

# 실전 프로젝트1





18

# JSP CRUD



# 공지

- 12주차 공지
  - 수업 전 LMS 동영상 시청 필수!
  - 각 동영상 별로 시청마감 기한 준수!!
  - 2차시 동영상 시청마감 : 11월 19일(금) 수업 시작 전까지 시청 완료!
- 12주차 수업 실습과제 제출 마감 : 11월 21일(일) 밤 12시
- 13주차 수업 준비
  - 13주차 1차시 수업준비 동영상 필수 시청 : 11월 23일 화요일 수업 시작 전까지 시청 완료!
- Hisnet 온라인 출석부 수시 체크
  - 수강과목 선택 > 강의정보 > 강의출석현황 메뉴에서 확인

# JSP Directives

- JSP 페이지를 **servlet 클래스**로 변환할 때 필요한 여러 정보들을 기술하기 위해 사용

```
<%@ directive attribute="value"%>
```

- **page directive**
  - jsp 전체 페이지에 적용되는 정보를 기술하기 위해 사용
- **include directive**
  - 다른 jsp or html 페이지의 소스를 가져와 현재 페이지의 일부로 포함시키기 위해 사용
- **taglib directive**
  - JSP 문법 중 Action 또는 JSTL 등 사용자 정의 태그를 사용할 때 필요
  - <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

# page directive

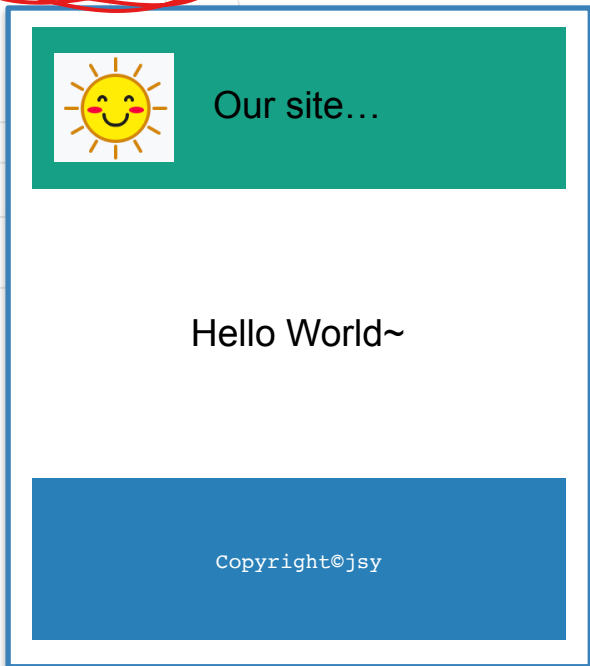
- contentType 속성

```
<%@ page language="java" contentType="text/html; charset=UTF-8"  
pageEncoding="UTF-8"%>
```

- import 속성

- ```
<%@ page import="java.util.ArrayList" %>
```
- ```
<%@ page import="java.util.*" %>
```
- ```
<%@ page import="java.util.*, java.io.*" %>
```
- ```
<%@ page contentType="text/html; charset=euc-kr" import="java.util.*" %>
```
- buffer, autoFlush, isThreadSafe, session, errorPage, isErrorPage, isELIgnored, pageEncoding ...

# include directive



main.jsp

```
<%@page contentType="text/html; charset=euc-kr" %>
<html>
  <head><title>오늘의 메뉴</title></head>
  <body>

    <%@include file="top.jsp" %>

    Hello World~

    <%@include file="footer.jsp" %>

  </body>
</html>
```

# JSP Action Tags

- **jsp:forward** 다른 페이지나 리소스로 이동할 때 사용

```
<jsp:forward page="list.jsp" />
```

- **jsp:param** 파라미터를 설정할 때 사용

```
<jsp:forward page="list.jsp">  
  <jsp:param name="name" value="ok!!" />  
</jsp:forward>
```

- **jsp:include** 다른 페이지나 리소스를 포함하여 사용

```
<jsp:include page="top.jsp" />
```

Forward

현재폴더/list.jsp? name=ok!!

include

# JSP Action Tags

자바빈즈(JavaBeans)는 자바로  
작성된 소프트웨어 컴포넌트이다.  
자바로 작성된 객체이며,  
데이터 표현을 목적으로 한다.

- `jsp:useBean`

```
<jsp:useBean id= "instanceName" scope= "page | request | session | application"  
class= "packageName.className" type= "packageName.className"  
beanName="packageName.className | <%= expression >" >  
</jsp:useBean>
```

- 예제 : Bean 생성 -> Bean 사용

```
package com.mycompany.common;  
  
public class Calculator{  
    public int sum(int n1, int n2){  
        return n1+n2;}  
}
```



```
<body>  
<jsp:useBean id= "calculator" class= "com.mycompany.common.Calculator" />  
<%  
    out.println("10 + 20 =" + calculator.sum(10, 20));  
%>  
</body>
```

10 + 20 =30



# JSP Action Tags

- **jsp:setProperty**

- `<jsp:setProperty name="bean" property="*" />`
- `<jsp:setProperty name="bean" property="username" />`
- `<jsp:setProperty name="bean" property="username" value="Kumar" />`

- **jsp:getProperty**

- `<jsp:getProperty name="obj" property="name" />`

```
package com.crud.bean;
```

```
public class User {  
    private String userid;  
    private String pwd;  
    public String getUserId() {  
        return userid;  
    }  
    public void setUserid(String userid) {  
        this.userid = userid;  
    }  
    public String getPwd() {  
        return pwd;  
    }  
    public void setPwd(String pwd) {  
        this.pwd = pwd;  
    }  
}
```

```
<body>
```

```
<jsp:useBean id="user" class="com.crud.bean.User" />
```

```
<jsp:setProperty property="userid" name="user" value="홍길동"/>
```

사용자는 `<jsp:getProperty property="userid" name="user" />`님입니다.

```
</body>
```

사용자는 홍길동님입니다.

# JSP Action Tags 예제 (1/2)

login.html

User ID:

Password:

```
<body>
<form action="login_ok.jsp" method="post">
User ID: <input type="text" name="name"> <br>
Password: <input type="password" name="password"> <br>
<input type="submit" value="login">
</form>
```

Java bean(UserInfo)

```
public class UserInfo {
    private String name;
    private String password;
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
}
```

# JSP Action Tags 예제 (2/2)

login\_ok.jsp

```
<body>
  <jsp:useBean id="user" class="com.mycompany.common.UserInfo" scope="session" />
  <jsp:setProperty property="*" name="user"/>

  Login info:<br>
  User ID : <jsp:getProperty property="name" name="user"/><br>
  Password : <jsp:getProperty property="password" name="user"/>
  <br>
  <a href='welcome.jsp' >Welcome page !</a>
</body>
```

Login info:  
User ID : guest  
Password : guest1234  
[Welcome page !](#)

welcome.jsp

```
<body>

  <jsp:useBean id="user" class="com.mycompany.common.UserInfo" scope="session" />
  [<jsp:getProperty property="name" name="user"/>]님 환영합니다.

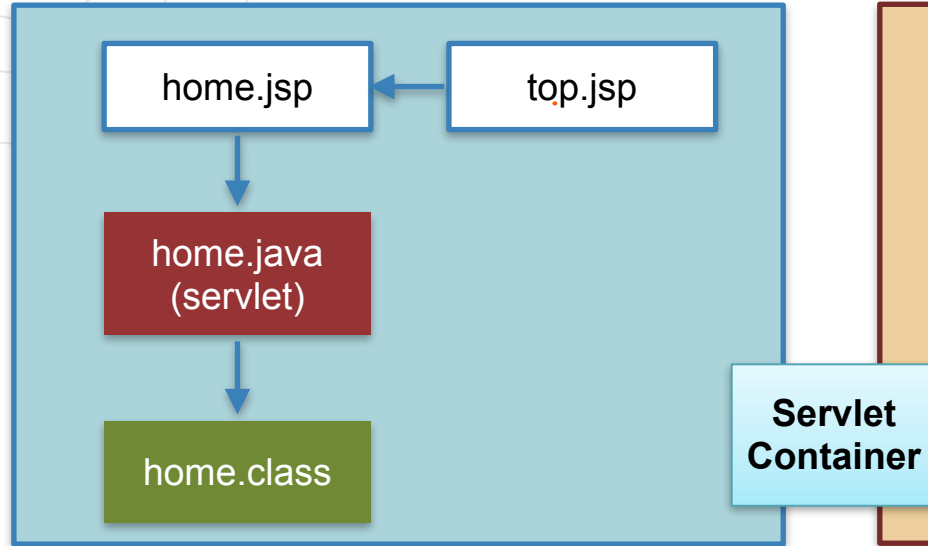
</body>
```

[guest]님 환영합니다.

# Include directive vs Include action

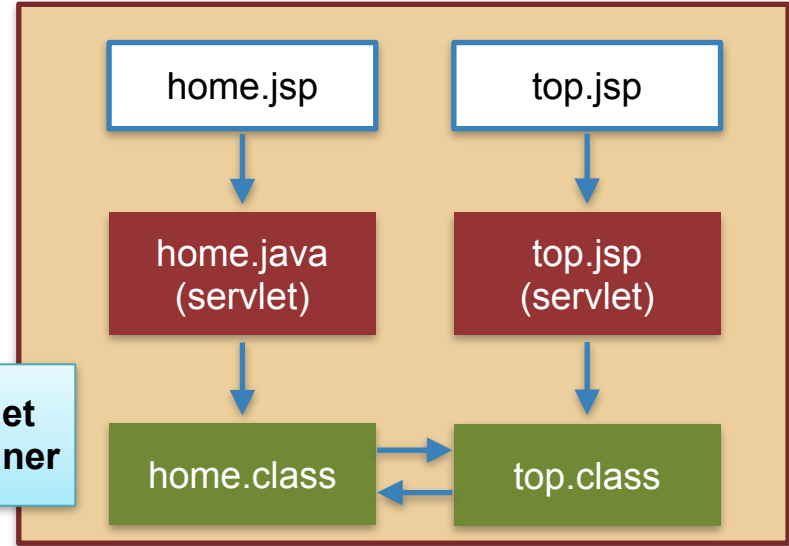
Include directive

```
<%@ include file="top.jsp" %>
```



Include action

```
<jsp:include page="top.jsp" />
```



# Expression Language(EL) in JSP

- 자바빈의 프로퍼티와 값을 JSP 표현식을 사용하는 것보다 더 간단히 사용할 수 있는 기술
- Request, Session, Application에서 값을 꺼낼 때도 사용
- 변수를 가져올 때 영역을 지정하지 않으면 순차적으로 찾아온다.
  - pageScope / requestScope / sessionScope / applicationScope

**`${변수명}`**

```
Member obj = (Member) request.getAttribute("member");  
int value = obj.getNo();
```

`${sessionScope.user}`

`${requestScope.member.no}`

```
<% String userid = request.getParameter("userid");%>  
user id(getParameter) : <%=userid %><br />  
user id(el) : ${param.userid} <br/>
```

- EL Operator 제공
  - [], (), 산술연산, 관계연산, 조건연산, 조건(a? b:c), 논리연산(&&, ||), empty

# EL 예제



날씨 search



검색 키워드는 [날씨]입니다.

기존방식

```
<form action='form_ok.jsp'>
  <input type='text' name="keyword" placeholder="검색 키워드 입력" />
  <input type='submit' value='search'>
</form>
```

```
<body>
```

```
<%
```

```
    String keyword = request.getParameter("keyword");
```

```
%>
```

```
검색 키워드는 [<%=keyword%>]입니다.
```

```
</body>
```

EL 사용

```
<body>
```

```
검색 키워드는 [${param.keyword}]입니다.
```

```
</body>
```

# CRUD methods

- 대부분의 소프트웨어가 가지는 기본적인 데이터 처리기능
- C(Create), R(Read), U(Update), D>Delete) 기능 메소드
- Database or File 을 이용하여 데이터를 저장함



# CRUD 게시판 서비스 제작

1. Database 선택 및 설치 : MySQL 사용
2. 테이블 생성 및 쿼리 생성
3. 새 프로젝트 생성 : dynamic web project
  - Maven Project로 변환 : 프로젝트 관리 도구 사용
4. Maven(pom.xml) 라이브러리 추가 : MySQL Connector. JSTL
5. JDBC(Java database Connectivity)를 사용한 클래스 추가
6. 웹페이지 제작
  - 목록페이지 : list.jsp
  - 새 글 폼 페이지 : add.html
  - 새 데이터 추가 처리 페이지 : add\_ok.jsp
  - 수정 폼 페이지 : edit.html
  - 수정 데이터 처리 페이지 : edit\_ok.jsp
  - 삭제 페이지 : delete\_ok.jsp



**Maven**<sup>TM</sup>



**MySQL Connector/J**

JDBC Type 4 driver for MySQL



# 1.MySQL 연습

- DB 서버에서 부여받은 계정을 사용해야 함.
- 무료 연습 사이트 : <https://www.w3schools.com/sql/>
- Try it Yourself >>

The screenshot shows the W3Schools website interface. At the top, there is a navigation bar with links for Tutorials, References, Exercises, Videos, Website (highlighted in pink), and Paid Courses. Below this is a dark navigation bar with links for HTML, CSS, JAVASCRIPT, SQL (highlighted in green), PYTHON, PHP, BOOTSTRAP, HOW TO, and W3.CSS. On the left side, there is a sidebar menu for the SQL Tutorial, listing links for SQL HOME, SQL Intro, SQL Syntax, SQL Select, and SQL Select Distinct. The main content area is titled 'Example' and displays a SQL query: `SELECT * FROM Customers;`. Below the query is a green button labeled 'Try it Yourself »'. The W3Schools logo is visible in the top left corner of the website interface.

## 2.테이블 및 쿼리 연습

- BOARD 테이블 생성

```
create table BOARD (  
    seq int auto_increment,  
    title varchar(100),  
    writer varchar(20),  
    content varchar(1000),  
    regdate timestamp default current_timestamp,  
    cnt int default 0,  
    primary key(seq)  
);
```

- 데이터 추가

- insert into BOARD (title,writer,content) values ('제목','홍길동','내용');

- 데이터 수정 update BOARD set title='제목2', content='내용' where seq=10

- 데이터 삭제 delete from BOARD where seq=10

- 데이터 목록

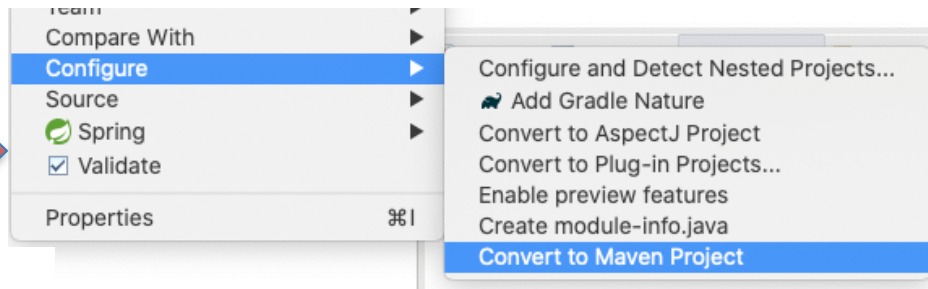
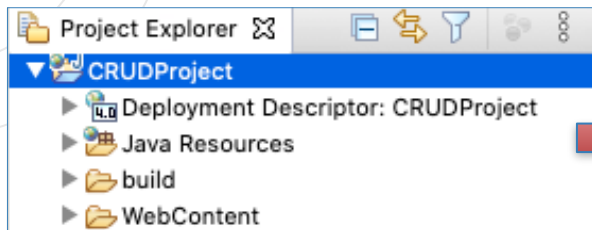
- select \* from BOARD order by regdate desc

- select \* from BOARD where seq=10

- select title, content from BOARD where seq=10

# 3.새 프로젝트(Maven)

- Dynamic web project 새 프로젝트 생성
- Maven project로 변환



## Maven POM

This wizard creates a new POM (pom.xml) descriptor for Maven.



Project: /CRUDProject

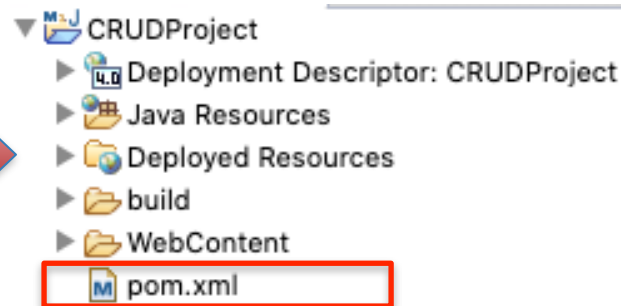
Artifact

Group Id: CRUDProject

Artifact Id: CRUDProject

Version: 0.0.1-SNAPSHOT

Packaging: war



# 4.라이브러리 추가1 : MySQL Connector

- Maven mysql 검색
- MVNRepository에서 MySQL Connector버전 선택 및 복사
- pom.xml 에 붙여넣기

**MySQL Connector/J » 8.0.19**  
JDBC Type 4 driver for MySQL

License: [GPL 2.0](#)

Categories: [MySQL Drivers](#)

Organization: [Oracle Corporation](#)

HomePage: <http://dev.mysql.com/doc/connector-j/en/>

Date: (Dec 04, 2019)

Files: [jar \(2.2 MB\)](#) | [View All](#)

Repositories: [Central](#) | [Spring Plugins](#)

Used By: [4,740 artifacts](#)

Note: There is a new version for this artifact

New Version: [8.0.21](#)

[Maven](#) | [Gradle](#) | [SBT](#) | [Ivy](#) | [Graco](#) | [Leiningen](#) | [Buildr](#)

```
<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>8.0.19</version>
</dependency>
```

CRUDProject/pom.xml

```
27 </build>
28 <dependencies>
29 <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->
30 <dependency>
31 <groupId>mysql</groupId>
32 <artifactId>mysql-connector-java</artifactId>
33 <version>8.0.19</version>
34 </dependency>
35 </dependencies>
```

Java Resources

- src
- Libraries
  - Apache Tomcat v9.0 [Apache Tomca
  - JRE System Library [JavaSE-14]
- Maven Dependencies
  - mysql-connector-java-8.0.19.jar**

## 4. 라이브러리 추가2 : JSTL

- JSTL (Java standard tag library)
- MVNRepository에서 JSTL 선택 및 복사
- pom.xml 에 붙여넣기



**JSTL » 1.2**

JSTL

Date	(May 12, 2006)
Files	<a href="#">pom (141 bytes)</a> <a href="#">jar (404 KB)</a> <a href="#">View All</a>
Repositories	<a href="#">Central</a> <a href="#">SciJava Public</a> <a href="#">Spring Lib Release</a>
Used By	<b>254 artifacts</b>

Maven

Gradle

SBT

Ivy

Grape

Leiningen

Builder

```
<!-- https://mvnrepository.com/artifact/jstl/jstl -->
<dependency>
  <groupId>jstl</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```

CRUDProject/pom.xml

```
<!-- https://mvnrepository.com/artifact/jstl/jstl -->
<dependency>
  <groupId>jstl</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```



Maven Dependencies

- mysql-connector-java-8.0.19.jar -
- protobuf-java-3.6.1.jar - /Users/jen
- jstl-1.2.jar - /Users/jerry1004/.m2/

# 4. JSTL 예제

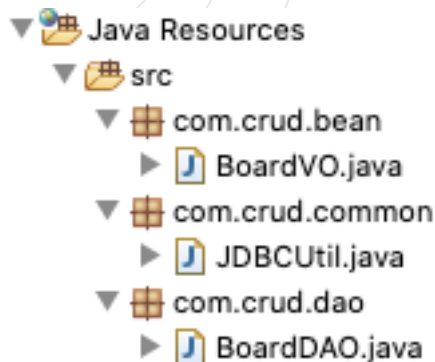
## JSTL (Java standard tag library)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@page import="com.crud.dao.BoardDAO, com.crud.bean.BoardVO,java.util.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

    <%
        BoardDAO boardDAO = new BoardDAO();
        List<BoardVO> list = boardDAO.getBoardList();
        request.setAttribute("list",list);
    %>
    <c:forEach items="${list}" var="u">
        <tr>
            <td>${u.getSeq()}</td>
            <td>${u.getTitle()}</td>
            <td>${u.getWriter()}</td>
            <td>${u.getContent()}</td>
            <td>${u.getRegdate()}</td>
            <td><a href="editform.jsp?id=${u.getSeq()}">Edit</a></td>
            <td><a href="javascript:delete_ok('${u.getSeq()}')">Delete</a></td>
        </tr>
    </c:forEach>
```

# 5. JDBCUtil class 생성

- MySQL 접속 기능을 클래스의 함수로 제작 후 MySQL 연결테스트
- 웹 페이지에서 사용



```
public class JDBCUtil {  
    public static Connection getConnection(){  
        Connection con=null;  
        try{  
            Class.forName("com.mysql.jdbc.Driver");  
            con= DriverManager.getConnection("jdbc:mysql://db4free.net:3306/  
dbname","dbid","dbpwd");  
        }catch(Exception e){  
            System.out.println(e);  
        }  
        return con;  
    }  
}
```

Loading class `com.mysql.jdbc.Driver`. This is deprecated.  
DB 연결됨!

```
// public static void main(String ars[]) {  
//     Connection conn = getConnection();  
//     if(conn != null)  
//         System.out.println("DB 연결됨!");  
//     else  
//         System.out.println("DB 연결중 오류 !");  
// }  
}
```

## 6. Java bean 생성 : BoardVO class

- 테이블에 들어가는 한 레코드를 구성하는 필드를 멤버변수로 bean 생성
- Getters 와 Setters 생성

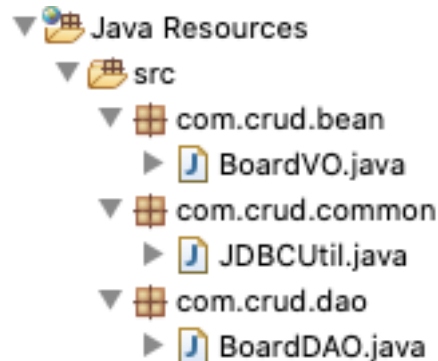
```
create table BOARD (  
    seq int AUTO_INCREMENT,  
    title varchar(100),  
    writer varchar(20),  
    content varchar(1000),  
    regdate timestamp default current_timestamp,  
    cnt int default 0,  
    primary key(seq)  
);
```

```
package com.crud.bean;
```

```
import java.util.Date;
```

```
public class BoardVO {  
    private int seq;  
    private String title;  
    private String writer;  
    private String content;  
    private Date regdate;  
    private int cnt;
```

```
    public int getSeq() {  
        return seq;  
    }  
    public void setSeq(int seq) {  
        this.seq = seq;  
    }  
    public String getTitle() {  
        return title;  
    }  
}
```





# 7.BoardDAO class 생성

- MySQL DB에서 실행하기 위한 CRUD 쿼리 구현

```
Connection conn = null;  
PreparedStatement stmt = null;  
ResultSet rs = null;
```

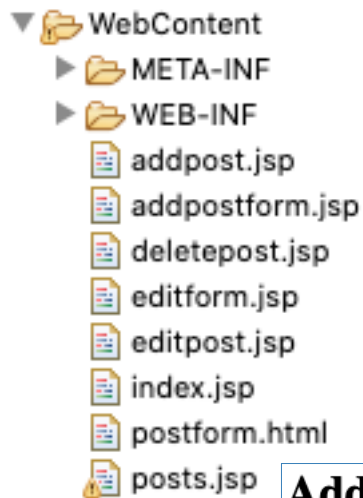
```
private final String BOARD_INSERT = "insert into BOARD (title, writer, content) values (?, ?, ?)";  
private final String BOARD_UPDATE = "update BOARD set title=?, writer=?, content=? where seq=?";  
private final String BOARD_DELETE = "delete from BOARD where seq=?";  
private final String BOARD_GET = "select * from BOARD where seq=?";  
private final String BOARD_LIST = "select * from BOARD order by seq desc";
```

```
public int insertBoard(BoardVO vo)  
public void deleteBoard(BoardVO vo);  
public int updateBoard(BoardVO vo);  
public BoardVO getBoard(int seq);  
public List<BoardVO> getBoardList();
```

## 7.BoardDAO class 생성:insertBoard();

```
public int insertBoard(BoardVO vo) {  
    System.out.println("==> JDBC로 insertBoard() 기능 처리");  
    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();  
        return 1;  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
    return 0;  
}
```

# 8.JSP를 이용한 웹페이지 생성



## 자유게시판

Id	Title	Writer	Content	Regdate	Edit	Delete
7	새로운 제목	새로운 작성자	새로운 내용	2020-10-09	<a href="#">Edit</a>	<a href="#">Delete</a>
5	제목	hong	내용입니다....	2020-10-08	<a href="#">Edit</a>	<a href="#">Delete</a>
4	제목	hong	내용입니다....	2020-10-08	<a href="#">Edit</a>	<a href="#">Delete</a>
3	제목 3	hong 3	내용입니다.... 3	2020-10-05	<a href="#">Edit</a>	<a href="#">Delete</a>
2	제목2	저자2	내용2	2020-10-05	<a href="#">Edit</a>	<a href="#">Delete</a>
1	가입인사	관리자	hello~	2020-10-04	<a href="#">Edit</a>	<a href="#">Delete</a>

[Add New Post](#)

## Add New Post

Title:

Writer:

Content:

[View All Records](#)


## Edit Form

Title:

Writer:

Content:

# 9. CRUD 프로젝트 생성

- 참고 소스 다운로드 
- STS4 - 새로운 프로젝트 생성 (dynamic web project)
- Library 추가 (MySQL Connector, JSTL)
- 클래스 제작
  - JDBCUtil class 제작 및 테스트 (DB 접속 정보 변경)
    - 각 분반별 제공
  - Java bean 제작 : BoardVO class 생성 (getters & setters 생성)
  - BoardDAO class 제작
- 웹페이지 소스 다운로드 및 저장
  - webapp 폴더 아래
- Tomcat 서버에서 실행 테스트

자유게시판

Id	Title	Writer	Content	Regdate	Edit	Delete
7	새로운 제목	새로운 작성자	새로운 내용	2020-10-09	<a href="#">Edit</a>	<a href="#">Delete</a>
5	제목	hong	내용입니다....	2020-10-08	<a href="#">Edit</a>	<a href="#">Delete</a>
4	제목	hong	내용입니다....	2020-10-08	<a href="#">Edit</a>	<a href="#">Delete</a>
3	제목 3	hong 3	내용입니다.... 3	2020-10-05	<a href="#">Edit</a>	<a href="#">Delete</a>
2	제목2	저자2	내용2	2020-10-05	<a href="#">Edit</a>	<a href="#">Delete</a>
1	가입인사	관리자	hello-	2020-10-04	<a href="#">Edit</a>	<a href="#">Delete</a>

[Add New Post](#)