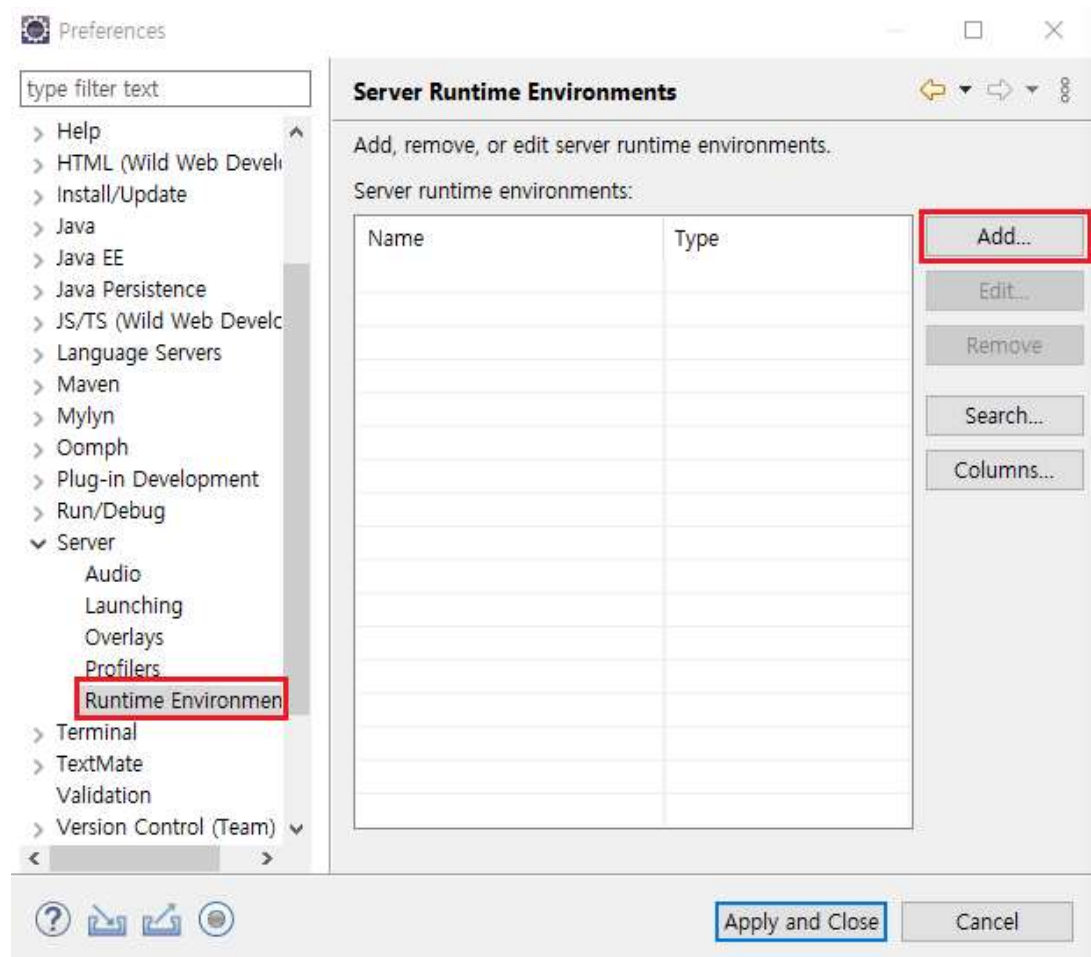
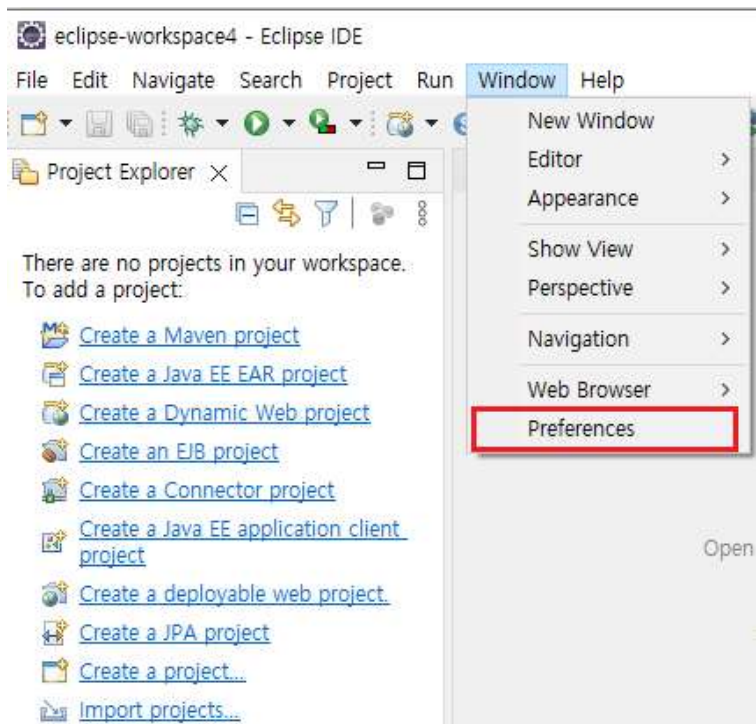
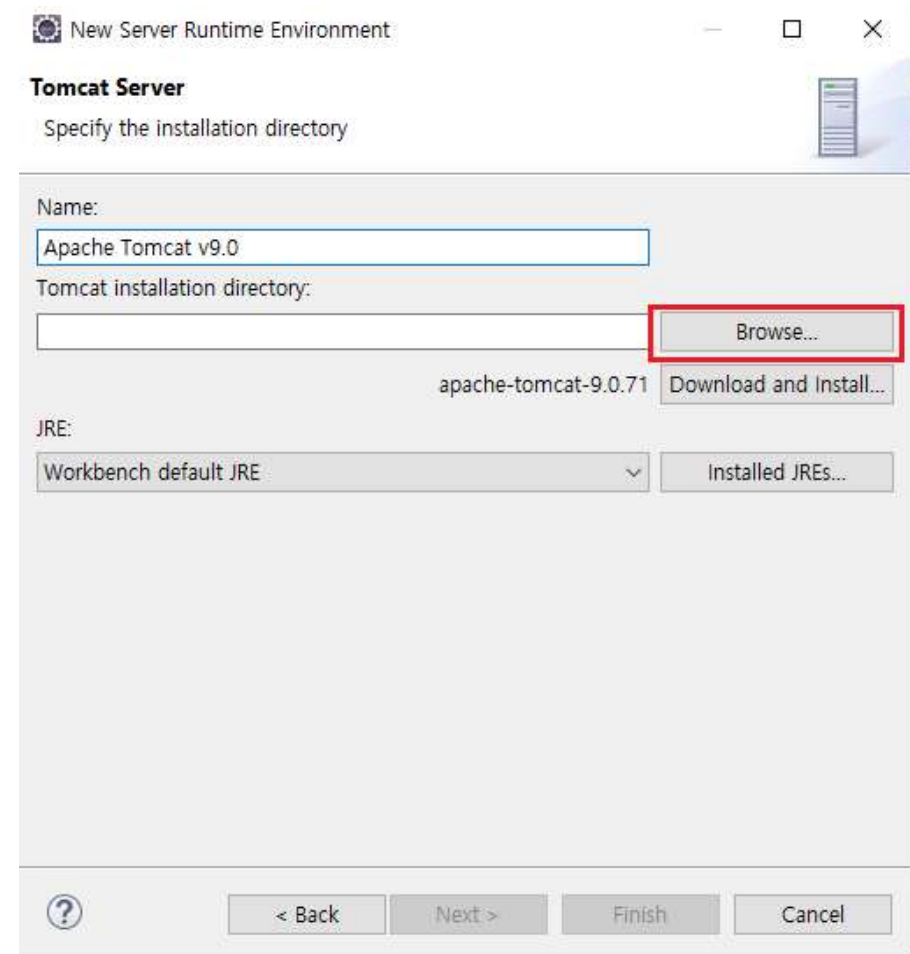
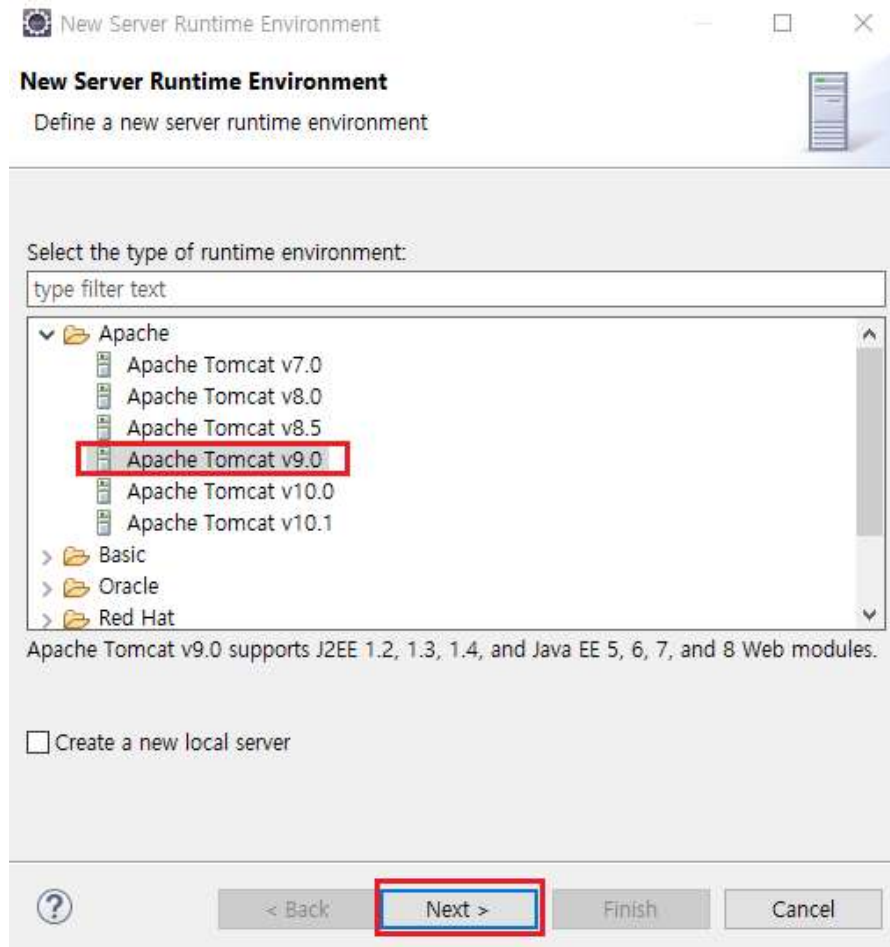
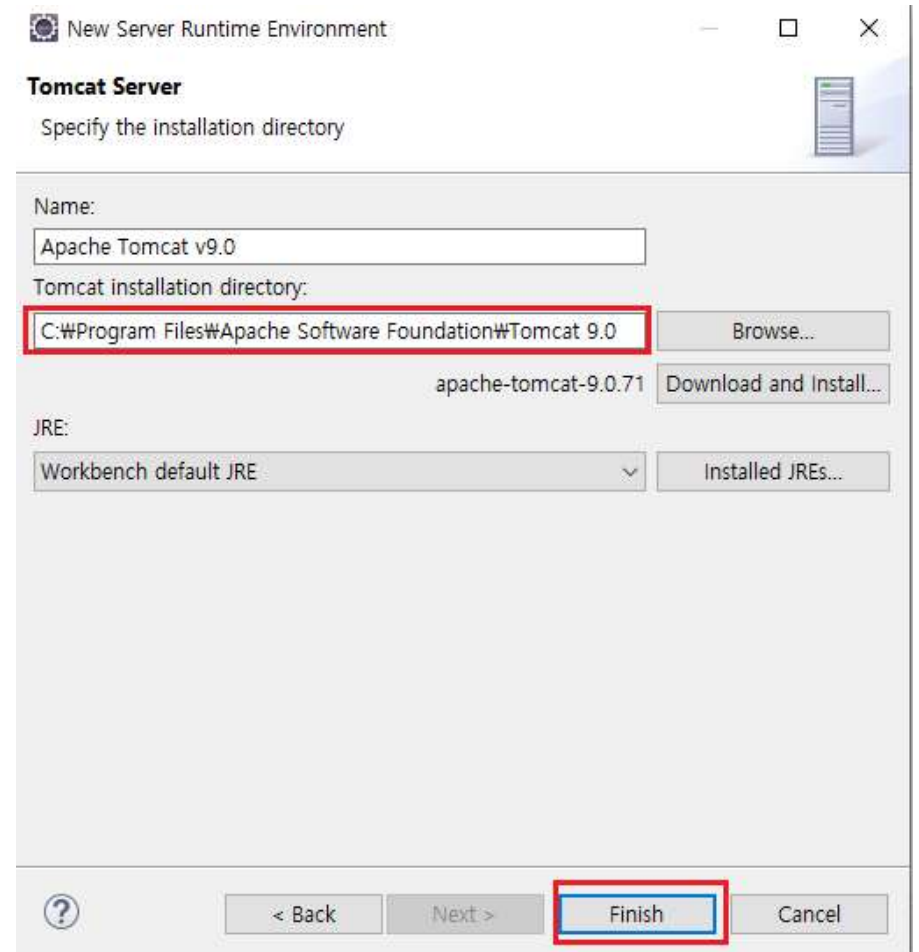
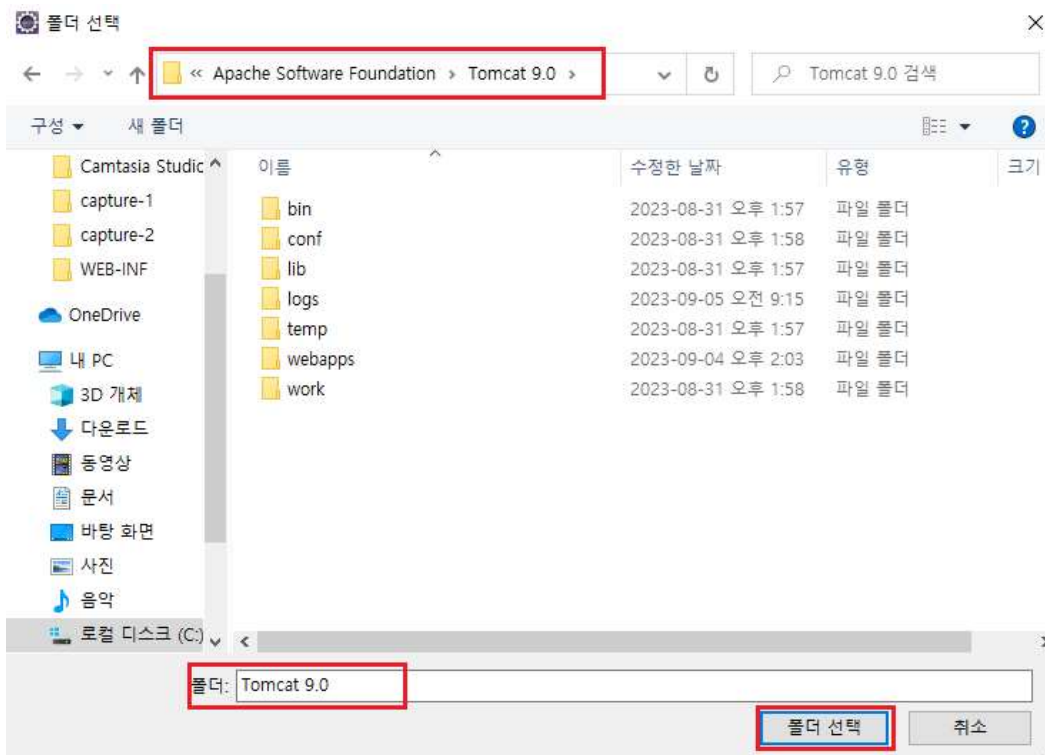
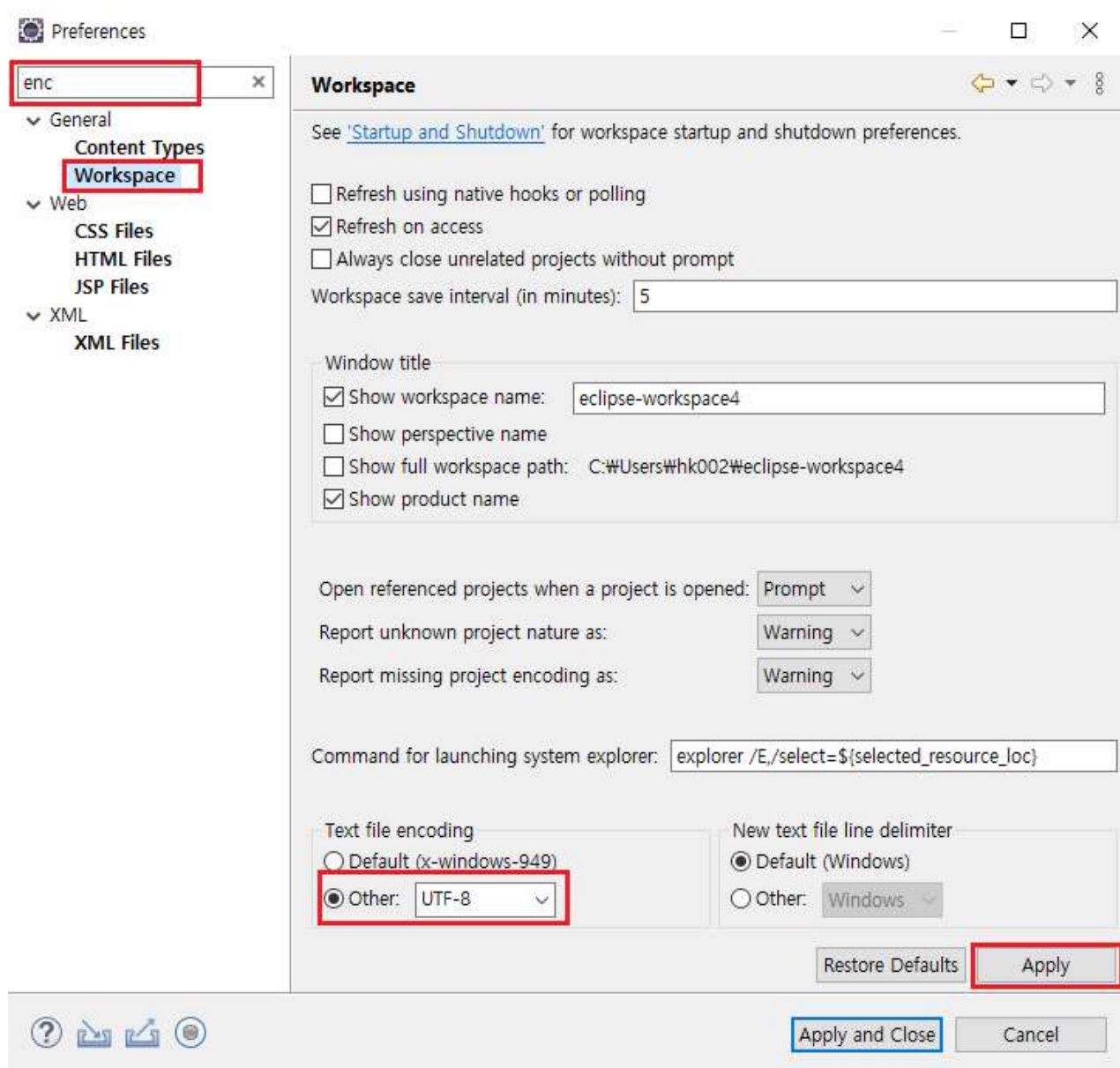
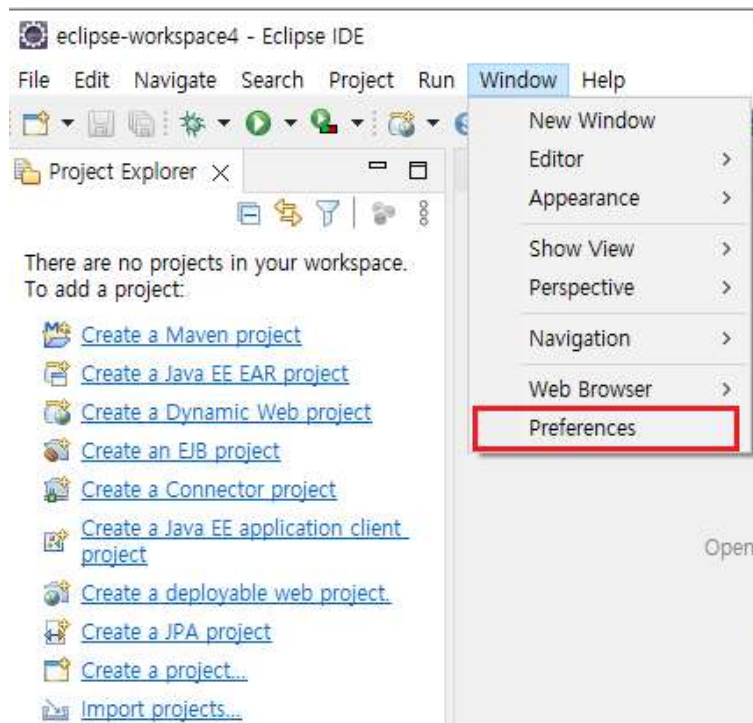


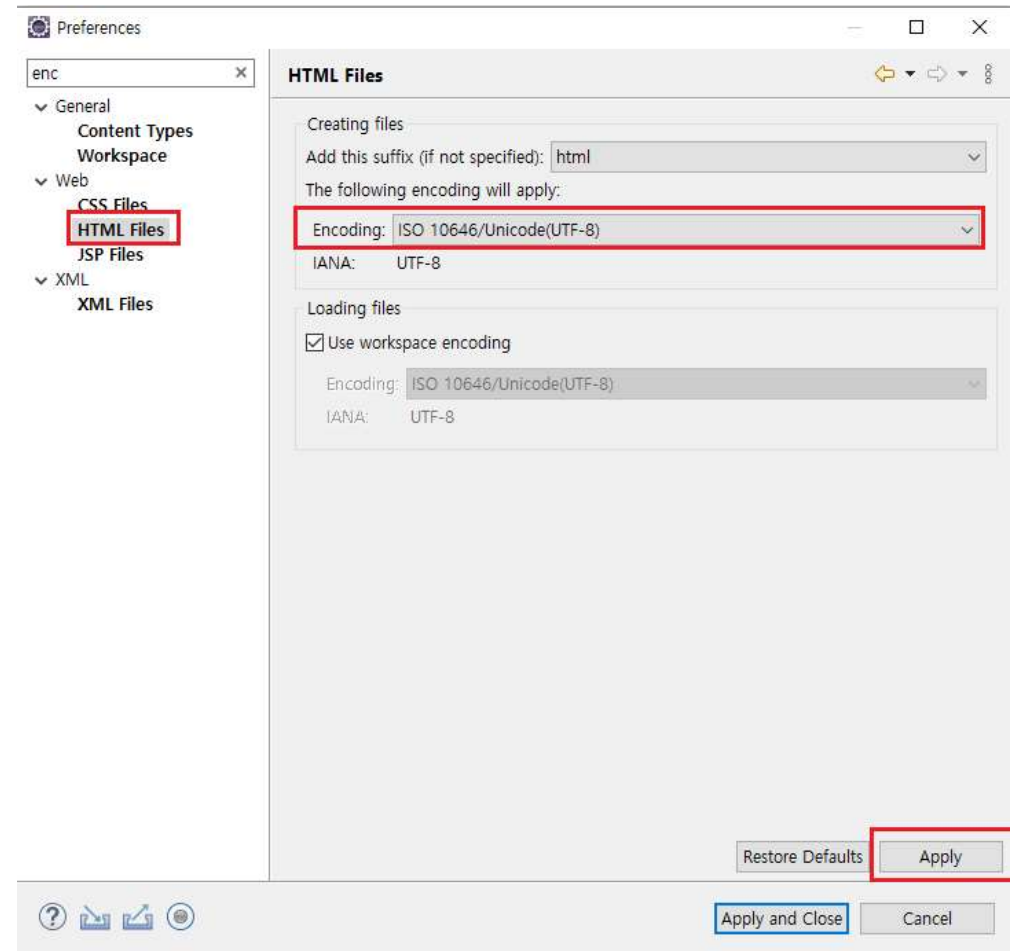
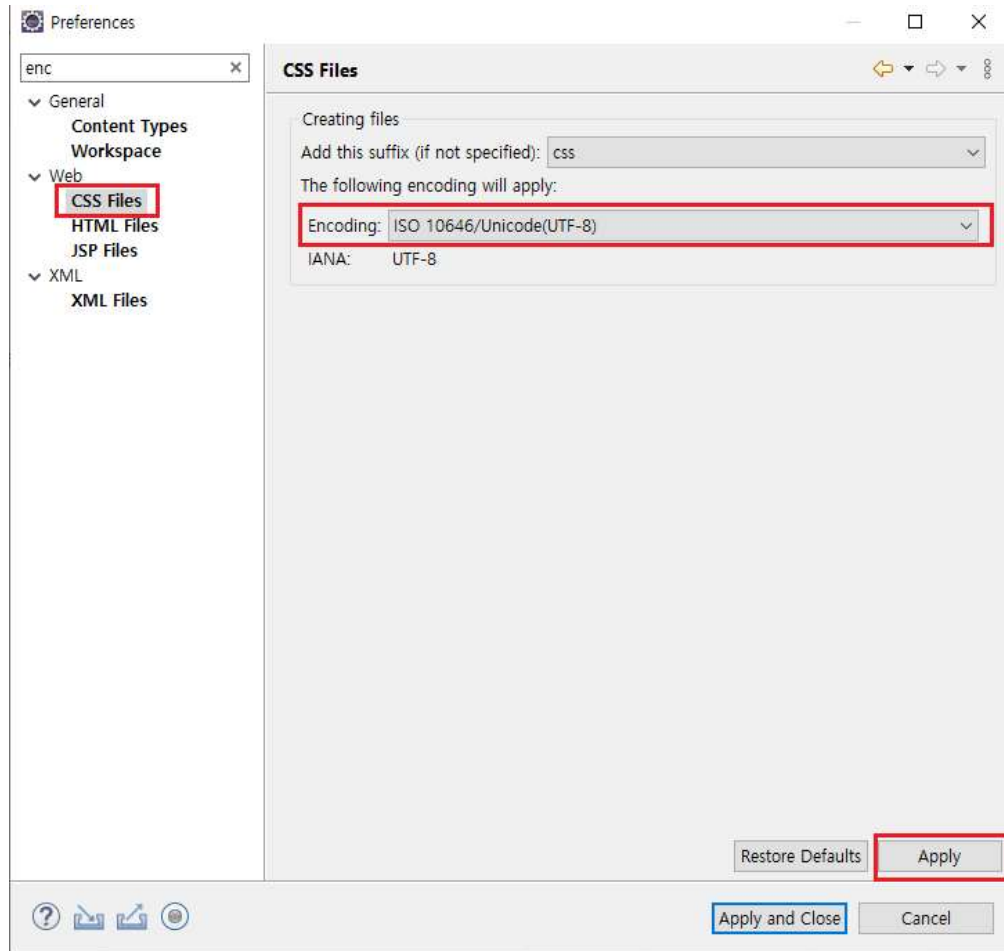
게시판 만들기

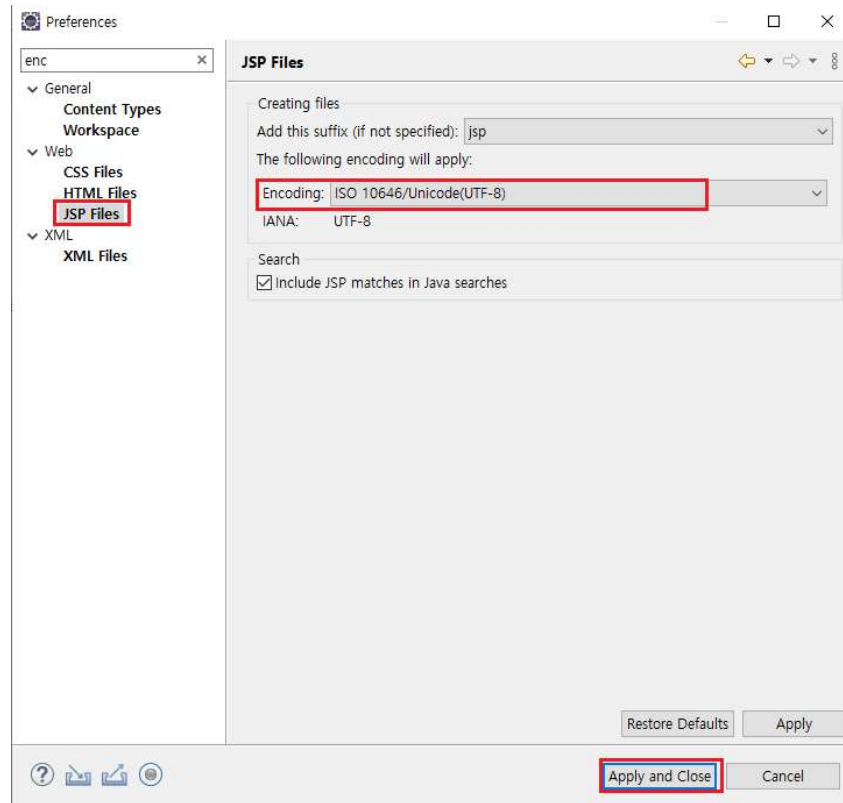


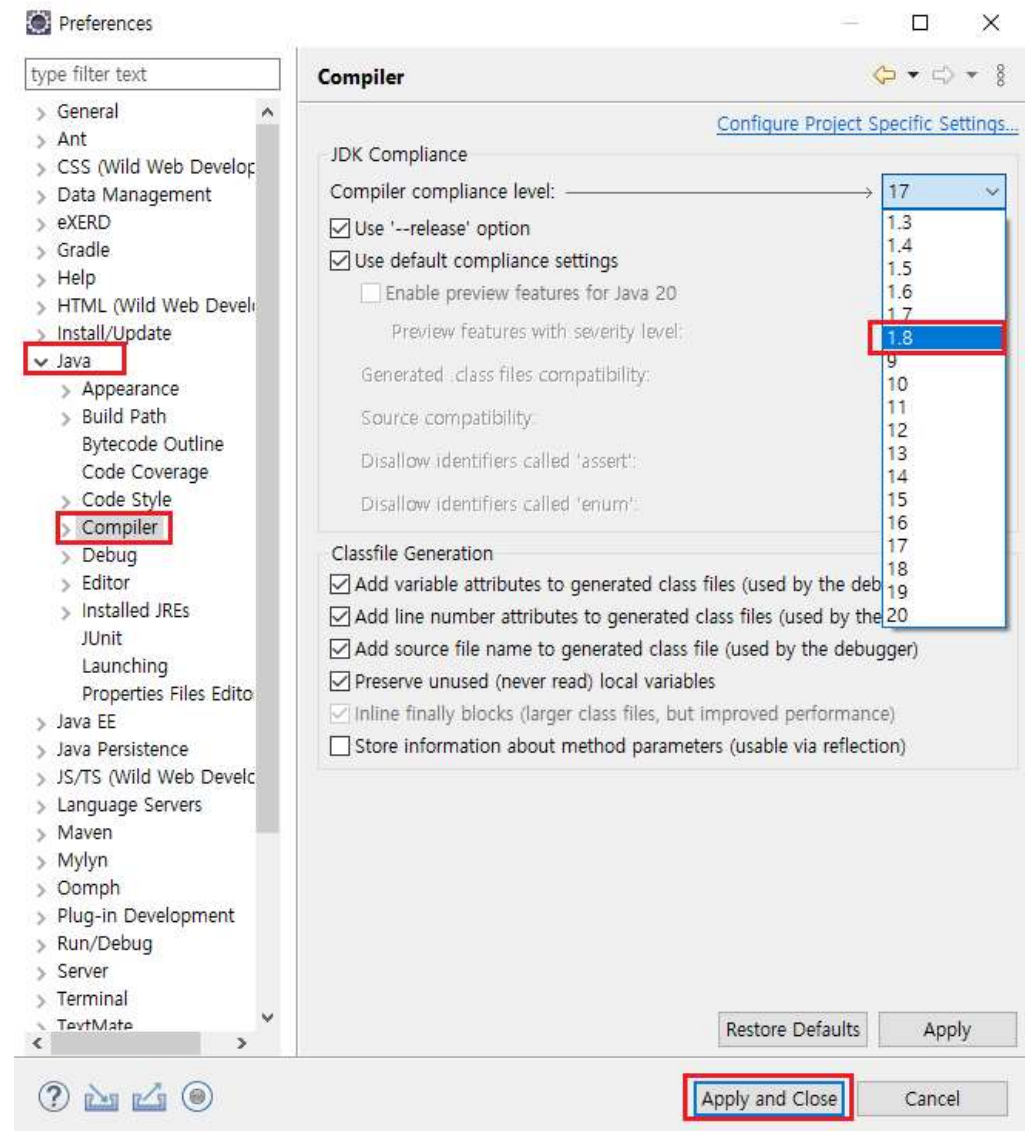
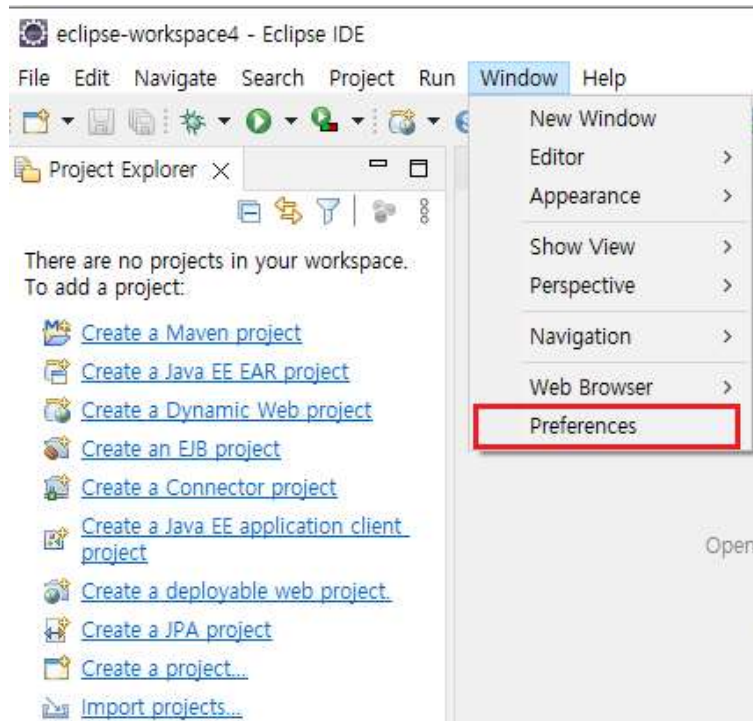




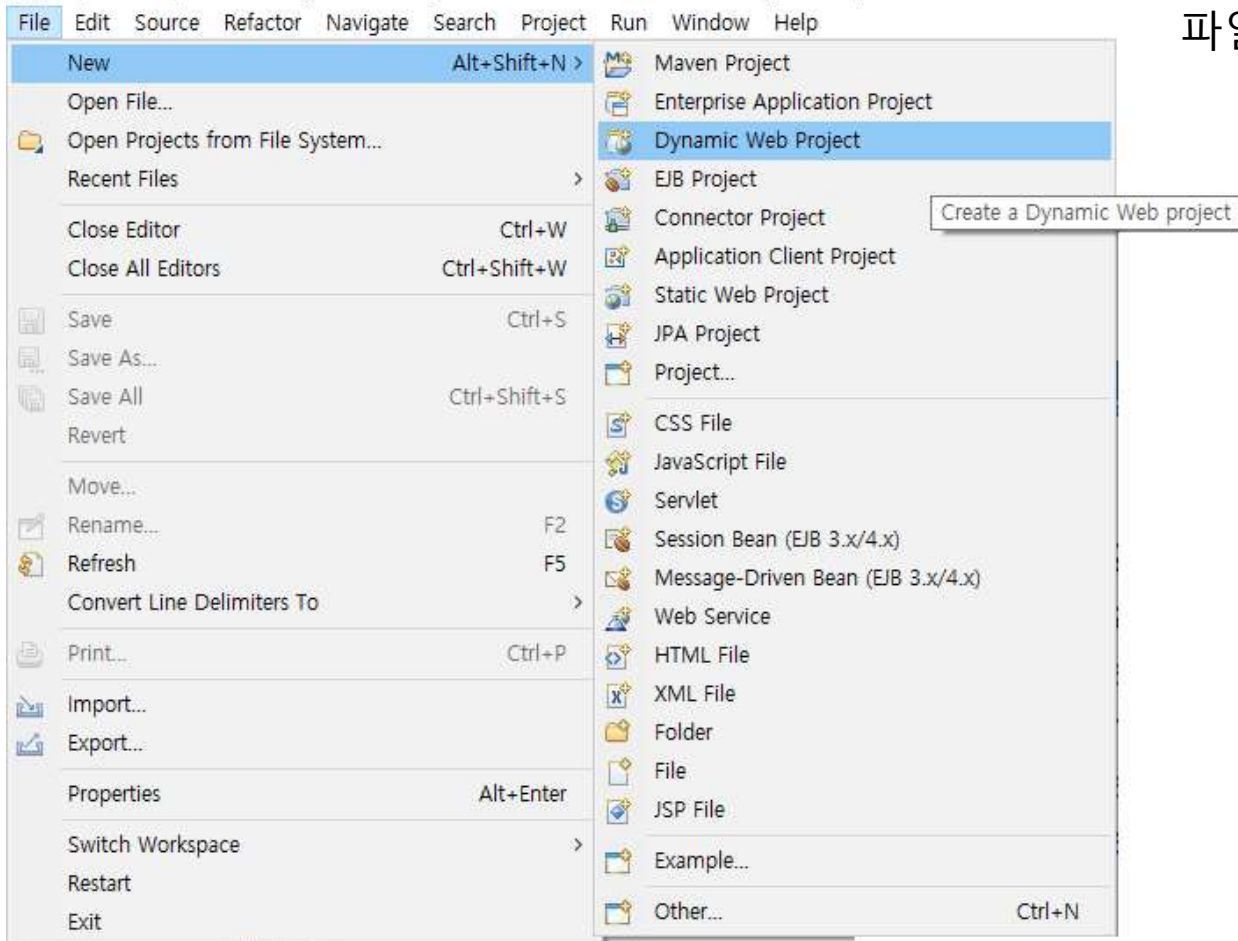








eclipse-workspace - testProject/src/main/java/controller/TestFrontController.java - Eclipse IDE



파일 => New => Dynamic Web Project를 선택

New Dynamic Web Project

Dynamic Web Project

Create a standalone Java-based Web Application or add it to a new or existing Enterprise Application.

Project name: board

Project location

☒ Use default location

Location: C:\Users\Whk002\workspace\board Browse...

Target runtime

Apache Tomcat v9.0 New Runtime...

Dynamic web module version

4.0

Configuration

Default Configuration for Apache Tomcat v9.0 Modify...

A good starting point for working with Apache Tomcat v9.0 runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

☐ Add project to an EAR

EAR project name: EAR New Project...

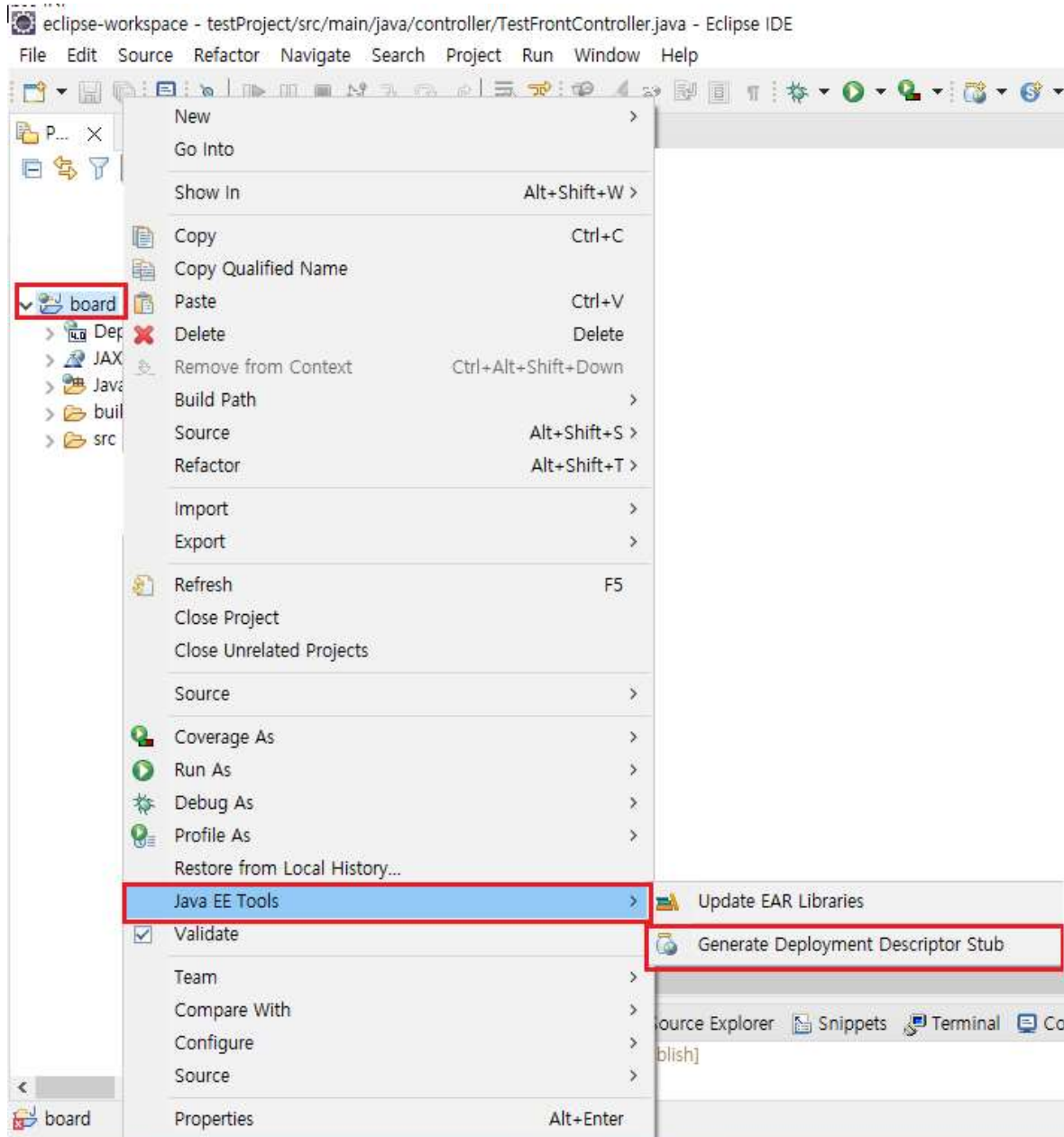
Working sets

☐ Add project to working sets New...

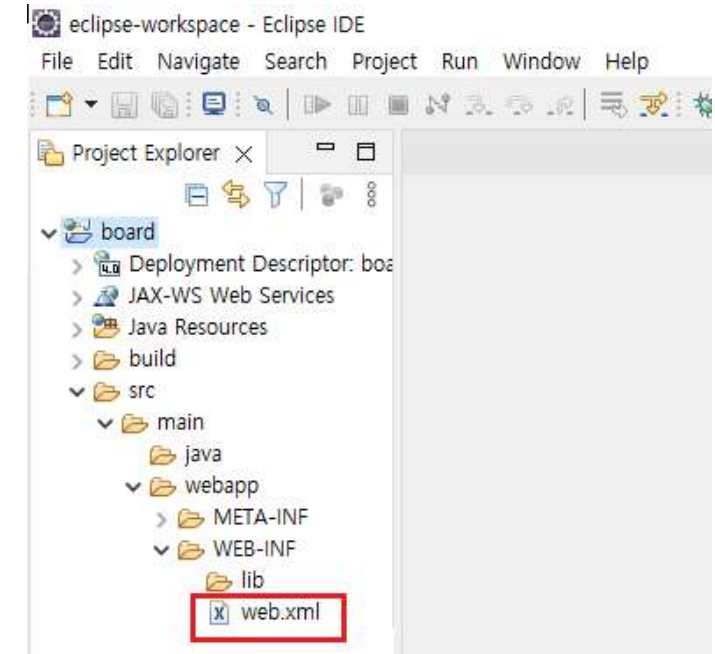
Working sets: Select...

< Back Next > Finish Cancel

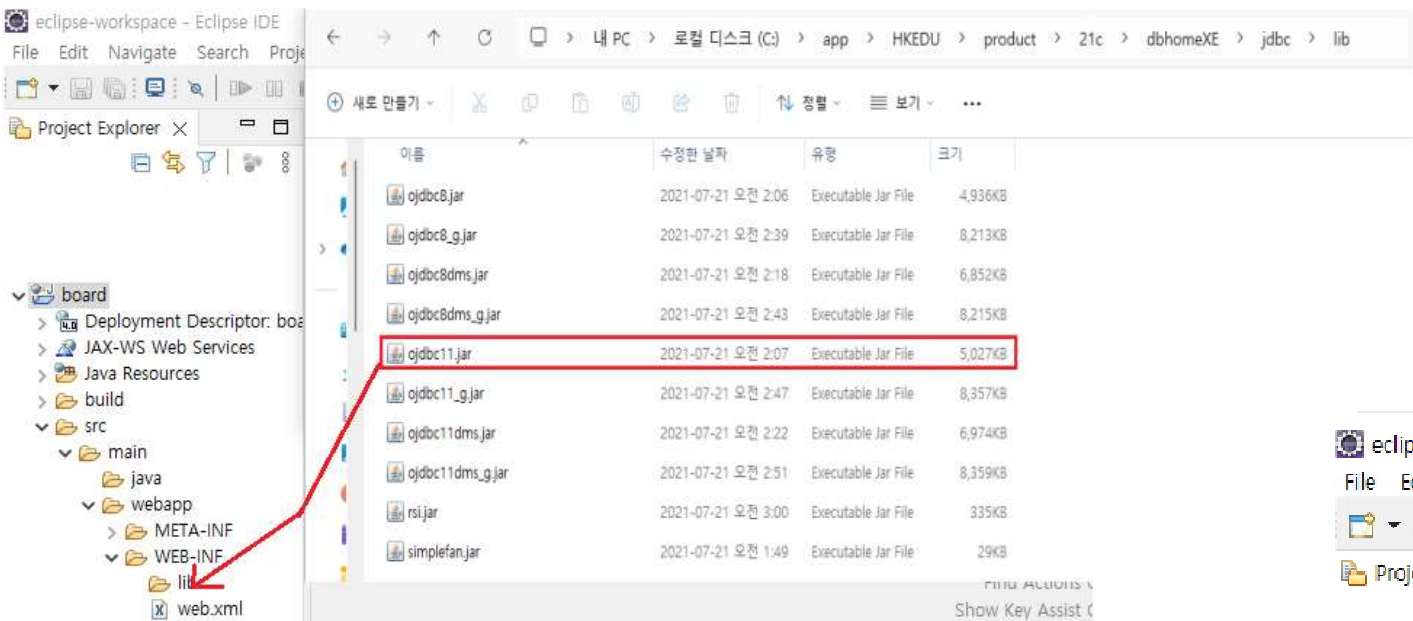
Project name에 "board"라는 이름을 주고 [Finish]를 클릭



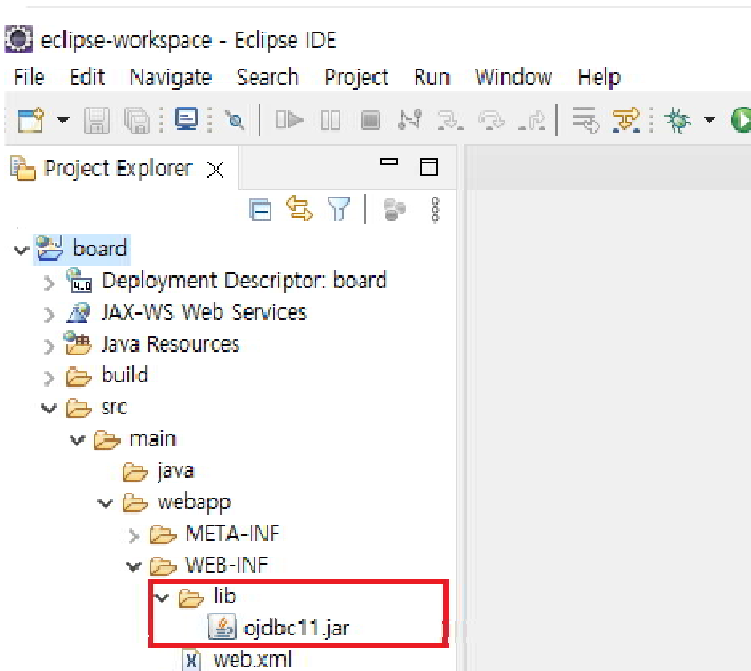
프로젝트명에서 마우스 오른쪽키를 눌러 팝업메뉴가 나오도록 한다.



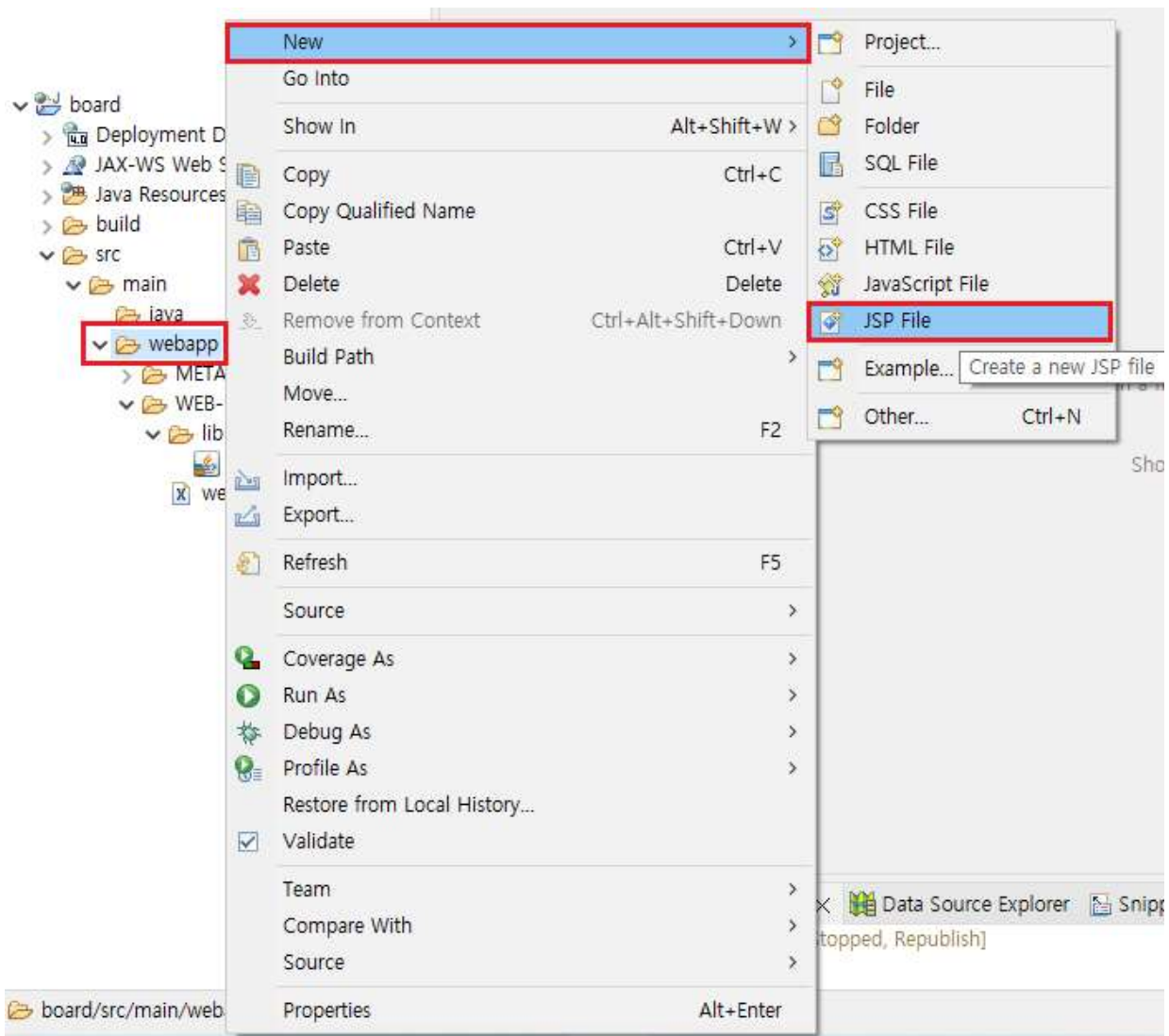
WEB-INF밑에 web.xml파일이 만들어졌는지 확인한다.

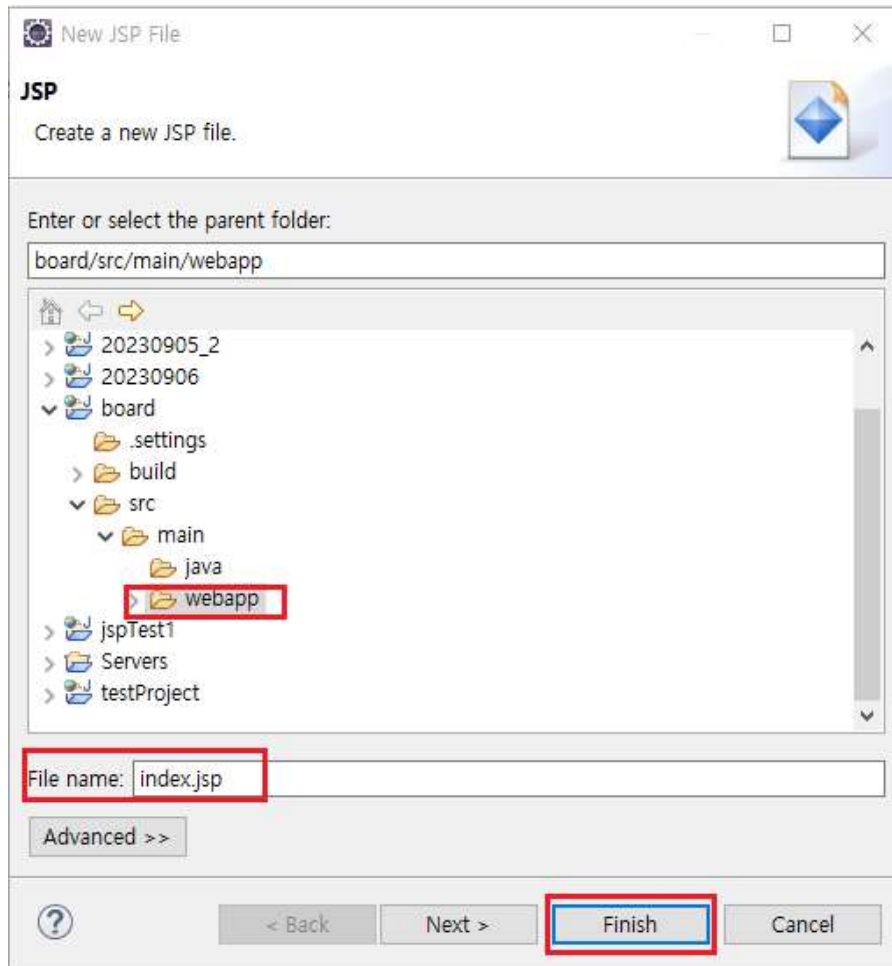


ojdbc11.jar 파일을 다운받아 WEB-INF폴더 밑에 있는 lib폴더에 드래그하여 파일을 복사한다.



ojdbc11.jar 파일이 복사되었는지 확인



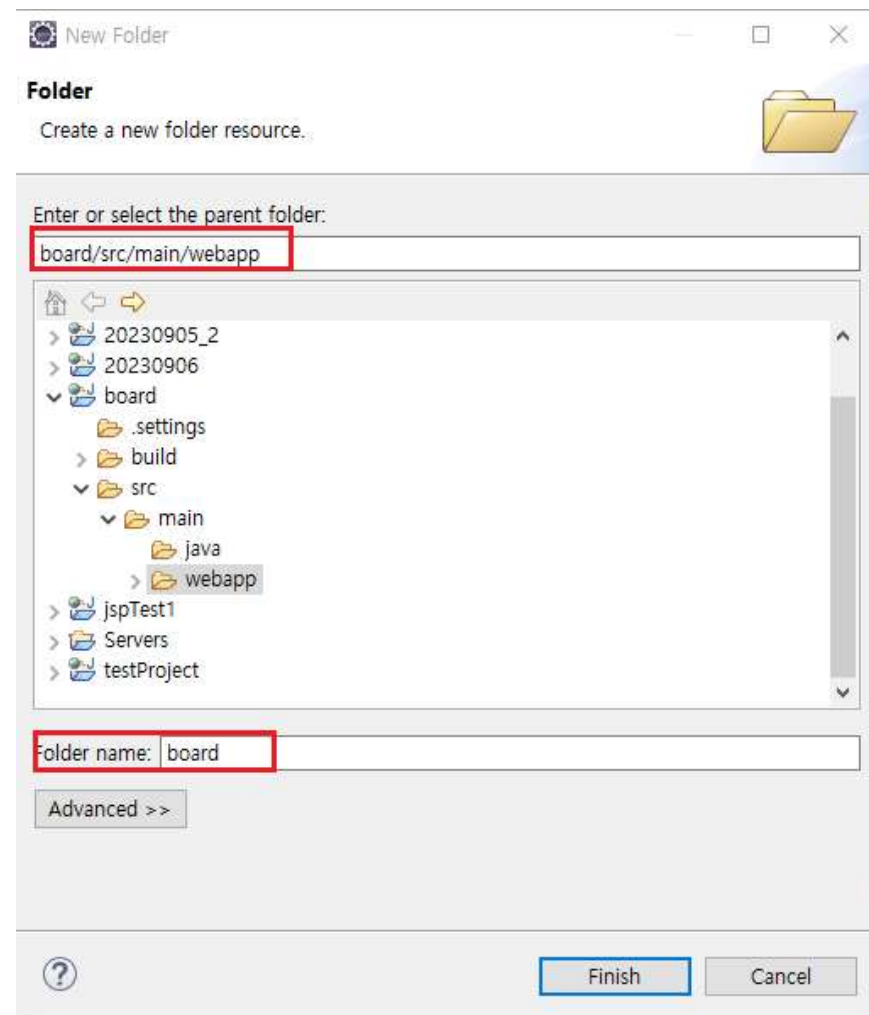
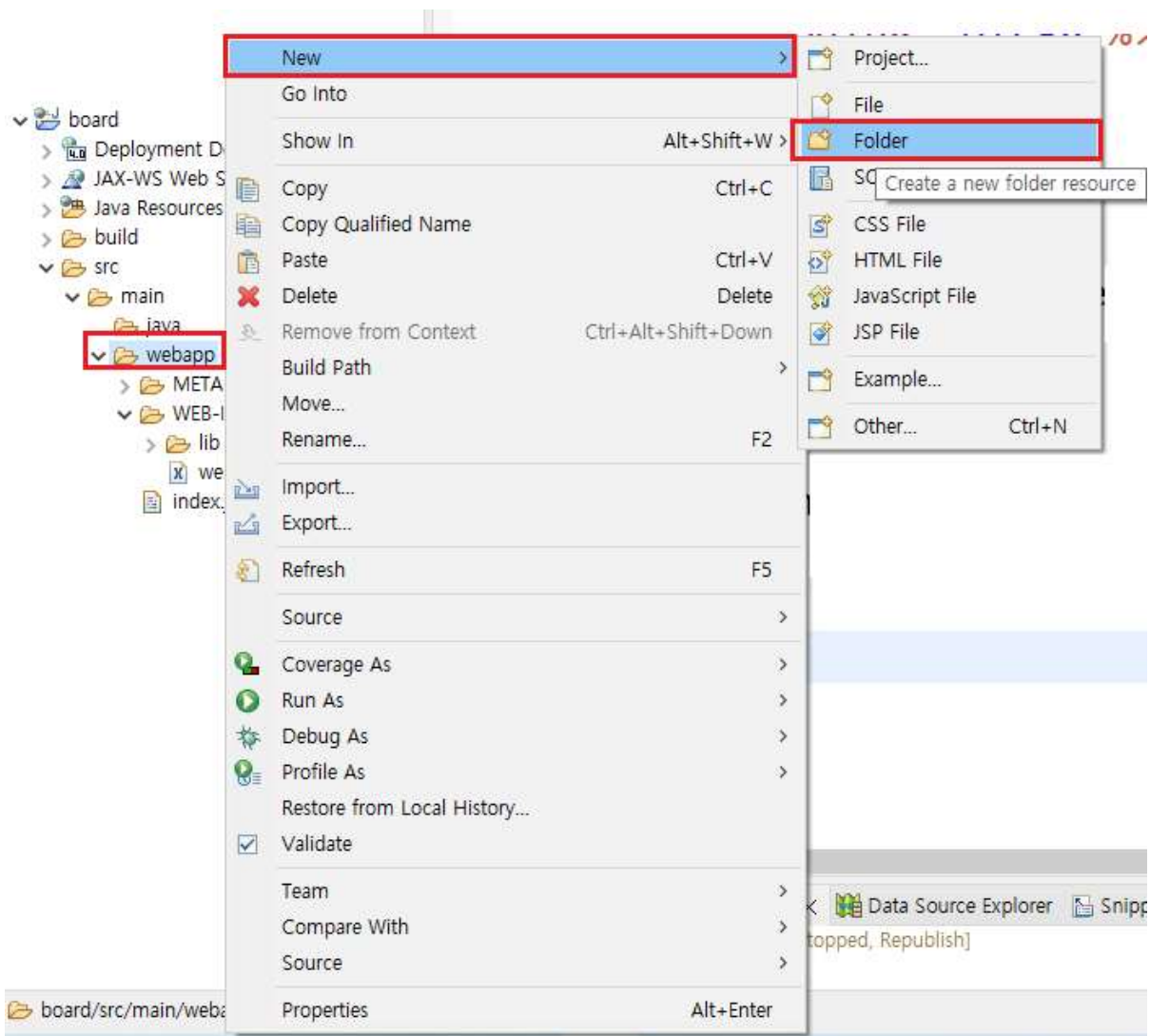


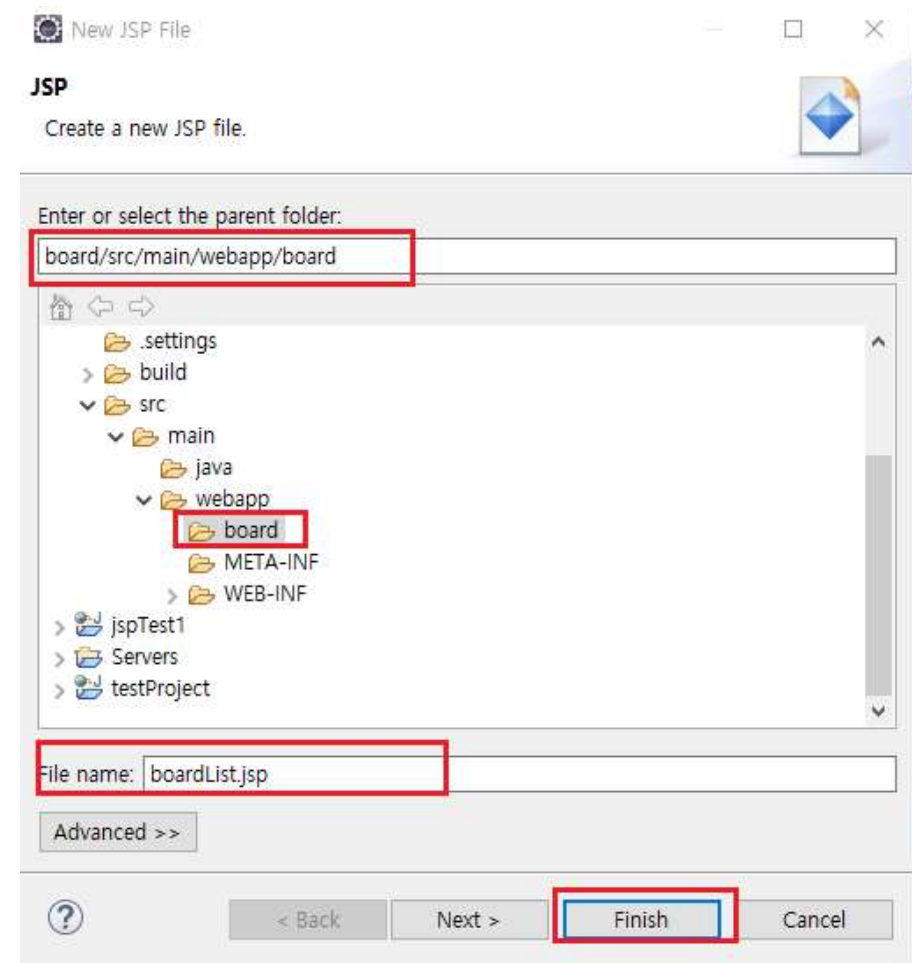
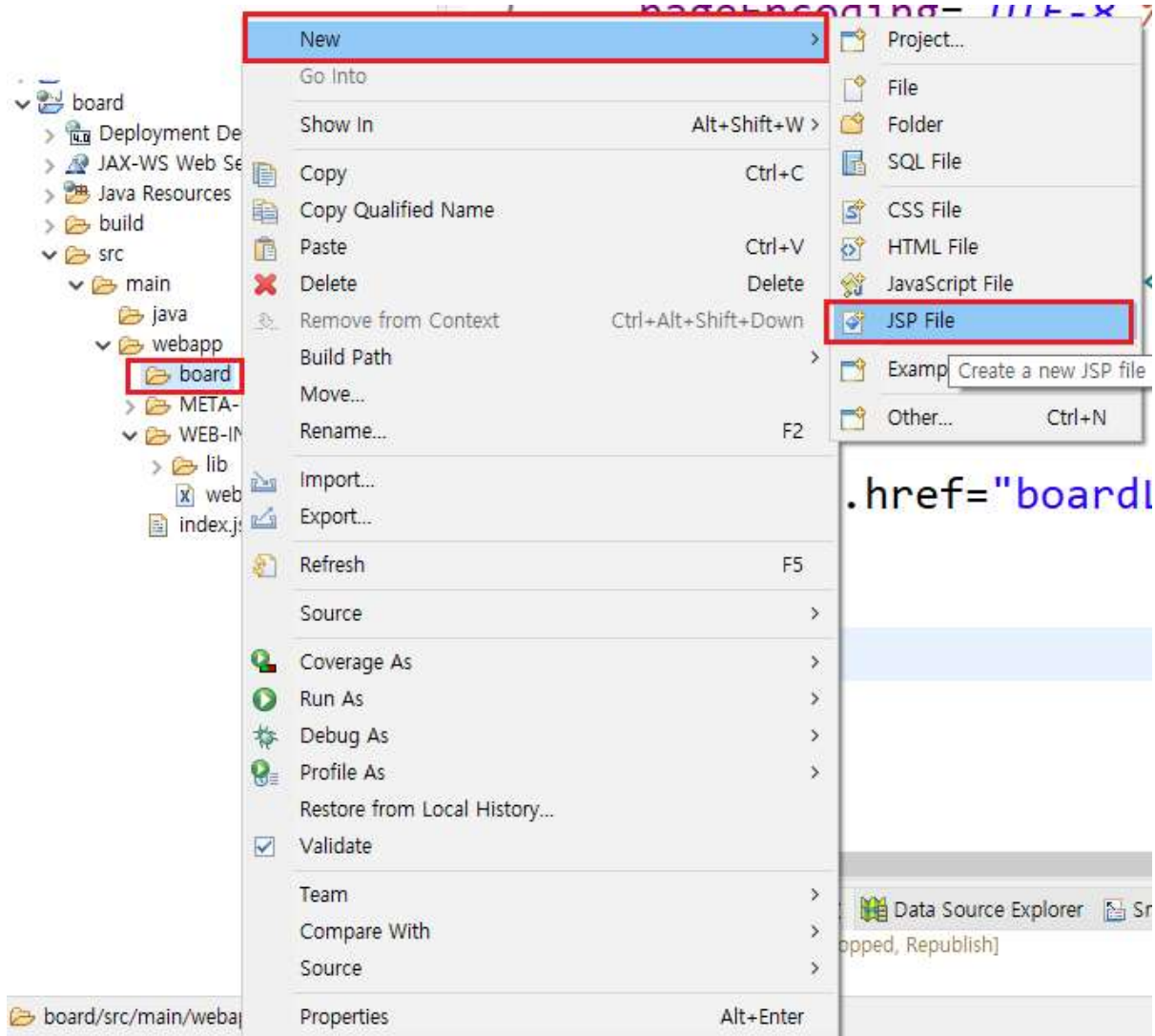
index.jsp파일을 만들어준다.

index.jsp파일을 만들어주면 첫 페이지로 자동으로 열리게 된다.

Index.jsp파일에 첫페이지가 될 주소를 적어준다.

```
<!DOCTYPE html>
<html>
<head>
<meta charset= "UTF-8">
<title>Insert title here</title>
</head>
  <body>
    <script>
      location.href="boardList.naver"
    </script>
  </body>
</html>
```

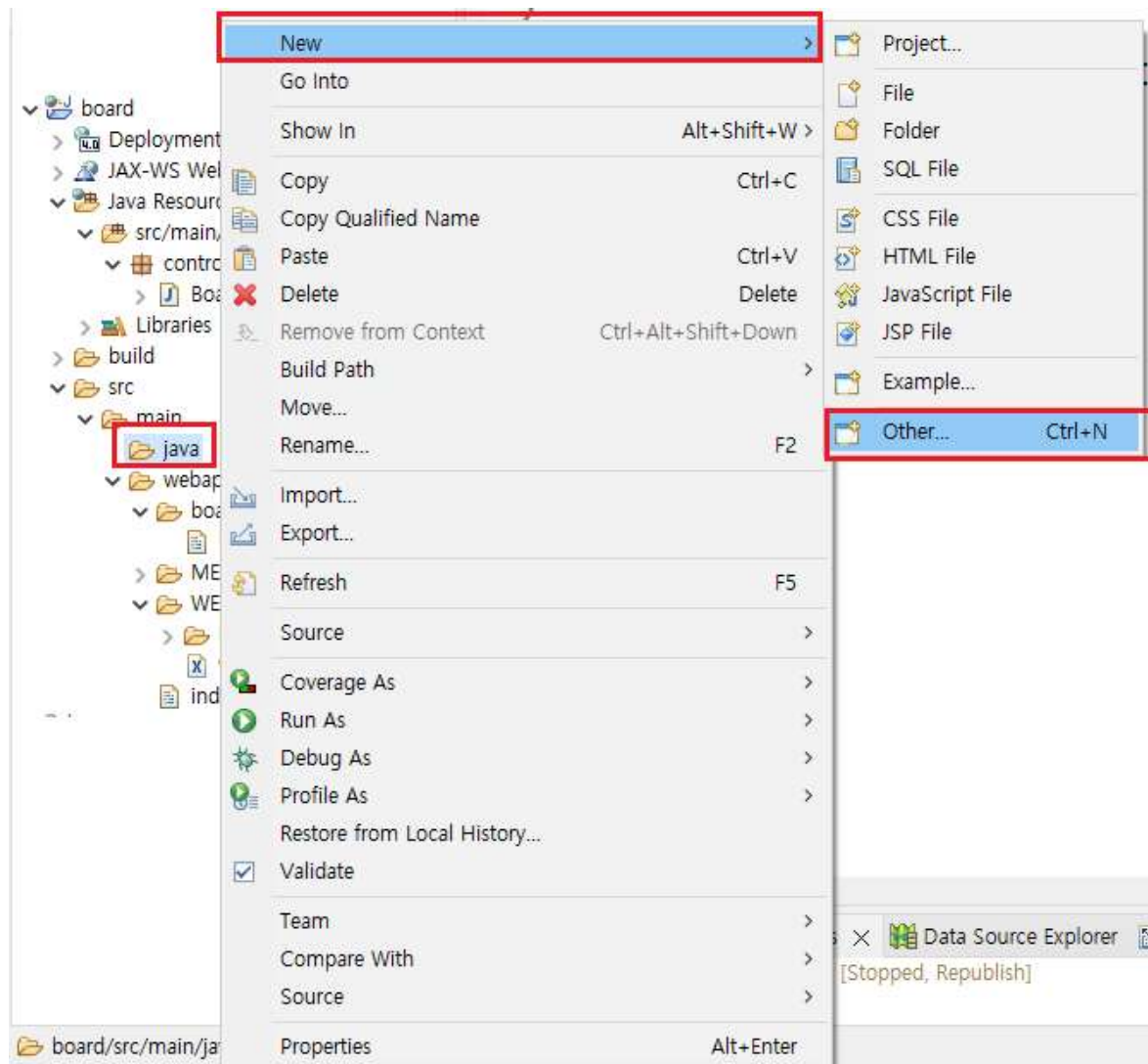





먼저 게시물 쓰기가 있어야 한다.

게시판 리스트 페이지가 필요하므로 board/boardList.jsp 파일을 생성시킨다.

```
<!DOCTYPE html>
<html>
<head>
<meta charset= "UTF-8">
<title>Insert title here</title>
</head>
  <body>
    게시물 목록<br />
  </body>
</html>
```



New Java Class

Java Class
Create a new Java class.

Source folder: board/src/main/java Browse...

Package: controller Browse...

☐ Enclosing type: Browse...

Name: BoardFrontController

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static
☒ none ☐ sealed ☐ non-sealed ☐ final

Superclass: java.lang.Object Browse...

Interfaces: Add... Remove

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

? < Back Next > Finish Cancel

리스트 페이지가 열리도록하기 위해 FrontController를 만들어주어야 한다.

java resource에서 BoardFrontController클래스 파일을 만들어준다.

패키지는 controller.board로 하도록 하겠다.

```
package controller.board;
```

```
public class BoardFrontController {
```

```
}
```

extends HttpServlet **implements** javax.servlet.Servlet를 클래스에 상속을 시켜야한다.

```
public class BoardFrontController extends HttpServlet  
    implements Servlet{
```

```
}
```

상속을 시킨 후 ctrl + shift + o를 눌러서 import를 시킨다.

Ctrl + space키를 눌러

@Override

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)  
throws ServletException, IOException {
```

```
}
```

@Override

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)  
throws ServletException, IOException {
```

```
}
```

를 추가한다. 그리고 ctrl + shift + o를 눌러서 import를 시킨다.

먼저 boardList.jsp 파일이 웹브라우저에서 열리도록 컨트롤러에 코드를 작성한다. 처음 페이지는 get방식이므로 doGet메서드에 작성을 한다.

먼저 주소를 찾기 위해 URI와 contextPath를 불러온 후 뒤 주소만 가져오기 위해 substring을 사용해서 가지고 온다.

```
protected void doGet(HttpServletRequest request,  
                     HttpServletResponse response) throws ServletException,  
                     IOException {
```

```
    String requestURI = request.getRequestURI();  
    String contextPath = request.getContextPath();  
    String command = requestURI.substring(contextPath.length());
```

```
}
```


주소를 가지고 온후 주소가 /boardList.board인지 확인 할수 있게 조건문을 사용한다.

```
protected void doGet(HttpServletRequest request,  
                     HttpServletResponse response) throws  
ServletException, IOException {
```

```
...
```

```
if(command.equals("/boardList.naver")) {
```

```
    RequestDispatcher dispatcher =  
        request.getRequestDispatcher("/board/boardList.jsp");  
    dispatcher.forward(request, response);
```

```
}
```

```
}
```

WEB-INF밑에 있는 web.xml에 servlet을 추가한다.

... 상략

```
</welcome-file-list>
```

```
<servlet>
```

```
<servlet-name>board</servlet-name>
```

```
<servlet-class>controller.BoardFrontController</servlet-class>
```

```
</servlet>
```

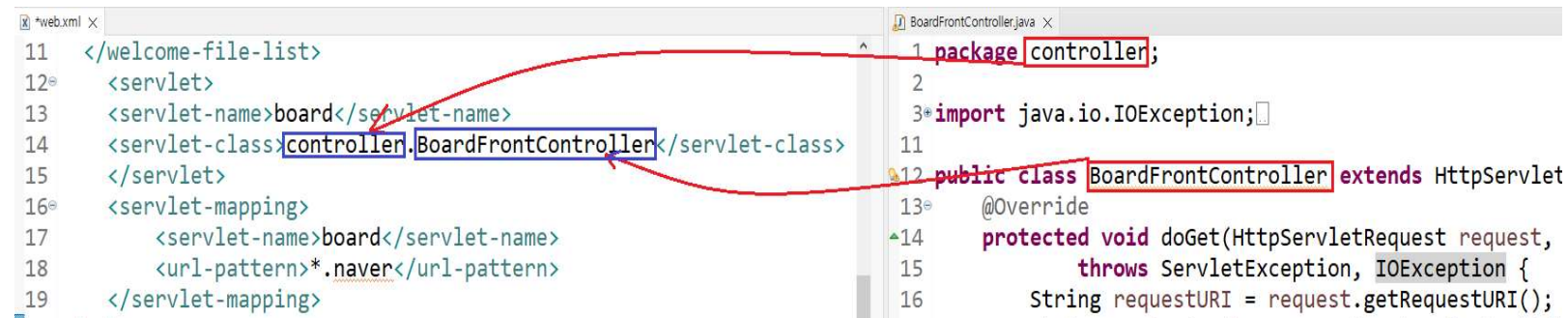
```
<servlet-mapping>
```

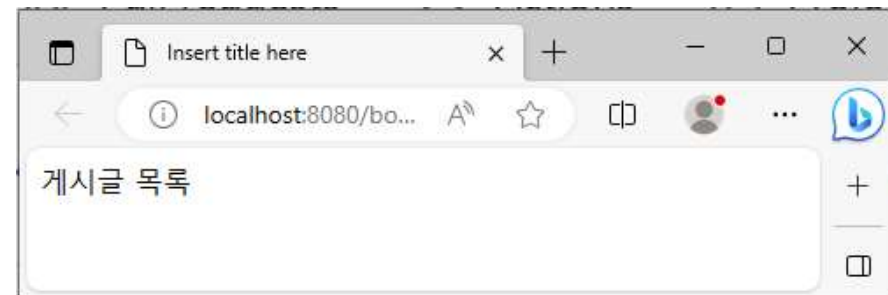
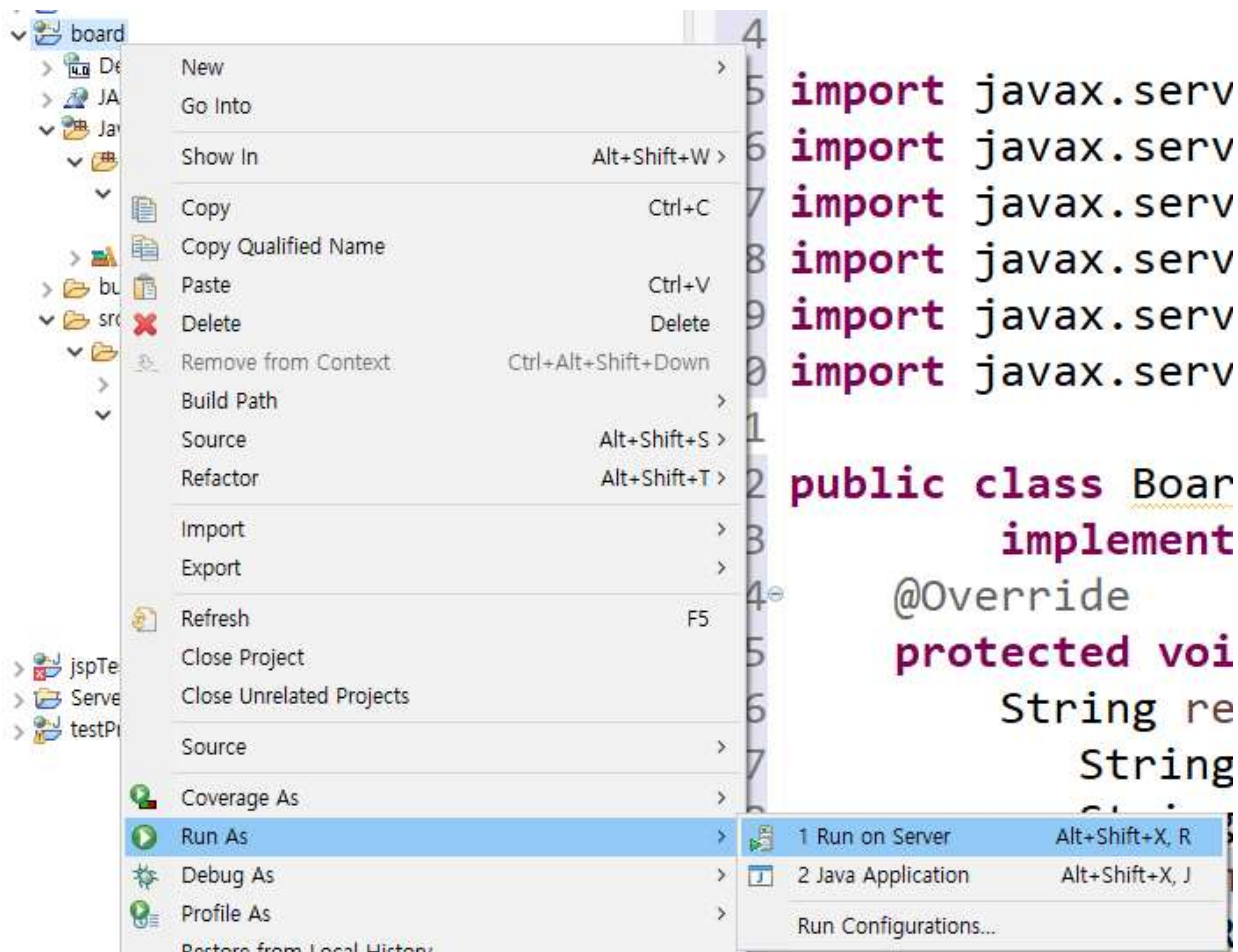
```
<servlet-name>board</servlet-name>
```

```
<url-pattern>*.naver</url-pattern>
```

```
</servlet-mapping>
```

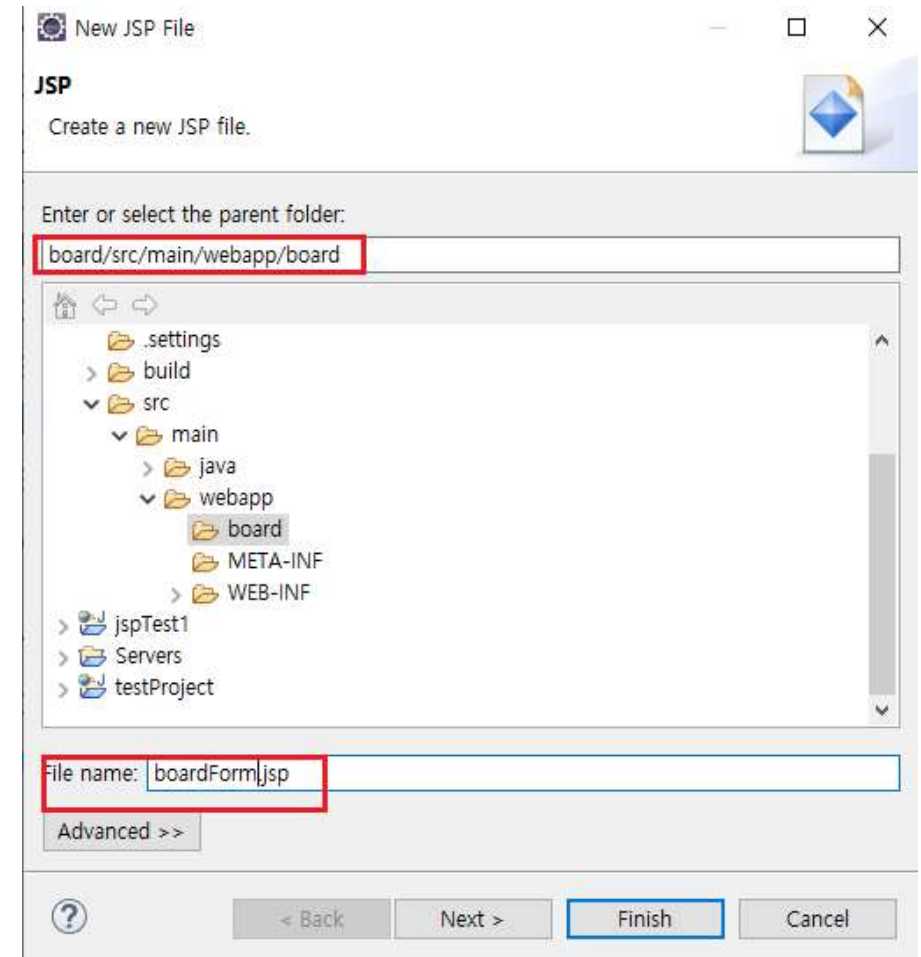
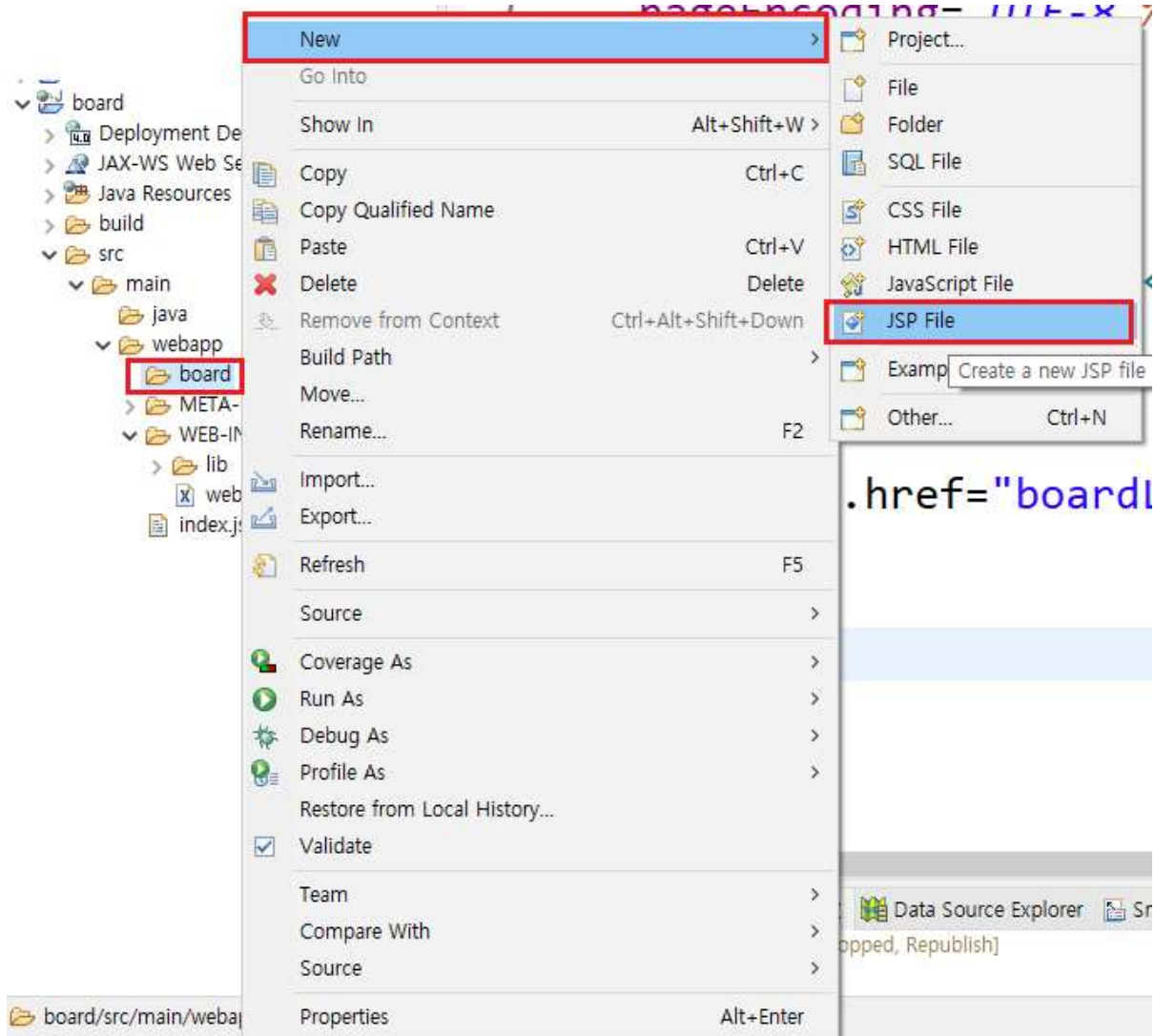
```
</web-app>
```





boardList.jsp에 글쓰기 링크가 있다.

```
<!DOCTYPE html>
<html>
<head>
<meta charset= "UTF-8">
<title>Insert title here</title>
</head>
<body>
  게시글 목록<br />
  <a href="boardWrite.naver">글쓰기</a>
</body>
</html>
```



먼저 "boardWrite.naver"주소에서 열릴 boardForm.jsp파일을 만들어야 할 것이다.

Webapps 밑에boardFrom.jsp를 만든다.

```
<form action= "#" method= "#">
```

```
    <table>
```

```
        <caption>게시글 쓰기</caption>
```

```
        <tr><td>글쓴이</td>
```

```
            <td><input type= "text" name= "boardWriter"> </td>
```

```
        </tr>
```

```
        <tr><td>제목</td>
```

```
            <td><input type= "text" name= "boardSubject"> </td>
```

```
        </tr>
```

```
        <tr><td>내용</td>
```

```
            <td><textarea rows= "6" cols= "40" name= "boardContent"> </textarea> </td>
```

```
        </tr>
```

```
        <tr><th colspan=2>
```

```
            <input type= "submit" value= "게시글 등록">
```

```
        </th> </tr>
```

```
    </table>
```

```
</form>
```


다시 BoardFrontController 를 열어서 내용을 추가한다.

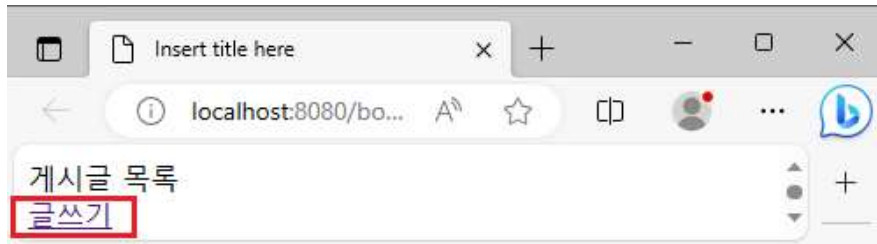


```
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11
12 public class BoardFrontController extends HttpServlet
13     implements Servlet{
14     @Override
15     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
16         String requestURI = request.getRequestURI();
17         String contextPath = request.getContextPath();
18         String command = requestURI.substring(contextPath.length());
19         if(command.equals("/boardList.naver")) {
20             RequestDispatcher dispatcher =
21                 request.getRequestDispatcher("/board/boardList.jsp");
22             dispatcher.forward(request, response);
23         }
24     }
25
26     @Override
27     protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
28         // TODO Auto-generated method stub
29         super.doPost(req, resp);
30     }
31 }
```

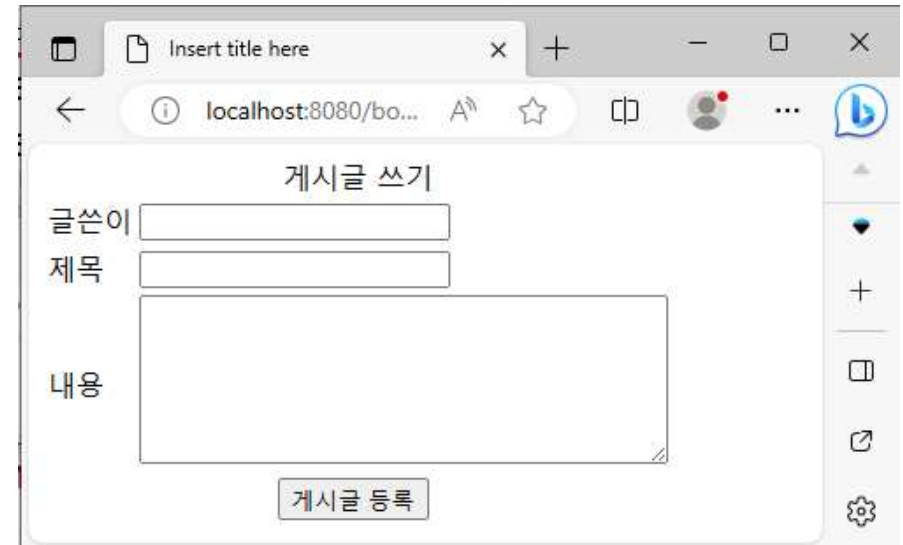
FrontController에서 boardForm.jsp파일을 웹브라우저에 전송하는 코드를 작성해준다.

```
public class BoardFrontController extends HttpServlet
    implements javax.servlet.Servlet{
@Override
    protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws.
        ServletException, IOException {
...
    if(){
...
    }else if(command.equals("/boardWrite.naver")){
        RequestDispatcher dispatcher =
            request.getRequestDispatcher("/board/boardForm.jsp");
        dispatcher.forward(request, response);
    }
}
```


현 페이지를 새로 고침(F5)해서 다음처럼 나오는지 확인



글쓰기(boardWrite.naver)를 클릭하면 다음페이지가 열리는 지 확인 해보자




먼저 "boardWrite.naver"주소에서 열릴 boardForm.jsp파일을 만들어야 할 것이다.

Webapps 밑에boardFrom.jsp를 만든다.

```
<form action= "boardRegist.naver" method= "post">
  <table>
    <caption>게시글 쓰기</caption>
    <tr><td>글쓴이</td>
      <td><input type= "text" name= "boardWriter"> </td>
    </tr>
    <tr><td>제목</td>
      <td><input type= "text" name= "boardSubject"> </td>
    </tr>
    <tr><td>내용</td>
      <td><textarea rows= "6" cols= "40" name= "boardContent"> </textarea> </td>
    </tr>
    <tr><th colspan=2>
      <input type= "submit" value= "게시글 등록">
    </th> </tr>
  </table>
</form>
```

다시 BoardFrontController 를 열어서 내용을 추가한다.



```
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11
12 public class BoardFrontController extends HttpServlet
13     implements Servlet{
14     @Override
15     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
16         String requestURI = request.getRequestURI();
17         String contextPath = request.getContextPath();
18         String command = requestURI.substring(contextPath.length());
19         if(command.equals("/boardList.naver")) {
20             RequestDispatcher dispatcher =
21                 request.getRequestDispatcher("/board/boardList.jsp");
22             dispatcher.forward(request, response);
23         }
24     }
25
26     @Override
27     protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
28         // TODO Auto-generated method stub
29         super.doPost(req, resp);
30     }
31 }
```

post방식이므로 doPost메서드에 작성을 한다.
먼저 주소를 찾기 위해 URI와 contextPath를 불러온 후 뒤 주소만 가져
오기 위해 substring을 사용해서 가지고 온다.

```
protected void doPost(HttpServletRequest request,  
IOException { HttpServletResponse response) throws ServletException,
```

```
String requestURI = request.getRequestURI();  
String contextPath = request.getContextPath();  
String command = requestURI.substring(contextPath.length());  
  
}
```

FrontController에 의해 boardForm.jsp가 웹브라우저에 열리면 자료항목에 자료를 작성을 하고 submit을 하게 된다.

Submit이 된 데이터는 서버단에서 저장이 된 후 목록페이지로 넘어가게 된다.

일단 boardForm.jsp에서 submit을 하면 *boardRegist.board*로 전송이 되고 다시 목록 페이지가 열려야 하므로 *boardRegist.board*

에서 목록 페이지로 가게 FrontController에 작성을 한다,

protected void doPost(HttpServletRequest request,

HttpServletResponse response) **throws** ServletException,

IOException {

... 중략

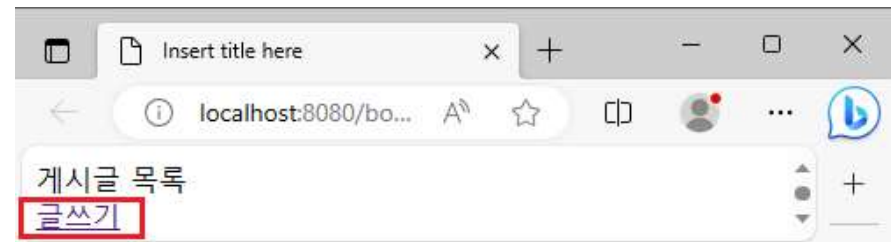
```
        if(command.equals("/boardRegist.naver")){  
            response.sendRedirect("boardList.naver");  
        }
```

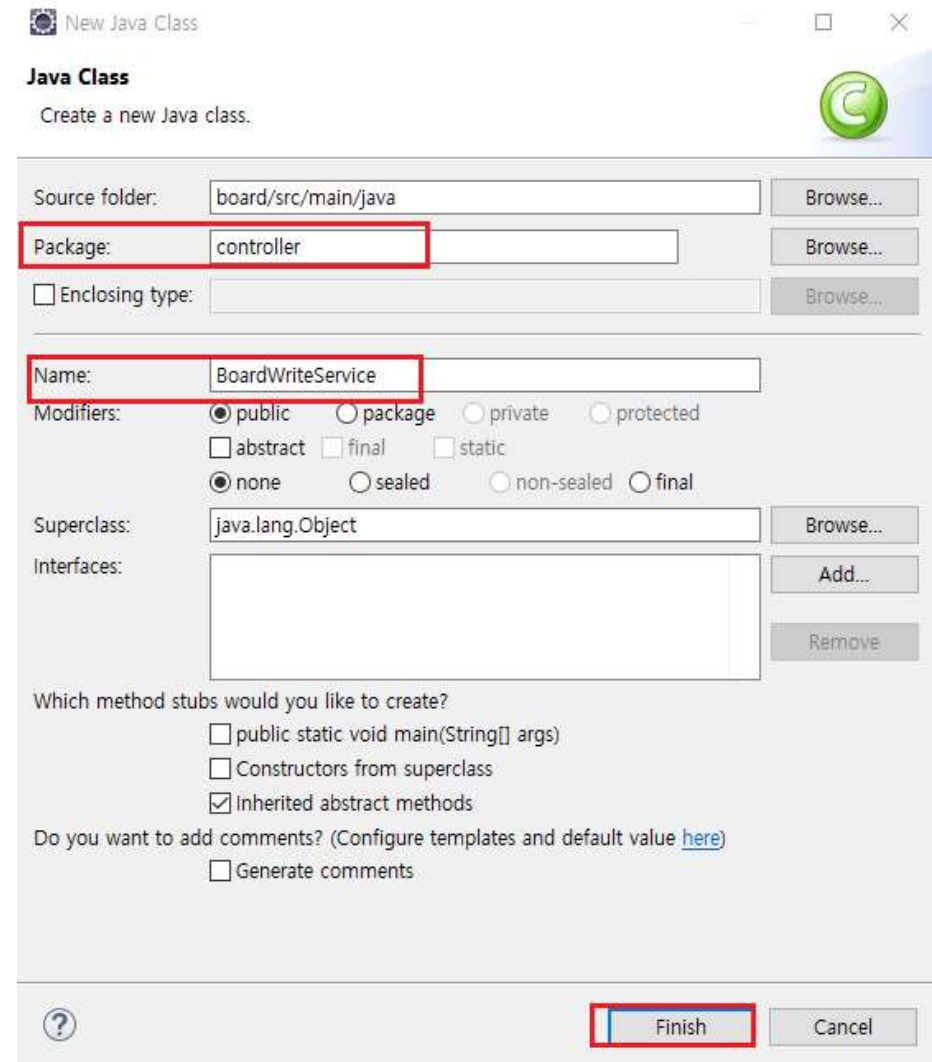
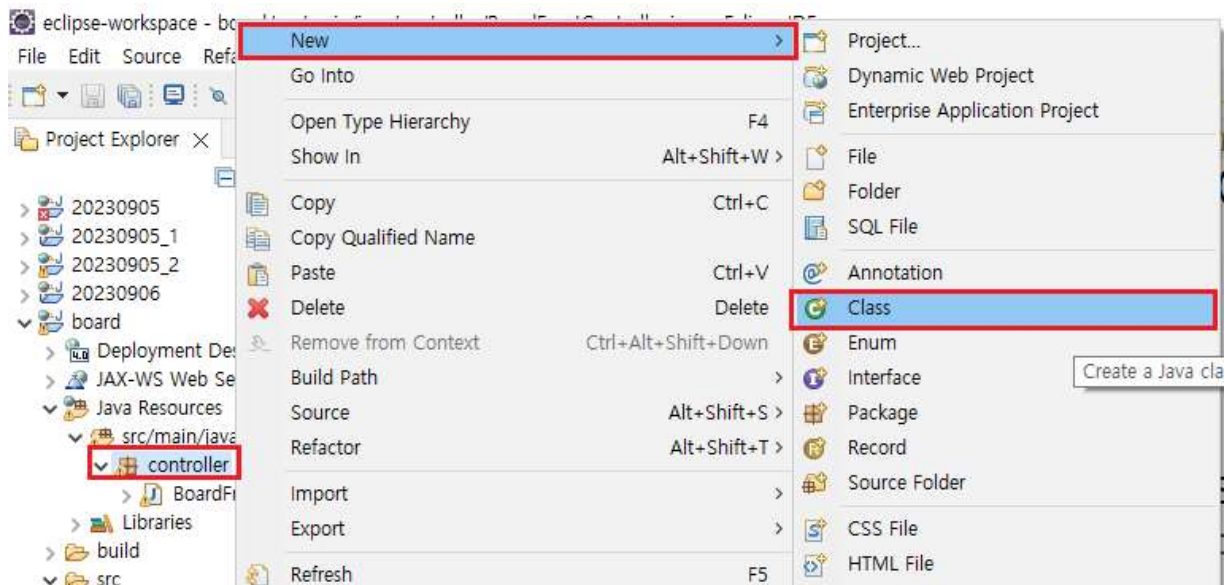
```
}
```

이때는 페이지 이동이므로 response.sendRedirect()를 이용하여 목록 페이지로 이동이 되게 만든다.

현 페이지를 새로 고침(F5) 한 후 게시글 등록을 클릭

A screenshot of a web browser window with the address bar showing 'localhost:8080/bo...'. The page title is '게시글 쓰기' (Write Post). The form contains three input fields: '글쓴이' (Author), '제목' (Title), and '내용' (Content). The '내용' field is a large text area. At the bottom right of the form is a button labeled '게시글 등록' (Register Post). The browser's address bar and navigation buttons are visible at the top.





목록 페이지로 이동이 되는 것을 확인되었다면 전송된 데이터를 저장하기 위한 page-controller를 만들어준다.

Page-controller의 이름은 BoardWriteService라고 만들고 패키지는 controller로 하자.

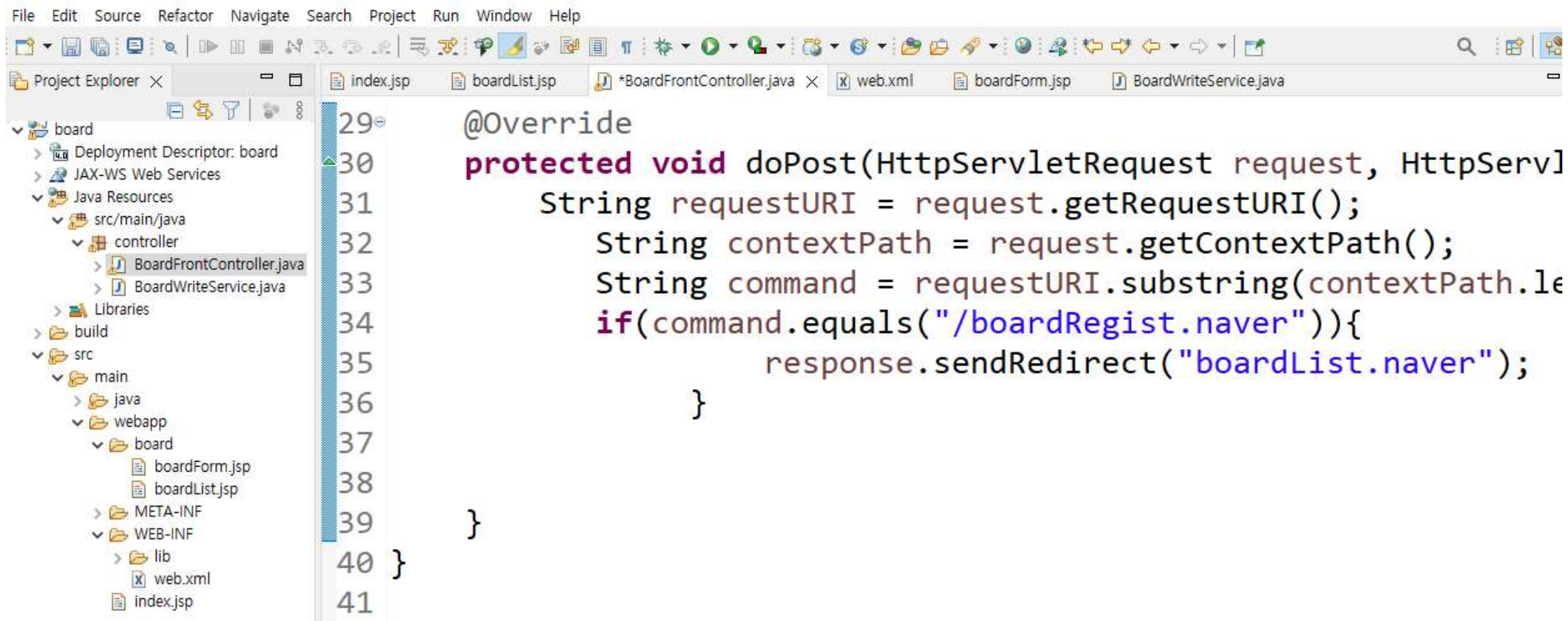
```
package controller;
```

```
public class BoardWriteService {  
}
```

Front-controller로 부터 데이터를 전달 받기 위한 메서드를 작성한다,

```
public class BoardWriteService {  
    public void execute(HttpServletRequest request) {  
  
    }  
}
```


다시 BoardFrontController 를 열어서 내용을 추가한다.



The screenshot shows an IDE window with the following components:

- Project Explorer (Left):** Displays the project structure. The 'board' package is expanded, showing 'src/main/java/controller'. Inside the controller package, 'BoardFrontController.java' is selected.
- Editor (Right):** Shows the code in 'BoardFrontController.java'. The code is as follows:

```
29 @Override
30 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
31     String requestURI = request.getRequestURI();
32     String contextPath = request.getContextPath();
33     String command = requestURI.substring(contextPath.length());
34     if(command.equals("/boardRegist.naver")){
35         response.sendRedirect("boardList.naver");
36     }
37 }
38
39 }
40 }
41
```

이제 Front-controller에서 데이터를 전달할 수 있게
BoardWriteController객체를 생성하고 메서드를 실행시켜준다.

```
protected void doPost(HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException {
...   중략
    if(command.equals("/boardRegist.naver")) {
        BoardWriteService action = new BoardWriteService();
        action.execute(request);

        response.sendRedirect("boardList.board");
    }
}
```

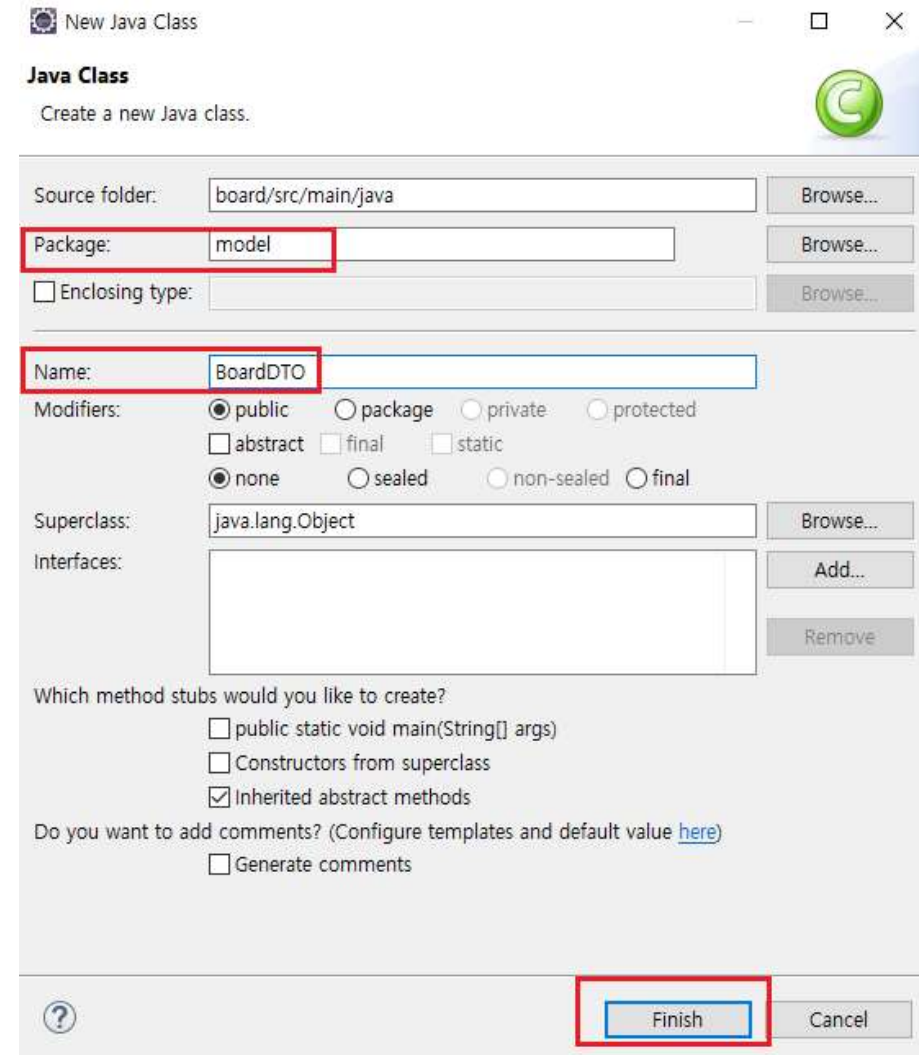
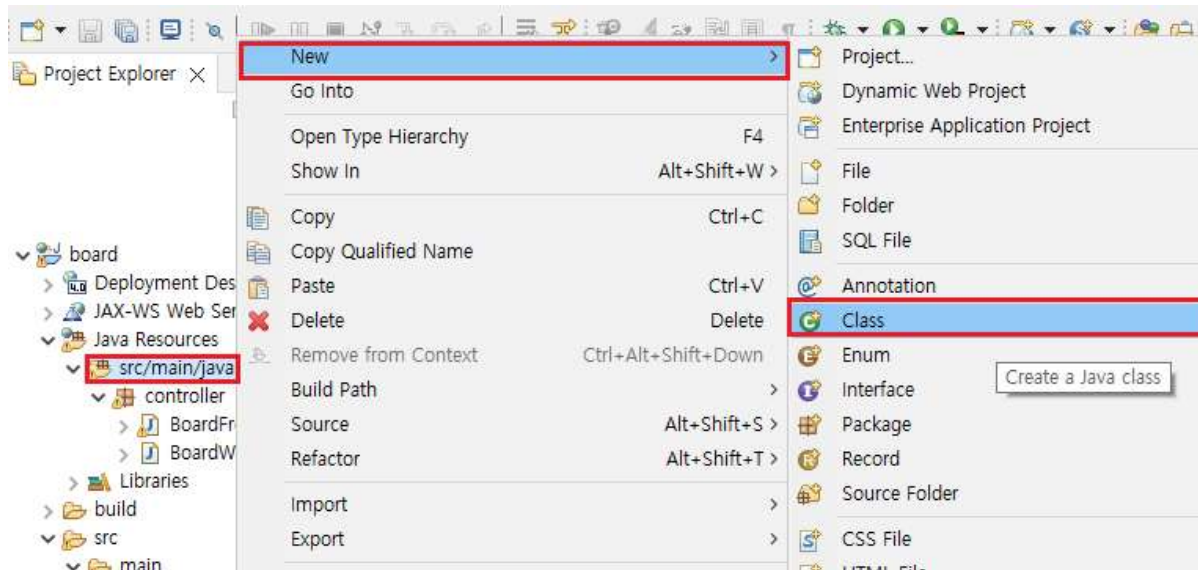
다시 BoardWriteService.java 를 열어 boardForm.jsp로 부터 전송된 내용을 requsrto로 부터 받도록 하자.



```
1 package controller;
2
3 import javax.servlet.http.HttpServletRequest;
4
5 public class BoardWriteService {
6     public void execute(HttpServletRequest request) {
7
8     }
9 }
10
```

BoardWriteService에 전달된 값을 받을 수 있게 변수를 정의하고 각 변수에는 boardForm.jsp의 input으로 부터 전송된 값을 request.getParameter() 메서드를 이용하여 받는다.

```
public class BoardWriteService {  
    public void execute(HttpServletRequest request) {  
        try {  
            request.setCharacterEncoding("utf-8");  
        } catch (UnsupportedEncodingException e) {}  
  
        /// request.getParameter("input의 name을 적어준다.")  
        String boardWriter = request.getParameter("boardWriter");  
        String boardSubject = request.getParameter("boardSubject");  
        String boardContent = request.getParameter("boardContent");  
    }  
}
```



각 변수에 저장된 값은 디비에 전송하기 위해 먼저 DTO에 저장을 해야하므로 DTO를 만든다.

```
package model;
```

```
public class BoardDTO {  
    Integer boardNum;  
    String boardWriter;  
    String boardSubject;  
    String boardContent;  
    // getter/setter 만든다.  
}
```

DTO를 만들었다면 BoardWriteService에서 DTO를 객체 생성후 변수에 저장되어 있는 값을 DTO에 저장한다.

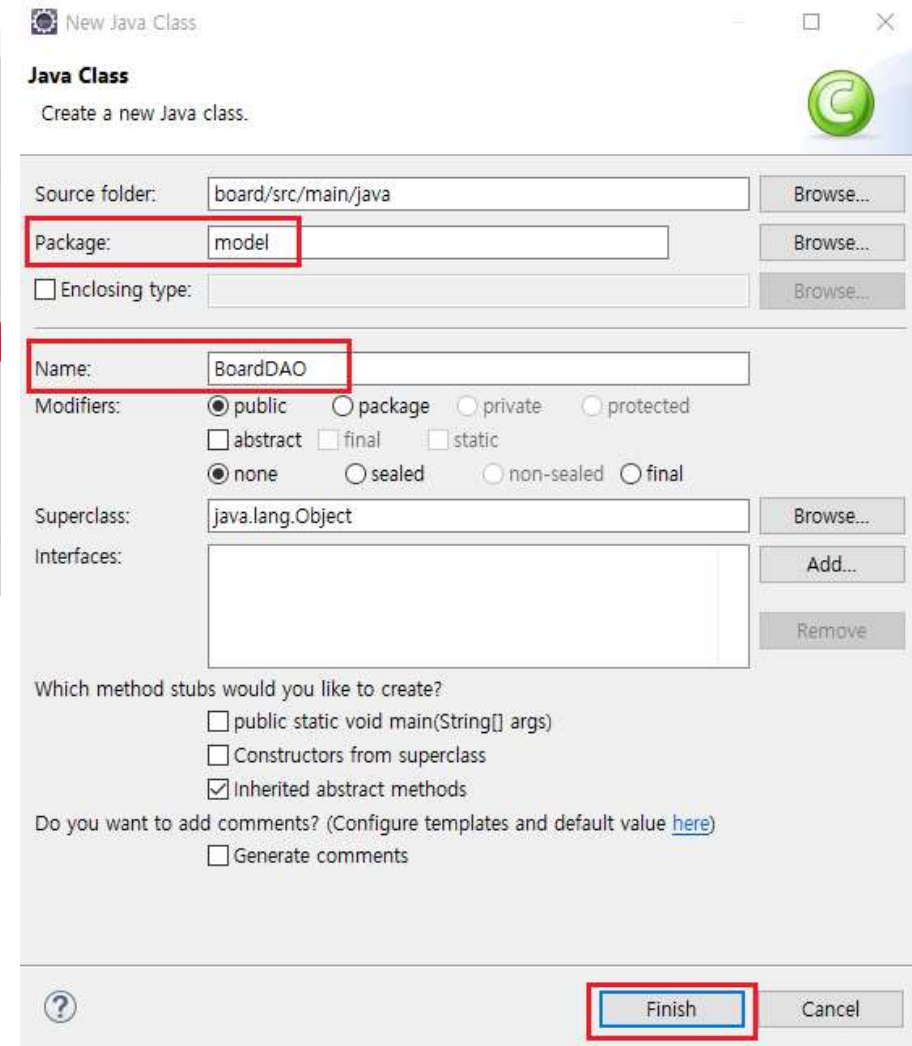
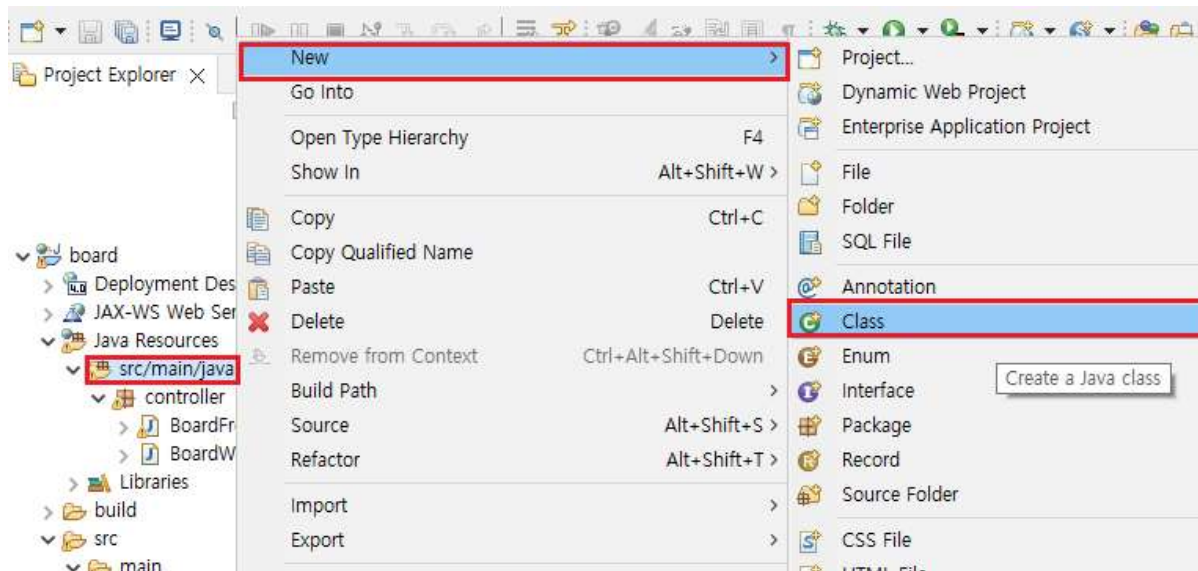
```
public class BoardWriteService {  
    public void execute(HttpServletRequest request) {
```

...중략

```
        BoardDTO dto = new BoardDTO();  
        dto.setBoardContent(boardContent);  
        dto.setBoardSubject(boardSubject);  
        dto.setBoardWriter(boardWriter);
```

```
    }
```

```
}
```

DTO에 있는 값을 DAO로 전달해야 한다,
전달 받을 DAO를 만든다. 그리고 데이터베이스 정보를 적어준다.

```
package model.DAO;
public class BoardDAO{
    String jdbcURL;
    String jdbcDriver;
    Connection con;
    PreparedStatement pstmt;
    ResultSet rs;
    public BoardDAO() {
        jdbcDriver = "oracle.jdbc.driver.OracleDriver";
        jdbcURL = "jdbc:oracle:thin:@localhost:1521:xe";
    }
    public Connection getConnection() {
        Connection conn = null;
        try {
            Class.forName(jdbcDriver);
            conn = DriverManager.getConnection(jdbcURL,"rhee","1234");
        }catch(Exception e) {e.printStackTrace();}
        return conn;
    }
}
```

그리고 데이터를 BoardWriteService에서 DTO로 전달 받은 값을 저장하기 위한 메서드를 추가한다.

```
public class BoardDAO{
```

```
... 중략
```

```
    public void boardInsert(BoardDTO dto) {  
        con = getConnection();  
        String sql = " insert into board(WRITER,SUBJECT, CONTENTS)"  
            + " values(?,?,?)";  
        try {  
            pstmt= con.prepareStatement(sql);  
            pstmt.setString(1, dto.getBoardWriter());  
            pstmt.setString(2, dto.getBoardSubject());  
            pstmt.setString(3, dto.getBoardContent());  
            int i = pstmt.executeUpdate();  
            System.out.println(i + " 개 행이(가) 삽입되었습니다.");  
        } catch (SQLException e) {e.printStackTrace();  
        } finally {  
            if(pstmt != null) try{pstmt.close();}catch(Exception e) {}  
            if(con != null) try{con.close();}catch(Exception e) {}  
        }  
    }  
}
```

DAO객체를 생성한 후 boardInsert를 통해 DTO를 전달해준다.

```
public class BoardWriteService {  
    public void execute(HttpServletRequest request) {  
  
        ...  
        BoardDTO dto = new BoardDTO();  
        dto.setBoardContent(boardContent);  
        dto.setBoardSubject(boardSubject);  
        dto.setBoardWriter(boardWriter);  
  
        BoardDAO dao = new BoardDAO();  
        dao.boardInsert(dto);  
    }  
}
```

게시글 쓰기

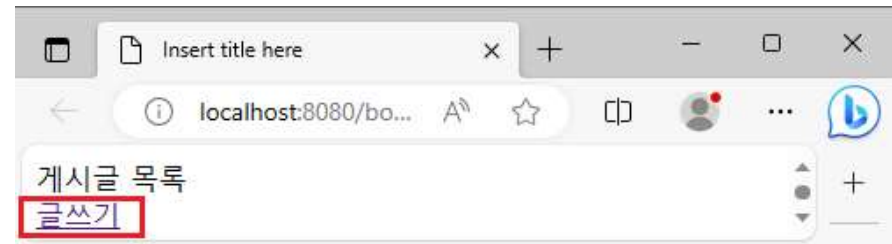
글쓴이 이승무

제목 제목

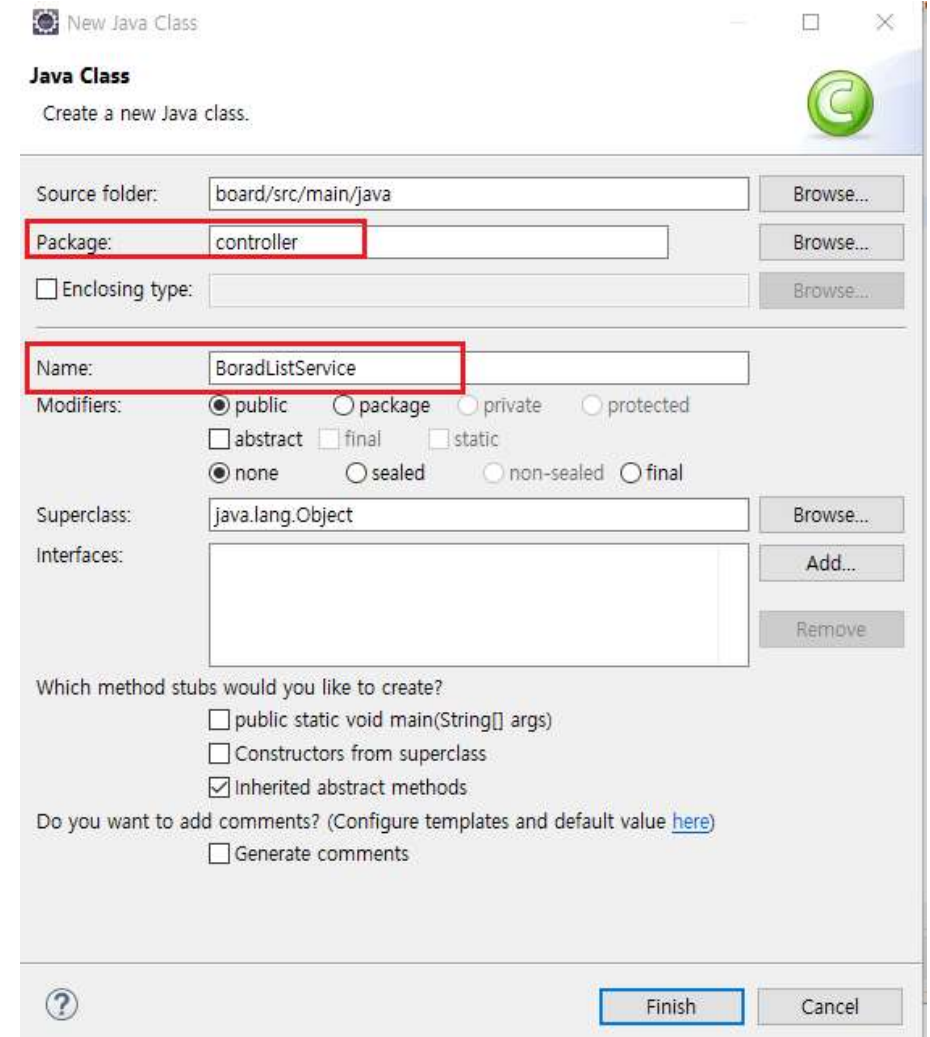
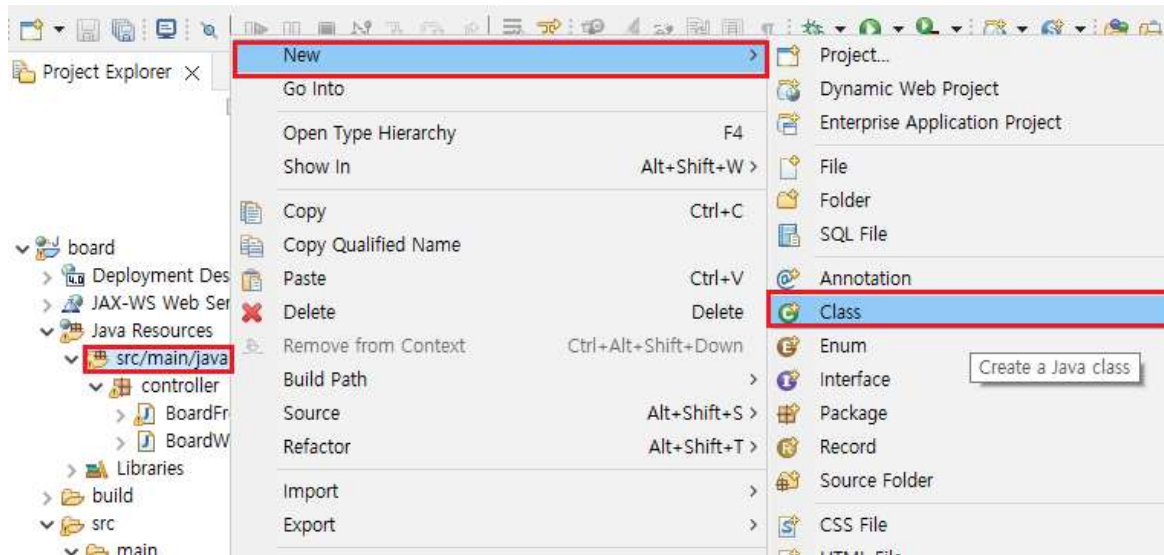
내용

게시글 등록

게시글 등록을 누르면 게시글이 디비에 저장
이 된 후 게시글 목록 페이지로 이동한다.

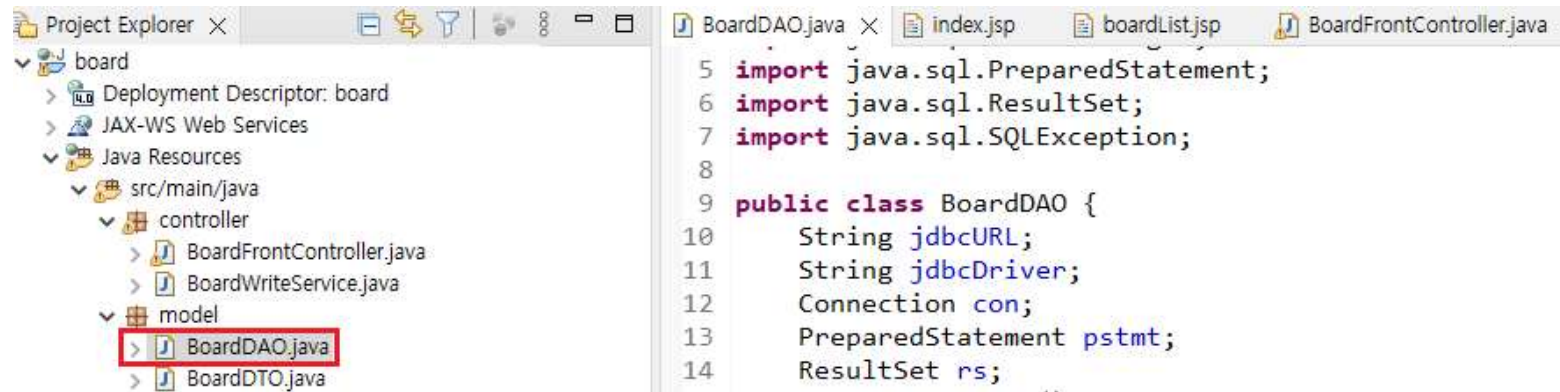


게시글 목록 페이지에서는 입력한 리스트가 출력이
되어야 한다.
그러기 위해서는 DB로부터 데이터를 가지고 와야
할 것이다.



```
public class BoradListService {  
    public void execute(HttpServletRequest request) {  
        BoardDAO dao = new BoardDAO();  
    }  
}
```

DAO 를 열어서 리스트를 가지고 오도록 메서드를 추가한다.



...상략

```
public class BoardDAO {
```

... 중략

```
    public List<BoardDTO> selectAll() {  
        List<BoardDTO> list = new ArrayList<BoardDTO>();  
        con = getConnection();  
        String sql="select num,writer,subject,contents"  
            + " from board"  
            + " order by num desc";
```

///// 다음 슬라이드에 있는 녹색 내용을 추가

```
        return list;
```

```
    }
```

```
}
```



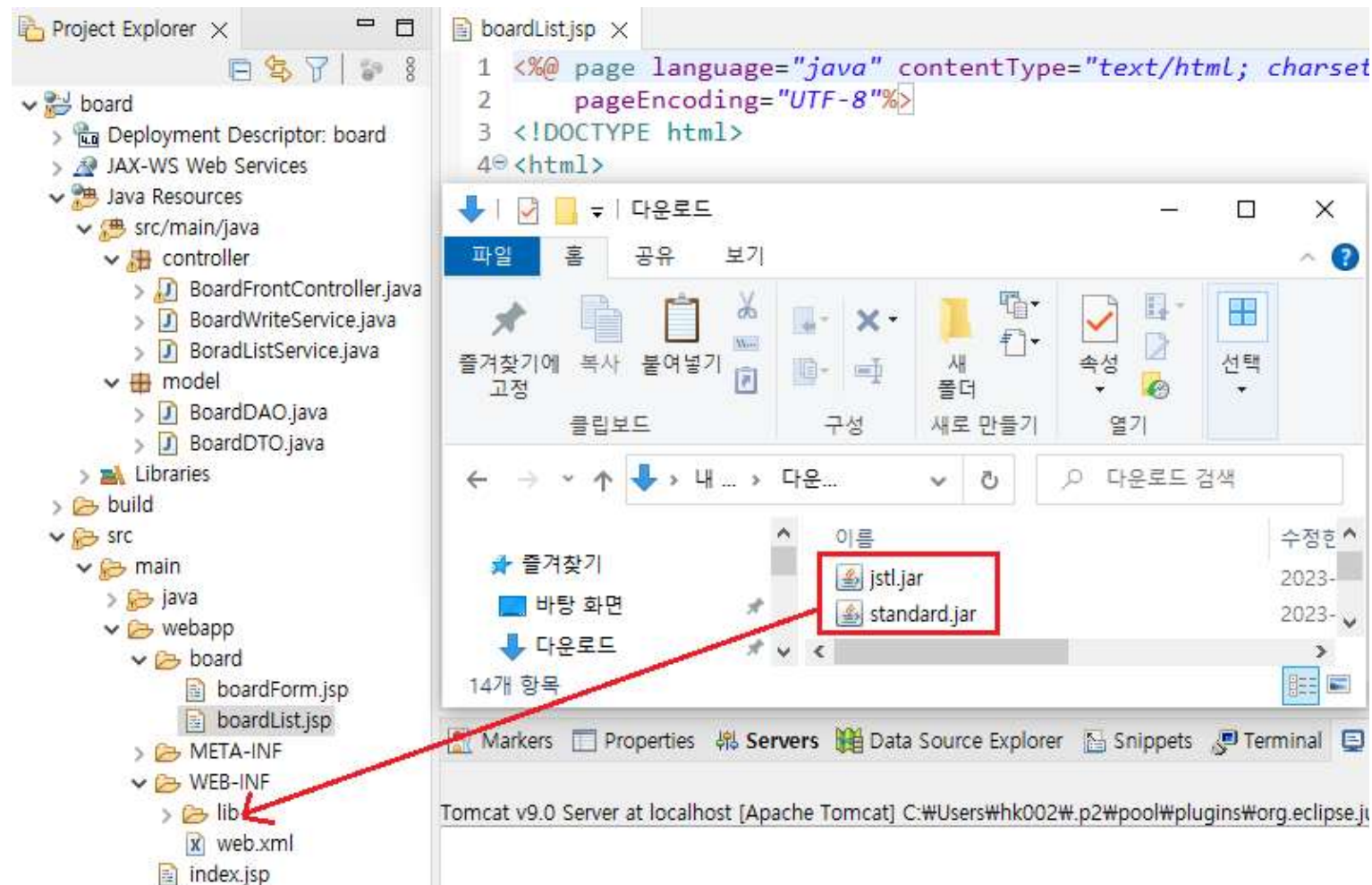
```
try {  
    pstmt = con.prepareStatement(sql);  
    rs = pstmt.executeQuery();// 출력될 모든 레코드를갖지고 옮  
    while(rs.next()) {  
        BoardDTO dto = new BoardDTO();  
        dto.setBoardContent(rs.getString("contents"));  
        dto.setBoardNum(rs.getInt("NUM"));  
        dto.setBoardSubject(rs.getString("SUBJECT"));  
        dto.setBoardWriter(rs.getString("WRITER"));  
        list.add(dto);  
    }  
} catch (Exception e) {  
    e.printStackTrace();  
}
```

```
public class BoradListService {  
    public void execute(HttpServletRequest request) {  
        BoardDAO dao = new BoardDAO();  
        ///// 추가  
        List<BoardDTO> list = dao.selectAll();  
        request.setAttribute("lists", list);  
    }  
}
```

BoardFrontController에 boardList.naver에 아래 내용을 추가한다.

```
public class BoardFrontController extends  
HttpServlet implements Servlet {  
... 중략  
    if(command.equals("/boardList.naver")) {  
        //// 추가  
        BoradListService action = new BoradListService();  
        action.execute(request);  
        ///  
        RequestDispatcher dispatcher =  
            request.getRequestDispatcher("/board/boardList.jsp");  
        dispatcher.forward(request, response);  
    }  
... 하략  
}
```

“스크립트릿” 대신 “jstl”을 사용하기 위해선
Jstl.jar와 standard.jar파일을 WEB-INF/lib폴더에 복사한다.

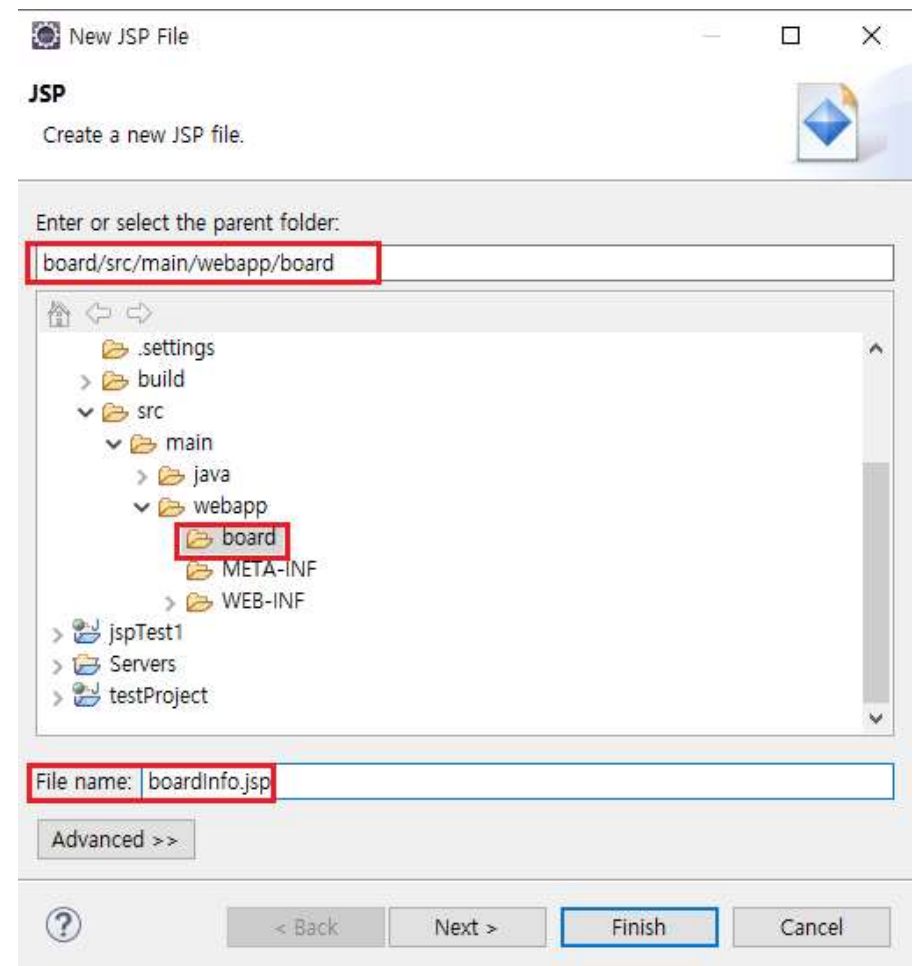
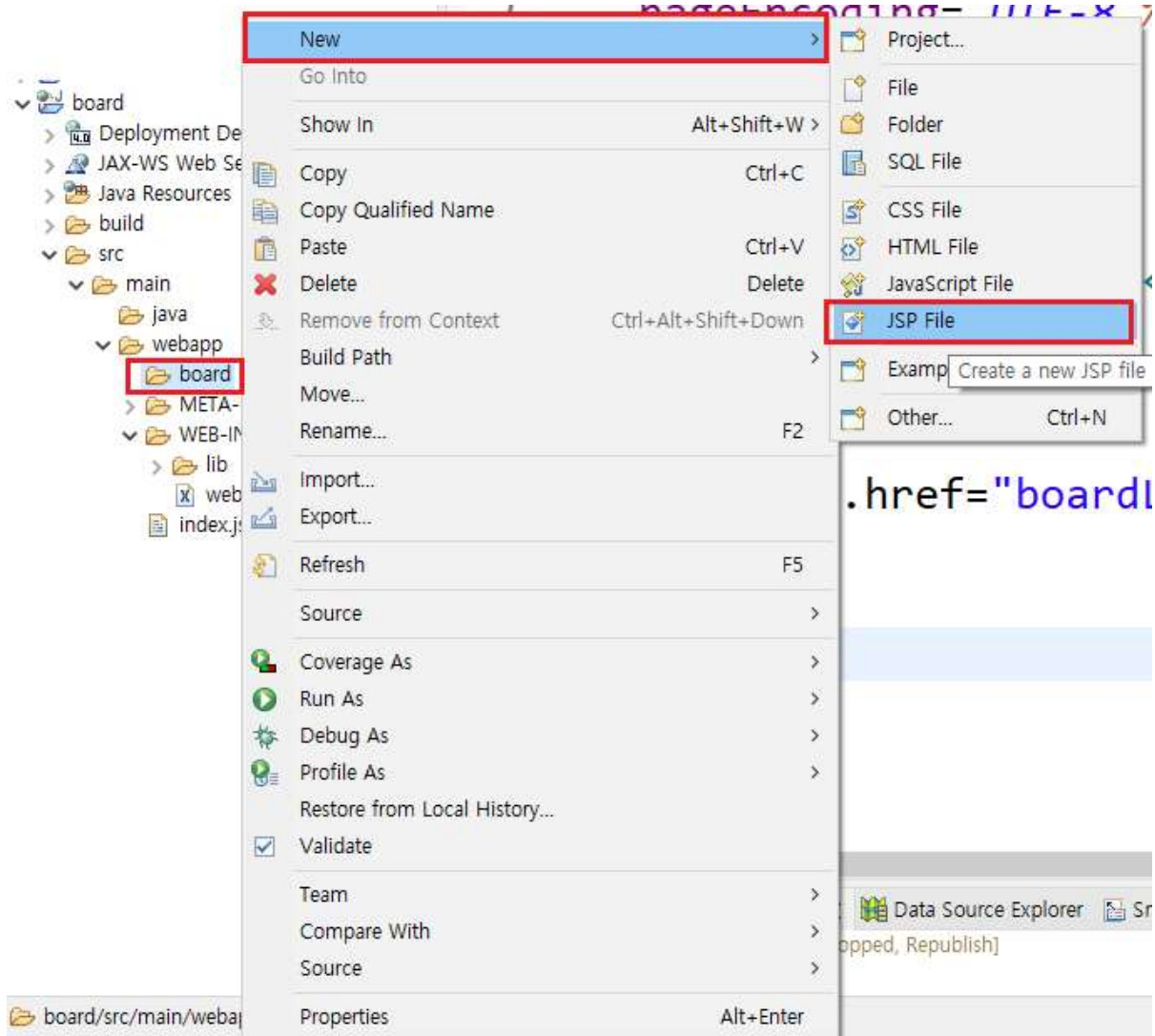


boardList.jsp에서 BoradListServicer에서 전달된 List에 있는 값을 출력하자.

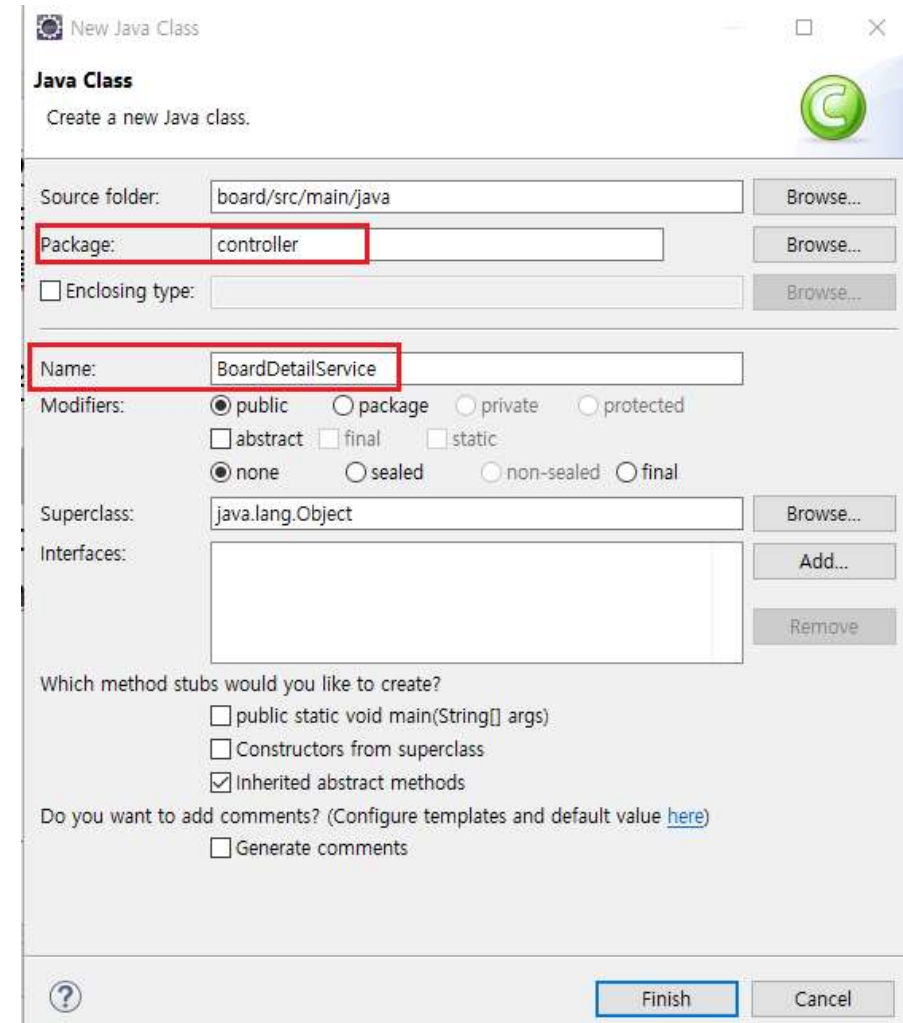
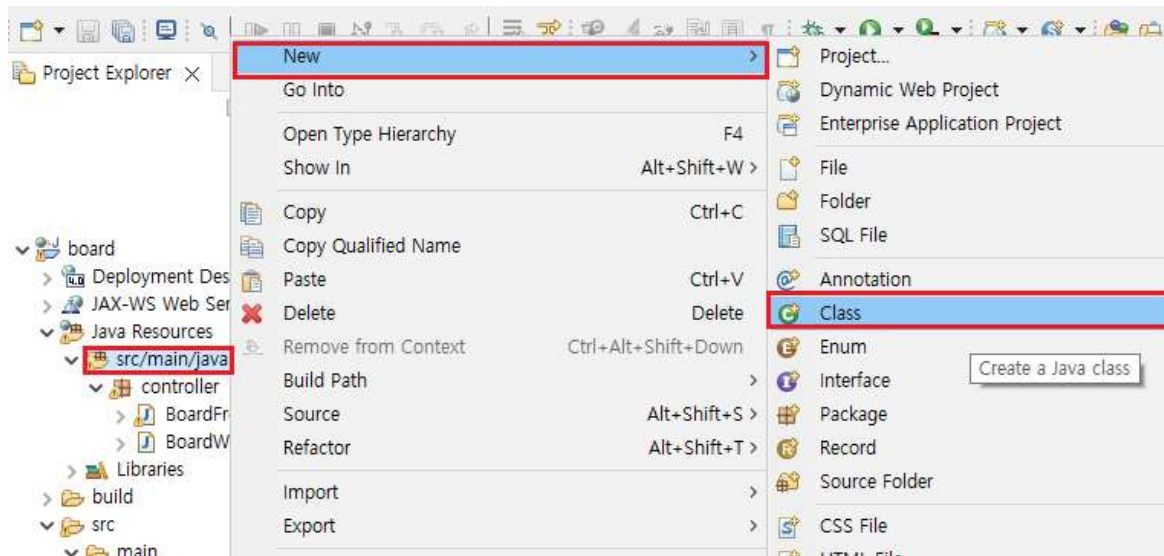
```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
...
<body>
게시글 목록<br />
<table border= 1 width= "600px">
<thead>
<tr><th>글번호</th><th>글쓴이</th><th>제목</th></tr>
</thead>
<tbody>
<c:forEach items= "${lists }" var= "dto">
<tr><td><a href="boardDetail.naver?num=${dto.boardNum }">${dto.boardNum }</a></td>
<td>${dto.boardWriter }</td>
<td><a href="boardDetail.naver?num=${dto.boardNum }">${dto.boardSubject }</a></td></tr>
</c:forEach>
</tbody>
</table><br />
<a href= "boardWrite.naver">글쓰기</a>
</body>
```

BoardFrontController에서 boardDetail.board 주소에서 보여줄 /board/boardInfo.jsp 파일을 전송할 수 있게 한다.

```
public class BoardFrontController extends HttpServlet
    implements Servlet {
    protected void doGet(HttpServletRequest request,
        HttpServletResponse response){
        ...중략
        else if(command.equals("/boardDetail.naver")) {
            RequestDispatcher dispatcher =
            request.getRequestDispatcher("/board/boardInfo.jsp");
            dispatcher.forward(request, response);
        }
    }
    ... 중략
}
```



```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
상세페이지 <br />
글번호 : 0<br />
글쓴이 : 0<br />
제목 : 0<br />
내용 : 0<br />
수정 | 삭제 | 게시물 리스트
</body>
</html>
```

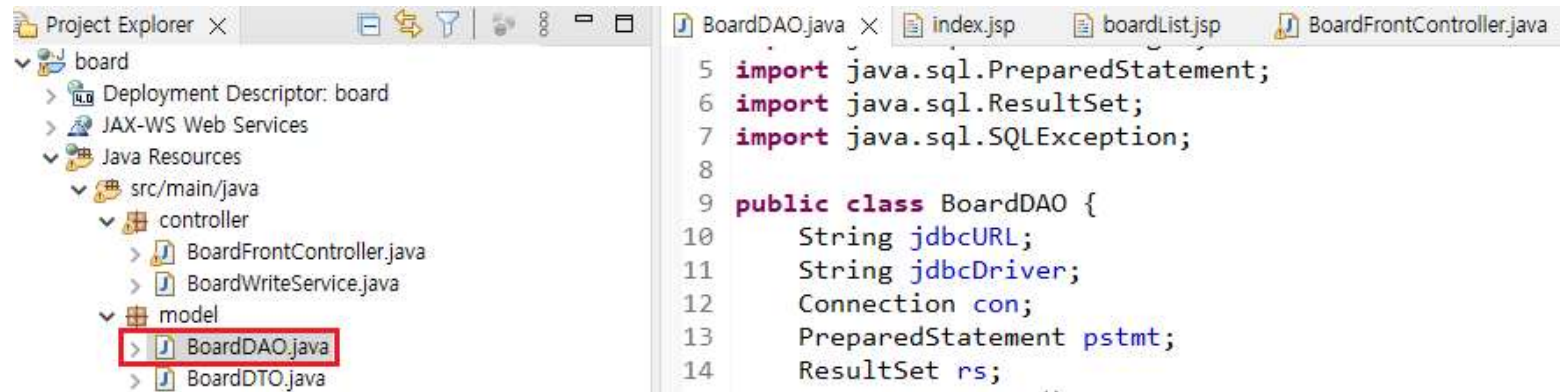
BoardFrontController의 boardDetail.naver에 데이터를 가지고 올 수 있게 page-controller가 있어야 한다.

BoardDetailController를 만들어 상세정보를 가지고 오도록한다.

쿼리스트링으로 받아온 num값을 이용해서 상세정보를 가지고 오자.

```
public class BoardDetailService {  
    public void execute(HttpServletRequest request) {  
        String num = request.getParameter("num");  
        BoardDAO dao = new BoardDAO();  
    }  
}
```

DAO 를 열어서 리스트를 가지고 오도록 메서드를 추가한다.



```
public class BoardDAO {
```

```
    ...중략
```

```
    public BoardDTO selectOne(String num) {
```

```
        BoardDTO dto = new BoardDTO();
```

```
        con = getConnection();
```

```
        String sql = "select NUM, WRITER, SUBJECT, CONTENTS"
```

```
            + " from board "
```

```
            + " where NUM = ?";
```

```
        //// 다음 슬라이드에 있는 녹색 내용을 추가
```

```
        return dto;
```

```
    }
```

```
}
```

```
try {  
    pstmt = con.prepareStatement(sql);  
    pstmt.setInt(1, Integer.parseInt(num));  
    rs = pstmt.executeQuery();  
    if(rs.next()) {  
        dto.setBoardContent(rs.getString("CONTENTS"));  
        dto.setBoardNum(rs.getInt("NUM"));  
        dto.setBoardSubject(rs.getString("SUBJECT"));  
        dto.setBoardWriter(rs.getString("WRITER"));  
    }  
} catch (Exception e) {  
    e.printStackTrace();  
}  
finally {  
    if(rs != null) try{rs.close();}catch(Exception e) {}  
    if(pstmt != null) try{pstmt.close();}catch(Exception e) {}  
    if(con != null) try{con.close();}catch(Exception e) {}  
}
```

```
public class BoardDetailService {  
    public void execute(HttpServletRequest request) {  
        String num = request.getParameter("num");  
        BoardDAO dao = new BoardDAO();  
  
        //// 추가  
        BoardDTO dto = dao.selectOne(num);  
        request.setAttribute("dto", dto);  
    }  
}
```

```

public class BoardFrontController extends HttpServlet
    implements Servlet {
    protected void doGet(HttpServletRequest request,
        HttpServletResponse response){
        ...중략
        else if(command.equals("/boardDetail.naver")) {
            //// 추가
            BoardDetailService action =
                new BoardDetailService();
            action.execute(request);
            RequestDispatcher dispatcher =
                request.getRequestDispatcher("/board/boardInfo.jsp");
            dispatcher.forward(request, response);
        }
    }
    ... 중략
}

```

/board/boardInfo.jsp에 수정페이지로 가도록 링크를 만들어준다.

```
<body>
```

```
상세페이지 <br />
```

```
글번호 : ${dto.boardNum}<br />
```

```
글쓴이 : ${dto.boardWriter }<br />
```

```
제목 : ${dto.boardSubject }<br />
```

```
내용 : ${dto.boardContent }<br />
```

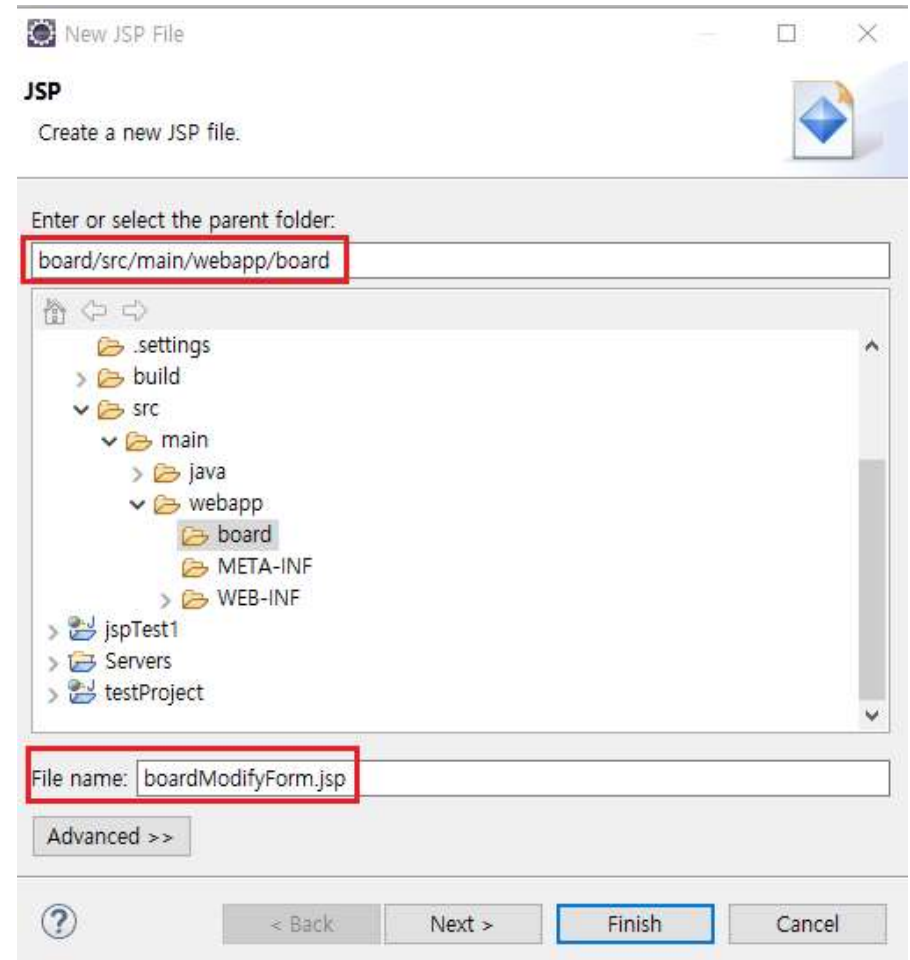
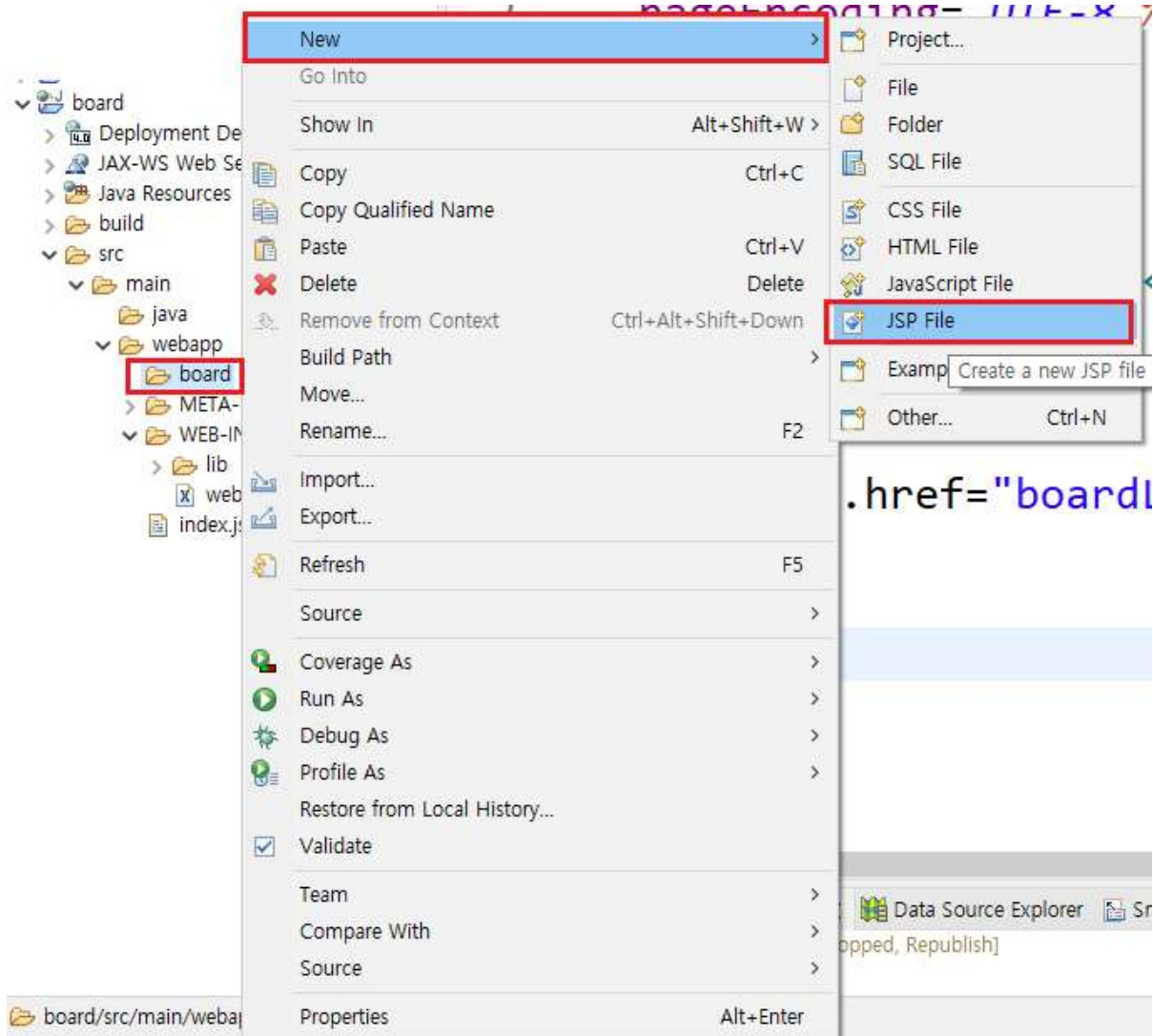
```
<a href= "boardUpdate.naver?num=${dto.boardNum}">수정 </a>
```

```
| 삭제 | 게시물 리스트
```

```
</body>
```


수정하기 위한 주소 *boardUpdate.naver*에서 *boardModifyForm.jsp*파일이 열리도록 *Front-Controller*에 작성하자.

```
public class BoardFrontController extends HttpServlet implements Servlet {  
    protected void doGet(HttpServletRequest request,  
        HttpServletResponse response){  
        ...중략  
        else if(command.equals("/boardUpdate.naver")) {  
            RequestDispatcher dispatcher =  
            request.getRequestDispatcher("/board/boardModifyForm.jsp");  
            dispatcher.forward(request, response);  
        }  
    }  
    ... 중략  
}
```



boardModifyForm.jsp 페이지를 만든다.

```
<form action= "boardModify.naver" method= "post">
  <input type= "hidden" name= "boardNum" value= ""/>
  <table>
    <caption>게시글 수정</caption>
    <tr> <td>글쓴이</td>
      <td> <input type= "text" name= "boardWriter" value= ""> </td>
    </tr>
    <tr> <td>제목</td>
      <td> <input type= "text" name= "boardSubject" value= ""> </td>
    </tr>
    <tr> <td>내용</td>
      <td> <textarea rows= "6" cols= "40" name= "boardContent"> </textarea> </td>
    </tr>
    <tr> <th colspan= 2>
      <input type= "submit" value= "게시글 수정 완료" />
      <input type= "button" value= "뒤로가기" onclick= "javascript:history.back()" />
    </th> </tr>
  </table>
</form>
```

```

public class BoardFrontController extends HttpServlet implements Servlet {
    protected void doGet(HttpServletRequest request,
        HttpServletResponse response){
        ...중략
        else if(command.equals("/boardUpdate.naver")) {
            BoardDetailService action = new BoardDetailService();
            action.execute(request);

            RequestDispatcher dispatcher =
            Request.getRequestDispatcher("/board/boardModifyForm.jsp");
            dispatcher.forward(request, response);
        }
    }
    ... 중략
}

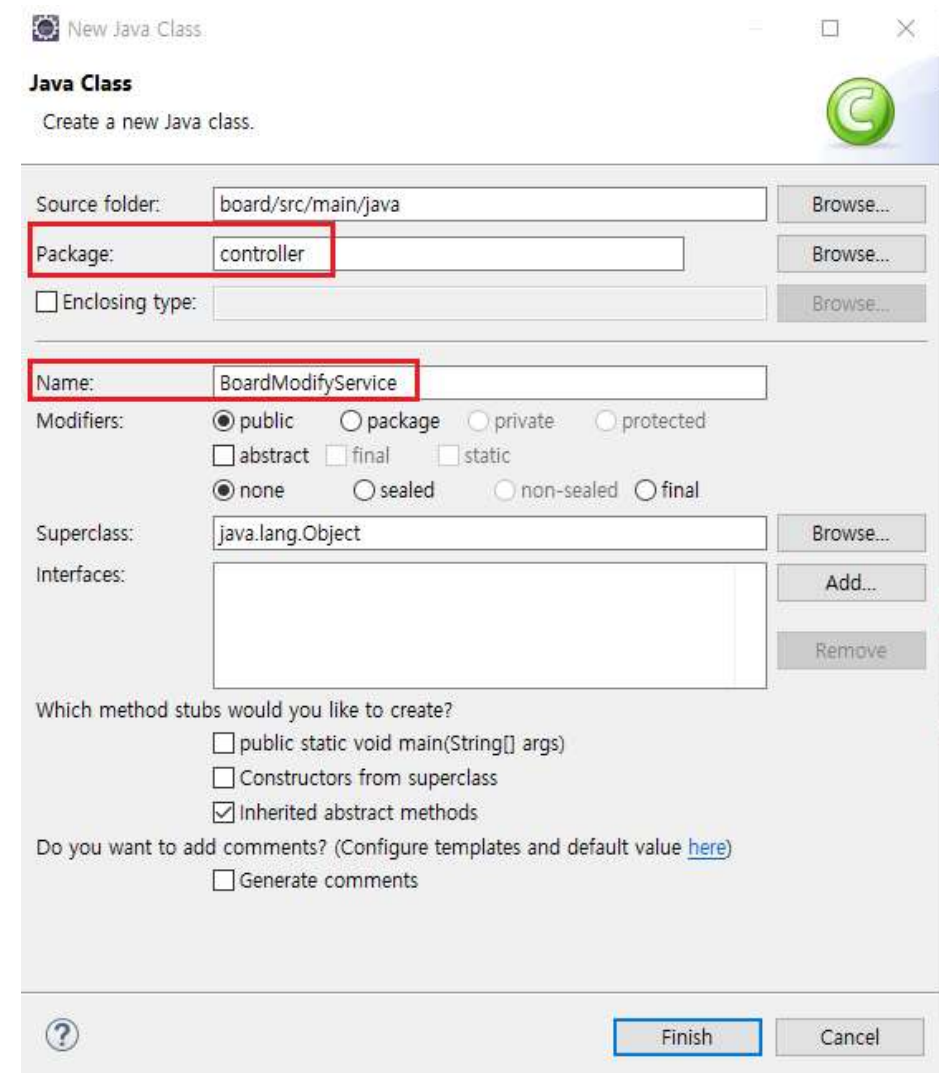
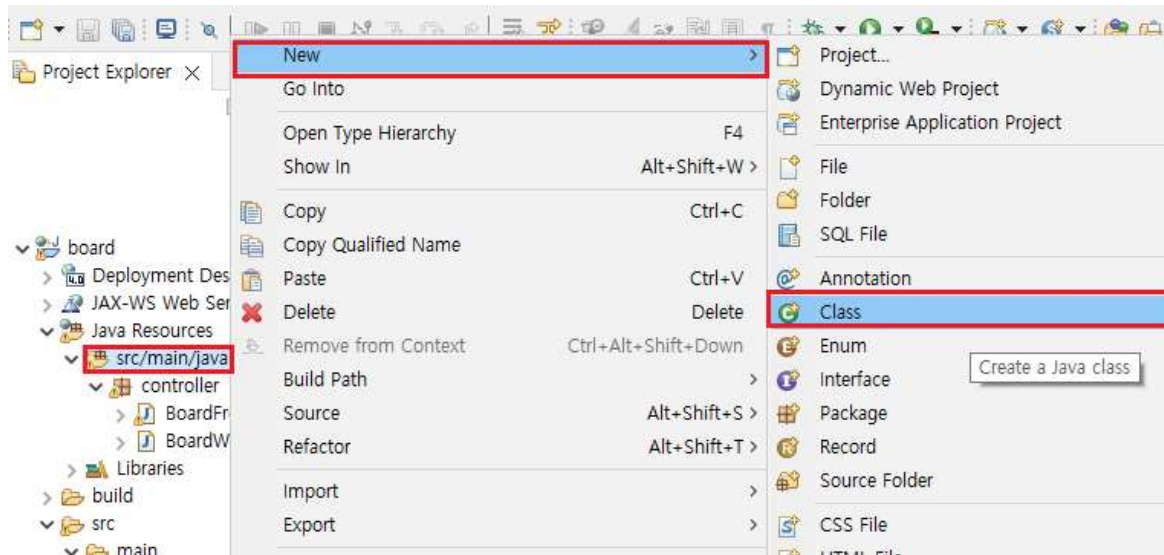
```

boardModifyForm.jsp 페이지를 만든다.

```
<form action= "boardModify.naver" method= "post">
  <input type= "hidden" name= "boardNum" value= "${dto.boardNum}"/>
  <table>
    <caption>게시글 수정</caption>
    <tr>
      <td>글쓴이</td>
      <td><input type= "text" name= "boardWriter" value= "${dto.boardWriter}"/></td>
    </tr>
    <tr>
      <td>제목</td>
      <td><input type= "text" name= "boardSubject" value= "${dto.boardSubject}"/></td>
    </tr>
    <tr>
      <td>내용</td>
      <td><textarea rows= "6" cols= "40" name= "boardContent"> ${dto.boardContent}</textarea></td>
    </tr>
    <tr>
      <th colspan= 2>
        <input type= "submit" value= "게시글 수정 완료" />
        <input type= "button" value= "뒤로가기" onclick= "javascript:history.back()" />
      </th>
    </tr>
  </table>
</form>
```

Front-Controller에서 수정완료를 위한 주소를 작성한다,
수정이 완료가 되면 다시 상세페이지로 이동할 수 있게 만든다.

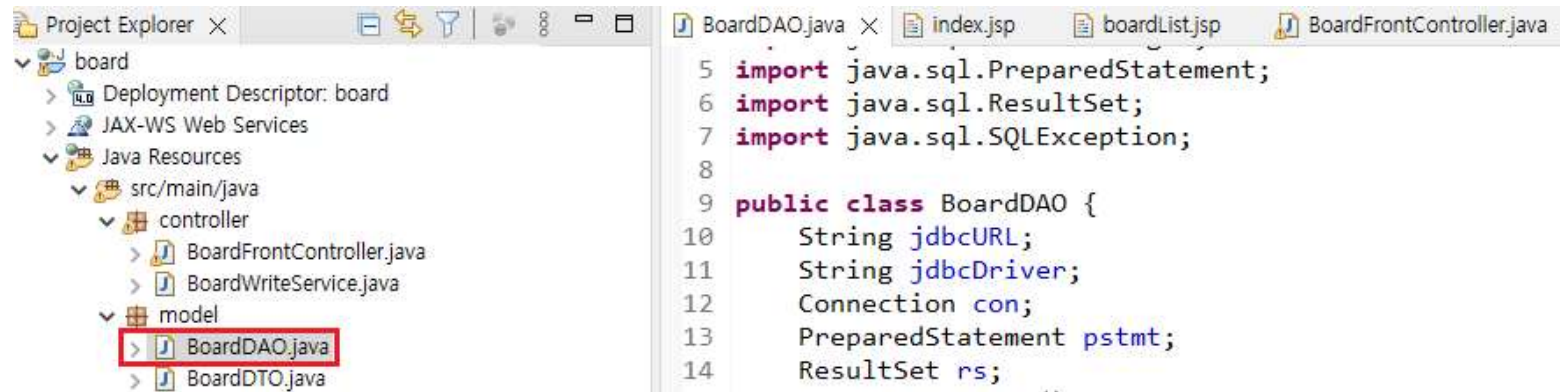
```
public class BoardFrontController extends HttpServlet implements Servlet {  
    protected void doPost(HttpServletRequest request,  
        HttpServletResponse response){  
        ...중략  
        else if(command.equals("/boardModify.naver")) {  
            response.sendRedirect("boardDetail.naver?num=" +  
                request.getParameter("boardNum"));  
        }  
    }  
    ... 중략  
}
```



데이터베이스에서 수정할 수 있게 page-controller를 만들어 준다.
BoardModifyService를 만든다.

```
public class BoardModifyService {  
    public void execute(HttpServletRequest request) {  
        try {  
            request.setCharacterEncoding("utf-8");  
        }catch(Exception e) {}  
        BoardDTO dto = new BoardDTO();  
        dto.setBoardContent(request.getParameter("boardContent"));  
        dto.setBoardNum( Integer.parseInt(request.getParameter("boardNum")));  
        dto.setBoardSubject(request.getParameter("boardSubject"));  
        dto.setBoardWriter(request.getParameter("boardWriter"));  
        BoardDAO dao = new BoardDAO();  
    }  
}
```


DAO 를 열어서 리스트를 가지고 오도록 메서드를 추가한다.



```

public void boardUpdate(BoardDTO dto) {
    con = getConnection();
    String sql = "update board"
        + " set WRITER = ? , SUBJECT = ?, CONTENTS = ? "
        + " where num = ?";

    try {
        pstmt = con.prepareStatement(sql);
        pstmt.setString(1, dto.getBoardWriter());
        pstmt.setString(2, dto.getBoardSubject());
        pstmt.setString(3, dto.getBoardContent());
        pstmt.setInt(4, dto.getBoardNum());
        int i = pstmt.executeUpdate();
        System.out.println(i + "개 행이(가) 수정되었습니다.");
    }catch(Exception e) {e.printStackTrace();}
    finally {
        if(pstmt != null) try{pstmt.close();}catch(Exception e) {}
        if(con != null) try{con.close();}catch(Exception e) {}
    }
}

```

```
public class BoardModifyService {  
    public void execute(HttpServletRequest request) {  
        try {  
            request.setCharacterEncoding("utf-8");  
        } catch (Exception e) {}  
        BoardDTO dto = new BoardDTO();  
        ...  
        BoardDAO dao = new BoardDAO();  
        /// 추가  
        dao.boardUpdate(dto);  
    }  
}
```

Front-Comntroller 에 추가한다.

```
public class BoardFrontController extends HttpServlet implements Servlet {  
    protected void doPost(HttpServletRequest request,  
        HttpServletResponse response){  
        ...중략  
        else if(command.equals("/boardModify.naver")) {  
            BoardModifyService action = new  
                BoardModifyService();  
            action.execute(request);  
            response.sendRedirect("boardDetail.naver?num="+  
                request.getParameter("boardNum"));  
        }  
    }  
    ... 중략  
}
```

/board/boardInfo.jsp에 삭제를 할 수 있게 링크를 만들어준다.

```
<body>
```

```
상세페이지 <br />
```

```
글번호 : ${dto.boardNum}<br />
```

```
글쓴이 : ${dto.boardWriter }<br />
```

```
제목 : ${dto.boardSubject }<br />
```

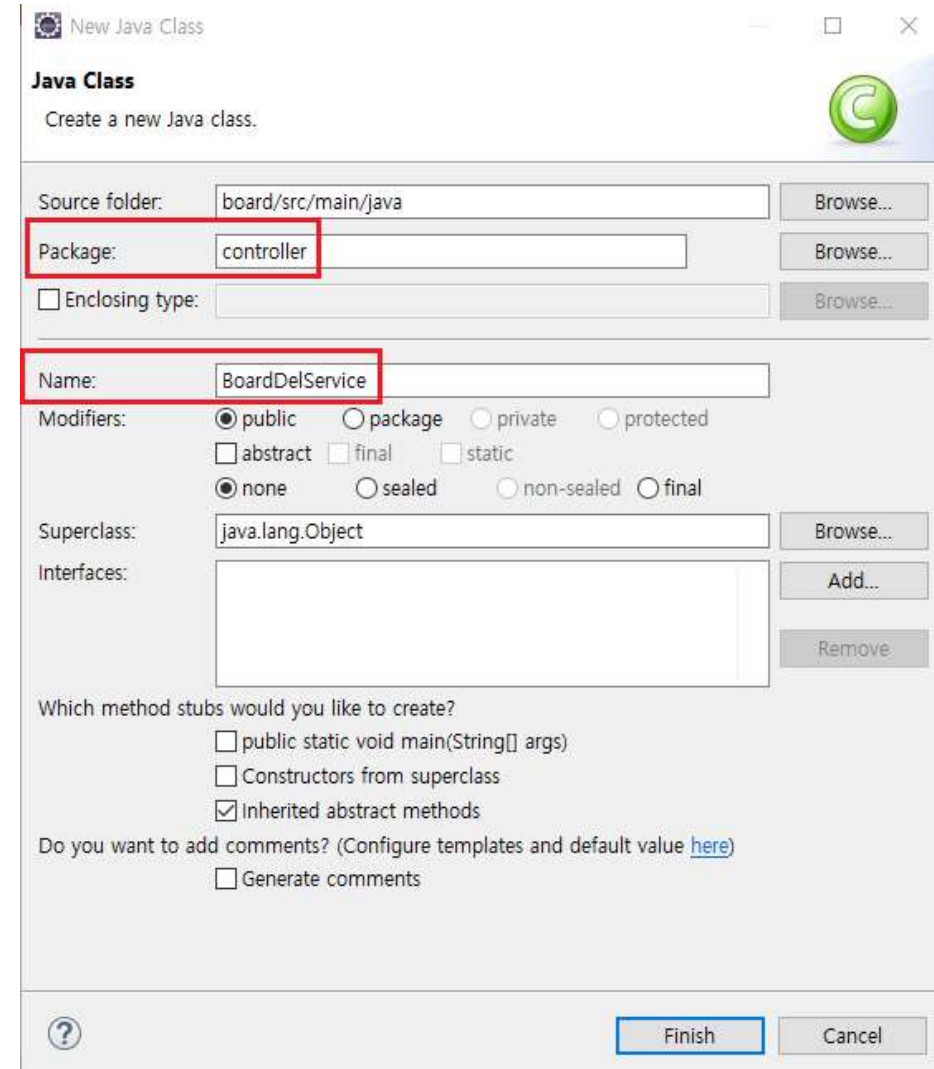
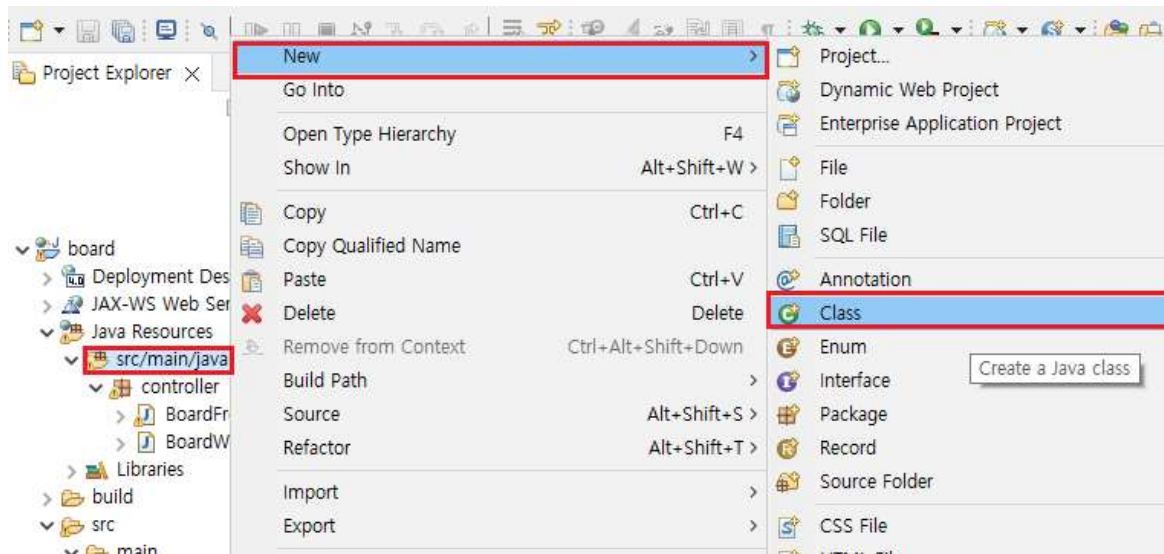
```
내용 : ${dto.boardContent }<br />
```

```
<a href= "boardUpdate.naver?num=${dto.boardNum}">수정</a>
```

```
| <a href= "boardDel.naver?num=${dto.boardNum}">삭제</a>
```

```
| 게시물 리스트
```

```
</body>
```



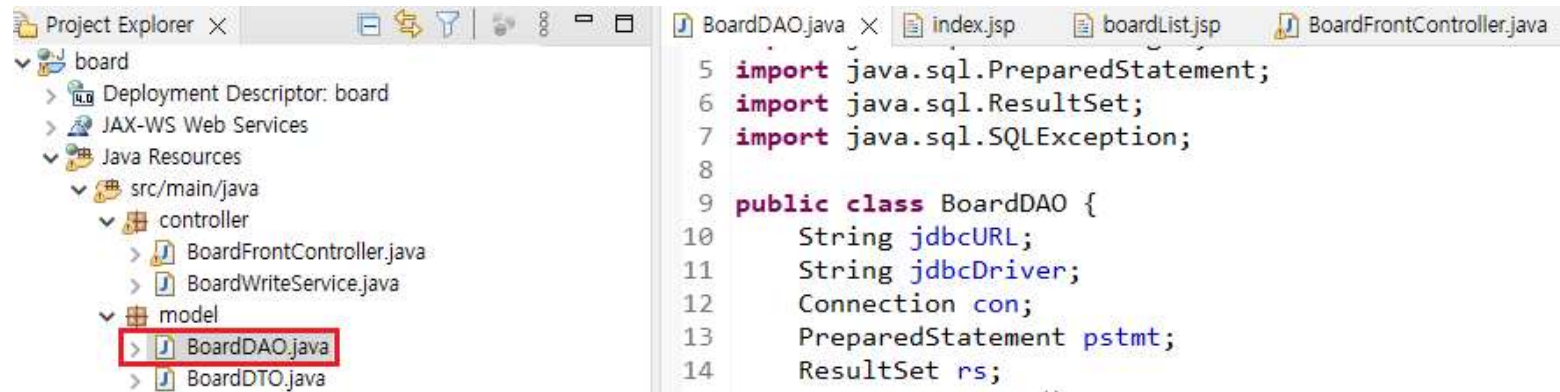
Front-Controller에서 삭제가 되면 목록 페이지로 이동하도록한다.

```
else if(command.equals("/boardDel.naver")) {  
    response.sendRedirect("boardList.naver");  
}
```

데이터베이스에서 삭제가 되도록 page-controller를 만들어준다.

```
public class BoardDelService {  
    public void execute(HttpServletRequest request) {  
        String num = request.getParameter("num");  
        BoardDAO dao = new BoardDAO();  
    }  
}
```

DAO 를 열어서 리스트를 가지고 오도록 메서드를 추가한다.



DAO에 메서드를 추가해준다.

```
public void boardDel(String num) {  
    con = getConnection();  
    String sql = "delete from board where num = ?";  
    try {  
        pstmt = con.prepareStatement(sql);  
        pstmt.setInt(1, Integer.parseInt(num));  
        int i = pstmt.executeUpdate();  
        System.out.println(i + " 개 행이(가) 삭제되었습니다.");  
    }catch(Exception e) {e.printStackTrace();}  
    } finally {  
        if(pstmt != null) try{pstmt.close();}catch(Exception e) {}  
        if(con != null) try{con.close();}catch(Exception e) {}  
    }  
}
```

page-controller메서드를 호출할 수 있게 추가해준다.

```
public class BoardDelService {  
    public void execute(HttpServletRequest request) {  
        String num = request.getParameter("num");  
        BoardDAO dao = new BoardDAO();  
        /// 추가  
        dao.boardDel(num);  
    }  
}
```

Front-Controller에서 BoardDelController의 메서드를 추가한다.

```
else if(command.equals("/boardDel.naver")) {  
    //// 추가  
    BoardDelService action = new BoardDelService();  
    action.execute(request);  
    ////  
  
    response.sendRedirect("boardList.naver");  
}
```