Devops 10

Board

Code Analysis

NAVER Cloud Camp

2조 | 엄지훈, 김태웅, 김수정, 홍화연, 신지현

CHAPTER 1

CODE LIST

Code Analysis

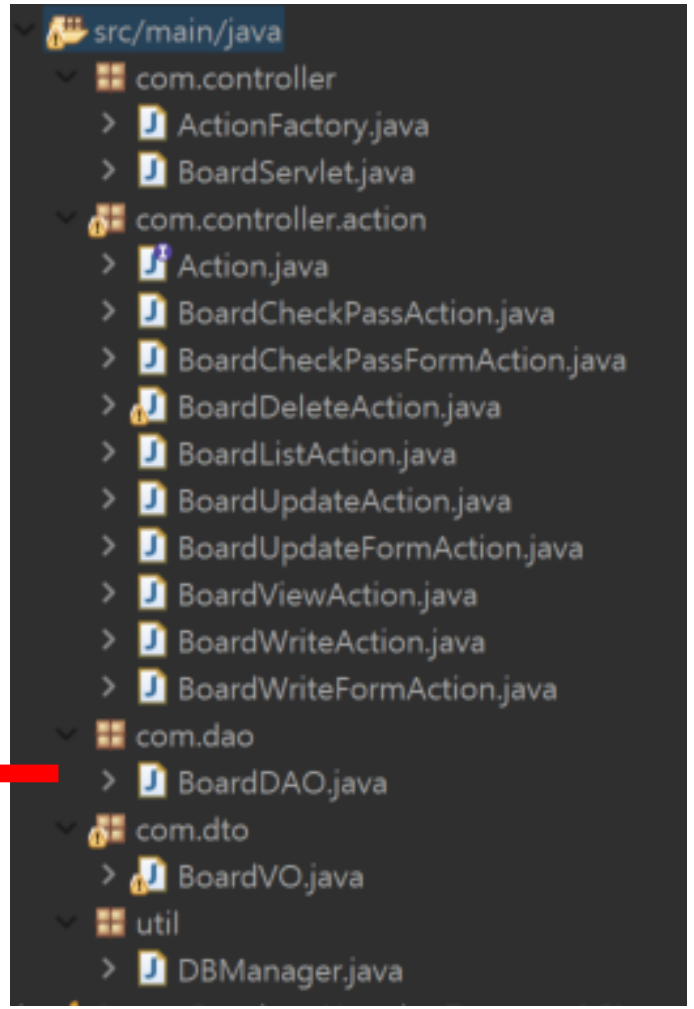
Table

Field	Type	Null	Key	Default	Extra
num	int	NO	PRI	NULL	auto_increment
pass	varchar(30)	YES		NULL	
name	varchar(30)	YES		NULL	
email	varchar(30)	YES		NULL	
title	varchar(50)	YES		NULL	
content	varchar(1000)	YES		NULL	
readcount	int	YES		0	
writedate	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED

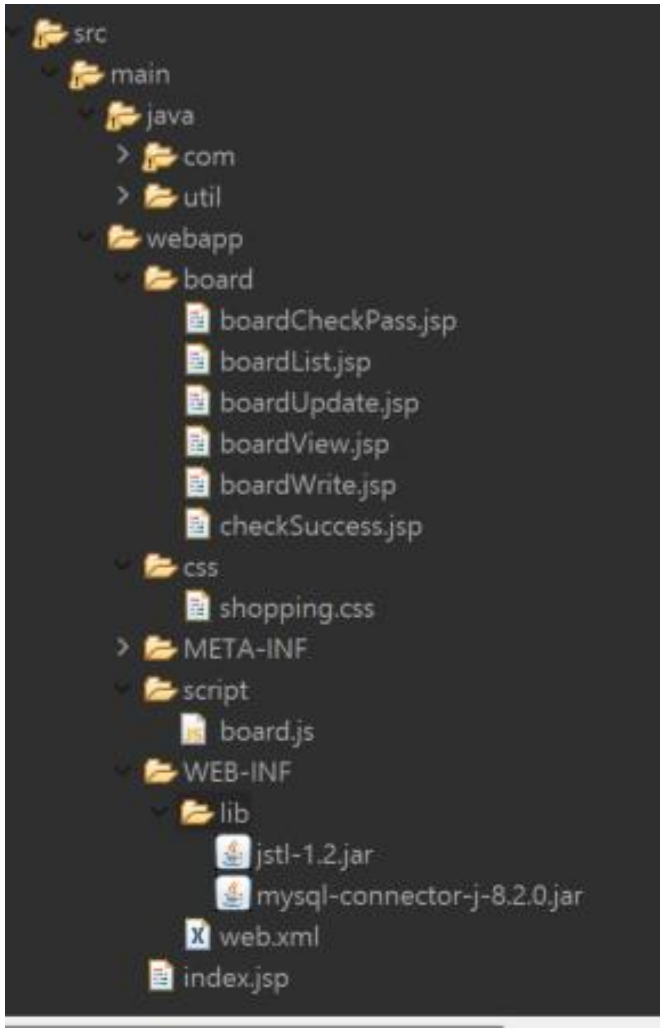
DAO method

class	반환형	메소드
DAO		BoardDAO()
	List<BoardVO>	selectAllBoards()
	void	insertBoard(BoardBean bean)
	int	getAllCount()
	BoardBean	oneBoard(int num)
	BoardBean	oneUpdateBoard(int num)
	String	getPass(int num)
	void	updateBoard(BoardBean bean)
	Vector<BoardBean>	allBoard(int startRow, int endRow)
	void	deleteBoard(int num)
	void	reWriteBoard(BoardBean bean)

Servlet, class



Jsp, css, js



도식도

https://www.figma.com/file/B3jwaF3jF4rNXoZXhXpy25/analyzation?type=design&node-id=0%3A1&mode=design&t=pGgRdX9GjSa3hNHc-1

DB 정보

1. Table 생성

Field	Type	Null	Key	Default	Extra
num	int	NO	PRI	NULL	auto_increment
pass	varchar(30)	YES		NULL	
name	varchar(30)	YES		NULL	
email	varchar(30)	YES		NULL	
title	varchar(50)	YES		NULL	
content	varchar(1000)	YES		NULL	
readcount	int	YES		0	
writedate	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED

2. DBManager.java

- Server에 DB Resorce 입력
- DBManager Class : DB 연결

```
<Context docBase="Test" path="/Test" reloadable="true" source="org.eclipse.jst.jee.server:Test">
    <Resource auth="Container" driverClassName="com.mysql.cj.jdbc.Driver" name="jdbc/pool" password="1234" type="javax.sql.DataS
</Context>
```

```
</Host>
</Engine>
</Service>
```

```
package util;

import java.sql.Connection;

public class DBManager {
    public static Connection getConnection() {
        Connection conn = null;
        try {
             Context ctx=new InitialContext();
               Context env=(Context)ctx.lookup("java:comp/env");
               DataSource ds=(DataSource)env.lookup("jdbc/pool");

               conn=ds.getConnection();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return conn;
    }

    // select을 수행한 후 리소스 해제를 위한 메소드
    public static void close(Connection conn, Statement stmt, ResultSet rs) {
        try {
            rs.close();
            stmt.close();
            conn.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    // DML(insert, update, delete)을 수행한 후 리소스 해제를 위한 메소드
    public static void close(Connection conn, Statement stmt) {
```

2. BoardVO

- DB 정보를 받아올 Bean 생성

```
package com.dto;

import java.sql.Timestamp;

public class BoardVO {
    private int num;
    private String name;
    private String email;
    private String pass;
    private String title;
    private String content;
    private int readcount;
    private Timestamp writedate;

    public int getNum() {
        return num;
    }
    public void setNum(int num) {
        this.num = num;
    }
    public String getName() {
        return name;
    }
}
```

페이지 이동

1. BoardServlet

- command 정보를 받아 ActionFactory 전달

```
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    String command = request.getParameter("command");
    System.out.println("BoardServlet에서 요청을 받음을 확인 : " + command);
    ActionFactory af=ActionFactory.getInstance();
    Action action=af.getAction(command);
    if(action != null){
    action.execute(request, response);
    }
}
```

기능	Page	Action
게시판 목록	(Action을 통해 BoardList.jsp이동)	BoardListAction()
게시물 상세	(Action을 통해 BoardView.jsp이동)	BoardViewAction()
게시물 작성	BoardWriteFormAction()	BoardWriteAction()
비밀번호 확인	BoardCheckPassFormAction()	BoardDeleteAction()
게시물 수정	BoardUpdateFormAction()	BoardUpdateAction()
게시물 삭제	BoardCheckPassFormAction()	BoardCheckPassAction()

2. ActionFactory

- command 정보로 이동할 페이지 또는 Action 지정

```
public Action getAction(String command) {
    Action action = null;
    System.out.println("ActionFactory :" + command);
    /* 추가된 부분 */
    if (command.equals("board_list")) {
        action = new BoardListAction();
    } else if (command.equals("board_write_form")) {
        action = new BoardWriteFormAction();
    } else if (command.equals("board_write")) {
        action = new BoardWriteAction();
    } else if (command.equals("board_view")) {
        action = new BoardViewAction();
    } else if (command.equals("board_check_pass_form")) {
        action = new BoardCheckPassFormAction();
    } else if (command.equals("board_check_pass")) {
        action = new BoardCheckPassAction();
    } else if (command.equals("board_update_form")) {
        action = new BoardUpdateFormAction();
    } else if (command.equals("board_update")) {
        action = new BoardUpdateAction();
    } else if (command.equals("board_delete")) {
        action = new BoardDeleteAction();
    }
    return action;
}
```

BoardServlet

```
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    String command = request.getParameter("command");
    System.out.println("BoardServlet에서 요청을 받음을 확인 : " + command);
    ActionFactory af=ActionFactory.getInstance();
    Action action=af.getAction(command);
    if(action != null){
    action.execute(request, response);
    }
}
```

- command를 request 객체에서 받은 뒤 해당 커맨드를 매개변수로 getAction() 메소드를 호출한다.

- getAction() 메소드가 반환한 객체의 execute() 메소드를 호출하여 페이지를 이동한다.

ActionFactory

```java
public class ActionFactory {  5 usages
    private static ActionFactory instance = new ActionFactory();  1 u

    private ActionFactory() { super(); }

    public static ActionFactory getInstance() { return instance; }

    public Action getAction(String command) {  1 usage
        Action action = null;
        System.out.println("ActionFactory :" + command);
        /* 추가된 부분 */
        if (command.equals("board_list")) {
            action = new BoardListAction();
        } else if (command.equals("board_write_form")) {
            action = new BoardWriteFormAction();
        } else if (command.equals("board_write")) {
            action = new BoardWriteAction();
        } else if (command.equals("board_view")) {
            action = new BoardViewAction();
        } else if (command.equals("board_check_pass_form")) {
            action = new BoardCheckPassFormAction();
        } else if (command.equals("board_check_pass")) {
            action = new BoardCheckPassAction();
        } else if (command.equals("board_update_form")) {
            action = new BoardUpdateFormAction();
        } else if (command.equals("board_update")) {
            action = new BoardUpdateAction();
        } else if (command.equals("board_delete")) {
            action = new BoardDeleteAction();
        }
        return action;
    }
}
```

ActionFactory

- ActionFactory는 싱글톤 방식으로 하나의 인스턴스를 생성하고 있다.
다른 파일에서 ActionFactory의 메소드를 사용할 때 생성된 인스턴스를 사용해서 호출함으로써 실행 속도를 향상시키고 메모리 관리를 용이하게 하고 있다.

- getAction 메소드는 BoardServlet 혹은 다른 파일에서 호출하고 있다.
getAction 메소드는 매개변수에 따라 다양한 객체들을 반환한다.

- getAction 메소드가 반환하는 객체들은 서로 다른 클래스지만, 동일한 Action 인터페이스를 상속받고, execute 메소드를 구현하고 있기 때문에 자동 형변환을 이용해서 Action 객체에 담을 수 있다. 이는 코드의 다형성을 증가시키는 효과를 가지고 있다.

-각각의 객체들이 구현하고 있는 execute 메소드들은 주로 데이터베이스와 연결되어 검색, 수정, 삭제 등의 작업을 수행한 뒤 다른 jsp 페이지로 이동을 시키는 기능을 가지고 있다.

BoardListAction

BoardListAction

```java
public class BoardListAction implements Action{  5 usages
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardList.jsp";
        BoardDAO bDao = BoardDAO.getInstance();
        List<BoardVO> boardList = bDao.selectAllBoards();
        request.setAttribute("boardList", boardList);
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

- BoardListAction은 BoardDAO 객체의 selectAllBoards() 메소드를 이용하여 데이터베이스 내에 존재하는 모든 튜플들을 BoardVO 객체의 배열 형식으로 받아온다.

- BoardListAction은 해당 배열을 request 객체에 담은 뒤 forward 방식으로 boardList.jsp로 이동한다.

BoardWriteFormAction

```java
public class BoardWriteFormAction implements Action {  2 usages
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardWrite.jsp";
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

- BoardWriteFormAction은 forward 방식으로 boardWrite.jsp로 이동시키는 객체이다.
- forward 방식으로 전송되기 때문에 파라미터로 받아온 값은 유지된다.

BoardWriteAction

```java
public class BoardWriteAction implements Action {  2 usages
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        BoardVO bVo = new BoardVO();
        bVo.setName(request.getParameter("name"));
        bVo.setPass(request.getParameter("pass"));
        bVo.setEmail(request.getParameter("email"));
        bVo.setTitle(request.getParameter("title"));
        bVo.setContent(request.getParameter("content"));
        BoardDAO bDao = BoardDAO.getInstance();
        bDao.insertBoard(bVo);
        new BoardListAction().execute(request, response);
    }
}
```

- BoardWriteAction은 BoardVO 객체를 생성한 뒤 request 객체에서 게시글의 정보를 받아서 생성한 객체에 저장한다.
- BoardDAO의 insertBoard 메소드를 통해서 게시글의 데이터들은 데이터베이스에 저장되게 된다.
- 데이터베이스에 게시글의 정보가 저장된 후, BoardListAction 객체를 생성하여 게시글 목록 페이지로 이동한다.

BoardViewAction

BoardViewAction

```java
public class BoardViewAction implements Action {  2 usages

    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardView.jsp";
        String num = request.getParameter("num");
        BoardDAO bDao = BoardDAO.getInstance();
        bDao.updateReadCount(num);
        BoardVO bVo = bDao.selectOneBoardByNum(num);
        request.setAttribute("board", bVo);
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

- BoardViewAction은 request 객체에서 게시글의 번호를 받아서 변수에 저장한다.
- 게시글의 번호를 매개변수로 이용하는 두 개의 메소드를 호출한다.
- updateReadCount 메소드는 해당 게시글의 조회수를 하나 증가시킨다.
- selectOneBoardByNum 메소드는 매개변수로 받은 번호에 해당하는 게시글의 정보들을 BoardVO 객체에 담아서 반환한다.
- 가져온 객체를 request 객체에 담은 뒤 forward 방식으로 boardView.jsp 페이지로 이동한다.

BoardCheckPassFormAction

```java
public class BoardCheckPassFormAction implements Action {  2 usages
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardCheckPass.jsp";
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

- BoardCheckPassFormAction은 forward 방식으로 boardCheckPass.jsp로 이동시키는 객체이다.
- forward 방식으로 전송되기 때문에 파라미터로 받아온 num값은 유지된다.

BoardCheckPassAction

BoardCheckPassAction

```java
public class BoardCheckPassAction implements Action {  2 usages
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = null;
        String num = request.getParameter("num");
        String pass = request.getParameter("pass");
        BoardDAO bDao = BoardDAO.getInstance();
        BoardVO bVo = bDao.selectOneBoardByNum(num);
        if (bVo.getPass().equals(pass)) { // 성공
            url = "/board/checkSuccess.jsp";
        } else {// 실패
            url = "/board/boardCheckPass.jsp";
            request.setAttribute("message", "비밀번호가 틀렸습니다.");
        }
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

- BoardCheckPassAction은 request 객체로부터 게시글의 번호와 비밀번호를 문자열 형태로 받아와서 변수에 저장한다.
- 저장한 게시글의 번호를 매개변수로 이용하여 selectOneBoardByNum 메소드를 호출하고 게시글의 정보를 BoardVO 객체 형식으로 받아온다. 사용자가 입력한 비밀번호와 BoardVO 객체로부터 받아온 비밀번호를 비교하는 유효성 검사를 실시한다.
- 비밀번호가 일치할 경우 forward 방식으로 checkSuccess.jsp 페이지로 이동한다.
- 비밀번호가 일치하지 않을 경우 request 객체에 오류 메시지를 저장 후 forward 방식으로 boardCheckPass.jsp 페이지로 이동한

BoardUpdateFormAction

BoardUpdateFormAction

```
public class BoardUpdateFormAction implements Action {  2 usages
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardUpdate.jsp";
        String num = request.getParameter("num");
        BoardDAO bDao = BoardDAO.getInstance();
        bDao.updateReadCount(num);
        BoardVO bVo = bDao.selectOneBoardByNum(num);
        request.setAttribute("board", bVo);
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

- BoardUpdateFormAction의 경우 request 객체로부터 게시글의 번호를 받아와서 변수에 저장한다.
- BoardDAO의 인스턴스를 사용해 updateReadCount 메소드를 호출하여 해당 게시글의 조회수를 업데이트한다.
- 저장한 게시글의 번호를 매개변수로 이용하여 selectOneBoardByNum 메소드를 호출하고 게시글의 정보를 BoardVO 객체 형식으로 받아온다.
- 가져온 객체를 request 객체에 담은 뒤 forward 방식으로 boardUpdate.jsp 페이지로 이동한다.

BoardUpdateAction

BoardUpdateAction

```java
public class BoardUpdateAction implements Action {  2 usages
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        BoardVO bVo = new BoardVO();
        bVo.setNum(Integer.parseInt(request.getParameter("num")));
        bVo.setName(request.getParameter("name"));
        bVo.setPass(request.getParameter("pass"));
        bVo.setEmail(request.getParameter("email"));
        bVo.setTitle(request.getParameter("title"));
        bVo.setContent(request.getParameter("content"));
        BoardDAO bDao = BoardDAO.getInstance();
        bDao.updateBoard(bVo);
        new BoardListAction().execute(request, response);
    }
```

- BoardUpdateAction의 경우 BoardVO 객체를 생성 후 request 객체로부터 받아온 게시글의 정보들을 저장한다.
- BoardDAO의 updateBoard 메소드를 이용하여 데이터베이스에 있는 게시글의 정보를 수정한다.
- BoardListAction 객체를 생성하여 boardList.jsp 페이지로 이동한다.

BoardDeleteAction

```
public class BoardDeleteAction implements Action {  2 usages
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String num=request.getParameter("num");
        BoardDAO bDao=BoardDAO.getInstance();
        bDao.deleteBoard(num);
        new BoardListAction().execute(request, response);
    }
}
```

- BoardDeleteAction의 경우 request 객체로부터 게시글의 번호를 받아와서 변수에 저장한다.

- 가져온 게시글의 번호와 BoardDAO의 인스턴스를 사용하여 deleteBoard 메소드를 호출해 게시글을 삭제한다
.

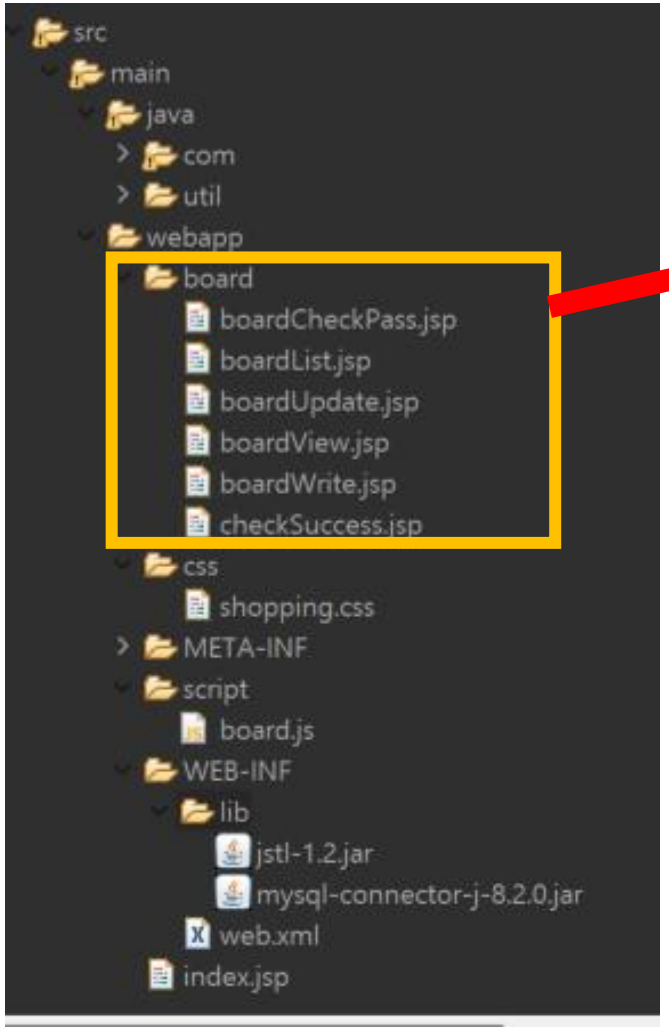
- 게시글을 삭제한 후 BoardListAction 객체를 생성하여 boardList.jsp 페이지로 이동한다.

CHAPTER 2

PAGE DETAILS

Code Analysis

페이지



1. 게시글 목록

게시글 리스트

게시글 등록

번호	제목	작성자	작성일	조회
1	첫방문	홍길동	2024. 4. 29.	1

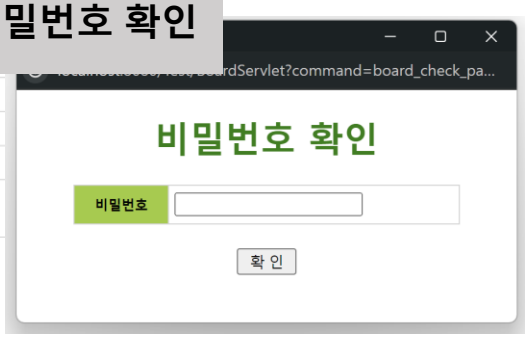
2. 게시글 상세

게시글 상세보기

작성자	홍길동	이메일	aa@naver.com
작성일	2024. 4. 29.	조회수	5
제목	첫방문		
내용	안녕		

게시글 수정 | 게시글 삭제 | 게시글 리스트 | 게시글 등록

3. 비밀번호 확인



4. 게시글 작성

게시글 등록

작성자	* 필수
비밀번호	* 필수 (게시물 수정 삭제시 필요합니다.)
이메일	
제목	* 필수
내용	

등록 | 다시 작성 | 목록

5. 게시글 수정

게시글 수정

작성자	홍길동 * 필수
비밀번호	* 필수 (게시물 수정 삭제시 필요합니다.)
이메일	aa@naver.com
제목	첫방문
내용	안녕

등록 | 다시 작성 | 목록

게시판 목록

참조: Index.jsp, BoardServlet, ActionFactory, BoardListAction, BoardDAO, BoardList.jsp

1. Index.jsp 호출

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dt
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>
<%
response.sendRedirect("http://localhost:8080/Test/BoardServlet?command=board_list");
%>
</body>
</html>
```

- BoardServlet 호출
- 이동 페이지(command): **board_list**

2. BoardServlet

Command 정보 전달

3. ActionFactory

BoardListAction 호출

4. BoardListAction

- BoardDAO로부터 게시판 정보를 List객체로 받아 출력

```java
public class BoardListAction implements Action{
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardList.jsp";
        BoardDAO bDao = BoardDAO.getInstance();
        List<BoardVO> boardList = bDao.selectAllBoards();
        request.setAttribute("boardList", boardList);
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

5. BoardDAO

- selectAllBoards()로 DB 정보를 List객체로 반환

```java
public List<BoardVO> selectAllBoards() {
    String sql = "select * from board order by num desc";
    List<BoardVO> list = new ArrayList<BoardVO>();
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    try {
        conn = DBManager.getConnection();
        stmt = conn.createStatement();
        rs = stmt.executeQuery(sql);
        while (rs.next()) {
            BoardVO bVo = new BoardVO();
            bVo.setNum(rs.getInt("num"));
            bVo.setName(rs.getString("name"));
            bVo.setEmail(rs.getString("email"));
            bVo.setPass(rs.getString("pass"));
            bVo.setTitle(rs.getString("title"));
            bVo.setContent(rs.getString("content"));
            bVo.setReadcount(rs.getInt("readcount"));
            bVo.setWritedate(rs.getTimestamp("writedate"));
            list.add(bVo);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        DBManager.close(conn, stmt, rs);
    }
    return list;
}
```

6. BoardList 화면 출력

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dt
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
<link rel="stylesheet" type="text/css" href="css/shopping.css">
</head>
<body>
    <div id="wrap" align="center">
        <h1>게시글 리스트</h1>
        <table class="list">
            <tr>
                <td colspan="5" style="border: white; text-align: right"><a
                    href="BoardServlet?command=board_write_form">게시글 등록</a></td>
            </tr>
            <tr>
                <th>번호</th>
                <th>제목</th>
                <th>작성자</th>
                <th>작성일</th>
                <th>조회</th>
```

화면 출력

BoardList.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
<link rel="stylesheet" type="text/css" href="css/shopping.css">
</head>
<body>
    <div id="wrap" align="center">
        <h1>게시글 리스트</h1>
        <table class="list">
            <tr>
                <td colspan="5" style="border: white; text-align: right"><a
                    href="BoardServlet?command=board_write_form">게시글 등록</a></td>
            </tr>
            <tr>
                <th>번호</th>
                <th>제목</th>
                <th>작성자</th>
                <th>작성일</th>
                <th>조회</th>
            </tr>
            <!-- JSTL 반복문을 활용하여 서버로부터 받아온 boardList객체를 전부 브라우저에 출력한다. -->
            <c:forEach var="board" items="${boardList }">
                <tr class="record">
                    <td>${board.num }</td>
                    <td><a href="BoardServlet?command=board_view&num=${board.num}">
                            ${board.title } </a></td>
                    <td>${board.name}</td>
                    <td><fmt:formatDate value="${board.writedate }" /></td>
                    <td>${board.readcount}</td>
                </tr>
            </c:forEach>
        </table>
    </div>
</body>
</html>
```

- BoardListAction에서 데이터베이스에 있는 모든 게시글들의 정보가 들어있는 boardList 배열을 받아온다.
- forEach 태그를 이용하여 boardList 안에 들어있는 모든 게시글들의 정보를 테이블 형식으로 출력한다.
- 게시글의 제목에는 a태그를 설정하여 클릭이 될 경우 url을 이용하여 BoardServlet으로 요청을 보낸다.
- 해당 url은 쿼리스트링을 이용하여 command와 게시글의 번호를 패러미터로 가진다.
- 게시글 등록 버튼을 클릭할 경우 url을 이용하여 BoardServlet으로 요청을 보낸다.
- 해당 url은 쿼리스트링을 이용하여 command를 패러미터로 가진다.

게시판 상세

참조: BoardList.jsp, BoardServlet, ActionFactory, BoardViewAction, BoardDAO, BoardWiew

1. BoardList 게시물 선택

게시글 리스트

게시글 등록

번호	제목	작성자	작성일	조회
1	첫방문	홍길동	2024. 4. 29.	10

```
<td>
    <a href="BoardServlet?command=board_view&num=${board.num}">${board.title } </a>
</td>
<td>${board.name}</td>
<td><fmt:formatDate value="${board.writedate }" /></td>
<td>${board.readcount}</td>
```

- BoardList에서 게시물 제목 선택
- 이동 페이지(command): **board_View**
- board.num 정보 쿼리스트링으로 전달

2. BoardServlet → 3. ActionFactory

Command 정보 전달 → BoardViewAction 호출

4. BoardViewAction

- BoardList에서 받은 num값을 DAO에 저장
- BoardDAO에서 조회수(updateReadCount), 게시글 읽기(selectOneBoardByNum) 실행
- 위의 작업들 수행 후 게시글 정보를 담아 (board) boardView.jsp로 이동

```
public class BoardViewAction implements Action {

    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardView.jsp";
        String num = request.getParameter("num");
        BoardDAO bDao = BoardDAO.getInstance();
        bDao.updateReadCount(num);
        BoardVO bVo = bDao.selectOneBoardByNum(num);
        request.setAttribute("board", bVo);
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

5. BoardDAO

- 매개변수로 받은 num값에 맞는 게시글 조회수 (updateReadCount) 증가하여 DB에 저장

```
public void updateReadCount(String num) {
    String sql = "update board set readcount=readcount+1 where num=?";
    Connection conn = null;
    PreparedStatement pstmt = null;
    try {
        conn = DBManager.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, num);
        pstmt.executeUpdate();
```

- 매개변수로 받은 num값에 맞는 게시글 정보 (selectOneBoardByNum)를 Bean에 저장

```
// 게시판 글 상세 내용 보기 :글번호로 찾아온다. : 실패 null,
public BoardVO selectOneBoardByNum(String num) {
    String sql = "select * from board where num = ?";
    BoardVO bVo = null;
    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;
    try {
        conn = DBManager.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, num);
```

6. BoardView

- bean에 저장한 게시물 정보(board)에서 데이터를 불러와 화면 출력

화면 출력

게시글 상세보기

작성자	홍길동	이메일	aa@naver.com
작성일	2024. 4. 29.	조회수	12
제목	첫방문		
내용	안녕		

게시글 수정 | 게시글 삭제 | 게시글 리스트 | 게시글 등록

출력내용	출력정보
작성자	${board.name}
이메일	${board.email}
작성일	${board.writedate}
조회수	${board.readcount }
제목	${board.title }
내용	${board.content }

BoardServlet → **ActionFactory**

Command 정보 전달 — BoardViewAction 호출

```
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    //각 페이지에서 요청(command) 값을 받아옴.
    String command = request.getParameter("command");
    System.out.println("BoardServlet에서 요청을 받음을 확인 : " + command);

    //싱글톤으로 ActionFactory 의 인스턴스를 받아옴.
    ActionFactory af=ActionFactory.getInstance();

    //ActionFactory의 getAction 메소드로 각 요청에 맞는 객체를 생성한 후 전송한다.
    Action action=af.getAction(command);
    if(action != null){
    action.execute(request, response);
    }
}
```

→

```
25      public Action getAction(String command) {
26          Action action = null;
27          System.out.println("ActionFactory :" + command);
28          /* 추가된 부분 */
29          if (command.equals("board_list")) {
30              action = new BoardListAction();
31          } else if (command.equals("board_write_form")) {
32              action = new BoardWriteFormAction();
33          } else if (command.equals("board_write")) {
34              action = new BoardWriteAction();
35          } else if (command.equals("board_view")) {
36              action = new BoardViewAction();
37          } else if (command.equals("board_check_pass_form")) {
38              action = new BoardCheckPassFormAction();
39          } else if (command.equals("board_check_pass")) {
40              action = new BoardCheckPassAction();
41          } else if (command.equals("board_update_form")) {
42              action = new BoardUpdateFormAction();
43          } else if (command.equals("board_update")) {
44              action = new BoardUpdateAction();
45          } else if (command.equals("board_delete")) {
46              action = new BoardDeleteAction();
47          }
48          return action;
49      }
50  }
```

BoardView.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
<link rel="stylesheet" type="text/css" href="css/shopping.css">
<script type="text/javascript" src="script/board.js"></script>
</head>
<body>
    <div id="wrap" align="center">
        <h1>게시글 상세보기</h1>
        <table>
            <tr>
                <th>작성자</th>
                <td>${board.name}</td>
                <th>이메일</th>
                <td>${board.email}</td>
            </tr>
            <tr>
                <th>작성일</th>
                <td><fmt:formatDate value="${board.writedate}" /></td>
                <th>조회수</th>
                <td>${board.readcount }</td>
            </tr>
            <tr>
                <th>제목</th>
                <td colspan="3">${board.title }</td>
            </tr>
            <tr>
                <th>내용</th>
                <td colspan="3"><pre>${board.content }</pre></td>
            </tr>
        </table>
        <br> <br> <input type="button" value="게시글 수정"
            onclick="open_win('BoardServlet?command=board_check_pass_form&num=${board.num}', 'update')">
        <input type="button" value="게시글 삭제"
            onclick="open_win('BoardServlet?command=board_check_pass_form&num=${board.num}', 'delete')">
        <input type="button" value="게시글 리스트"
            onclick="location.href='BoardServlet?command=board_list'">
        <input type="button" value="게시글 등록"
            onclick="location.href='BoardServlet?command=board_write_form'">
    </div>
</body>
</html>
```

- VO객체 board 에 저장된 데이터를 받아와서 '게시글 상세보기' 테이블에 출력한다.
- 테이블에는 작성자, 이메일, 작성일, 조회수, 제목, 내용 등의 정보가 출력된다.
- 게시글 수정과 게시글 삭제 버튼이 클릭될 경우 open_win 함수를 호출하여 새로운 창을 띄운 뒤 입력받은 url로 이동한다.
- 해당 url에은 command와 게시글의 번호를 패러미터로 가진다.
- 게시글 리스트와 게시글 등록 버튼을 누를 경우 각각 다른 command를 패러미터로 가진 채 BoardServlet으로 이동한다.

게시물 작성

참조: BoardList.jsp, BoardServlet, ActionFactory, BoardDAO, BoardWriteFormAction, BoardWriteFormAction,Board.js

1. BoardList 게시글 등록

```
<h1>게시글 리스트</h1>
<table class="list">
    <tr>
        <td colspan="5" style="border: white; text-align: right">
            <a href="BoardServlet?command=board_write_form">게시글 등록</a>
        </td>
    </tr>
```

- BoardServlet 호출
- 이동 페이지(command): **board_write_form**

2. BoardServlet

Command 정보 전달

3. ActionFactory

BoardWriteFormAction 호출

4. BoardWriteFormAction

```
public class BoardWriteFormAction implements Action {
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardWrite.jsp";
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

- boardWrite.jsp 화면 이동

5. BoardWrite 화면 출력

화면 출력

1. 등록: onclick="return boardCheck()"
2. 다시 작성: input type="reset"
3. 목록: onclick으로 BoardList 페이지 이동

입력정보	type	name
작성자	text	name
비밀번호	password	pass
이메일	text	email
제목	text	title
내용	textarea	content

6. Board.js - boardCheck()

```
function boardCheck() {
    if (document.frm.name.value.length == 0) {
        alert("작성자를 입력하세요.");
        return false;
    }
    if (document.frm.pass.value.length == 0) {
        alert("비밀번호를 입력하세요.");
        return false;
    }
    if (document.frm.title.value.length == 0) {
        alert("제목을 입력하세요.");
        return false;
    }
    return true;
}
```

입력 정보를 받아 유효성 체크

- 작성자, 비밀번호, 제목

boardWrite.jsp

```jsp
<%@ 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 PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
<link rel="stylesheet" type="text/css" href="css/shopping.css">
<script type="text/javascript" src="script/board.js"></script>
</head>
<body>
    <div id="wrap" align="center">
        <h1>게시글 등록</h1>
        <form name="frm" method="post" action="BoardServlet">
            <input type="hidden" name="command" value="board_write">
            <table>
                <tr>
                    <th>작성자</th>
                    <td><input type="text" name="name"> * 필수</td>
                </tr>
                <tr>
                    <th>비밀번호</th>
                    <td><input type="password" name="pass"> * 필수 (게시물 수정
                        삭제시 필요합니다.)</td>
                </tr>
                <tr>
                    <th>이메일</th>
                    <td><input type="text" name="email"></td>
                </tr>
                <tr>
                    <th>제목</th>
                    <td><input type="text" size="70" name="title"> * 필수</td>
                </tr>
                <tr>
                    <th>내용</th>
                    <td><textarea cols="70" rows="15" name="content"></textarea></td>
                </tr>
            </table>
            <br>
            <br> <input type="submit" value="등록"
                onclick="return boardCheck()"> <input type="reset"
                value="다시 작성"> <input type="button" value="목록"
                onclick="location.href='BoardServlet?command=board_list'">
        </form>
    </div>
</body>
</html>
```

```js
function boardCheck() {
    if (document.frm.name.value.length == 0) {
        alert("작성자를 입력하세요.");
        return false;
    }
    if (document.frm.pass.value.length == 0) {
        alert("비밀번호를 입력하세요.");
        return false;
    }
    if (document.frm.title.value.length == 0) {
        alert("제목을 입력하세요.");
        return false;
    }
    return true;
}
```

- BoardWrite.jsp에서 [등록] 버튼을 누르면(onclick), board.js 파일에서 boardCheck() 메서드를 실행시킨다.
- boardCheck 메서드는 클라이언트의 form(frm)의 각 input의 name의 값들이 입력되었는지 빈값인지를 확인하고, 빈값일 경우 경고창을 띄운다.
- 작성자,비밀번호,제목이 모두 입력되었다면 true를 반환한다.

BoardServlet → **ActionFactory**

Command 정보 전달

Board_write호출

```
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    //각 페이지에서 요청(command) 값을 받아옴.
    String command = request.getParameter("command");
    System.out.println("BoardServlet에서 요청을 받음을 확인 : " + command);

    //싱글톤으로 ActionFactory 의 인스턴스를 받아옴.
    ActionFactory af=ActionFactory.getInstance();

    //ActionFactory의 getAction 메소드로 각 요청에 맞는 객체를 생성한 후 전송한다.
    Action action=af.getAction(command);
    if(action != null){
    action.execute(request, response);
    }
}
```

→

```
public static ActionFactory getInstance() {
    return instance;
}

public Action getAction(String command) {
    Action action = null;
    System.out.println("ActionFactory :" + command);
    /* 추가된 부분 */
    if (command.equals("board_list")) {
        action = new BoardListAction();
    } else if (command.equals("board_write_form")) {
        action = new BoardWriteFormAction();
    } else if (command.equals("board_write")) {
        action = new BoardWriteAction();
    } else if (command.equals("board_view")) {
        action = new BoardViewAction();
    } else if (command.equals("board_check_pass_form")) {
        action = new BoardCheckPassFormAction();
    } else if (command.equals("board_check_pass")) {
        action = new BoardCheckPassAction();
    } else if (command.equals("board_update_form")) {
        action = new BoardUpdateFormAction();
    } else if (command.equals("board_update")) {
        action = new BoardUpdateAction();
    } else if (command.equals("board_delete")) {
        action = new BoardDeleteAction();
    }
    return action;
}
```

비밀번호 확인

참조: boardView.jsp, BoardServlet, ActionFactory, BoardCheckPassFormAction, BoardCheckPass, BoardDAO, Board.js, BoardCheckAction, checkSuccess.jsp

1. BoardView

게시글 상세보기

게시글 수정 | 게시글 삭제 | 게시글 리스트 | 게시글 등록

- 게시글 수정 및 삭제 onclick="open_win(url, name)" 버튼 선택
- url=이동 페이지(command): **board_check_pass_form**
  쿼리스트링: 게시글 번호 ${board.num} 이동
- name='update'

2. BoardServlet → 3. ActionFactory

Command 정보 전달

BoardCheckPassFormAction 호출

4. BoardCheckPassFormAction

boardCheckPass.jsp 페이지로 이동

```
public class BoardCheckPassFormAction implements Action {
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String url = "/board/boardCheckPass.jsp";
        RequestDispatcher dispatcher = request.getRequestDispatcher(url);
        dispatcher.forward(request, response);
    }
}
```

5. BoardCheckPass 페이지

- 비밀번호 확인 문자 입력(pass)
- 입력 값 유효성 검사 onclick="return passCheck()"
- form 입력 시 BoardServlet 이동
- hidden으로 num, command 값 넘김
- 이동 페이지(command): **board_check_pass**

```
<div align="center">
    <h1>비밀번호 확인</h1>
    <form action="BoardServlet" name="frm" method="get">
        <input type="hidden" name="command" value="board_check_pass">
        <input type="hidden" name="num" value="${param.num}">
        <table style="width: 80%">
            <tr>
                <th>비밀번호</th>
                <td><input type="password" name="pass" size="20"></td>
            </tr>
        </table>
        <br> <input type="submit" value=" 확 인 "
            onclick="return passCheck()"> <br>
        <br>${message}
    </form>
</div>
```

6-1. Board.js - passCheck()

- 비밀번호 문자 미입력 시 알림창

```
function passCheck() {
    if (document.frm.pass.value.length == 0) {
        alert("비밀번호를 입력하세요.");
        return false;
    }
    return true;
}
```

6-2. BoardServlet → ActionFactory → BoardCheckAction

- 전달받은 "num"값 기준 비밀번호와 입력한 비밀번호 "pass"값 비교
- 값이 같은 경우 checkSuccess.jsp 이동
- 값이 다른 경우 message에 값 set

비밀번호 다름

비밀번호 같음

7. checkSuccess.jsp

```
<script type="text/javascript">
if (window.name == "update") {
window.opener.parent.location.href = "BoardServlet?command=board_update_form&num=${param.num}";
} else if (window.name == 'delete') {
alert('삭제되었습니다.');
window.opener.parent.location.href = "BoardServlet?command=board_delete&num=${param.num}";
}
window.close();
</script>
```

- 전달받은 name값이 "update"일 경우
  → BoardServlet에 command: board_update_form 전달
- 전달받은 name값이 "delete"일 경우
  → BoardServlet에 command: board_delete 전달

BoardView.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
<link rel="stylesheet" type="text/css" href="css/shopping.css">
<script type="text/javascript" src="script/board.js"></script>
</head>
<body>
    <div id="wrap" align="center">
        <h1>게시글 상세보기</h1>
        <table>
            <tr>
                <th>작성자</th>
                <td>${board.name}</td>
                <th>이메일</th>
                <td>${board.email}</td>
            </tr>
            <tr>
                <th>작성일</th>
                <td><fmt:formatDate value="${board.writedate}" /></td>
                <th>조회수</th>
                <td>${board.readcount }</td>
            </tr>
            <tr>
                <th>제목</th>
                <td colspan="3">${board.title }</td>
            </tr>
            <tr>
                <th>내용</th>
                <td colspan="3"><pre>${board.content }</pre></td>
            </tr>
        </table>
        <br> <br> <input type="button" value="게시글 수정"
            onclick="open_win('BoardServlet?command=board_check_pass_form&num=${board.num}', 'update')">
        <input type="button" value="게시글 삭제"
            onclick="open_win('BoardServlet?command=board_check_pass_form&num=${board.num}', 'delete')">
        <input type="button" value="게시글 리스트"
            onclick="location.href='BoardServlet?command=board_list'">
        <input type="button" value="게시글 등록"
            onclick="location.href='BoardServlet?command=board_write_form'">
    </div>
</body>
</html>
```

```js
function open_win(url, name) {
    window.open(url, name, "width=500, height=230");
}
```

- BoardView.jsp에서 [게시글 수정] 혹은 [게시글 삭제] 버튼을 누르면(onclick), board.js 파일에서 open_win 메서드를 실행시킨다.
- open_win 메서드의 매개변수로 새 윈도우 창에 띄울 url 값 과 각 버튼의 기능을 구별하기 위한 window name 값을 넘겨준다.
- 실행결과 새 윈도우 창이 뜨면서 url 값으로 넘긴 command = “board_check_pass_form” 이 BoardServlet에서 실행된다. 이 때 수정 혹은 삭제 하는 게시글이 무엇인지를 판별하기 위해서 board.num 값을 같이 넘긴다.

BoardServlet → **ActionFactory**

Command 정보 전달

BoardCheckPassFormAction 호출

```
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    //각 페이지에서 요청(command) 값을 받아옴.
    String command = request.getParameter("command");
    System.out.println("BoardServlet에서 요청을 받음을 확인 : " + command);

    //싱글톤으로 ActionFactory 의 인스턴스를 받아옴.
    ActionFactory af=ActionFactory.getInstance();

    //ActionFactory의 getAction 메소드로 각 요청에 맞는 객체를 생성한 후 전송한다.
    Action action=af.getAction(command);
    if(action != null){
    action.execute(request, response);
    }
}
```

→

```
17    private ActionFactory() {
18        super();
19    }
20
21    public static ActionFactory getInstance() {
22        return instance;
23    }
24
25    public Action getAction(String command) {
26        Action action = null;
27        System.out.println("ActionFactory :" + command);
28        /* 추가된 부분 */
29        if (command.equals("board_list")) {
30            action = new BoardListAction();
31        } else if (command.equals("board_write_form")) {
32            action = new BoardWriteFormAction();
33        } else if (command.equals("board_write")) {
34            action = new BoardWriteAction();
35        } else if (command.equals("board_view")) {
36            action = new BoardViewAction();
37        } else if (command.equals("board_check_pass_form")) {
38            action = new BoardCheckPassFormAction();
39        } else if (command.equals("board_check_pass")) {
40            action = new BoardCheckPassAction();
41        } else if (command.equals("board_update_form")) {
42            action = new BoardUpdateFormAction();
43        } else if (command.equals("board_update")) {
44            action = new BoardUpdateAction();
45        } else if (command.equals("board_delete")) {
46            action = new BoardDeleteAction();
47        }
48        return action;
49    }
50 }
51
```

BoardCheckPass.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
<link rel="stylesheet" href="css/shopping.css">
<script type="text/javascript" src="script/board.js"></script>
</head>
<body>
    <div align="center">
        <h1>비밀번호 확인</h1>
        <form action="BoardServlet" name="frm" method="get">
            <input type="hidden" name="command" value="board_check_pass">
            <input type="hidden" name="num" value="${param.num}">
            <table style="width: 80%">
                <tr>
                    <th>비밀번호</th>
                    <td><input type="password" name="pass" size="20"></td>
                </tr>
            </table>
            <br> <input type="submit" value=" 확 인 "
                onclick="return passCheck()"> <br>
            <br>${message}
        </form>
    </div>
</body>
</html>
```

- 여러 번 데이터를 전송하면서 최종적으로 BoardCheckPass.jsp 폼이 새 윈도우 브라우저에 출력된다.
- 사용자가 비밀번호 입력창에 비밀번호를 적고 submit 하게 되면 board.js 에서 passCheck() 유효성 검사를 실시한다. (input type password 값이 null 이면 false를 반환)
- 유효성 검사를 통과하면 사용자가 입력한 비밀번호 및 BoardView.jsp 에서 넘어온 num 값을 BoardServlet 에 전송한다. (command = "board_check_pass")

```js
function passCheck() {
    if (document.frm.pass.value.length == 0) {
        alert("비밀번호를 입력하세요.");
        return false;
    }
    return true;
}
```

BoardServlet → **ActionFactory**

Command 정보 전달

BoardCheckPassAction 호출

```
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
    //각 페이지에서 요청(command) 값을 받아옴.
    String command = request.getParameter("command");
    System.out.println("BoardServlet에서 요청을 받음을 확인 : " + command);

    //싱글톤으로 ActionFactory 의 인스턴스를 받아옴.
    ActionFactory af=ActionFactory.getInstance();

    //ActionFactory의 getAction 메소드로 각 요청에 맞는 객체를 생성한 후 전송한다.
    Action action=af.getAction(command);
    if(action != null){
    action.execute(request, response);
    }
}
```

→

```
17    private ActionFactory() {
18        super();
19    }
20
21    public static ActionFactory getInstance() {
22        return instance;
23    }
24
25    public Action getAction(String command) {
26        Action action = null;
27        System.out.println("ActionFactory :" + command);
28        /* 추가된 부분 */
29        if (command.equals("board_list")) {
30            action = new BoardListAction();
31        } else if (command.equals("board_write_form")) {
32            action = new BoardWriteFormAction();
33        } else if (command.equals("board_write")) {
34            action = new BoardWriteAction();
35        } else if (command.equals("board_view")) {
36            action = new BoardViewAction();
37        } else if (command.equals("board_check_pass_form")) {
38            action = new BoardCheckPassFormAction();
39        } else if (command.equals("board_check_pass")) {
40            action = new BoardCheckPassAction();
41        } else if (command.equals("board_update_form")) {
42            action = new BoardUpdateFormAction();
43        } else if (command.equals("board_update")) {
44            action = new BoardUpdateAction();
45        } else if (command.equals("board_delete")) {
46            action = new BoardDeleteAction();
47        }
48        return action;
49    }
50 }
51
```

checkSuccess.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>
<script type="text/javascript">
if (window.name == "update") {
window.opener.parent.location.href = "BoardServlet?command=board_update_form&num=${param.num}";
} else if (window.name == 'delete') {
alert('삭제되었습니다.');
window.opener.parent.location.href = "BoardServlet?command=board_delete&num=${param.num}";
}
window.close();
</script>
</body>
</html>
```

- window.name 이 update 일 경우, board_update_form 으로 num값을 같이 전송한다.
- window.name 이 delete 일 경우, board_delete 로 num값을 같이 전송하고 '삭제되었습니다' 라는 alert 창을 띄운다.

게시물 수정

참조: boardUpdate.jsp, BoardServlet, BoardUpdateFormAction, BoardListAction, BoardDAO, boardUpdate, board.js

1. checkSuccess.jsp

```
<script type="text/javascript">
if (window.name == "update") {
window.opener.parent.location.href = "BoardServlet?command=board_update_form&num=${param.num}";
} else if (window.name == 'delete') {
alert('삭제되었습니다.');
window.opener.parent.location.href = "BoardServlet?command=board_delete&num=${param.num}";
}
window.close();
</script>
```

- 전달받은 name값이 "update"일 경우 이동 페이지(command): **board_update_form**

↓

2. BoardServlet

Command 정보 전달

↓

3. ActionFactory

BoardUpdateFormAction 호출

4. BoardUpdateFormAction.jsp

- BoardList에서 받은 num값을 DAO에 저장
- BoardDAO에서 조회수(updateReadCount), 게시글 읽기(selectOneBoardByNum) 실행
- 위의 작업들 수행 후 게시글 정보를 담아 (board) boardUpdate.jsp로 이동

```
public void execute(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    String url = "/board/boardUpdate.jsp";
    String num = request.getParameter("num");
    BoardDAO bDao = BoardDAO.getInstance();
    bDao.updateReadCount(num);
    BoardVO bVo = bDao.selectOneBoardByNum(num);
    request.setAttribute("board", bVo);
    RequestDispatcher dispatcher = request.getRequestDispatcher(url);
    dispatcher.forward(request, response);
}
}
```

5. BoardDAO

- 매개변수로 받은 num값에 맞는 게시글 조회수 (updateReadCount) 증가하여 DB에 저장

```
public void updateReadCount(String num) {
    String sql = "update board set readcount=readcount+1 where num=?";
    Connection conn = null;
    PreparedStatement pstmt = null;
    try {
        conn = DBManager.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, num);
        pstmt.executeUpdate();
```

- 매개변수로 받은 num값에 맞는 게시글 정보 (selectOneBoardByNum)를 Bean에 저장

```
// 게시판 글 상세 내용 보기 :글번호로 찾아온다. : 실패 null,
public BoardVO selectOneBoardByNum(String num) {
    String sql = "select * from board where num = ?";
    BoardVO bVo = null;
    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;
    try {
        conn = DBManager.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, num);
```

6. boardUpdate

- bean에 저장한 게시물 정보(board)에서 데이터를 불러와 화면 출력(작성자, 이메일, 제목 , 내용)

화면 출력

1. 등록: onclick="return boardCheck()"
2. 다시 작성: input type="reset"
3. 목록: onclick으로 BoardList 페이지 이동

입력정보	type	name
작성자	text	name
비밀번호	password	pass
이메일	text	email
제목	text	title
내용	textarea	content

게시물 수정

```
<br> <br> <input type="button" value="게시글 수정"
    onclick="open_win('BoardServlet?command=board_check_pass_form&num=${board.num}', 'update')">
```

```
function open_win(url, name) : void  {  Show usages
    window.open(url, name,  features: "width=500, height=230");
}
```

```
function passCheck() : boolean  {  Show usages
    if (document.frm.pass.value.length == 0) {
        alert("비밀번호를 입력하세요.");
        return false;
    }
    return true;
}
```

- boardView.jsp의 게시글 수정 버튼을 클릭하면 board.js의 open_win 함수가 호출되어 새로운 팝업 창을 열고, 해당 창에서 비밀번호 확인 절차를 진행하기 위해 BoardServlet으로 요청을 보낸다.
- url은 요청을 보낼 서블릿의 URL(BoardCheckPassFormAction)과 해당 서블릿에서 처리할 명령 및 추가적인 파라미터를 포함한다 (num - 게시글 번호).
- 사용자가 수정을 원하는 게시글의 번호를 서블릿에 전달한다.
- 사용자는 새로 열린 창에서 비밀번호를 입력하여 해당 게시글의 비밀번호와 일치하는 검증받는다(passCheck()).
- 비밀번호가 확인되면 BoardUpdateFormAction을 호출한다.

updateBoard

```
<input type="submit" value="등록" onclick="return boardCheck()">
<input type="reset" value="다시 작성">
<input type="button" value="목록" onclick="location.href='BoardServlet?command=board_list'">
```

```
public void updateBoard(BoardVO bVo) {  1 usage
    String sql = "update board set name=?, email=?, pass=?, "
            + "title=?, content=? where num=?";
    Connection conn = null;
    PreparedStatement pstmt = null;
    try {
        conn = DBManager.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString( parameterIndex: 1, bVo.getName());
        pstmt.setString( parameterIndex: 2, bVo.getEmail());
        pstmt.setString( parameterIndex: 3, bVo.getPass());
        pstmt.setString( parameterIndex: 4, bVo.getTitle());
        pstmt.setString( parameterIndex: 5, bVo.getContent());
        pstmt.setInt( parameterIndex: 6, bVo.getNum());
        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        DBManager.close(conn, pstmt);
    }
}
```

- BoardUpdateFormAction을 호출하여 선택한 게시글의 번호를 가져와 BoardDAO의 updateReadCount메소드와 selectOneBoardByNum메소드를 호출하여 해당 게시글의 상세 내용과 조회수를 조회하고,
해당 내용을 bVo에 저장하여 boardUpdate.jsp에 전달한다. (${board.~}로 출력)

- boardUpdate로 출력한 화면에 필수 항목인 작성자명과 비밀번호를 입력하고, 수정을 원하는 내용을 작성하여 게시글 수정 버튼을 클릭하면, BoardUpdateAction이 호출해 bVo에 작성한 내용이 저장되고,
해당 내용을 updateBoard(bVo) 메소드를 호출하여 게시글 내용이 수정된다.

- 다시 작성 버튼을 누르면 입력된 내용이 초기화되고,
목록 버튼을 누르면 BoardListAction이 호출되어 boardList.jsp 창으로 이동한다.

게시물 삭제

참조: checkSucces.jsp, BoardServlet, ActionFactory, BoardDeleteAction, BoardDAO

1. checkSuccess.jsp

```
<script type="text/javascript">
if (window.name == "update") {
window.opener.parent.location.href = "BoardServlet?command=board_update_form&num=${param.num}";
} else if (window.name == 'delete') {
alert('삭제되었습니다.');
window.opener.parent.location.href = "BoardServlet?command=board_delete&num=${param.num}";
}
window.close();
</script>
```

- 전달받은 name값이 "delete"일 경우 이동 페이지(command): **board_delete**

↓

2. BoardServlet

Command 정보 전달

↓

3. ActionFactory

BoardDeleteAction 호출

4. BoardDeleteAction

- 전달받은 num값을 DAO의 deleteBoard(num)의 매개변수로 저장

```
public class BoardDeleteAction implements Action {
    @Override
    public void execute(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String num=request.getParameter("num");
        BoardDAO bDao=BoardDAO.getInstance();
        bDao.deleteBoard(num);
        new BoardListAction().execute(request, response);
    }
}
```

5. BoardDAO

- 매개변수로 받은 num값에 맞는 게시글 DB 정보 삭제

```
public void deleteBoard(String num) {
    String sql = "delete from board where num=?";
    Connection conn = null;
    PreparedStatement pstmt = null;
    try {
        conn = DBManager.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, num);
        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

화면 출력

게시글 상세보기

작성자	홍길동	이메일	aa@naver.com
작성일	2024. 4. 29.	조회수	14
제목	첫방문		
내용	안녕		

게시글 수정

Insert title here - Chrome

localhost:8080/Test/BoardServlet?command=board_check_pass_form&...

비밀번호 확인

비밀번호

확 인

게시물 삭제

```
<input type="button" value="게시글 삭제"
    onclick="open_win('BoardServlet?command=board_check_pass_form&num=${board.num}', 'delete')">
```

```
function open_win(url, name) : void  {  Show usages
    window.open(url, name,  features: "width=500, height=230");
}
```

```
function passCheck() : boolean  {  Show usages
    if (document.frm.pass.value.length == 0) {
        alert("비밀번호를 입력하세요.");
        return false;
    }
    return true;
}
```

- boardView.jsp의 게시글 삭제 버튼을 클릭하면 board.js의 open_win 함수가 호출되어 새로운 팝업 창을 열고, 해당 창에서 비밀번호 확인 절차를 진행하기 위해 BoardServlet으로 요청을 보낸다.
- url은 요청을 보낼 서블릿의 URL(BoardCheckPassFormAction)과 해당 서블릿에서 처리할 명령 및 추가적인 파라미터를 포함한다 (num - 게시글 번호).
- 사용자가 삭제를 원하는 게시글의 번호를 서블릿에 전달한다.
- 사용자는 새로 열린 창에서 비밀번호를 입력하여 해당 게시글의 비밀번호와 일치하는 검증받는다(passCheck()).
- 비밀번호가 확인되면 BoardDeleteAction을 호출한다.

deleteBoard

```java
public void deleteBoard(String num) {  1 usage
    String sql = "delete from board where num=?";
    Connection conn = null;
    PreparedStatement pstmt = null;
    try {
        conn = DBManager.getConnection();
        pstmt = conn.prepareStatement(sql);
        pstmt.setString( parameterIndex: 1, num);
        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

- BoardDeleteAction을 호출하여 삭제하려는 게시글의 번호를 다시 가져온 후,
  deleteBoard(num) 메소드를 호출해 해당 게시글을 삭제한다.

- 삭제한 후 new BoardListAction().execute(request, response) 메소드를 호출해
  게시글 목록 페이지(boardList.jsp)로 이동하여 해당 게시글이 삭제되었는지 확인할 수 있다.