페이지네이션 구현 프로세스

Spring Boot 교안에 있는 boardProject(게시판)를 기반으로 진행합니다.

프로젝트 자료 깃허브 주소: https://github.com/sjgopromer2025/spring_projects_data

페이지네이션 동작 흐름도

1. 사용자
    - 페이지 번호 클릭
    - 예: **3페이지 클릭**
2. 브라우저
    - HTTP GET 요청 전송

```
/board/list?page=3
```

3. **Controller**
    - `@RequestParam`으로 `page` 값 읽기
    - `size`는 고정값(예: 10) 설정
    - `page`와 `size`를 **Service**로 전달
4. **Service**
    1. 전체 게시글 수(`totalCount`) 조회
    2. 시작 인덱스(`offset`) 계산

```
offset = (page - 1) * size
```

    3. Mapper 호출 → 해당 범위 게시글 목록 조회
    4. 페이지 정보 계산 → `PaginationDTO` 생성

페이지네이션 동작 흐름도

5. **Mapper / DB**
   - 총 게시글 수 조회

```
SELECT COUNT(*) FROM board;
```

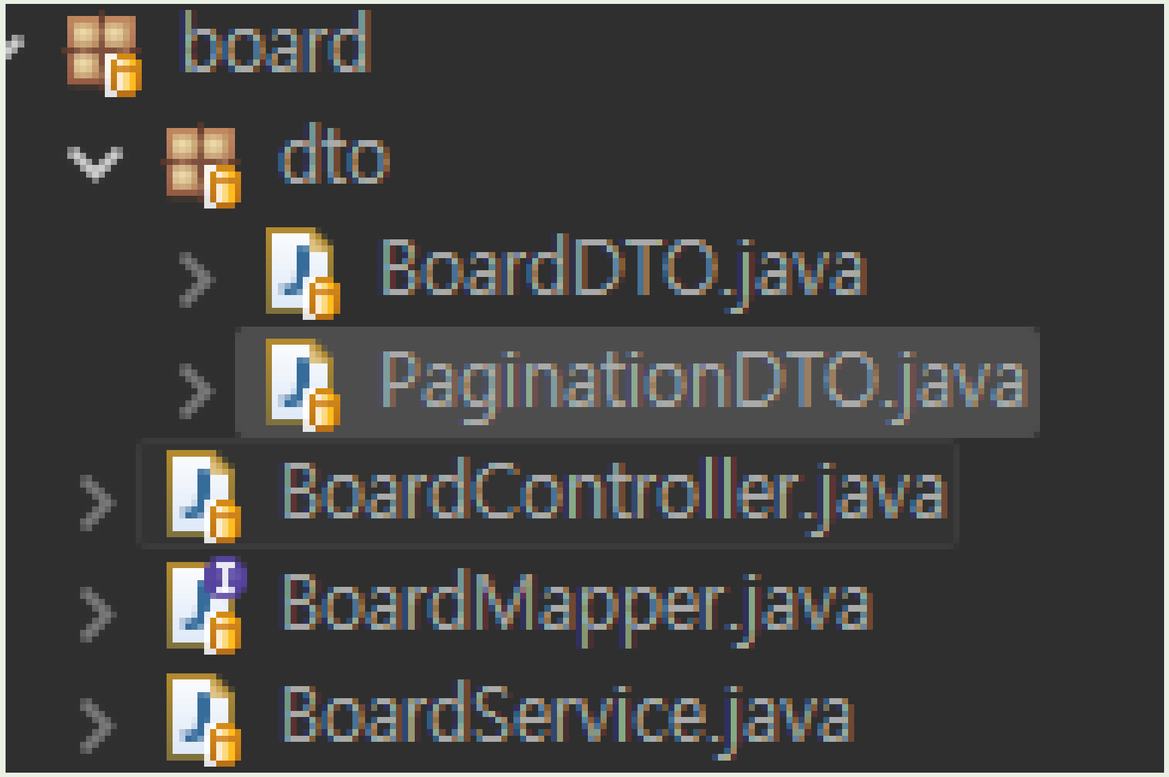
   - 해당 범위 게시글 조회

```
SELECT *
FROM board
ORDER BY idx DESC
LIMIT #{offset}, #{size};
```

6. **Controller → View(JSP)**
   - `boards` : 게시글 목록
   - `pagination` : 페이지 정보
   - `model.addAttribute()`로 전달

7. **View(JSP)**
   1. 게시글 목록 출력 (`<c:forEach>`)
   2. 페이지 번호 버튼 생성 (`startPage` ~ `endPage`)
   3. 이전(prev), 다음(next) 버튼 표시 여부 결정

dto 패키지 생성



1. 페이지네이션 관련 정보는 board(게시판)과 연결이 되므로 board패키지 안에 생성합니다.

2. 다만, DTO의 파일이 많아지면 파일 구조가 복잡해질 수 있으므로 dto 패키지를 생성하고 따로 관리

PaginationDTO 생성

```java
// 페이지네이션 정보를 담는 DTO 클래스
// 현재 페이지, 전체 페이지 수, 시작/끝 페이지, 이전/다음 페이지 존재 여부를 관리
public class PaginationDTO {

    private int currentPage; // 현재 페이지 번호
    private int totalPages; // 전체 페이지 개수
    private int startPage; // 현재 페이지 블록의 시작 페이지 번호
    private int endPage; // 현재 페이지 블록의 마지막 페이지 번호
    private boolean hasPrev; // 이전 페이지 블록 존재 여부
    private boolean hasNext; // 다음 페이지 블록 존재 여부

    // 생성자
    // 생략..
    // getter & setter ~
    // 생략..
    // toString
    // 생략..
```

PaginationDTO 생성자

```java
// 페이지네이션 계산 로직
// currentPage 현재 페이지 번호
// totalCount 전체 데이터 개수
// size 한 페이지당 보여줄 데이터 개수
// blockLimit 한 번에 보여줄 페이지 번호 개수 (예: 1~10, 11~20)
public PaginationDTO(int currentPage, int totalCount, int size, int blockLimit) {
    this.currentPage = currentPage;

    // 전체 페이지 수 계산 (소수점이 있으면 올림 처리)
    // 예: 데이터 25개, size 10 → 3페이지 필요
    this.totalPages = (int) Math.ceil((double) totalCount / size);

    // 현재 페이지 블록의 시작 페이지 계산
    // 예: currentPage=7, blockLimit=10 → startPage=1
    // currentPage=15, blockLimit=10 → startPage=11
    this.startPage = ((currentPage - 1) / blockLimit) * blockLimit + 1;

    // 현재 페이지 블록의 끝 페이지 계산
    // 블록의 마지막 페이지가 전체 페이지 수보다 클 수 없으므로 min 사용
    this.endPage = Math.min(startPage + blockLimit - 1, totalPages);

    // 이전 페이지 블록이 있는지 여부
    this.hasPrev = startPage > 1;

    // 다음 페이지 블록이 있는지 여부
    this.hasNext = endPage < totalPages;
}
```

PaginationDTO 생성자 설명 - 1

```
// 전체 페이지 수 계산 (소수점이 있으면 올림 처리)
// 예: 데이터 25개, size 10 → 3페이지 필요
this.totalPages = (int) Math.ceil((double) totalCount / size);
```

1. 전체 데이터 개수를 한 페이지에 보여줄 개수(size)로 나누어서 총 몇 페이지가 필요한지 계산
2. 소수점이 생기면 올림 처리 (Math.ceil)
3. 예시: 데이터 25개, 한 페이지에 10개씩 → 25 / 10 = 2.5 → 올림 → 3페이지 필요

PaginationDTO 생성자 설명 - 2

```
// 현재 페이지 블록의 시작 페이지 계산
// 예: currentPage=7, blockLimit=10 → startPage=1
// currentPage=15, blockLimit=10 → startPage=11
this.startPage = ((currentPage - 1) / blockLimit) * blockLimit + 1;
```

1. 한 번에 보여줄 페이지 버튼 묶음(=블록)의 첫 번째 페이지 번호 구하기
2. 예시 :
    1. 블록 크기(blockLimit) = 10
    2. 현재 페이지가 7 → (7 - 1) / 10 = 0 → 첫 페이지 = 1
    3. 현재 페이지가 15 → (15 - 1) / 10 = 1 → 첫 페이지 = 11

PaginationDTO 생성자 설명 - 3

```
        // 현재 페이지 블록의 끝 페이지 계산
        // 블록의 마지막 페이지가 전체 페이지 수보다 클 수 없으므로 min 사용
        this.endPage = Math.min(startPage + blockLimit - 1, totalPages);
```

1. 현재 페이지 블록의 마지막 페이지 번호를 구함
2. 단, 전체 페이지 수를 넘어가면 안 되니까 Math.min으로 조정
3. 예시
    1. 총 페이지 = 13, 시작 페이지 = 11, 블록 크기 = 10
    2. 원래 끝 페이지 = 20이지만, 총 페이지 13이 더 작으니 13으로 제한

PaginationDTO 생성자 설명 - 4

```
    // 이전 페이지 블록이 있는지 여부
    this.hasPrev = startPage > 1;
```

1. 현재 페이지 블록 앞에 다른 페이지 블록이 있으면 true
2. 시작 페이지가 1이면 이전 블록 없음 (false)
3. 시작 페이지가 11이면 이전 블록 있음 (true)

PaginationDTO 생성자 설명 - 5

```
        // 다음 페이지 블록이 있는지 여부
        this.hasNext = endPage < totalPages;
    }
```

1. 현재 페이지 블록 뒤에 다른 페이지 블록이 있으면 true
2. 끝 페이지가 전체 페이지 수보다 작으면 true (다음 버튼 활성화)
3. 끝 페이지가 전체 페이지 수와 같으면 false (다음 버튼 비활성화)

BoardMapper.xml

```xml
<!-- <mapper namespace="com.ysj.practice.board.BoardMapper"> -->
<mapper namespace="com.example.boardProject.board.BoardMapper">

    <!-- 전체 게시글 수 조회 -->
    <select id="getTotalCount" resultType="int">
        SELECT COUNT(*) FROM board
    </select>

    <!-- 페이지별 게시글 조회 -->
    <select id="findPage" resultType="boarddto">
        SELECT *
        FROM board
        ORDER BY idx DESC
        LIMIT #{offset}, #{size}
    </select>
```

BoardMapper.java

```
//  <select id = "getTotalCount" resultType="int">
    // 전체 게시물 수 조회
    int getTotalCount();

//  <select id="findPage" resultType="boarddto">
    // 페이지별 게시글 조회
//  List<BoardDTO> findPage(@Param("size") int size, @Param("offset") int offset);
    List<BoardDTO> findPage(int size, int offset);
```

1. MyBatis는 메서드 파라미터가 1개일 때는 parameterType으로 XML에 명시 가능
2. 2개 이상일 때는 자동으로 param1, param2 또는 @Param으로 지정한 이름을 사용
3. XML의 parameterType은 하나만 설정할 수 있어서, 2개 이상이면 의미가 없어집니다.

BoardService - 1

```java
public int getTotalCount() {
    try {
        return boardMapper.getTotalCount();
    } catch (Exception e) {
        // 예외 발생 시 로그 출력하고 0 리턴
        System.err.println("[ERROR] 게시글 총 개수 조회 실패: " + e.getMessage());
        return 0;
    }
}
```

1. Board(게시판)의 총 게시글의 개수(count)를 가져오는 메서드입니다.

BoardService - 2

```java
public List<BoardDTO> findPage(int page, int size) {
    // 기본값 처리
    if (size <= 0) {
        size = 10;
    }
    if (page <= 0) {
        page = 1;
    }

    // 전체 게시글 수 조회
    int totalCount = getTotalCount();
    int totalPages = (int) Math.ceil((double) totalCount / size);

    // 요청한 페이지가 최대 페이지보다 크면 마지막 페이지로 조정
    if (totalPages > 0 && page > totalPages) {
        page = totalPages;
    }

    int offset = (page - 1) * size;

    try {
        return boardMapper.findPage(size, offset);
    } catch (Exception e) {
        // 예외 발생 시 로그 출력하고 빈 리스트 반환
        System.err.println("[ERROR] 게시글 페이지 조회 실패: " + e.getMessage());
        return Collections.emptyList();
    }
}
```

1. Size, page의 값이 0일 때 기본 값으로 대체 하는 코드
2. 총 게시글의 개수를 조회, 조회한 게시글의 수를 기준으로 나올 수 있는 최대 페이지의 수를 계산 (totalPages)

   if 문에서는 만약 요청한 page가 최대페이지(totalPages)보다 크면 넘어가지 못하게 page를 totalPages로 재할당
3. Offset 시작 지점을 구하고 그에 맞게 게시글을 조회하는 코드

```java
public List<BoardDTO> findPage(int page, int size) {
    // 기본값 처리
    if (size <= 0) {
        size = 10;
    }
    if (page <= 0) {
        page = 1;
    }

    // 전체 게시글 수 조회
    int totalCount = getTotalCount();
    int totalPages = (int) Math.ceil((double) totalCount / size);

    // 요청한 페이지가 최대 페이지보다 크면 마지막 페이지로 조정
    if (totalPages > 0 && page > totalPages) {
        page = totalPages;
    }

    int offset = (page - 1) * size;

    try {
        return boardMapper.findPage(size, offset);
    } catch (Exception e) {
        // 예외 발생 시 로그 출력하고 빈 리스트 반환
        System.err.println("[ERROR] 게시글 페이지 조회 실패: " + e.getMessage());
        return Collections.emptyList();
    }
}
```

BoardController

```
// 게시글 목록 페이지
@GetMapping("/list")
public String listPage(@RequestParam(defaultValue = "1") int page, // 현재 페이지 (기본값 1)
        @RequestParam(defaultValue = "10") int size, // 한 페이지당 게시글 수 (기본값 10)
        Model model) {

    // 전체 게시글 수 조회
    int totalCount = boardService.getTotalCount();

    // 페이지네이션 정보 생성 (blockLimit = 10 → 1~10, 11~20)
    PaginationDTO pagination = new PaginationDTO(page, totalCount, size, 10);

    // 현재 페이지에 맞는 게시글 목록 조회
    List<BoardDTO> boards = boardService.findPage(page, size);

    // View로 데이터 전달
    model.addAttribute("boards", boards); // 게시글 목록
    model.addAttribute("pagination", pagination); // 페이지네이션 정보

    return "board/list_jstl"; // /WEB-INF/view/board/list_jstl.jsp
}
```

1. 쿼리스트링은 RequestParam, 혹은 dto를 사용하여 데이터를 받을 수 있습니다.
2. PaginationDTO의 생성자를 이용하여 페이지네이션 정보를 쉽게 구할 수 있습니다.
3. 요청한 page에 맞게 해당하는 게시글 정보를 조회하는 메서드 호출
4. Model에 각각 게시글 정보와 페이지 네이션 정보를 설정합니다.

```java
// 게시글 목록 페이지
@GetMapping("/list")
public String listPage(@RequestParam(defaultValue = "1") int page, // 현재 페이지 (기본값 1)
        @RequestParam(defaultValue = "10") int size, // 한 페이지당 게시글 수 (기본값 10)
        Model model) {

    // 전체 게시글 수 조회
    int totalCount = boardService.getTotalCount();

    // 페이지네이션 정보 생성 (blockLimit = 10 → 1~10, 11~20)
    PaginationDTO pagination = new PaginationDTO(page, totalCount, size, 10);

    // 현재 페이지에 맞는 게시글 목록 조회
    List<BoardDTO> boards = boardService.findPage(page, size);

    // View로 데이터 전달
    model.addAttribute("boards", boards); // 게시글 목록
    model.addAttribute("pagination", pagination); // 페이지네이션 정보

    return "board/list_jstl"; // /WEB-INF/view/board/list_jstl.jsp
}
```

JSP 뷰 수정 - 게시글 목록 페이지

```jsp
<!-- 페이지 버튼 -->
<div>
    <c:if test="${pagination.hasPrev}">
        <a href="?page=${pagination.startPage - 1}">이전</a>
    </c:if>

    <c:forEach var="page" begin="${pagination.startPage}" end="${pagination.endPage}">

        <a href="?page=${page}">${page}</a>
    </c:forEach>

    <c:if test="${pagination.hasNext}">
        <a href="?page=${pagination.endPage + 1}">다음</a>
    </c:if>
</div>
```

📋 게시글 목록

글쓰기

번호	제목	작성자
100	게시글 제목 5	작성자5
99	게시글 제목 4	작성자4
98	게시글 제목 3	작성자3
97	게시글 제목 2	작성자2
96	게시글 제목 1	작성자1
95	게시글 제목 15	작성자5
94	게시글 제목 14	작성자4
93	게시글 제목 13	작성자3
92	게시글 제목 12	작성자2
91	게시글 제목 11	작성자1

CSS 적용 버전 - 1

```css
/* 페이지네이션 스타일 */
.pagination {
    margin-top: 20px;
    text-align: center;
}
.pagination a, .pagination span {
    display: inline-block;
    margin: 0 5px;
    padding: 8px 12px;
    border: 1px solid #ddd;
    text-decoration: none;
    color: #333;
}
.pagination .active {
    background-color: #007bff;
    color: white;
    border-color: #007bff;
}
.pagination a:hover {
    background-color: #f2f2f2;
}
```

CSS 적용 버전

```
<!-- 페이지네이션 -->
<div class="pagination">
    <!-- 이전 버튼 -->
    <c:if test="${pagination.hasPrev}">
        <a href="/board/list?page=${pagination.startPage - 1}">&laquo; 이전</a>
    </c:if>

    <!-- 페이지 번호 출력 -->
    <c:forEach begin="${pagination.startPage}" end="${pagination.endPage}" var="p">
        <c:choose>
            <c:when test="${p == pagination.currentPage}">
                <span class="active">${p}</span>
            </c:when>
            <c:otherwise>
                <a href="/board/list?page=${p}">${p}</a>
            </c:otherwise>
        </c:choose>
    </c:forEach>

    <!-- 다음 버튼 -->
    <c:if test="${pagination.hasNext}">
        <a href="/board/list?page=${pagination.endPage + 1}">다음 &raquo;</a>
    </c:if>
</div>
```

글쓰기

번호	제목	작성자
100	게시글 제목 5	작성자5
99	게시글 제목 4	작성자4
98	게시글 제목 3	작성자3
97	게시글 제목 2	작성자2
96	게시글 제목 1	작성자1
95	게시글 제목 15	작성자5
94	게시글 제목 14	작성자4
93	게시글 제목 13	작성자3
92	게시글 제목 12	작성자2
91	게시글 제목 11	작성자1