# 4장. 화면 개발 - JSP

*jsp*

# 화면 개발

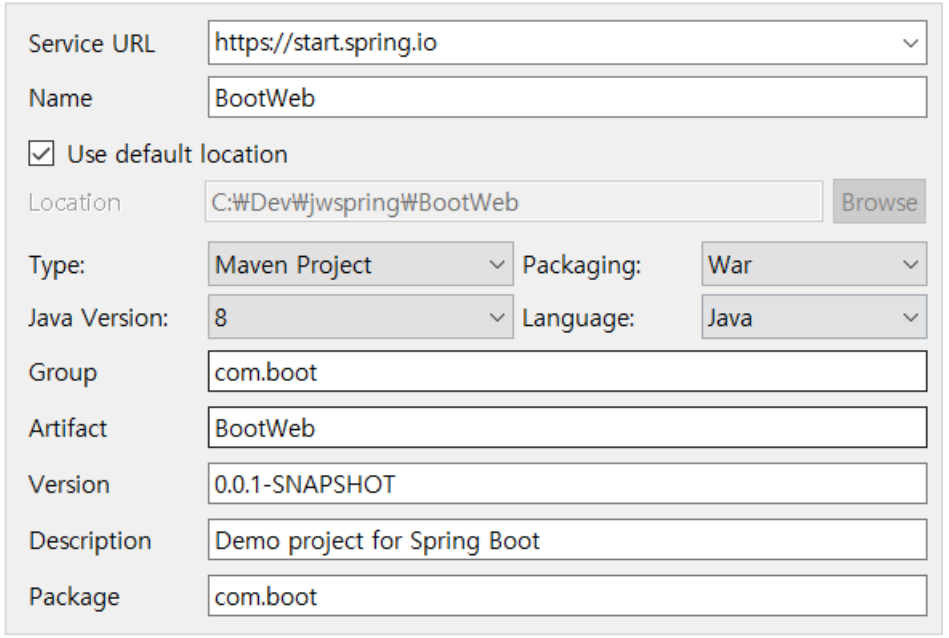
- **웹 애플리케이션 화면 개발**

  스프링 부트는 템플릿 엔진을 이용한 화면 처리를 지원한다.
  스프링 부트가 지원하는 템플릿 엔진은 타임리프(Thymeleaf)를 비롯하여 Freemaker, Mustache, Groovy Templates이 있다.
  템플릿 엔진을 이용하면 데이터와 거의 완벽하게 분리된 화면을 개발 할 수 있기 때문에 순수하게 HTML 파일 만을 이용한 화면 개발이 가능하고, 운영 과정에서 쉽게 화면을 변경할 수도 있다.

  최종적으로 스프링 부트가 제공하는 템플릿 엔진 중에서 타임리프를 적용하는데, 먼저 JSP 기반의 화면을 만들고 타임리프를 적용한다.
  JSP는 자바의 표준이며, JSP로 여전히 화면을 제공하는 시스템들이 존재하기 때문이다.

# 화면 개발

- **BootWeb 프로젝트 생성 및 환경 설정**




Packing – War

- 모듈 추가

Lombok, Spring Web, DevTools

Spring Data JPA, H2 Database

# 화면 개발

- **JSP를 사용하기 위한 라이브러리 추가 및 설정 파일 변경**

```
<dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>jstl</artifactId>
</dependency>
<dependency>
    <groupId>org.apache.tomcat.embed</groupId>
    <artifactId>tomcat-embed-jasper</artifactId>
</dependency>
```

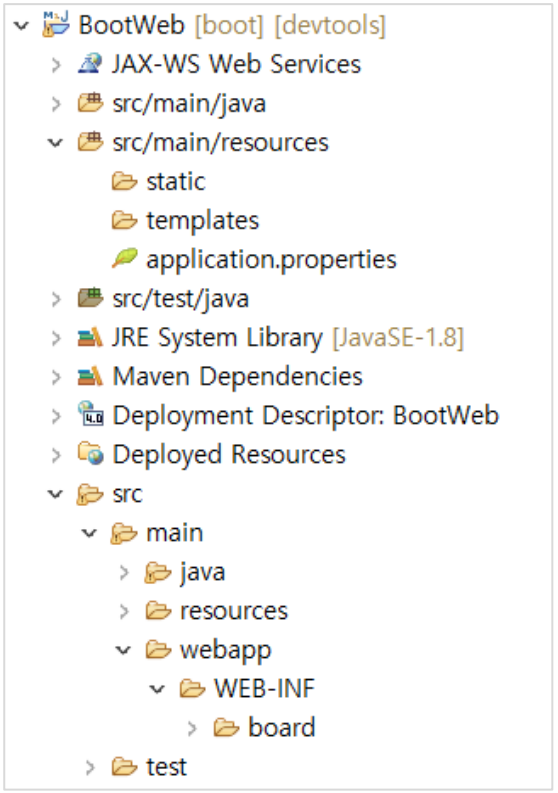
tomcat-embed-jasper는 JSP 파일을 컴파일 할 때 사용하는 라이브러리이다.

application.properties 파일

```
## ViewResolver Setting
spring.mvc.view.prefix=/WEB-INF/board/
spring.mvc.view.suffix=.jsp
```

# 화면 개발

- **JSP를 사용하기 위한 설정 추가**




Webapp 밑에

WEB-INF와 하위에 board 폴더를

직접 만들어 줌

# 화면 개발

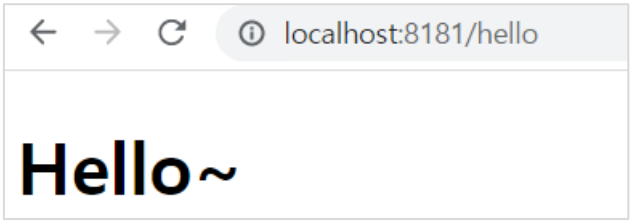
- **BoardController 클래스 – hello.jsp**

```java
@Controller
public class BoardController {

    @GetMapping("/hello")
    public String hello() {
        return "hello";
    }
```

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Hello~</title>
</head>
<body>
    <h1>Hello~</h1>
</body>
</html>
```

# 화면 개발

- Spring Boot App 실행 및 브라우저 요청




```java
@SpringBootApplication
public class BootWebApplication {

    public static void main(String[] args) {
        SpringApplication.run(BootWebApplication.class, args);
    }

}
```

# 화면 개발

- 게시글 목록 조회하기

Board 클래스

```
@Getter
@Setter
public class Board {
    private int seq;
    private String title;
    private String writer;
    private String content;
    private Date createDate;
    private int cnt;
}
```

# 화면 개발

- **BoardController 클래스**

```
@GetMapping("/getBoardList")
public String getBoardList(Model model) {
    List<Board> boardList = new ArrayList<>();

    for(int i=1; i<=10; i++) {
        Board board = new Board();
        board.setSeq((long)i);
        board.setTitle("게시판 프로그램 테스트");
        board.setWriter("아기상어");
        board.setContent("게시판 프로그램 테스트입니다...");
        board.setCreateDate(new Date());
        board.setCnt(0L);

        boardList.add(board);
    }
    model.addAttribute("boardList", boardList);

    return "getBoardList";
}
```

# 화면 개발

- **BoardController 클래스**

## 게시글 목록

| 번호 | 제목 | 작성자 | 등록일 | 조회수 |
|---|---|---|---|---|
| 1 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 2 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 3 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 4 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 5 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 6 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 7 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 8 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 9 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |
| 10 | 게시판 프로그램 테스트 | 아기상어 | Sun Sep 18 06:04:33 KST 2022 | 0 |

새글 등록

# 화면 개발

- **getBoardList.jsp**

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
```

```
<div id="container">
    <h2>게시글 목록</h2>
    <table>
        <thead>
            <tr>
                <th>번호</th><th>제목</th><th>작성자</th><th>등록일</th><th>조회수</th>
            </tr>
        </thead>
        <tbody>
            <c:forEach items="${boardList}" var="board">
            <tr>
                <td><c:out value="${board.seq}" /></td>
                <td>
                     <a href='/getBoard?seq=<c:out value="${board.seq}" />'>
                        <c:out value="${board.title}" />
                     </a>
                 </td>
                <td><c:out value="${board.writer}" /></td>
```

# 화면 개발

- **getBoardList.jsp**

```
                <td>
                    <fmt:formatDate value="${board.createDate}" pattern="yyyy/MM/dd HH:mm:ss"/>
                </td>
                <td><c:out value="${board.cnt}" /></td>
            </tr>
            </c:forEach>
        </tbody>
    </table>
    <p><a href="/insertBoard">새글 등록</a></p>
</div>
```

# 화면 개발

- **JPA와 DB 연동하기**

(1) 엔티티 클래스로 변환 - Board.java

```
@Getter
@Setter
@Entity
public class Board {
    @Id @GeneratedValue
    private Long seq;

    private String title;

    @Column(updatable=false)
    private String writer;
    private String content;

    @Column(insertable=false, updatable=false,
            columnDefinition = "timestamp default current_timestamp")
    private Date createDate;

    @Column(insertable=false, updatable=false,
            columnDefinition = "bigint default 0")
    private Long cnt;
}
```

# 화면 개발

- 엔티티 클래스로 변환 - Board.java

  - @Entity : Board 클래스를 엔티티로 처리하기 위해 클래스 위에 를 명시
  - @Id : seq 변수를 식별자 필드로 처리하도록 추가함
  - @GeneratedValue : 자동으로 증가된 값 할당 위해 추가
  - @Column

    writer, regDate, cnt 칼럼은 제외하도록 updatable 속성 추가

    regDate, cnt 칼럼은 기본값을 설정했으므로 insertable 속성을 false로 함

    또한 default 속성을 추가하여 NULL 대신 기본값이 설정되도록 함

# 화면 개발

- **JPA와 DB 연동하기**

(2) 테이블 생성 설정

```
# JPA Setting
spring.jpa.hibernate.ddl-auto=create
spring.jpa.generate-ddl=false
spring.jpa.show-sql=true
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.properties.hibernate.format_sql=true
```

# 화면 개발

- **JPA와 DB 연동하기**

(3) 리포지터리 인터페이스

```
package com.boot.persistence;

import org.springframework.data.jpa.repository.JpaRepository;

import com.boot.domain.Board;

public interface BoardRepository extends JpaRepository<Board, Long>{

}
```

# 화면 개발

- **비즈니스 컴포넌트 만들기**

(1) Service 인터페이스

```
public interface BoardService {

    List<Board> getBoardList();  //목록 보기

    void insertBoard(Board board);  //새글 등록

    Board getBoard(long seq);  //글 상세보기

    void updateBoard(Board board); //글 수정하기

    void deleteBoard(Board board); //글 삭제하기

    void updateCnt(Long seq);  //조회수
}
```

# 화면 개발

- **Service 구현 클래스**

```java
@Service
public class BoardServiceImpl implements BoardService {

    @Autowired
    private BoardRepository boardRepo;

    //게시글 목록 보기
    @Override
    public List<Board> getBoardList() {
        return (List<Board>) boardRepo.findAll();
    }

    //게시글 등록
    @Override
    public void insertBoard(Board board) {
        boardRepo.save(board);
    }

    //게시글 상세 조회
    @Override
    public Board getBoard(long seq) {
        return boardRepo.findById(seq).get();
    }
```

# 화면 개발

```
//게시글 수정
@Override
public void updateBoard(Board board) {
    /*Board findBoard = boardRepo.findById(board.getSeq()).get();
    findBoard.setTitle(board.getTitle());
    findBoard.setContent(board.getContent());*/
    boardRepo.save(board);
}

//게시글 삭제
@Override
public void deleteBoard(Board board) {
    boardRepo.delete(board);
}

//조회수 증가
@Transactional  //트랜잭션 처리하는 어노테이션
@Override
public void updateCnt(Long seq) {
    boardRepo.updateCnt(seq);
}
```

# 화면 개발

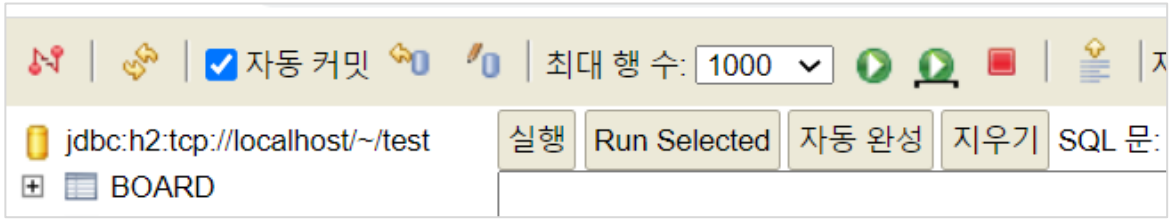
- 비즈니스 컴포넌트 사용하기

```
@Controller
public class BoardController {

    @Autowired
    private BoardService service;

    @GetMapping("/getBoardList")
    public String getBoardList(Model model) {
        List<Board> boardList = service.getBoardList();

        model.addAttribute("boardList", boardList);

        return "getBoardList";
    }
```

# 화면 개발

- **게시글 목록 확인**


H2에 Board 테이블 생성 확인




# 화면 개발

- JSP 화면 디자인 – style.css

```
# static resource
spring.mvc.static-path-pattern=/static/**
```

```
@charset "UTF-8";

/* common 스타일 */
#container{width: 1000px; margin: 20px auto; text-align: center}
a{text-decoration: none;}
h2{padding-top: 20px;}

/* getBoardList 스타일 */
.tbl_list{width: 700px; margin: 0 auto; text-align: center;}
.tbl_list, th, td{border: 1px solid #ccc; border-collapse: collapse;}
.tbl_list th, td{padding: 10px;}
.tbl_list thead{background: #eee;}
```

# 화면 개발

- JSP 화면 디자인 – css 사용

```css
/* insertBoard 스타일 */
.tbl_reg{width: 700px; margin: 0 auto;}
.tbl_reg, td{border: 1px solid #ccc; border-collapse: collapse;}
.tbl_reg td{padding: 10px;}
.tbl_reg input{width: 580px; height: 25px;}
.tbl_reg textarea{width: 580px;}

/* getBoard 스타일*/
.tbl_view{width: 700px; margin: 0 auto;}
.tbl_view, td{border: 1px solid #ccc; border-collapse: collapse;}
.tbl_view td{padding: 10px;}
.tbl_view input{width: 580px; height: 25px;}
.tbl_view textarea{width: 580px;}
```

# 화면 개발

- **게시글 쓰기**

ddl-auto 설정을 update로 변경함

```
# JPA Setting
spring.jpa.hibernate.ddl-auto=update
spring.jpa.generate-ddl=false
spring.jpa.show-sql=true
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.properties.hibernate.format_sql=true
```

# 화면 개발

- **게시글 쓰기**

(1) BoardController 작성

```
//게시글 등록
@Override
public void insertBoard(Board board) {
    boardRepo.save(board);
}
```

```
//게시글 등록 폼 요청
@GetMapping("/insertBoard")
public String insertBoard() {
    return "insertBoard";
}

//게시글 등록 처리
@PostMapping("/insertBoard")
public String insertBoard(Board board) {
    service.insertBoard(board);
    return "redirect:getBoardList";
}
```

# 화면 개발

- **게시글 쓰기**

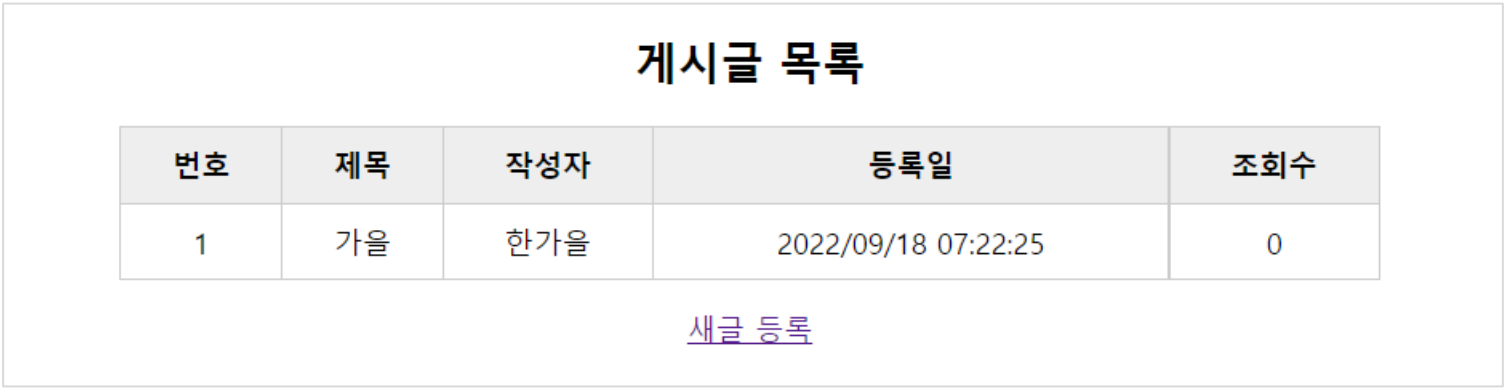
(2) 등록 화면

```
Hibernate:
    insert
    into
        board
        (content, title, writer, seq)
    values
        (?, ?, ?, ?)
```

# 화면 개발

- **게시글 쓰기**

(2) 목록 화면




# 화면 개발

- **게시글 쓰기**

(2) 등록 화면 – insertBoard.jsp

```
<title>글쓰기</title>
<link rel="stylesheet" href="/static/css/style.css">
</head>
<body>
    <div id="container">
        <h2>글쓰기</h2>
        <form action="/insertBoard" method="post">
            <table class="tbl_reg">
                <tr>
                    <td>제목</td>
                    <td><input type="text" name="title"></td>
                </tr>
                <tr>
                    <td>작성자</td>
                    <td><input type="text" name="writer"></td>
                </tr>
```

# 화면 개발

- **게시글 쓰기**

(2) 등록 화면 – insertBoard.jsp

```
        <tr>
            <td>내용</td>
            <td>
                <textarea name="content" rows="10" cols="50"></textarea>
            </td>
        </tr>
        <tr>
            <td colspan="2">
               <button type="submit">새글 등록</button>
            </td>
        </tr>
    </table>
</form>
```

# 화면 개발

- **게시글 상세 조회**

(2) 상세 조회 처리

```
//게시글 상세 조회
@GetMapping("/getBoard")
public String getBoard(Long seq, Model model) {
    Board board = service.getBoard(seq);
    model.addAttribute(board);
    return "getBoard";
}
```

```
//게시글 상세 조회
@Override
public Board getBoard(long seq) {
    return boardRepo.findById(seq).get();
}
```

# 화면 개발

- **게시글 상세 조회**

(2) 상세 조회 처리

제목에 링크 걸기 – 상세 화면 이동

```
<tbody>
    <c:forEach items="${boardList}" var="board">
    <tr>
        <td><c:out value="${board.seq}" /></td>
        <td>
            <a href='/getBoard?seq=<c:out value="${board.seq}" />'>
               <c:out value="${board.title}" />
            </a>
         </td>
```

```
Hibernate:
    select
        board0_.seq as seq1_0_0_,
        board0_.cnt as cnt2_0_0_,
        board0_.content as content3_0_0_,
        board0_.create_date as create_d4_0_0_,
        board0_.title as title5_0_0_,
        board0_.writer as writer6_0_0_
    from
        board board0_
    where
        board0_.seq=?
```

# 화면 개발

- **게시글 상세 조회**

(2) 상세 조회 화면 – getBoard.jsp


# 화면 개발

- **게시글 상세 조회**

(2) 상세 조회 화면 – getBoard.jsp

```
<div id="container">
    <h2>글 상세 보기</h2>
    <form action="/updateBoard" method="post">
        <!-- 글 수정시에 반드시 기본키를 컨트롤러에 보내야 함 -->
        <input type="hidden" name="seq" value="${board.seq}">
        <table>
            <tr>
                <td>제목</td>
                <td><input type="text" name="title" value="${board.title}"></td>
            </tr>
            <tr>
                <td>작성자</td>
                <td><input type="text" name="writer" value="${board.writer}"></td>
            </tr>
            <tr>
                <td>내용</td>
                <td>
                    <textarea name="content" rows="10"
                        cols="50"><c:out value="${board.content}" /></textarea>
                </td>
            </tr>
```

# 화면 개발

- **게시글 상세 조회**

(2) 상세 조회 화면 – getBoard.jsp

```
        <tr>
            <td>조회수</td>
            <td><input type="text" name="writer" value="${board.cnt}"></td>
        </tr>
        <tr>
            <td colspan="2">
               <button type="submit">글수정</button>
               <a href='/deleteBoard?seq=<c:out value="${board.seq}" />'
                  onclick="return confirm('정말로 삭제하시겠습니까?')">
                    <button type="button">글삭제</button>
                </a>
               <a href="/getBoardList"><button type="button">글목록</button></a>
            </td>
        </tr>
    </table>
</form>
```

# 화면 개발

- **게시글 삭제**

(1) 글 삭제 처리

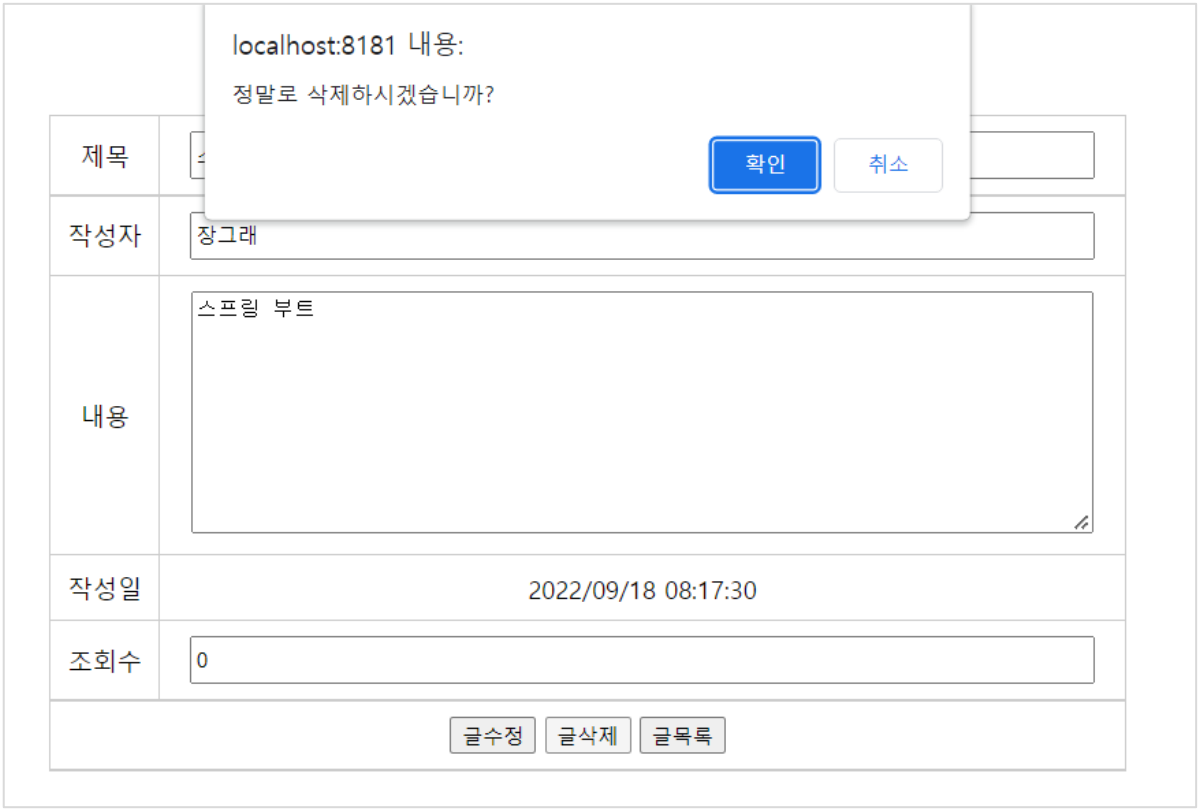
```
//게시글 삭제
@Override
public void deleteBoard(Board board) {
    boardRepo.delete(board);
}
```

```
//게시글 삭제
@GetMapping("/deleteBoard")
public String deleteBoard(Board board) {
    service.deleteBoard(board);
    return "redirect:getBoardList";
}
```

# 화면 개발

- **게시글 삭제**

(2) 글 삭제 화면




# 화면 개발

- **게시글 수정**

(1) 글 수정 처리

```
//게시글 수정
@Override
public void updateBoard(Board board) {
    Board findBoard = boardRepo.findById(board.getSeq()).get();
    findBoard.setTitle(board.getTitle());
    findBoard.setContent(board.getContent());
    boardRepo.save(findBoard);
}
```

```
//게시글 수정
@PostMapping("/updateBoard")
public String updateBoard(Board board) {
    service.updateBoard(board);
    return "redirect:getBoardList";
}
```

# 화면 개발

- **게시글 수정**

(2) 글 수정 화면

```
<h2>글 상세 보기</h2>
<form action="updateBoard" method="post">
    <!-- 글 수정시에 반드시 기본키를 컨트롤러에 보내야 함 -->
    <input type="hidden" name="seq" value="${board.seq}">
    <table>
        <tr>
            <td>제목</td>
            <td><input type="text" name="title" value="${board.title}"></td>
        </tr>
```

```
Hibernate:
    update
        board
    set
        content=?,
        title=?
    where
        seq=?
```

# 화면 개발

- **게시글 수정**

(2) 글 수정 화면


# 화면 개발

- 조회수 증가

## 게시글 목록

| 번호 | 제목 | 작성자 | 등록일 | 조회수 |
|---|---|---|---|---|
| 1 | 첫번째 게시글 | 관리자 | 2022/09/23 13:26:23 | 2 |
| 2 | test | tester | 2022/09/23 13:26:37 | 1 |
| 3 | 가을 | 김기용 | 2022/09/23 13:26:50 | 1 |

새글 등록

# 화면 개발

- 조회수 증가

```
public interface BoardRepository extends CrudRepository<Board, Long>{

    //제목에 특정 단어가 포함된 글 목록을 내림차순으로 조회
    @Query("SELECT b FROM Board b WHERE b.title LIKE %?1% ORDER BY b.seq DESC")
    List<Board> queryAnnotationTest1(String searchKeyword);
```

```
  //조회수 증가
  @Transactional  //트랜잭션 처리하는 어노테이션
  @Override
  public void updateCnt(Long seq) {
      boardRepo.updateCnt(seq);
  }
```

# 화면 개발

- 조회수 증가

```
//게시글 상세 조회
@GetMapping("/getBoard")
public String getBoard(Long seq, Model model) {
    service.updateCnt(seq);    //조회수 증가
    Board board = service.getBoard(seq); //상세 보기 처리
    model.addAttribute(board);
    return "getBoard";
}
```