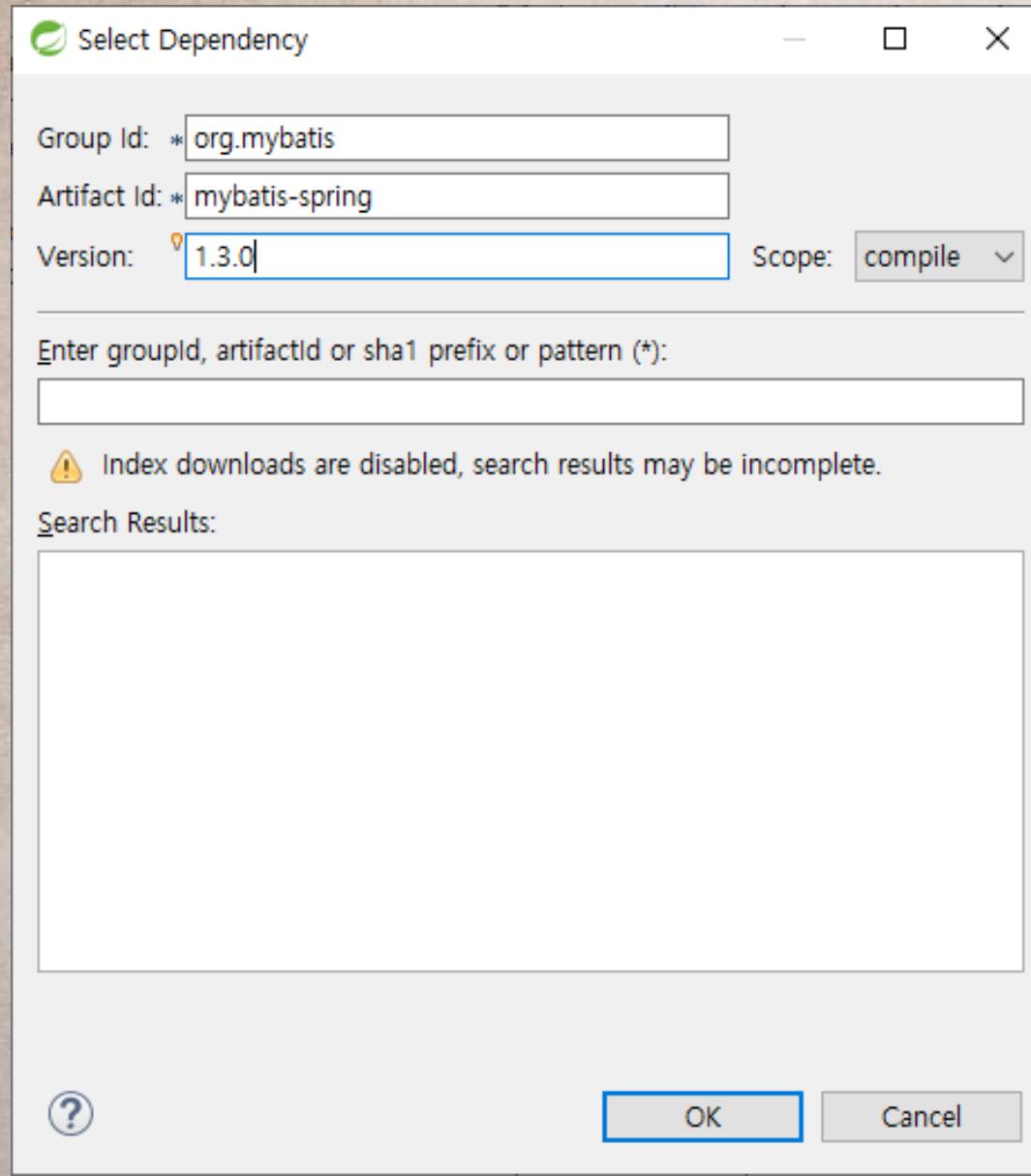
The screenshot shows the Eclipse IDE interface with the Maven Dependencies view open. The title bar includes tabs for 'servlet-context.xml', 'root-context.xml', and 'BroadWeb/pom.xml'. The main area is titled 'Dependencies' and contains a list of dependencies:

- spring-context : \${org.springframework-version}
- spring-webmvc : \${org.springframework-version}
- aspectjweaver : \${org.aspectj-version}
- slf4j-api : \${org.slf4j-version}
- jcl-over-slf4j : \${org.slf4j-version} [runtime]
- slf4j-log4j12 : \${org.slf4j-version} [runtime]
- log4j : 1.2.15 [runtime]
- javax.inject : 1
- servlet-api : 2.5 [provided]
- jsp-api : 2.1 [provided]
- jstl : 1.2
- junit : 4.7 [test]
- h2 : 1.4.197
- commons-dbcop2 : 2.1.1
- spring-jdbc : 4.3.15.RELEASE

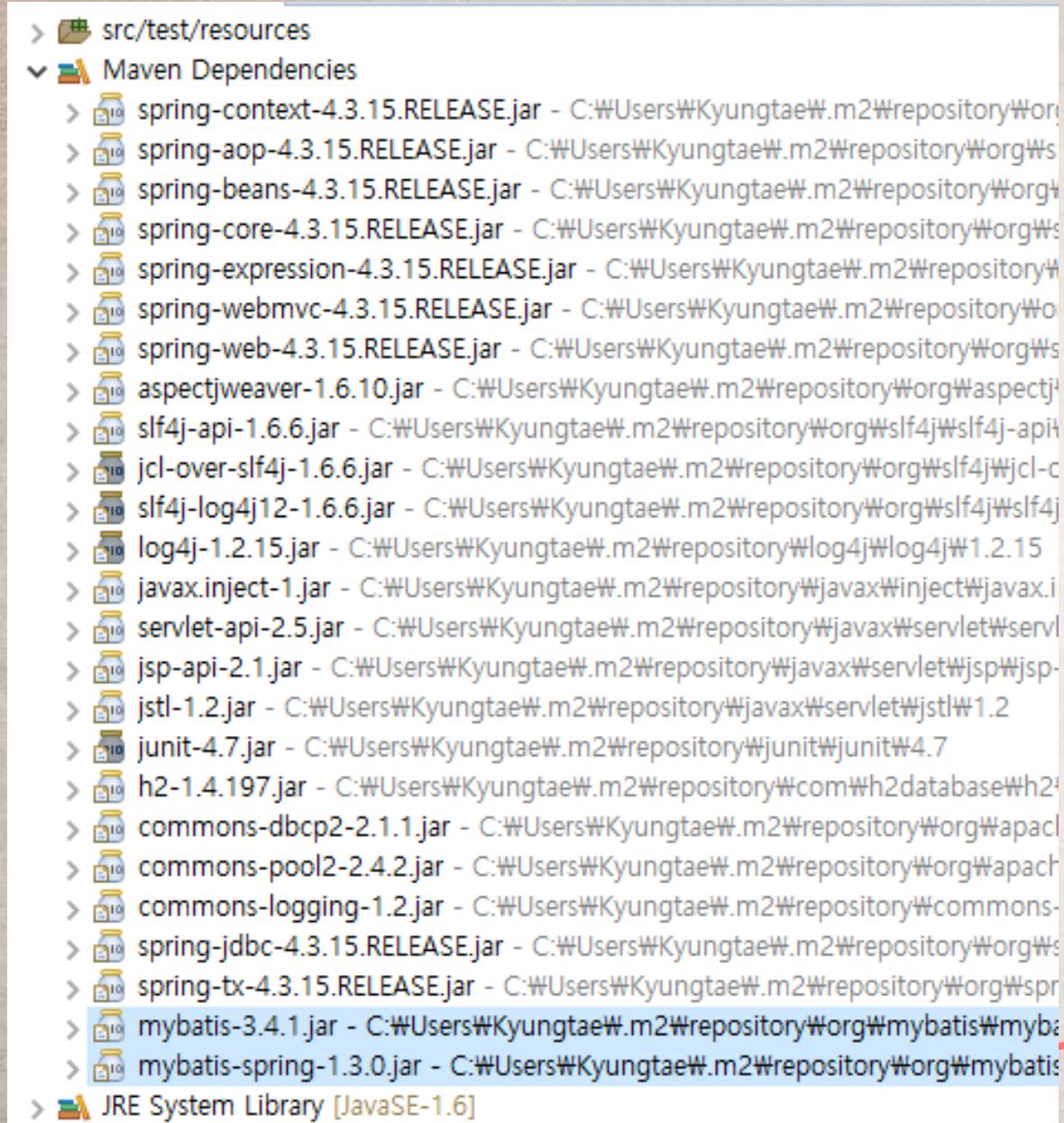
Below the list is a central toolbar with several icons. One icon, labeled 'Add...', is highlighted with a red box. To the right of the toolbar is a 'Dependency Management' section with its own toolbar and three buttons: 'Add...', 'Remove', and 'Properties...'. At the bottom of the view, there is a note: 'To manage your transitive dependency exclusions, please use the [Dependency Hierarchy](#) page.' The navigation bar at the bottom includes tabs for 'Overview', 'Dependencies', 'Dependency Hierarchy', 'Effective POM', and 'pom.xml'.



Group Id: org.mybatis  
Artifact Id: mybatis  
Version: 3.4.1



Group Id: org.mybatis  
Artifact Id: mybatis-spring  
Version: 1.3.0



mybatis-3.4.1.jar  
mybatis-spring-1.3.0.jar

servlet-context.xml \*root-context.xml

### Namespaces

Configure Namespaces  
Select XSD namespaces to use in the configuration file

- AOP - http://www.springframework.org/schema/aop
- beans - http://www.springframework.org/schema/beans
- C - http://www.springframework.org/schema/c
- cache - http://www.springframework.org/schema/cache
- context - http://www.springframework.org/schema/context
- JDBC - http://www.springframework.org/schema/jdbc
- JEE - http://www.springframework.org/schema/jee
- lang - http://www.springframework.org/schema/lang
- MVC - http://www.springframework.org/schema/mvc
- mybatis-spring - http://mybatis.org/schema/mybatis-spring
- P - http://www.springframework.org/schema/p
- task - http://www.springframework.org/schema/task
- tx - http://www.springframework.org/schema/tx
- util - http://www.springframework.org/schema/util

Source Namespaces Overview beans context jdbc mvc Beans Graph

# root-context.xml 파일

- Namespace 항목 추가하기
  - jdbc
  - mybatis-spring

# 1.4 root-context.xml 수정-DataSource 추석 추가

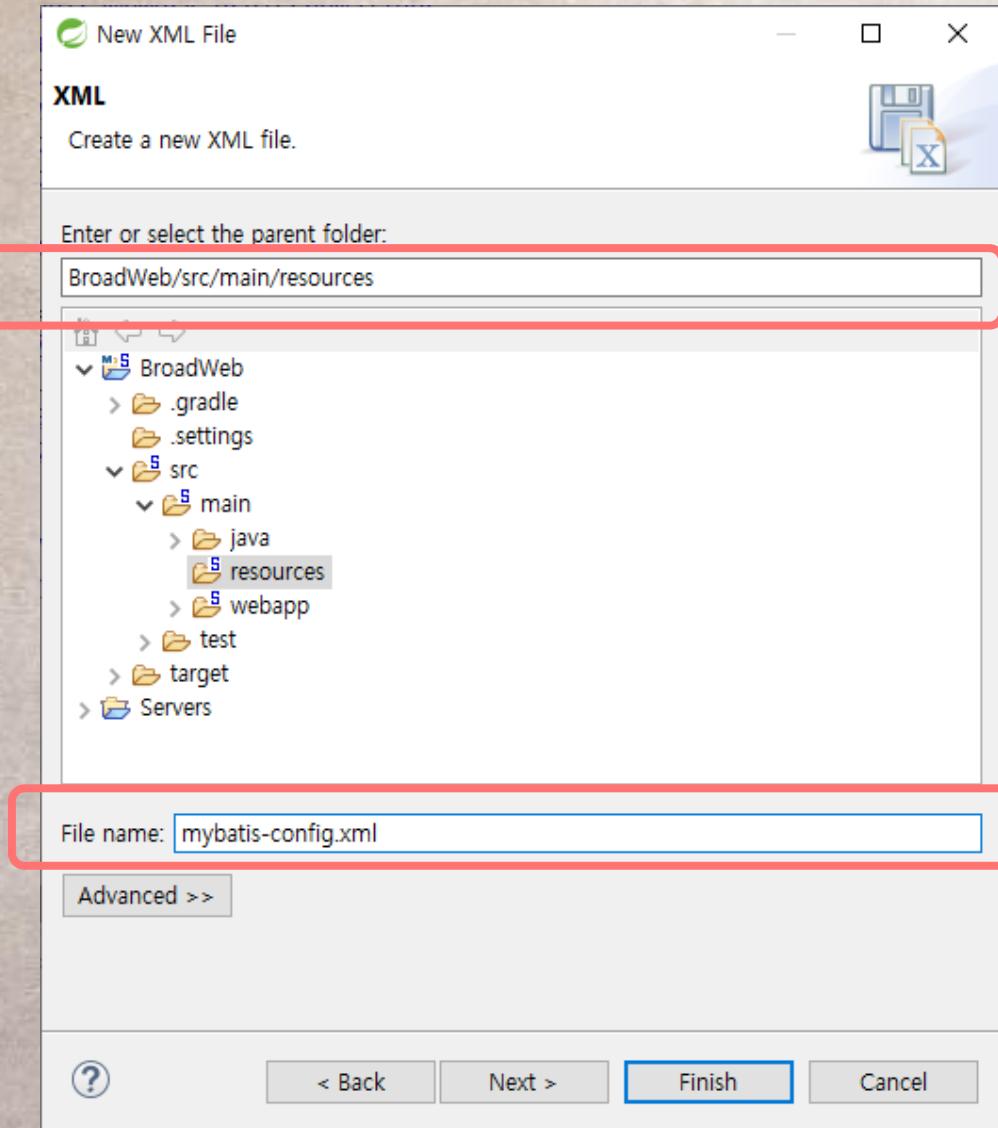
```
servlet-context.xml root-context.xml applicationContext.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2<beans xmlns="http://www.springframework.org/schema/beans"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:context="http://www.springframework.org/schema/context"
5   xmlns:mvc="http://www.springframework.org/schema/mvc"
6   xmlns:mybatis-spring="http://mybatis.org/schema/mybatis-spring"
7   xmlns:jdbc="http://www.springframework.org/schema/jdbc"
8   xsi:schemaLocation="http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-4.3.xsd
9     http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-4.3.xsd
10    http://mybatis.org/schema/mybatis-spring http://mybatis.org/schema/mybatis-spring-1.2.xsd
11    http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.3.xsd
12    http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.3.xsd">
13
14 <context:component-scan base-package="kr.ac.inje.comsi.board"></context:component-scan>
15
16<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
17   <property name="driverClassName" value="org.h2.Driver"></property>
18   <property name="url" value="jdbc:h2:D:/workspace/h2db/db"></property>
19   <property name="username" value="sa"></property>
20   <property name="password" value=""></property>
21 </bean>
22
23</beans>
24
25
```



주석 처리하기(삭제)

# 1.5 SQL Mapping 프레임워크

- MyBatis는 SQL Mapping 프레임워크로 별도의 설정파일을 가짐

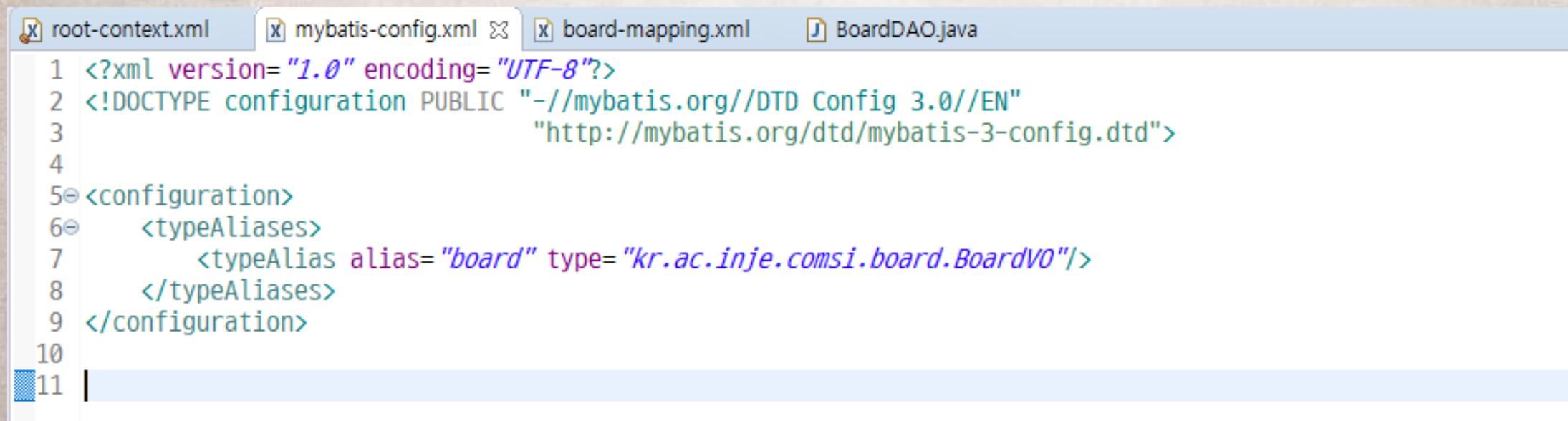


BoardWeb/src/main/resources

mybatis-config.xml

# mybatis-config.xml 파일 수정 – typeAlias 설정

- <typeAliases> 엘리먼트는 <typeAlias>를 여러 개 가질 수 있으며,
- <typeAlias>를 사용하여 특정 클래스의 별칭(alias)을 선언할 수 있다.
- “kr.ac.inje.comsi.board.BoardVO”클래스를 “board” 별칭으로 선언
- \*.mapping.xml 파일에서 접근 가능



```
root-context.xml mybatis-config.xml board-mapping.xml BoardDAO.java
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
   "http://mybatis.org/dtd/mybatis-3-config.dtd">
3
4
5<configuration>
6  <typeAliases>
7    <typeAlias alias="board" type="kr.ac.inje.comsi.board.BoardVO"/>
8  </typeAliases>
9</configuration>
10
11 |
```

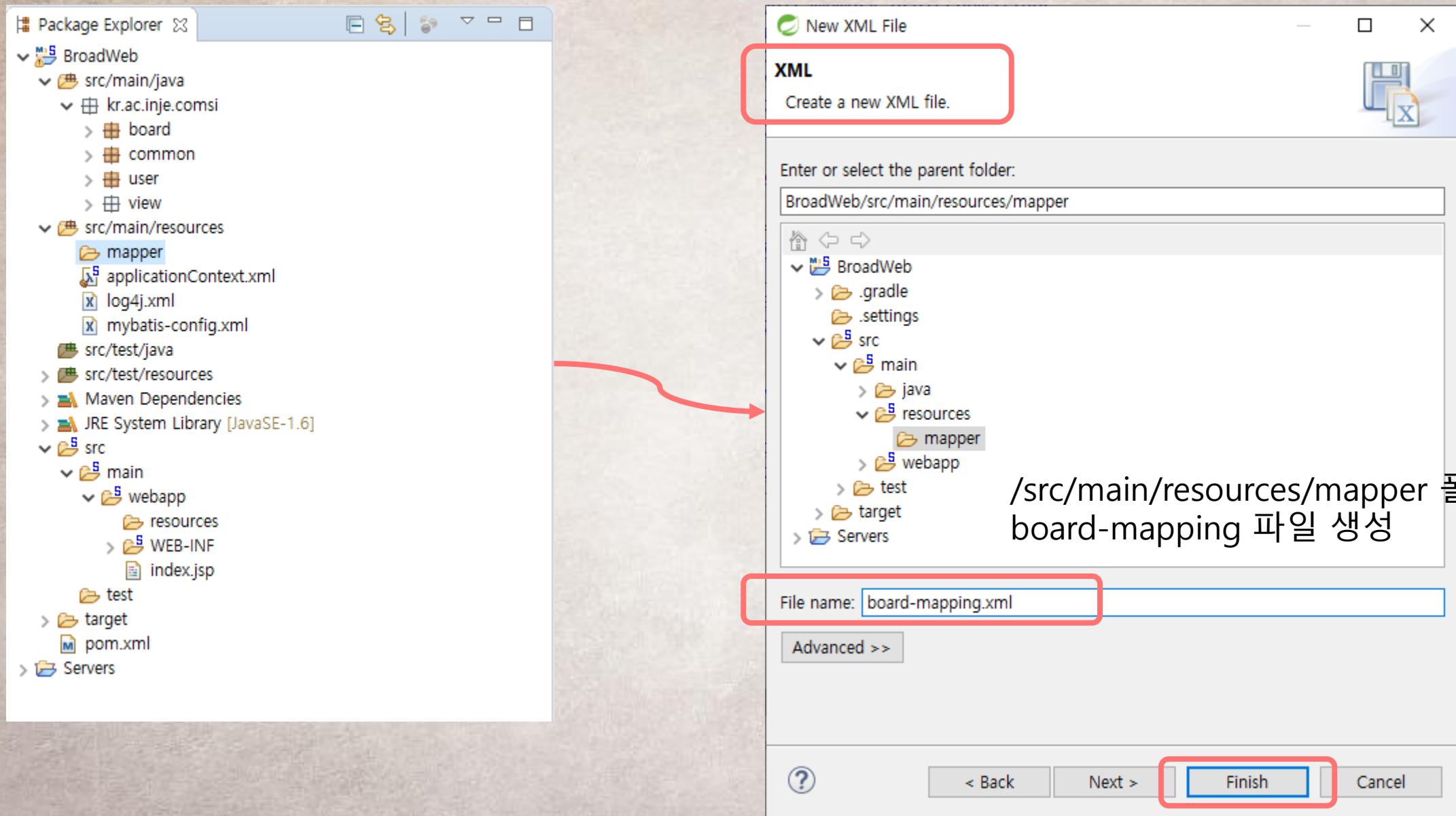
# mybatis-config.xml 내용 파일 추가

```
root-context.xml mybatis-config.xml board-mapping.xml BoardDAO.java SqlSessionFactoryBean.java BoardServiceClient.java
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
3           "http://mybatis.org/dtd/mybatis-3-config.dtd">
4
5<configuration>
6   <!-- <properties resource="db.properties"/> -->
7   <typeAliases>
8     <typeAlias alias="board" type="kr.ac.inje.comsi.board.BoardVO"/>
9   </typeAliases>
10  <environments default="development">
11    <environment id="development">
12      <transactionManager type="JDBC"/>
13      <dataSource type="POOLED">
14        <property name="driver" value="org.h2.Driver"/></property>
15        <property name="url" value="jdbc:h2:D:/workspace/h2db/db"/></property>
16        <property name="username" value="sa"/></property>
17        <property name="password" value=""></property>
18      </dataSource>
19    </environment>
20  </environments>
21
22  <mappers>
23    <mapper resource="mapper/board-mapping.xml"/> sql mapper 파일 위치 설정
24  </mappers>
25</configuration>
```

SQL mapper 파일에서 호출 사용

→ <http://www.mybatis.org/mybatis-3/ko/getting-started.html>

# 1.6 SQL Mapper 파일 설정 – Board Table



/src/main/resources/mapper 폴더 안에  
board-mapping 파일 생성

# board-mapping.xml의 DTD 및 <mapper>root 엘리먼트 설정

The screenshot shows a code editor window with three tabs at the top: 'root-context.xml', 'mybatis-config.xml', and 'board-mapping.xml'. The 'board-mapping.xml' tab is active, displaying the following XML code:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
3           "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
4<mapper>
5
6</mapper>
7
```

# board-mapping.xml 내용 파일 추가

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
3           "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
4<mapper namespace="BoardDAO">
5  <insert id="insertBoard">
6    insert into board(seq, title, writer, content)
7      values((select nvl(max(seq),0)+1 from board),#{title}, #{writer},#{content})
8  </insert>
9
10 <select id="getBoard" resultType="board">
11   select * from board where seq=#{seq}
12 </select>
13
14 <select id="getBoardList" resultType="board">
15   select * from board order by seq desc
16 </select>
17
18 <delete id="deleteBoard">
19   delete board where seq=#{seq}
20 </delete>
21
22 <update id="updateBoard">
23   update board set title=#{title}, content=#{content} where seq=#{seq}
24 </update>
25 </mapper>
26
```

# 1.7 SqlSession 객체 생성하기-SqlSessionFactoryBean 클래스

- Mybatis에서 DAO를 구현할려면 SqlSession 객체가 필요
- SqlSession 객체를 얻으려면 SqlSessionFactory객체가 필요

```
root-context.xml mybatis-config.xml board-mapping.xml BoardDAO.java SqlSessionFactoryBean.java BoardServiceClient.java
1 package kr.ac.inje.comsi.board;
2
3 import java.io.Reader;
4
5 import org.apache.ibatis.io.Resources;
6 import org.apache.ibatis.session.SqlSession;
7 import org.apache.ibatis.session.SqlSessionFactory;
8 import org.apache.ibatis.session.SqlSessionFactoryBuilder;
9
10 public class SqlSessionFactoryBean {
11     private static SqlSessionFactory sessionFactory = null;
12
13     static {
14         try {
15             if (sessionFactory == null) {
16                 Reader reader = Resources.getResourceAsReader("mybatis-config.xml");
17                 sessionFactory = new SqlSessionFactoryBuilder().build(reader);
18             }
19         } catch (Exception e) {
20             e.printStackTrace();
21         }
22     }
23
24     public static SqlSession getSqlSessionInstance() {
25         return sessionFactory.openSession(); → SqlSession 객체를 넘긴다.
26     }
27 }
```

# BoardDAO 수정

```
root-context.xml mybatis-config.xml board-mapping.xml BoardDAO.java SqlSessionFactoryBean.java BoardServiceClient.java
1 package kr.ac.inje.comsi.board.impl;
2
3 import java.util.List;
4
5 import org.apache.ibatis.session.SqlSession;
6
7 import kr.ac.inje.comsi.board.BoardVO;
8 import kr.ac.inje.comsi.board.SqlSessionFactoryBean;
9
10 public class BoardDAO {
11
12     private SqlSession mybatis;
13
14     public BoardDAO(){
15         mybatis = SqlSessionFactoryBean.getSqlSessionInstance();
16     }
17
18     public void insertBoard(BoardVO vo) {
19         mybatis.insert("BoardDAO.insertBoard", vo);
20         mybatis.commit();
21     }
22
23
24     public void updateBoard(BoardVO vo) {
25         mybatis.update("BoardDAO.updateBoard", vo);
26         mybatis.commit();
27     }
28 }
```

# 이어서...

```
29✉ public void deleteBoard(BoardVO vo) {  
30    mybatis.delete("BoardDAO.deleteBoard", vo);  
31    mybatis.commit();  
32}  
33  
34  
35✉ public BoardVO getBoard(BoardVO vo) {  
36    return (BoardVO) mybatis.selectOne("BoardDAO.getBoard", vo);  
37}  
38  
39✉ public List<BoardVO> getBoardList(BoardVO vo) {  
40    return mybatis.selectList("BoardDAO.getBoardList", vo);  
41}  
42  
43}  
44
```

# 테스트 클라이언트-BoardServiceClient 클래스 수정

The screenshot shows an IDE interface with the following details:

- Project Structure:** The tabs at the top include root-context.xml, mybatis-config.xml, board-mapping.xml, BoardDAO.java, SqlSessionFactoryBean.java, and BoardServiceClient.java (which is currently selected).
- Java Code:** The code in BoardServiceClient.java is as follows:

```
1 package kr.ac.inje.comsi.board;
2
3 import java.util.List;
4
5 import kr.ac.inje.comsi.board.impl.BoardDAO;
6
7 public class BoardServiceClient {
8
9     public static void main(String[] args) {
10         BoardDAO boardDAO = new BoardDAO();
11
12         BoardVO vo = new BoardVO();
13         vo.setTitle("myBatis 제목");
14         vo.setWriter("김길동");
15         vo.setContent("myBatis 내용입니다....");
16         //boardDAO.insertBoard(vo);
17
18         List<BoardVO> boardList = boardDAO.getBoardList(vo);
19         for (BoardVO board:boardList) {
20             System.out.println("-->" + board.toString());
21         }
22     }
23 }
24 }
```

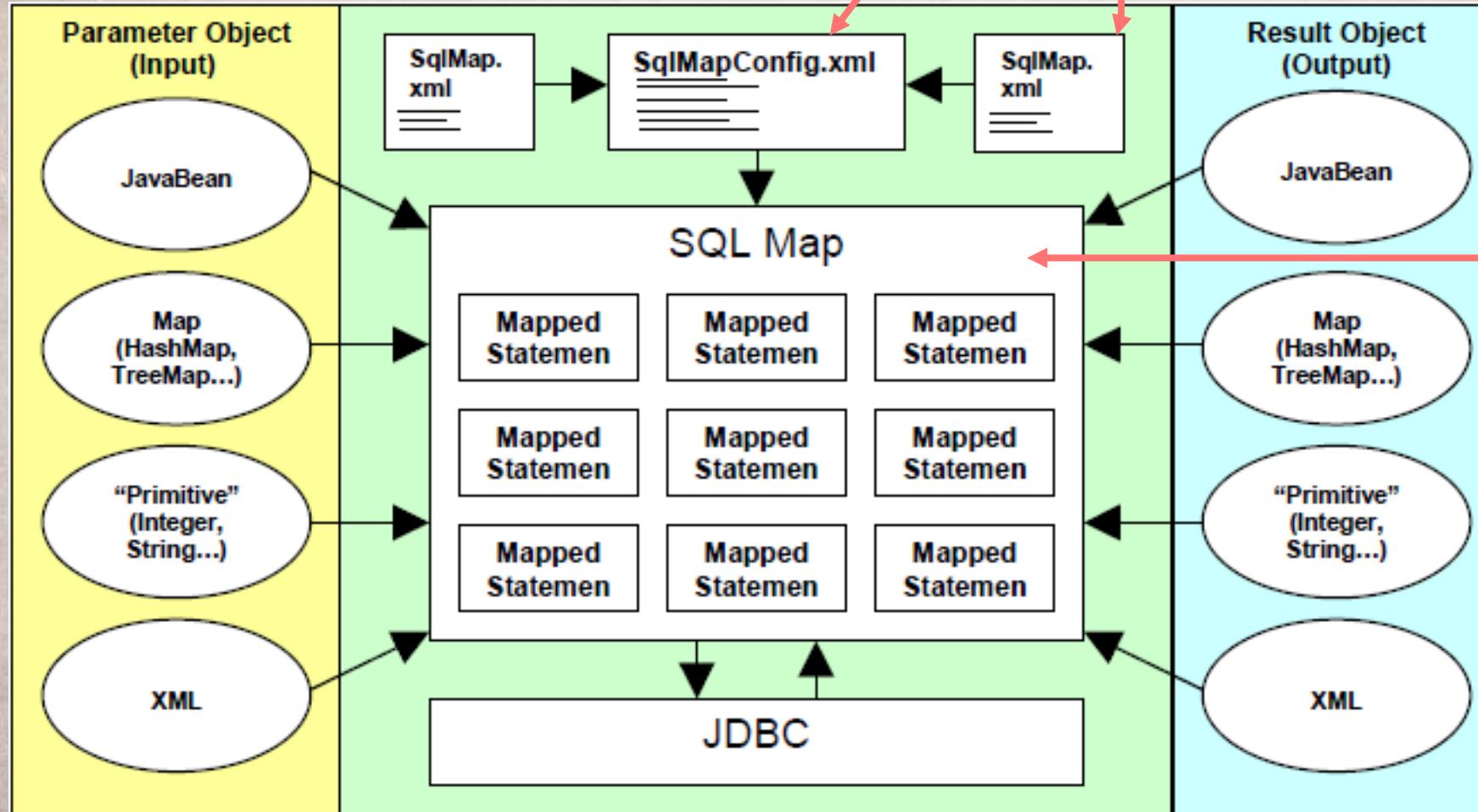
- Execution Output:** The Console tab shows the application's output after running the main method. It lists five BoardVO objects with their attributes.

```
<terminated> BoardServiceClient [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (2018. 11. 12. 오후 10:16:53)
-->BoardVO [seq=5, title=글 등록, writer=글 등록, content=글 수정테스트, regDate=2018-09-10, cnt=0]
-->BoardVO [seq=4, title=JDBC 테스트2, writer=관리자, content=JDBC 테스트2....., regDate=2018-05-27, cnt=0]
-->BoardVO [seq=3, title=JDBC 테스트, writer=관리자, content=JDBC 테스트....., regDate=2018-05-27, cnt=0]
-->BoardVO [seq=2, title=임시 제목, writer=홍길동, content=임시 내용 ....., regDate=2018-05-12, cnt=0]
-->BoardVO [seq=1, title=가입인사, writer=관리자, content=잘 부탁드립니다...., regDate=2018-04-08, cnt=0]
```

## 02. Mapper XML 파일 설정

## 2.1.1 Mybatis 프레임워크 구조

입력



출력

Mybatis 메인 환경 설정 파일  
(mybatis-config.xml)

SQL Mapper파일  
(board-mapping.xml)

실행할 SQL명령어가  
Map 구조로 저장관리

## 2.1.2 SQL Mapper 파일 구조

```
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"  
      "http://mybatis.org/dtd/mybatis-3-mapper.dtd">  
<mapper namespace="BoardDAO"> mapper 루트 엘리먼트  
  
    <insert id="insertBoard">  
        insert into board(seq, title, writer, content)  
            values((select nvl(max(seq),0)+1 from board),#{title}, #{writer},#{content})  
    </insert>  
  
    <select id="getBoard" resultType="board">  
        select * from board where seq=#{seq}  
    </select>  
  
    <select id="getBoardList" resultType="board">  
        select * from board order by seq desc  
    </select>  
  
    <delete id="deleteBoard">  
        delete board where seq=#{seq}  
    </delete>  
  
    <update id="updateBoard">  
        update board set title=#{title}, content=#{content} where seq=#{seq}  
    </update>  
  
</mapper>
```

DTD 선언 부분

SQL 매팅 부분

# Mapper 파일과 DAO 클래스 연결

Mapper XML	<pre>&lt;mapper namespace="BoardDAO"&gt;     &lt;insert id="insertBoard"&gt;         insert into board(seq, title, writer, content)         values((select nvl(max(seq),0)+1 from board),#{title}, #{writer},#{content})     &lt;/insert&gt;</pre>
DAO 클래스	<pre>public void insertBoard(BoardVO vo) {     mybatis.insert("BoardDAO.insertBoard", vo);     mybatis.commit(); }</pre>

- Mapper 파일에 SQL 명령어를 등록할 때는 SQL 구문의 종류에 따라 적절한 엘리먼트를 사용한다.
- INSERT 구문은 <insert>, SELECT구문은 <select> 엘리먼트를 사용
- 각 엘리먼트에서 사용할 수 있는 속성들이 다르므로 그 용도와 의미를 이해해야 한다.

## 2.1.3 <select> 엘리먼트

Mapper XML

```
<select id="getBoard" parameterType="board" resultType="board">
    select * from board where seq=#{seq}
</select>

<select id="getBoardList" resultType="board">
    select * from board order by seq desc
</select>
```

- id – 필수 속성으로 Mapper 파일들 내에서 유일해야하며 DAO에서 호출 가능하다.
- parameterType – SQL 실행에 필요한 데이터를 외부로부터 받아야 할 때 사용하며 일반적으로 기본형이나 VO형태의 클래스를 지정
- resultType – SQL 구문이 실행된 ResultSet 결과를 맵핑할 객체 지정

## 2.1.4 기타 엘리먼트

<insert> 엘리먼트	<pre>&lt;insert id="insertBoard" parameterType="board"&gt;     insert into board(seq, title, writer, content)         values((select nvl(max(seq),0)+1 from board),#{title}, #{writer},#{content}) &lt;/insert&gt;</pre>
<update> 엘리먼트	<pre>&lt;update id="updateBoard" parameterType="board"&gt;     update board set title=#{title}, content=#{content} where seq=#{seq} &lt;/update&gt;</pre>
<delete> 엘리먼트	<pre>&lt;delete id="deleteBoard" parameterType="board"&gt;     delete board where seq=#{seq} &lt;/delete&gt;</pre>

## 2.2 SQL Mapper XML 추가 설정

### • 2.2.1 resultMap 속성 설정

- 일반적으로 resultType 속성으로 SQL 실행결과를 넘겨주는 방식을 사용한다.
  - 하나의 테이블 결과를 넘겨줄 때 테이블의 VO객체 사용
  - 2개 테이블을 join하여 넘겨줄 때는 적용할 수 없다. → resultMap 사용

### • 2.2.2 CDATA Section 사용

```
<select id="getBoard" parameterType="board" resultType="board">
    select * from board where seq=#{seq}
</select>
```

- CDATA(Character DATA)는 XML 파서에 의해서 해석되지 않는다. 즉, CDATA 안의 내용이 그대로 DB에 전달되어 실행된다.

```
<select id="getBoard" parameterType="board" resultType="board">
    <![CDATA[
        select * from board where seq=#{seq} < 5
    ]]>
</select>
```

## 2.3 Mybatis JAVA API

### • 2.3.1 SqlSessionFactoryBuilder 클래스

- Mybatis로 DAO 클래스의 CRUD메소드를 구현하려면 Mybatis에서 제공하는 SqlSession객체를 사용해야 함.
- SqlSession객체는 SqlSessionFactory로부터 얻어야한다.
- 따라서, 가장 먼저 할 작업은 SqlSessionFactory 객체를 생성하는 일임.
- SqlSessionFactory는 SqlSessionFactoryBuild의 build()메소드를 이용
  - Mybatis 설정 파일()을 로딩하여 SqlSessionFactory객체를 생성

```
Reader reader = Resources.getResourceAsReader("mybatis-config.xml");
sessionFactory = new SqlSessionFactoryBuilder().build(reader);
```

### • 2.3.2 SqlSessionFactory 클래스

- SqlSession 객체에 대한 생성 역할을 수행
- SqlSessionFactory객체는 openSession()이라는 메소드를 통해 SqlSession 객체를 획득

```
sessionFactory.openSession();
```

### • 2.3.3 유ти리티 클래스 작성 – 싱글톤 디자인 패턴으로 작성



The screenshot shows a Java code editor with the following tabs at the top: root-context.xml, mybatis-config.xml, board-mapping.xml, BoardDAO.java, SqlSessionFactoryBean.java (which is the active tab), and BoardServiceClient.java.

```
1 package kr.ac.inje.comsi.board;
2
3 import java.io.Reader;
4
5 import org.apache.ibatis.io.Resources;
6 import org.apache.ibatis.session.SqlSession;
7 import org.apache.ibatis.session.SqlSessionFactory;
8 import org.apache.ibatis.session.SqlSessionFactoryBuilder;
9
10 public class SqlSessionFactoryBean {
11     private static SqlSessionFactory sessionFactory = null;
12
13     static {
14         try {
15             if (sessionFactory == null) {
16                 Reader reader = Resources.getResourceAsReader("mybatis-config.xml");
17                 sessionFactory = new SqlSessionFactoryBuilder().build(reader);
18             }
19         } catch (Exception e) {
20             e.printStackTrace();
21         }
22     }
23
24     public static SqlSession getSqlSessionInstance() {
25         return sessionFactory.openSession();
26     }
27 }
```

## 2.3.4 SqlSession 객체

(1) selectOne() : 오직 하나의 데이터를 검색하는 SQL구문 실행할 때

- public Object selectOne(String statement)
- public Object selectOne(String statement, Object parameter)
- statement 매개변수는 Mapper XML 파일에 등록된 SQL의 아이디(ID)

(2) selectList() : 여러 개의 데이터가 검색되는 SQL구문을 실행할 때

- public Object selectList(String statement)
- public Object selectList(String statement, Object parameter)

(3) insert(), update(), delete() : 각각 INSERT, UPDATE, DELETE SQL 구문을 실행할 때

- public int insert(String statement, Object parameter)
- public int update(String statement, Object parameter) throws SQLException
- public int delete(String statement, Object parameter) throws SQLException

→ 사용 방법은 BoardDAO 클래스에서 확인

# 03. 스프링과 Mybatis 연동

## 3.2 Mybatis 설정 파일 복사 및 설정

The screenshot shows an IDE interface with several tabs at the top: root-context.xml, mybatis-config.xml (which is the active tab), board-mapping.xml, BoardDAO.java, SqlSessionFactoryBean.java, and BoardServiceClient.java.

The code in the mybatis-config.xml tab is as follows:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
3 "http://mybatis.org/dtd/mybatis-3-config.dtd">
4
5<configuration>
6    <!-- <properties resource="db.properties"/> -->
7    <typeAliases>
8        <typeAlias alias="board" type="kr.ac.inje.comsi.board.BoardVO"/>
9    </typeAliases>
10   <!-- <environments default="development">
11      <environment id="development">
12          <transactionManager type="JDBC"/>
13          <dataSource type="POOLED">
14              <property name="driver" value="org.h2.Driver"/></property>
15              <property name="url" value="jdbc:h2:D:/workspace/h2db/db"/></property>
16              <property name="username" value="sa"/></property>
17              <property name="password" value=""></property>
18          </dataSource>
19      </environment>
20  </environments> -->
21
22  <mappers>
23      <mapper resource="mapper/board-mapping.xml"/>
24  </mappers>
25</configuration>
26
```

A red curly brace on the right side of the code groups the environment configuration section (lines 10-18) under the heading "주석 처리" (Commented Out).

### 3.3 스프링 연동: root-context.xml 파일 수정

SqlSessionFactoryBean 클래스를 Bean으로 등록해서 사용

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:p="http://www.springframework.org/schema/p"
5   xmlns:context="http://www.springframework.org/schema/context"
6   xmlns:aop="http://www.springframework.org/schema/aop"
7   xmlns:tx="http://www.springframework.org/schema/tx"
8   xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
9     http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.3.xsd
10    http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.3.xsd
11    http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.3.xsd">
12
13 <context:component-scan base-package="kr.ac.inje.comsi"></context:component-scan>
14
15 <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
16   <property name="driverClassName" value="org.h2.Driver"></property>
17   <property name="url" value="jdbc:h2:D:/workspace/h2db/db"></property>
18   <property name="username" value="sa"></property>
19   <property name="password" value=""></property>
20 </bean>
21
22 <!-- mybatis-config.xml 파일이 스프링 시작할 때, 같이 동작하도록 설정 -->
23 <bean id="sessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
24   <property name="dataSource" ref="dataSource"></property>
25   <property name="configLocation" value="classpath:mybatis-config.xml"></property>
26 </bean>
27 </beans>
28
```

DataSource 설정

SqlSessionFactoryBean 등록

## 3.4 DAO 클래스 구현-방법1

- SqlSessionDaoSupport 클래스 상속하여 구현

The screenshot shows a Java code editor with several tabs at the top: SqlSessionFactoryBean.java, BoardServiceClient.java, BoardDAOMybatis.java (which is the active tab), applicationContext.xml, and BoardServiceImpl.java.

```
1 package kr.ac.inje.comsi.board;
2
3 import java.util.List;
4
5 import org.apache.ibatis.session.SqlSessionFactory;
6 import org.mybatis.spring.support.SqlSessionDaoSupport;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.stereotype.Repository;
9
10 @Repository
11 public class BoardDAOMybatis extends SqlSessionDaoSupport {
12
13     @Autowired // 스프링 설정 파일의 SqlSessionFactoryBean을 인자로 받아 설정
14     public void setSqlSessionFactory(SqlSessionFactory sqlSessionFactory) {
15         super.setSqlSessionFactory(sqlSessionFactory);
16     }
17
18     public void insertBoard(BoardVO vo) {
19         System.out.println("==>Mybatis로 insertBoard기능 처리");
20         getSqlSession().insert("BoardDAO.insertBoard", vo);
21     }
22
23
24     public void updateBoard(BoardVO vo) {
25         System.out.println("==>Mybatis로 updateBoard기능 처리");
26         getSqlSession().update("BoardDAO.updateBoard", vo);
27     }
}
```

# 이어서...

```
28
29@  public void deleteBoard(BoardVO vo) {
30      System.out.println("==>Mybatis로 deleteBoard기능 처리");
31      getSqlSession().delete("BoardDAO.deleteBoard", vo);
32  }
33
34
35@  public BoardVO getBoard(BoardVO vo) {
36      System.out.println("==>Mybatis로 getBoard기능 처리");
37      return (BoardVO) getSqlSession().selectOne("BoardDAO.getBoard", vo);
38  }
39
40@  public List<BoardVO> getBoardList(BoardVO vo) {
41      System.out.println("==>Mybatis로 getBoardList기능 처리");
42      return getSqlSession().selectList("BoardDAO.getBoardList", vo);
43  }
44
45 }
46 }
```

# BoardServiceImpl 클래스 수정

BoardServiceClient.java BoardDAOMybatis.java applicationContext.xml BoardServiceImpl.java

```
1 package kr.ac.inje.comsi.board.impl;
2
3 import java.util.List;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.stereotype.Service;
7
8 import kr.ac.inje.comsi.board.BoardDAOMybatis;
9 import kr.ac.inje.comsi.board.BoardService;
10 import kr.ac.inje.comsi.board.BoardVO;
11
12 @Service("boardService")
13 public class BoardServiceImpl implements BoardService {
14
15     @Autowired
16     private BoardDAOMybatis boardDAOMybatis;
17     //private BoardDAO boardDAO;
18
19
20     @Override
21     public void insertBoard(BoardVO vo) {
22         //boardDAO.insertBoard(vo);
23         boardDAOMybatis.insertBoard(vo);
24     }
25 }
```

# 이어서...

```
26@override
△27 public void updateBoard(BoardVO vo) {
28     //boardDAO.updateBoard(vo);
29     boardDAOMybatis.updateBoard(vo);
30 }
31
32@override
△33 public void deleteBoard(BoardVO vo) {
34     //boardDAO.deleteBoard(vo);
35     boardDAOMybatis.deleteBoard(vo);
36 }
37
38@override
△39 public BoardVO getBoard(BoardVO vo) {
40     //return boardDAO.getBoard(vo);
41     return boardDAOMybatis.getBoard(vo);
42 }
43
44@override
△45 public List<BoardVO> getBoardList(BoardVO vo) {
46     //return boardDAO.getBoardList(vo);
47     return boardDAOMybatis.getBoardList(vo);
48 }
49 }
50 }
```

# BoardServiceClient 클래스...

```
BoardServiceClient.java  BoardDAOMybatis.java  applicationContext.xml  BoardServiceImpl.java

1 package kr.ac.inje.comsi.board;
2
3 import java.util.List;
4
5 import org.springframework.context.support.AbstractApplicationContext;
6 import org.springframework.context.support.GenericXmlApplicationContext;
7
8 public class BoardServiceClient {
9
10    public static void main(String[] args) {
11        // 1. Spring 컨테이너를 구동
12        AbstractApplicationContext container = new GenericXmlApplicationContext("applicationContext.xml");
13
14        // 2. Spring 컨테이너로부터 UserServiceImpl 객체를 Lookup
15        BoardService boardService = (BoardService) container.getBean("boardService");
16
17        BoardVO vo = new BoardVO();
18        vo.setTitle("myBatis 제목");
19        vo.setWriter("김길동");
20        vo.setContent("myBatis 내용입니다....");
21        //boardDAO.insertBoard(vo);
22
23        List<BoardVO> boardList = boardService.getBoardList(vo);
24        for (BoardVO board:boardList) {
25            System.out.println("-->" + board.toString());
26        }
27        container.close();
28    }
29
30 }
```

# 실행 결과...

```
Console Progress Problems Markers

<terminated> BoardServiceClient [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (2018. 11. 15. 오후 9:44:36)
INFO : org.springframework.beans.factory.xml.XmlBeanDefinitionReader - Loading XML bean definitions from class path resource [applicationContext.xml]
INFO : org.springframework.context.support.GenericXmlApplicationContext - Refreshing org.springframework.context.support.GenericXmlApplicationContext
INFO : org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor - JSR-330 'javax.inject.Inject' annotation found and supported in this bean
==>Mybatis로 getBoardList기능 처리
-->BoardVO [seq=5, title=글 등록, writer=글 등록, content=글 수정테스트, regDate=2018-09-10, cnt=0]
-->BoardVO [seq=4, title=JDBC 테스트2, writer=관리자, content=JDBC 테스트2....., regDate=2018-05-27, cnt=0]
-->BoardVO [seq=3, title=JDBC 테스트, writer=관리자, content=JDBC 테스트....., regDate=2018-05-27, cnt=0]
-->BoardVO [seq=2, title=임시 제목, writer=홍길동, content=임시 내용 ....., regDate=2018-05-12, cnt=0]
-->BoardVO [seq=1, title=가입인사, writer=관리자, content=잘 부탁드립니다...., regDate=2018-04-08, cnt=0]
```

## 3.4 DAO 클래스 구현-방법2

- SqlSessionTemplate 클래스를 Bean으로 등록하여 사용

The screenshot shows the Eclipse IDE interface with the following tabs at the top: BoardServiceClient.java, BoardDAOMybatis.java, applicationContext.xml (selected), and BoardServiceImpl.java.

```
1<beans xmlns="http://www.springframework.org/schema/beans"
2  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3  xmlns:p="http://www.springframework.org/schema/p"
4  xmlns:context="http://www.springframework.org/schema/context"
5  xmlns:aop="http://www.springframework.org/schema/aop"
6  xmlns:tx="http://www.springframework.org/schema/tx"
7  xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
8          http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.3.xsd
9          http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.3.xsd
10         http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.3.xsd">
11
12 <context:component-scan base-package="kr.ac.inje.comsi"></context:component-scan>
13
14 <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
15     <property name="driverClassName" value="org.h2.Driver"></property>
16     <property name="url" value="jdbc:h2:D:/workspace/h2db/db"></property>
17     <property name="username" value="sa"></property>
18     <property name="password" value=""></property>
19 </bean>
20
21
22 <!-- mybatis-config.xml 파일이 스프링 시작할 때, 같이 동작하도록 설정 -->
23 <bean id="sqlSession" class="org.mybatis.spring.SqlSessionFactoryBean">
24     <property name="dataSource" ref="dataSource"></property>
25     <property name="configLocation" value="classpath:mybatis-config.xml"></property>
26 </bean>
27 <bean class="org.mybatis.spring.SqlSessionTemplate">
28     <!-- SqlSessionTemplate 클래스는 setter 메소드가 없어서 생성자로 주입함 -->
29     <constructor-arg ref="sqlSession"></constructor-arg>
30 </bean>
31 </beans>
```

A red rectangular box highlights the configuration of the SqlSessionTemplate bean in lines 27 to 31. A red arrow points from the text "SqlSessionTemplate 클래스는 setter 메소드가 없어서 생성자로 주입함 -->" to the "constructor-arg" element in the XML code.

# BoardDAOMybatis 클래스 수정

```
BoardServiceClient.java BoardDAOMybatis.java applicationContext.xml BoardServiceImpl.java

1 package kr.ac.inje.comsi.board;
2
3 import java.util.List;
4
5 import org.mybatis.spring.SqlSessionTemplate;
6 import org.springframework.beans.factory.annotation.Autowired;
7 import org.springframework.stereotype.Repository;
8
9 @Repository
10 public class BoardDAOMybatis {
11
12     @Autowired
13     private SqlSessionTemplate mybatis;
14
15     public void insertBoard(BoardVO vo) {
16         System.out.println("==>Mybatis로 insertBoard기능 처리");
17         mybatis.insert("BoardDAO.insertBoard", vo);
18     }
19
20
21     public void updateBoard(BoardVO vo) {
22         System.out.println("==>Mybatis로 updateBoard기능 처리");
23         mybatis.update("BoardDAO.updateBoard", vo);
24     }
25 }
```

# 이어서...

```
26@ public void deleteBoard(BoardVO vo) {  
27     System.out.println("==>Mybatis로 deleteBoard기능 처리");  
28     mybatis.delete("BoardDAO.deleteBoard", vo);  
29 }  
30  
31  
32@ public BoardVO getBoard(BoardVO vo) {  
33     System.out.println("==>Mybatis로 getBoard기능 처리");  
34     return (BoardVO) mybatis.selectOne("BoardDAO.getBoard", vo);  
35 }  
36  
37@ public List<BoardVO> getBoardList(BoardVO vo) {  
38     System.out.println("==>Mybatis로 getBoardList기능 처리");  
39     return mybatis.selectList("BoardDAO.getBoardList", vo);  
40 }  
41  
42 }  
43  
44
```

# 실행 결과

```
Console Progress Problems Markers
<terminated> BoardServiceClient [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (2018. 11. 15. 오후 9:58:04)
INFO : org.springframework.beans.factory.xml.XmlBeanDefinitionReader - Loading XML bean definitions from class path resource [applicationContext.xml]
INFO : org.springframework.context.support.GenericXmlApplicationContext - Refreshing org.springframework.context.support.GenericXmlApplicationCo
INFO : org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor - JSR-330 'javax.inject.Inject' annotation found on field value of type java.util.List
==>Mybatis로 getBoardList기능 처리
-->BoardVO [seq=5, title=글 등록, writer=글 등록, content=글 수정테스트, regDate=2018-09-10, cnt=0]
-->BoardVO [seq=4, title=JDBD 테스트2, writer=관리자, content=JDBC 테스트2....., regDate=2018-05-27, cnt=0]
-->BoardVO [seq=3, title=JDBD 테스트, writer=관리자, content=JDBC 테스트....., regDate=2018-05-27, cnt=0]
-->BoardVO [seq=2, title=임시 제목, writer=홍길동, content=임시 내용 ....., regDate=2018-05-12, cnt=0]
-->BoardVO [seq=1, title=가입인사, writer=관리자, content=잘 부탁드립니다...., regDate=2018-04-08, cnt=0]
INFO : org.springframework.context.support.GenericXmlApplicationContext - Closing org.springframework.context.support.GenericXmlApplicationCo
```

WEB에서 보기.

# root-context.xml 파일 수정(applicationContext.xml과 동일)

```
BoardServiceClient.java BoardDAOMybatis.java applicationContext.xml BoardServiceImpl.java root-context.xml
5  xmlns:mvc="http://www.springframework.org/schema/mvc"
6  xmlns:mybatis-spring="http://mybatis.org/schema/mybatis-spring"
7  xmlns:jdbc="http://www.springframework.org/schema/jdbc"
8  xsi:schemaLocation="http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-4.3.xsd
9    http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-4.3.xsd
10   http://mybatis.org/schema/mybatis-spring http://mybatis.org/schema/mybatis-spring-1.2.xsd
11   http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.3.xsd
12   http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.3.xsd">
13
14 <context:component-scan base-package="kr.ac.inje.comsi.board"></context:component-scan>
15
16<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
17   <property name="driverClassName" value="org.h2.Driver"></property>
18   <property name="url" value="jdbc:h2:D:/workspace/h2db/db"></property>
19   <property name="username" value="sa"></property>
20   <property name="password" value=""></property>
21 </bean>
22
23 <!-- mybatis-config.xml 파일이 스프링 시작할 때, 같이 동작하도록 설정 -->
24<bean id="sqlSession" class="org.mybatis.spring.SqlSessionFactoryBean">
25   <property name="dataSource" ref="dataSource"></property>
26   <property name="configLocation" value="classpath:mybatis-config.xml"></property>
27 </bean>
28<bean class="org.mybatis.spring.SqlSessionTemplate">
29   <!-- SqlSessionTemplate클래스는 setter 메소드가 없어서 생성자로 주입함 -->
30   <constructor-arg ref="sqlSession"></constructor-arg>
31 </bean>
32
33</beans>
34
```

BoardDAOMyb... root-contex... 글 목록 BoardDAO.java BoardControl... BoardServic... mybatis-conf... board-mappin... »  
http://localhost:8080/Board/getBoardList.do

# 글 목록

환영합니다. [Logout](#)

번호	제목	작성자	등록일	조회수
5	<a href="#">글 등록</a>	글 등록	2018-09-10	0
4	<a href="#">JDBD 테스트2</a>	관리자	2018-05-27	0
3	<a href="#">JDBD 테스트</a>	관리자	2018-05-27	0
2	<a href="#">임시 제목</a>	홍길동	2018-05-12	0
1	<a href="#">가입인사</a>	관리자	2018-04-08	0

[새글 등록](#)

BoardDAOMyb... root-contex... 글 상세 BoardDAO.java BoardControl... BoardServic... mybatis-conf... board-mappin... »<sub>2</sub>

http://localhost:8080/Board/getBoard.do?seq=5

[Log-out](#)

제목	글 등록
작성자	글 등록
내용	글 수정테스트
등록일	2018-09-10
조회수	0
<a href="#">글 수정</a>	

[글등록](#) [글삭제](#) [글목록](#)