



박 경 태

comsi.java@gmail.com

고급 자바 프로그래밍

: STS를 이용한 Spring 프로그래밍

강의 내용

순서	내용
1	<ul style="list-style-type: none">Spring IoC를 이용한 비즈니스 컴포넌트 만들기
2	<ul style="list-style-type: none">Spring AOP(Aspect Oriented Programming)를 이용한 공통 서비스 만들기Spring DAO(Data Access Object)를 이용한 데이터베이스 연동 및 트랜잭션 처리
3	<ul style="list-style-type: none">Spring MVC를 이용한 MVC 아키텍쳐 적용하기
4	<ul style="list-style-type: none">Spring MVC의 부가 기능 사용하기(파일 업로드, 다국어, 예외 처리 등)
5	<ul style="list-style-type: none">Spring과 MyBatis 연동하기Spring과 JPA 연동하기

스프링 JDBC

- JDBC(Java Database Connectivity)
→ 자바에서 데이터베이스에 접속할 수 있도록 하는 자바 API이다. JDBC는 데이터베이스에서 자료를 쿼리하거나 업데이트하는 방법을 제공
- JDBC를 이용한 DB 연동 프로그램을 개발하면 데이터베이스에 비종속적인 DB 연동 로직을 구현할 수 있다.
 - But, JDBC 프로그램을 이용하면 개발자가 작성해야 할 코드가 너무 많다

UserDAO.java

```
public class UserDAO {  
    // JDBC 관련 변수  
    private Connection conn = null;  
    private PreparedStatement stmt = null;  
    private ResultSet rset = null;  
    // SQL 명령어들  
    private final String USER_GET = "select * from users where id=? and password=?";  
  
    // CRUD 기능의 메소드  
    // 회원 등록  
    public UserVO getUser(UserVO vo){  
        UserVO user = null;  
        try{  
            System.out.println("==> JDBC로 getUser() 기능 처리");  
            conn = JDBCUtil.getConnection();  
            stmt = conn.prepareStatement(USER_GET);  
            stmt.setString(1, vo.getId());  
            stmt.setString(2, vo.getPassword());  
            rset = stmt.executeQuery();  
  
            if (rset.next()){  
                user = new UserVO();  
                user.setId(rset.getString("ID"));  
                user.setName(rset.getString("NAME"));  
                user.setPassword(rset.getString("PASSWORD"));  
                user.setRole(rset.getString("ROLE"));  
            }  
        }catch(Exception e){  
            e.printStackTrace();  
        }finally{  
            JDBCUtil.close(rset, stmt, conn);  
        }  
        return user;  
    }  
}
```

JDBCUtil.java - 반복코드??

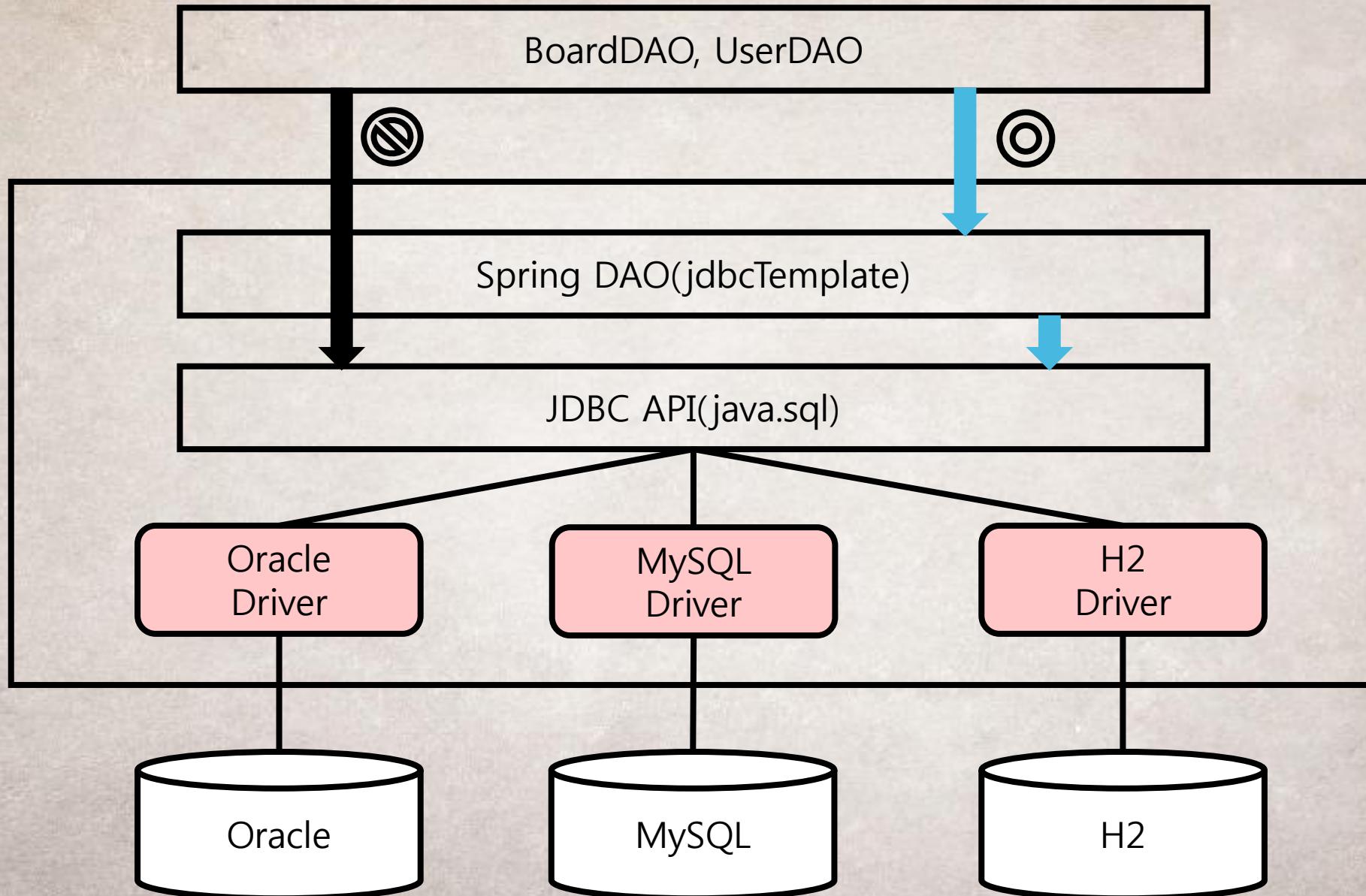
```
public static Connection getConnection(){
    try{
        Class.forName("org.h2.Driver");
        return DriverManager.getConnection("jdbc:h2:/d:/workspace/java/h2db/myDB", "sa", "");
    }catch(Exception e){
        e.printStackTrace();
    }
    return null;
}
// close connection
public static void close(PreparedStatement stmt, Connection conn){
    if(stmt != null){
        try{
            if(!stmt.isClosed()) stmt.close();
        }catch(Exception e){
            e.printStackTrace();
        }finally{
            stmt = null;
        }
    }

    if(conn != null){
        try{
            if(!conn.isClosed()) conn.close();
        }catch(Exception e){
            e.printStackTrace();
        }finally{
            conn = null;
        }
    }
}
```

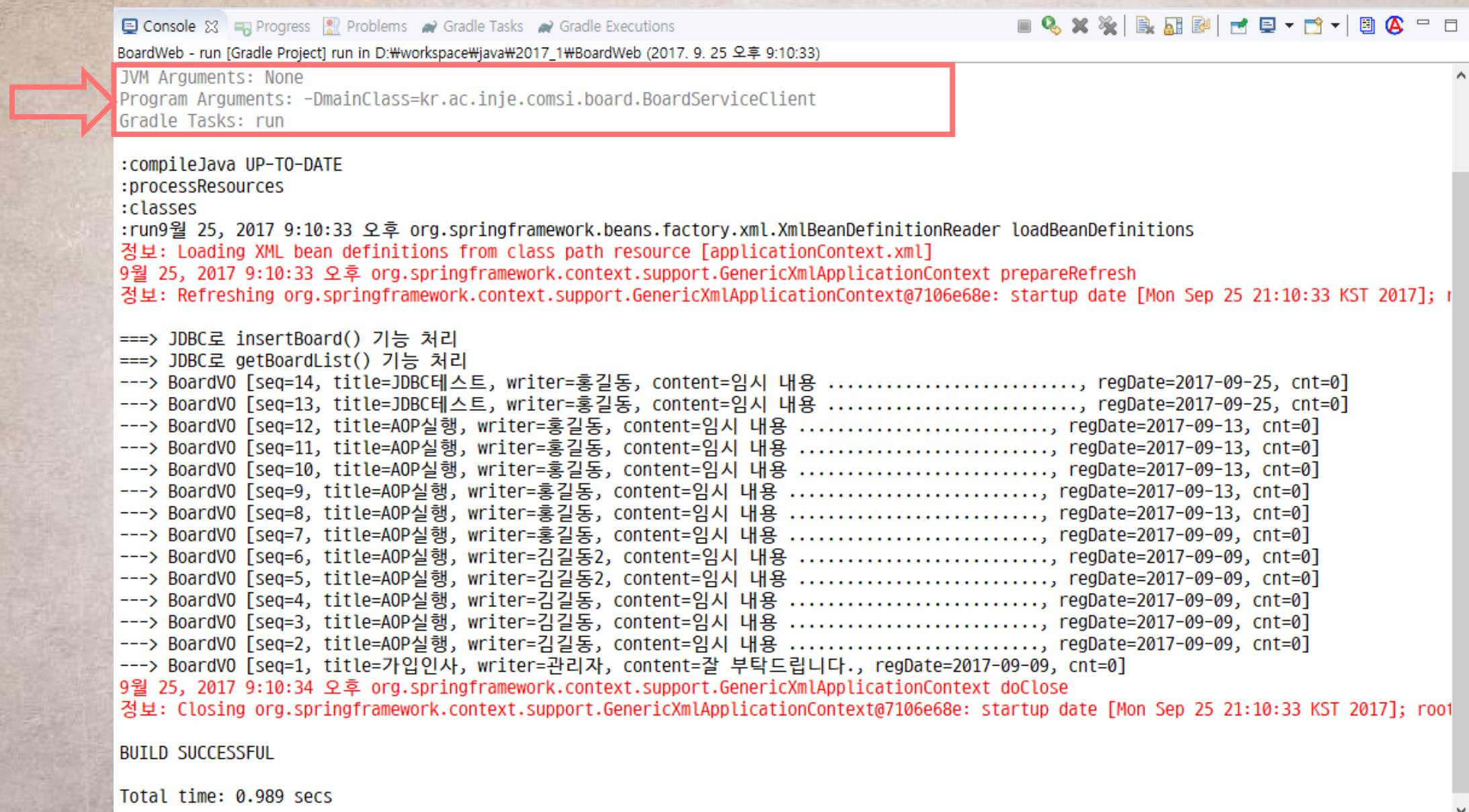
JdbcTemplate 클래스

- 스프링에서 JDBC 기반의 DB 연동 프로그램을 쉽게 개발할 수 있도록 jdbcTemplate 클래스 지원
- JdbcTemplate은 Gof 디자인 패턴 중 템플릿 메소드 패턴이 적용된 클래스
 - 템플릿 메소드 패턴은 복잡하고 반복되는 알고리즘을 캡슐화해서 재사용하는 패턴으로 정의
- JDBC처럼 코딩 순서가 정형화된 기술에서 유용하게 사용
 - 반복되는 DB 연동 로직은 JdbcTemplate 클래스의 템플릿 메소드가 제공
 - 개발자는 달라지는 SQL 구문과 설정값만 신경 씀

JdbcTemplate의 위치와 역할



JDBC 설정 후 DB 연동 결과



The screenshot shows the Eclipse IDE's Console view with the title "BoardWeb - run [Gradle Project] run in D:\workspace\java\2017_1\BoardWeb (2017. 9. 25 오후 9:10:33)". A red arrow points to the command line arguments section, which includes "JVM Arguments: None", "Program Arguments: -DmainClass=kr.ac.inje.comsi.board.BoardServiceClient", and "Gradle Tasks: run". The main output area displays the execution log:

```
:compileJava UP-TO-DATE
:processResources
:classes
:run
9월 25, 2017 9:10:33 오후 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
9월 25, 2017 9:10:33 오후 org.springframework.context.support.GenericXmlApplicationContext prepareRefresh
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@7106e68e: startup date [Mon Sep 25 21:10:33 KST 2017]; root of context hierarchy
====> JDBC로 insertBoard() 가능 처리
====> JDBC로 getBoardList() 가능 처리
---> BoardVO [seq=14, title=JDBC테스트, writer=홍길동, content=임시 내용 ....., regDate=2017-09-25, cnt=0]
---> BoardVO [seq=13, title=JDBC테스트, writer=홍길동, content=임시 내용 ....., regDate=2017-09-25, cnt=0]
---> BoardVO [seq=12, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=11, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=10, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=9, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=8, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=7, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=6, title=AOP실행, writer=김길동2, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=5, title=AOP실행, writer=김길동2, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=4, title=AOP실행, writer=김길동, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=3, title=AOP실행, writer=김길동, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=2, title=AOP실행, writer=김길동, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=1, title=가입인사, writer=관리자, content=잘 부탁드립니다., regDate=2017-09-09, cnt=0]
9월 25, 2017 9:10:34 오후 org.springframework.context.support.GenericXmlApplicationContext doClose
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@7106e68e: startup date [Mon Sep 25 21:10:33 KST 2017]; root of context hierarchy
BUILD SUCCESSFUL
Total time: 0.989 secs
```

스프링 JDBC 설정 - 라이브러리 추가

The screenshot shows an IDE interface with several tabs at the top: applicationContext.xml, UserDAO.java, JDBCUtil.java, and build.gradle. The build.gradle tab is currently selected, displaying the following Groovy code:

```
1 /*
2 * This build file was generated by the Gradle 'init' task.
3 *
4 * This generated file contains a sample Java project to get you started.
5 * For more details take a look at the Java Quickstart chapter in the Gradle
6 * user guide available at https://docs.gradle.org/3.3/userguide/tutorial_java_projects.html
7 */
8
9 // Apply the java plugin to add support for Java
10 apply plugin: 'application'
11
12 mainClassName = System.getProperty("mainClass")
13
14 compileJava.options.encoding = 'UTF-8'
15
16 // In this section you declare where to find the dependencies of your project
17 repositories {
18     mavenCentral()
19 }
20
21 dependencies {
22     compile 'org.springframework:spring-context:4.3.8.RELEASE'
23     compile 'com.h2database:h2:1.4.195'
24     compile 'org.aspectj:aspectjweaver:1.8.8'
25     compile 'org.apache.commons:commons-dbcop2:2.1' // Line 25, highlighted with a red box and arrow
26 }
27
28
```

A red arrow points from the highlighted line 25 to the word "추가하기" (Add) located below the code editor.

To the right of the code editor, a "Project and External Dependencies" tree view is shown, listing various JAR files:

- spring-context-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\spring\spring-context\4.3.8.RELEASE\43f3a...
- h2-1.4.195.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\com\h2database\h2\1.4.195\10...
- aspectjweaver-1.8.8.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\org\aspectj\aspectjweaver\1.8.8\10...
- commons-dbcop2-2.1.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\org\apache\commons\dbcop2\2.1\10...
- spring-aop-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\org\springframework\spring-aop\4.3.8.RELEASE\43f3a...
- spring-beans-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\org\springframework\spring-beans\4.3.8.RELEASE\43f3a...
- spring-core-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\org\springframework\spring-core\4.3.8.RELEASE\43f3a...
- spring-expression-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\org\springframework\spring-expression\4.3.8.RELEASE\43f3a...
- commons-pool2-2.3.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\org\apache\commons\commons-pool2\2.3\10...
- commons-logging-1.2.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\org\apache\commons\commons-logging\1.2\10...

JdbcTemplate 메소드

- update() 메소드: “?”에 값을 설정하는 방식에 따라 두 가지 형태가 사용
 - 첫번째: SQL 구문에 설정된 “?” 수만큼 값을 나열하는 방식

메소드	int update(String sql, Object ... args)
사용 예	// 글 수정 public void updateBoard(BoardVO vo) { String BOARD_UPDATE = "update board set title=?, content=? where seq=?"; int cnt = jdbcTemplate.update(BOARD_UPDATE, vo.getTitle(), vo.getContent(), vo.getSeq()); System.out.println(cnt + " 건 데이터 수정"); }

- 두번째: Object 배열 객체에 SQL 구문에 설정된 “?” 수만큼의 값을 세팅하여 배열 객체를 두번째 인자로 전달하는 방식

메소드	int update(String sql, Object ... args)
사용 예	// 글 수정 public void updateBoard(BoardVO vo) { String BOARD_UPDATE = "update board set title=?, content=? where seq=?"; Object[] args = {vo.getTitle(), vo.getContent(), vo.getSeq()}; int cnt = jdbcTemplate.update(BOARD_UPDATE, args); System.out.println(cnt + " 건 데이터 수정"); }

- **queryForInt() 메소드**
 - SELECT 구문으로 검색된 정수값을 받을 때 사용하며 매개변수의 의미는 앞에서 본 update() 메소드와 같다.

메소드	int queryForInt(String sql) int queryForInt(String sql, Object... args) int queryForInt(String sql, Object[] args)
사용 예	<pre>// 전체 게시글 수 조회 public int getBoardTotalCount(BoardVO vo) { String BOARD_TOT_COUNT = "select count(*) from board"; int cnt = jdbcTemplate.queryForInt(BOARD_TOT_COUNT); System.out.println("전체 게시글 수: " + cnt + " 건"); }</pre>

• queryForObject() 메소드

- SELECT 구문의 실행 결과를 특정 자바 객체(Value Object)로 매팅하여 리턴받을 때 사용한다.
- 검색 결과가 없거나 검색 결과가 두개 이상이면 예외 (IncorrectResultSizeDataAccessException)를 발생시킨다.

메소드	<code>int queryForObject(String sql)</code> <code>int queryForObject(String sql, RowMapper<T> rowMapper)</code> <code>int queryForObject(String sql, Object[] args, RowMapper<T> rowMapper)</code>
사용 예	// 글 상세 조회 public BoardVO getBoard(BoardVO vo) { String BOARD_GET = "select * from board where seq=?"; Object[] args = {vo.getSeq()}; return jdbcTemplate.queryForObject(BOARD_GET, args, new BoardRowMapper()); }

- 검색 결과를 특정 VO(Value Object) 객체에 매팅하여 리턴하려면 RowMapper 인터페이스를 구현한 RowMapper 클래스가 필요
- 따라서 RowMapper 클래스는 테이블당 하나씩 필요함
- RowMapper 인터페이스는 mapRow() 메소드가 있어서 검색 결과로 얻어낸 Row 정보를 어떤 VO에 어떻게 매팅할 것인지 구현하면 된다.

RowMapper 인터페이스 구현 클래스 만들기 예제

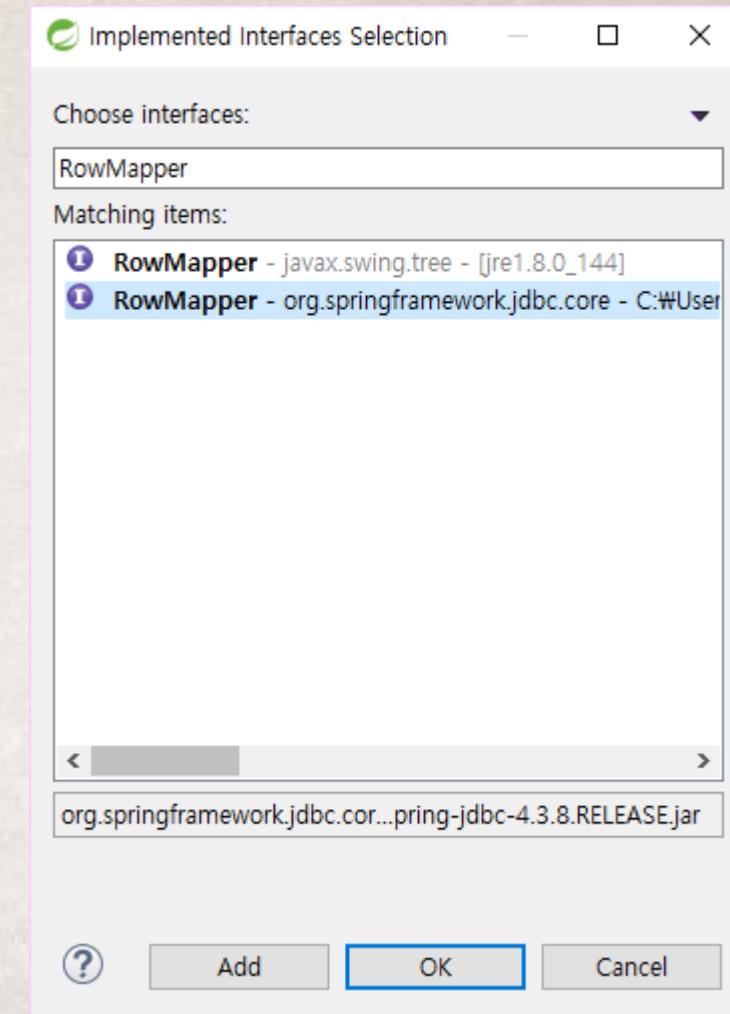
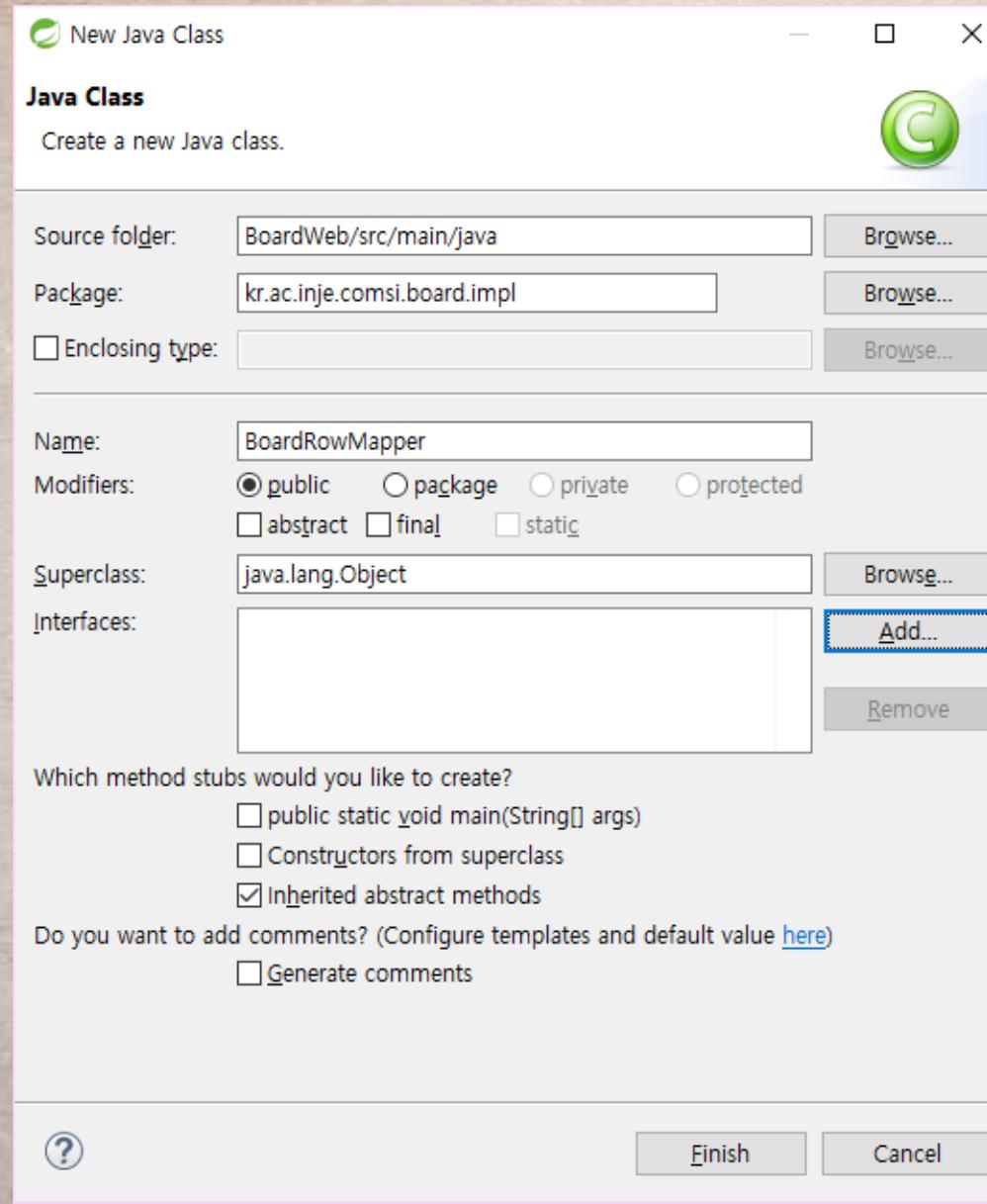
```
1 /*
2  * This build file was generated by the Gradle 'init' task.
3  *
4  * This generated file contains a sample Java project to get you started.
5  * For more details take a look at the Java Quickstart chapter in the Gradle
6  * user guide available at https://docs.gradle.org/3.3/userguide/tutorial\_java\_projects.html
7 */
8
9 // Apply the java plugin to add support for Java
10 apply plugin: 'application'
11
12 mainClassName = System.getProperty("mainClass");
13
14 compileJava.options.encoding = 'UTF-8'
15
16 // In this section you declare where to find the dependencies of your project
17 repositories {
18     mavenCentral()
19 }
20
21 dependencies {
22     compile 'org.springframework:spring-context:4.3.8.RELEASE'
23     compile 'com.h2database:h2:1.4.195'
24     compile 'org.aspectj:aspectjweaver:1.8.8'
25     compile 'org.apache.commons:commons-dbcop2:2.1'
26     compile 'org.springframework:spring-jdbc:4.3.8.RELEASE'
27 }
28
```

추가



추가

RowMapper 인터페이스 구현 클래스 만들기



New Java Class

Java Class

Source folder: BoardWeb/src/main/java

Package: kr.ac.inje.comsi.board.impl

Enclosing type:

Name: BoardRowMapper

Modifiers: public package private protected
 abstract final static

Superclass: java.lang.Object

Interfaces:

org.springframework.jdbc.core.RowMapper<T>

Which method stubs would you like to create?

public static void main(String[] args)
 Constructors from superclass
 Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

Generate comments

```
applicationC... UserDAO.java build.gradle BoardServic... BoardServic... BoardDAO.java *BoardRowMa... >
1 package kr.ac.inje.comsi.board.impl;
2
3 import org.springframework.jdbc.core.RowMapper;
4
5 public class BoardRowMapper implements RowMapper<T> {
6
7 }
8
```

```
applicationC... UserDAO.java build.gradle BoardServic... BoardServic... BoardDAO.java *BoardRowMa... >1
1 package kr.ac.inje.comsi.board.impl;
2
3 import java.sql.ResultSet;
4 import java.sql.SQLException;
5
6 import org.springframework.jdbc.core.RowMapper;
7
8 public class BoardRowMapper implements RowMapper<BoardVO> {
9
10    @Override
11    public BoardVO mapRow(ResultSet rs, int rowNum) throws SQLException {
12        // TODO Auto-generated method stub
13        return null;
14    }
15
16 }
17
```

완성된 RowMapper 클래스 - BoardRowMapper 클래스

```
applicationC... UserDAO.java build.gradle BoardServic... BoardServic... BoardDAO.java BoardRowMap... >>1
1 package kr.ac.inje.comsi.board.impl;
2
3 import java.sql.ResultSet;
4 import java.sql.SQLException;
5
6 import org.springframework.jdbc.core.RowMapper;
7
8 import kr.ac.inje.comsi.board.BoardVO;
9
10 public class BoardRowMapper implements RowMapper<BoardVO> {
11
12     @Override
13     public BoardVO mapRow(ResultSet rs, int rowNum) throws SQLException {
14
15         BoardVO board = new BoardVO();
16         board.setSeq(rs.getInt("SEQ"));
17         board.setTitle(rs.getString("TITLE"));
18         board.setWriter(rs.getString("WRITER"));
19         board.setContent(rs.getString("CONTENT"));
20         board.setRegDate(rs.getDate("REGDATE"));
21         board.setCnt(rs.getInt("CNT"));
22
23         return board;
24     }
25
26 }
27
```

- query 메소드

- queryForObject()가 SELECT문으로 객체 하나를 검색할 때 사용하는 메소드라면 query() 메소드는 SELECT문의 실행 결과가 목록일 때 사용한다.

메소드	<code>int query(String sql)</code> <code>int query(String sql, RowMapper<T> rowMapper)</code> <code>int query(String sql, Object[] args, RowMapper<T> rowMapper)</code>
사용 예	// 글 상세 조회 public List<BoardVO> getBoardList(BoardVO vo) { String BOARD_LIST = "select * from board order by seq desc"; return jdbcTemplate.query(BOARD_LIST, args, new BoardRowMapper()); }

- query() 메소드가 실행되면 여러 건의 row 정보가 검색되며, row 수 만큼 RowMapper 객체의 mapRow() 메소드가 실행
- row 정보가 매팅된 VO 객체 여러 개가 List 컬렉션에 저장되어 리턴된다.

JdbcTemplate를 이용한 DAO

- 스프링 JDBC 사용을 위한 설정이 마무리되었고
- JdbcTemplate를 이용한 DAO 클래스 구현(2가지 방법)
 - jdbcDaoSupport 클래스를 상속
 - **jdbcTemplate** 클래스를 <bean> 등록, 의존성 주입

JdbcTemplate 클래스를 <bean> 등록 및 의존성 주입

The screenshot shows the `applicationContext.xml` configuration file in a Java IDE. The file defines a Spring application context with various beans. A red box highlights the configuration for the `dataSource` and `jdbcTemplate` beans.

```
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   xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-bean.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
11 <context:component-scan base-package="kr.ac.inje.comsi"></context:component-scan>
12
13 <!-- DataSource 설정 -->
14 <bean id="dataSource" class="org.apache.commons.dbcp2.BasicDataSource" destroy-method="close">
15   <property name="driverClassName" value="org.h2.Driver"/>
16   <property name="url" value="jdbc:h2:/d:/workspace/java/h2db/myDB"/>
17   <property name="username" value="sa"/>
18   <property name="password" value="" />
19 </bean>
20 <!-- Spring JDBC 설정 -->
21 <bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
22   <property name="dataSource" ref="dataSource"/>
23 </bean>
24 </beans>
```

A red box surrounds the following code block:

```
<!-- DataSource 설정 -->
<bean id="dataSource" class="org.apache.commons.dbcp2.BasicDataSource" destroy-method="close">
  <property name="driverClassName" value="org.h2.Driver"/>
  <property name="url" value="jdbc:h2:/d:/workspace/java/h2db/myDB"/>
  <property name="username" value="sa"/>
  <property name="password" value="" />
</bean>
```

A red arrow points from the text "추가시킴" (Added) to the red box.

BoardDAOSpring 클래스 생성

```
applicationContext.xml BoardRowMapper.java BoardDAO.java BoardDAOSpring.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.jdbc.core.JdbcTemplate;
7 import org.springframework.stereotype.Repository;
8
9 import kr.ac.inje.comsi.board.BoardVO;
10
11 @Repository("boardDAOSpring")
12 public class BoardDAOSpring {
13
14     @Autowired
15     private JdbcTemplate jdbcTemplate; → applicationContext.xml에 등록된 bean
16
17     // SQL 명령어
18     private final String BOARD_INSERT = "insert into board(seq, title, writer,
19         + "content) values((select nvl(max(seq),0)+1 from board),?, ?, ?)";
20     private final String BOARD_UPDATE = "update board set title=?, content=? where seq=?";
21     private final String BOARD_DELETE = "delete board where seq=?";
22     private final String BOARD_GET = "select * from board where seq=?";
23     private final String BOARD_LIST = "select * from board order by seq desc";
24
25     // CRUD 기능의 메소드 구현
26     // 글 등록
27     public void insertBoard(BoardVO vo){
28         System.out.println("==> JDBC로 insertBoard() 기능 처리");
29         jdbcTemplate.update(BOARD_INSERT, vo.getTitle(), vo.getWriter(), vo.getContent());
30     }
31 }
```

```
32 // 글 수정
33 public void updateBoard(BoardVO vo){
34     System.out.println("==> JDBC로 updateBoard() 기능 처리");
35     jdbcTemplate.update(BOARD_UPDATE, vo.getTitle(), vo.getContent(), vo.getSeq());
36 }
37
38 // 글 삭제
39 public void deleteBoard(BoardVO vo){
40     System.out.println("==> JDBC로 deleteBoard() 기능 처리");
41     jdbcTemplate.update(BOARD_DELETE, vo.getSeq());
42 }
43
44 // 글 상세 조회
45 public BoardVO getBoard(BoardVO vo){
46     System.out.println("==> JDBC로 getBoard() 기능 처리");
47     Object[] args = {vo.getSeq()};
48     return jdbcTemplate.queryForObject(BOARD_GET, args, new BoardRowMapper());
49 }
50
51 // 글 목록 조회
52 public List<BoardVO> getBoardList(BoardVO vo){
53     System.out.println("==> JDBC로 getBoardList() 기능 처리");
54     return jdbcTemplate.query(BOARD_LIST, new BoardRowMapper());
55 }
56 }
57 }
```

BoardServiceImpl 클래스 수정

The screenshot shows a Java code editor with the tab bar containing: applicationContext.xml, BoardRowMapper.java, BoardDAO.java, BoardDAOSpring.java, and BoardServiceImpl.java. The BoardServiceImpl.java tab is active.

```
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.BoardService;
9 import kr.ac.inje.comsi.board.BoardVO;
10
11 @Service("boardService")
12 public class BoardServiceImpl implements BoardService {
13
14 // @Autowired
15 // private BoardDAO boardDAO;           ← 주석처리
16
17 @Autowired
18 private BoardDAOSpring boardDAOSpring;   ← 추가하기
19
20
21     @Override
22     public void insertBoard(BoardVO vo) {
23         boardDAOSpring.insertBoard(vo);
24     }           ← 모든 boardDAO에서 boardDAOSpring으로 변경
25
26     @Override
27     public void updateBoard(BoardVO vo) {
28         boardDAOSpring.updateBoard(vo);
29     }
}
```

Annotations and comments in the code:

- Line 14: `// @Autowired` and `// private BoardDAO boardDAO;` are both enclosed in a red box and have a red arrow pointing to the right labeled "주석처리" (commented out).
- Line 18: `@Autowired` and `private BoardDAOSpring boardDAOSpring;` are both enclosed in a red box and have a red arrow pointing to the right labeled "추가하기" (add).
- Line 24: A red arrow points from the code to the right, labeled "모든 boardDAO에서 boardDAOSpring으로 변경" (Change all boardDAO to boardDAOSpring).

```
30@override  
△31 public void deleteBoard(BoardVO vo) {  
32     boardDAOSpring.deleteBoard(vo);  
33 } ██████████  
34  
35@override  
△36 public void getBoard(BoardVO vo) {  
37     boardDAOSpring.getBoard(vo);  
38 } ██████████  
39  
40@override  
△41 public List<BoardVO> getBoardList(BoardVO vo) {  
42     return boardDAOSpring.getBoardList(vo);  
43 } ██████████  
44}  
45 |
```

실행 결과

```
Program Arguments: -DmainClass=kr.ac.inje.comsi.board.BoardServiceClient
```

```
Gradle Tasks: run
```

```
:compileJava
:processResources
:classes
:run
9월 26, 2017 10:13:08 오전 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
9월 26, 2017 10:13:08 오전 org.springframework.context.support.GenericXmlApplicationContext prepareRefresh
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@300ffa5d: startup date [Tue Sep 26 10:13:08 KST 2017]; root of context hierarchy

==> JDBC로 insertBoard() 기능 처리
==> JDBC로 getBoardList() 기능 처리
---> BoardVO [seq=15, title=JDBC테스트, writer=홍길동, content=임시 내용 ....., regDate=2017-09-26, cnt=0]
---> BoardVO [seq=14, title=JDBC테스트, writer=홍길동, content=임시 내용 ....., regDate=2017-09-25, cnt=0]
---> BoardVO [seq=13, title=JDBC테스트, writer=홍길동, content=임시 내용 ....., regDate=2017-09-25, cnt=0]
---> BoardVO [seq=12, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=11, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=10, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=9, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=8, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-13, cnt=0]
---> BoardVO [seq=7, title=AOP실행, writer=홍길동, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=6, title=AOP실행, writer=김길동2, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=5, title=AOP실행, writer=김길동2, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=4, title=AOP실행, writer=김길동, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
9월 26, 2017 10:13:08 오전 org.springframework.context.support.GenericXmlApplicationContext doClose
---> BoardVO [seq=3, title=AOP실행, writer=김길동, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@300ffa5d: startup date [Tue Sep 26 10:13:08 KST 2017]; root of context hierarchy
---> BoardVO [seq=2, title=AOP실행, writer=김길동, content=임시 내용 ....., regDate=2017-09-09, cnt=0]
---> BoardVO [seq=1, title=가입인사, writer=관리자, content=잘 부탁드립니다., regDate=2017-09-09, cnt=0]
```

```
BUILD SUCCESSFUL
```