

Spring Framework

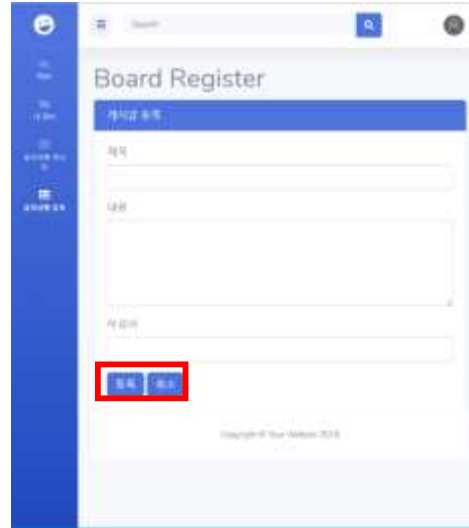
- 게시판
- 1. 구현 순서
- 2. 페이징 처리

1. 게시판 구현 순서



Board List Page showing a table of board entries. The '게시글' (Post) column is highlighted with a red box. A '게시글' button is also highlighted in the top right corner.

번호	제목	작성자	작성일	수정일
30	게시글	게시글	2019-04-22 06:01:00	2019-04-22
29	게시글	게시글	2019-04-22 06:01:00	2019-04-22
28	게시글	게시글	2019-04-22 06:01:00	2019-04-22
27	게시글	게시글	2019-04-22 06:01:00	2019-04-22
26	게시글	게시글	2019-04-22 06:01:00	2019-04-22
25	게시글	게시글	2019-04-22 06:01:00	2019-04-22
24	게시글	게시글	2019-04-22 06:01:00	2019-04-22
23	게시글	게시글	2019-04-22 06:01:00	2019-04-22
22	게시글	게시글	2019-04-22 06:01:00	2019-04-22
21	게시글	게시글	2019-04-22 06:01:00	2019-04-22



Board Register Page showing a form for creating a new board entry. The '등록' (Register) button is highlighted in a red box.

Board Register

게시글 등록

제목

내용

작성자

등록



Board Read Page showing a form for reading a board entry. The '게시글' (Post) button is highlighted in a red box.

Board Read Page

게시글 보기

제목

내용

작성자

게시글

게시글



Board Modify Page showing a form for modifying a board entry. The '게시글' (Post) button is highlighted in a red box.

Board Modify Page

게시글 수정

제목

내용

작성자

게시글

게시글

1. 게시판 구현 순서

1. 컨트롤러 생성(화면 확인)
 2. 등록 처리
 3. 테이블 생성
 4. DB관련 설정(root-xml작업)
 5. BoardVO 생성(DB컬럼명과 반드시 동일하게 생성)
 6. Service구현
 7. DAO 구현
 8. 마이바티스 DB작업
-
9. 상세보기 처리
 10. 변경 처리
 11. 삭제 처리
 12. 페이징 처리



테이블 SQL

```
create table freeboard(  
    bno number(10,0),  
    title varchar(200) not null,  
    content varchar2(2000) not null,  
    writer varchar2(50) not null,  
    regdate date DEFAULT sysdate,  
    update date DEFAULT sysdate  
);  
alter table freeboard add CONSTRAINT freeboard_pk PRIMARY key (bno);  
create SEQUENCE freeboard_seq;
```

2. 페이징

1. 반드시 GET 방식으로만 처리한다
2. 이동할 때 페이지 번호 를가지고 다닌다
3. 페이징 처리하는 로직을 클래스로 분류한다
 - > Criteria클래스, PageVO클래스

2. 페이징 (Criteria 클래스)

-페이징을 처리하는 기준

```
public class Criteria {
    private int pageNum;
    private int amount;

    public Criteria() {
        this(1, 10);
    }

    public Criteria(int pageNum, int amount) {
        this.pageNum = pageNum;
        this.amount = amount;
    }
}
```

아래 게터 세터

pageNum = 조회하는 페이지번호
amount = 몇개씩 보여줄건가



오라클 쿼리문

```
select *
from (select rownum as rn, bno, writer, title, content, regdate, updatedate
      from (select *
            from freeboard
            order by bno desc
            )
      ) where rn > 10 and rn <= 20;
```

← 1-10번 데이터가 조회된다

오라클 sql문

```
select *
from (select rownum as rn, bno, writer, title, content, regdate, updatedate
      from (select *
            from freeboard
            order by bno desc
            )
      ) where rn > ? and rn <= ?;
```

← ?는 뭐가 들어가는가?

$\#{pageNum-1} * \#{amount}$

$\#{pageNum} * \#{amount}$

2. 페이징 (PageVO 클래스) -페이징 계산 처리 클래스

- 1 total 게시판 글 전체 개수.
- 2 endPage: 게시판을 화면에 보여질 마지막 페이지 번호.
- 3 startPage: 게시판을 화면에 보여질 첫번째 페이지 번호.
- 4 realEnd: 게시판의 실제 마지막 페이지 번호.
- 5 prev: 이전 페이지 버튼 활성화 여부.
- 6 next: 다음 페이지 버튼 활성화 여부.



공식

2 endPage =

1~10 페이지 클릭시 10

11~20 페이지 클릭시 20

21~30 페이지 클릭시 30

3 startPage =

endPage가 10 일때 1

endPage가 20 일때 11

endPage가 30 일때 21

4 realEnd =

게시물이 50개 라면 5

게시물이 51개 라면 6

게시물이 101개 라면 11

공식

4 prev =

startPage가 1이면 비활성화

startPage가 11이면 활성화

5 next =

realEnd가 endPage보다 크면 활성화

Math.ceil() 함수를 이용한다

2. 페이징 (PageVO 클래스)

```
public class PageVO {
    private int startPage; //게시판 화면에 보여질 첫페이지 번호
    private int endPage; //게시판 화면에 보여질 끝페이지 번호
    private boolean prev; //다음 이전 활성화 여부
    private boolean next;

    private int pageNum; //현재 조회하는 페이지 번호
    private int amount; //한 페이지에서 몇개의 데이터를 보여줄건가
    private int total; //총 게시물 수

    private Criteria cri; //페이징 기준

    public PageVO(Criteria cri, int total) {
        this.cri = cri;
        this.total = total;
        this.pageNum = cri.getPageNum();
        this.amount = cri.getAmount();

        this.endPage = (int)(Math.ceil(this.pageNum / 10.0) ) * 10;

        this.startPage = endPage - 10 + 1;

        int realEnd = (int)Math.ceil(total / (double)this.amount );

        if(this.endPage > realEnd ) {
            this.endPage = realEnd;
        }

        this.prev = this.startPage > 1;

        this.next = realEnd > this.endPage}
}
```

2. 검색과 페이징

자유게시판

게시판의 **검색기능**과 **페이징** 두개의 폼으로 사용된다

총 113게시글

제목

검색

번호	제목	작성자	등록일	수정일
141	ㅎㅎ	두글	2019-11-07 01:44:14.0	2019-11-07 01:44:14.0
131	admin2	test2	2019-11-05 18:56:11.0	2019-11-05 18:56:11.0
130	admin2	test2	2019-11-05 18:56:11.0	2019-11-05 18:56:11.0
129	admin2	test2	2019-11-05 18:56:11.0	2019-11-05 18:56:11.0
128	admin2	test2	2019-11-05 18:56:11.0	2019-11-05 18:56:11.0
127	admin2	test2	2019-11-05 18:56:11.0	2019-11-05 18:56:11.0
126	admin2	test2	2019-11-05 18:56:11.0	2019-11-05 18:56:11.0
125	admin2	test2	2019-11-05 18:56:11.0	2019-11-05 18:56:11.0
124	admin1	test1	2019-11-05 18:56:06.0	2019-11-05 18:56:06.0
123	admin1	test1	2019-11-05 18:56:06.0	2019-11-05 18:56:06.0

1

2

3

4

5

6

7

8

9

10

다음

글쓰기

1. 페이징 처리를 a태그에서 -> 폼전송으로
2. Criteria에 search키워드 search타입 변수 추가
3. 페이징 쿼리를 **동적쿼리**로 변경
4. 전체게시글 수 **동적쿼리**로 변경
5. searc폼 과 page폼을 나눈다
6. 두 Form에서 pageNum, amount, searchType, searchName을 동시에 넘긴다(hidde이용)
7. **검색 후 화면에서 일어나는 문제해결**
3페이지에서 검색하면 3페이지로 이동되는 문제
(검색버튼 클릭시 pageNum은 1로 처리)

select박스에 검색 조건이 남게 처리한다
input박스에 검색 단어가 남게 처리한다

동적 쿼리란?

마이바티스의 태그로써 조건문을 써서 쿼리의 실행을 제어하는 방법입니다
jstl구문과 사용방법이 유사합니다.

마이바티스의 `test=""` 구문 안에 작성되는 값은 **VO의 getter나 , map의 key값**이 쓰입니다

대표적 태그의 종류

1. if
2. choose(when, otherwise)
3. foreach
4. include등

1. 페이징 처리를 a태그에서 -> 폼전송으로

페이징 태그

```
<ul class="pagination pagination-sm">
  <c:if test="${pageV0.prev }">
    <li><a href="${pageV0.startPage-1 }" onclick="page(${pageV0.startPage-1 })">이전</a></li>
  </c:if>
  <c:forEach var="num" begin="${pageV0.startPage }" end="${pageV0.endPage }">
    <li class="${pageV0.pageNum eq num ? 'active':'' }">
      <a href="${num }" onclick="page(${num})">${num }</a>
    </li>
  </c:forEach>
  <c:if test="${pageV0.next }">
    <li><a href="${pageV0.endPage+1}" onclick="page(${pageV0.endPage+1})">다음</a></li>
  </c:if>
</ul>
```

1ST - 기본이벤트방식

```
var searchBtn = document.getElementById("searchBtn");
searchBtn.onclick = function() {
  //3페이지에서 검색하면 3페이지로 이동하는 문제 해결
  document.getElementById("pageNum").setAttribute("value", 1);

  document.getElementById("searchForm").submit();
}
//검색처리 (페이징 클릭시 폼전송)
function page(num) {
  event.preventDefault(); //이벤트의 실행을 막는다

  document.querySelector("#pageForm #pageNum").setAttribute("value", num) //페이지 폼의 값을 value를 세팅한다
  document.getElementById("pageForm").submit(); //폼 서버밋
}
```

스크립트

1. 페이징 처리를 a태그에서 -> 폼전송으로

페이징 태그

```
<ul class="pagination pagination-sm" id="page">
  <c:if test="${pageVO.prev }">
    <li><a href="${pageVO.startPage-1 }">이전</a></li>
  </c:if>
  <c:forEach var="num" begin="${pageVO.startPage }" end="${pageVO.endPage }">
    <li class="${pageVO.pageNum eq num ? 'active':'' }">
      <a href="${num }">${num }</a>
    </li>
  </c:forEach>
  <c:if test="${pageVO.next }">
    <li><a href="${pageVO.endPage+1}">다음</a></li>
  </c:if>
</ul>
```

2ed – 이벤트위임방식 – 부모에 이벤트를 걸고 클릭시 event객체를 이용하여 자식태그에서 적용하는 방법

//이벤트 위임방식 2nd - 부모에 이벤트를 걸어놓고 자식태그 클릭시 이벤트를 위임하여 적용하는 방법

```
var page = document.getElementById("page");
page.onclick = function(event) {
  event.preventDefault();

  var value = event.target.getAttribute("href");
  document.querySelector("#pageForm #pageNum").setAttribute("value", value); //페이지 폼의 값을 value를 세팅한다
  document.getElementById("pageForm").submit();
}
```

2. Criteria에 search키워드 search타입 변수 추가

```
public class Criteria {  
  
    private int pageNum;  
    private int amount;  
  
    private String searchName;  
    private String searchType;  
  
    public Criteria() {  
        this(1, 10);  
    }  
  
    public Criteria(int pageNum, int amount) {  
        this.pageNum = pageNum;  
        this.amount = amount;  
    }  
  
}
```

3. 페이징 쿼리를 동적쿼리로 변경 전체게시글 쿼리를 동적쿼리로 변경

```
<select id="getList" resultType="FreeBoardVO">
  <![CDATA[
    select *
    from (select rownum as rn, bno, writer, title, content, regdate, updatedate
          from (select *
                from freeboard
                where
<if test="searchType == 'title' ">title like '%' || #{searchName} || '%</if>
<if test="searchType == 'content' ">content like '%' || #{searchName} || '%</if>
<if test="searchType == 'writer' ">writer like '%' || #{searchName} || '%</if>
<if test="searchType == 'titcont' ">title like '%' || #{searchName} || '% or content like '%' || #{searchName} || '%</if>
<if test="searchType == null or searchType == '' ">1=1</if>

          )
          order by bno desc
        )
    ) where rn > (#{pageNum}-1) * #{amount} and rn <= (#{pageNum}* #{amount})
  ]]>
</select>
```

5. 검색폼,페이징폼에서 pageNum, amount, searchType, searchName를 각각 전송한다 hidden을 이용

페이징폼

```
<input type="hidden" name="pageNum" id="pageNum" value="${pageVO.cri.pageNum }">
<input type="hidden" name="amount" id="amount" value="${pageVO.cri.amount }">
<input type="hidden" name="searchType" value="${pageVO.cri.searchType }">
<input type="hidden" name="searchName" value="${pageVO.cri.searchName }">
```

6. 검색 후 selec박스에 키워드 처리, 검색input에 키워드 처리

검색폼

```
<form action="freeList" method="get" name="searchForm" id="searchForm">
<div class="search-wrap" >
    <span>총 ${pageVO.total }게시글</span>
    <button type="button" class="btn btn-info search-btn" id="searchBtn">검색</button>
    <input type="text" class="form-control search-input" value="${pageVO.cri.searchName }" name="searchName" id="searchName">
    <select class="form-control search-select" name="searchType">
        <option value="title" ${pageVO.cri.searchType eq 'title' ? 'selected' : '' }>제목</option>
        <option value="content" ${pageVO.cri.searchType eq 'content' ? 'selected' : '' }>내용</option>
        <option value="writer" ${pageVO.cri.searchType eq 'writer' ? 'selected' : '' }>작성자</option>
        <option value="titcont" ${pageVO.cri.searchType eq 'titcont' ? 'selected' : '' }>제목+내용</option>
    </select>
</div>
<!-- 검색의 클릭시 pageNum은 1부터 시작하게 함 -->
<input type="hidden" name="pageNum" id="pageNum" value="1">
<input type="hidden" name="amount" id="amount" value="${pageVO.cri.amount }">
```

스프링1부 끝 수고하셨습니다