11. 게시판 구현 – 글 목록보기, 글 쓰기 구현

1) DAO 구현

BoardDAOjsp (: DB 에 접속해서 불러올 테이블의 CRUD 관련된 method 선언 > insert,update,delete,select)

```
package devtaco.board;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;

public class BoardDAO { // MemberDAO와 같이

	private DBConnectionMgr pool = null; // 1. 얻어올 DB 객체
	// 웹에서 공통으로 사용할 필드
	private Connection conn = null;
	private PreparedStatement pstmt = null; //stmt 객체보다 속도가 빠름
	private ResultSet rs = null;
	private String sql ="";

	// 생성자를 통해서 상대방의 객체를 생성해서 연결함
	public BoardDAO() {
		try {
			pool = DBConnectionMgr.getInstance();
			System.out.println("pool:"+pool); // BoardDAO() 체크
		}catch(Exception e) {
			System.out.println("pool:"+pool);
		}
	}
	// 1. 페이징처리를 위해서 '전체 레코드 수'를 구해와야함
	public int getArticleCount() {
		int x = 0;
		try {
			con = pool.getConnection();
			System.out.println("con:"+con);
			sql="select count(*) from board";   // select 구문 -> pstmt
			pstmt = con.prepareStatement(sql);
			rs = pstmt.executeQuery();
			// 검색된 레코드가 있다면 -> rs.next()
			if(rs.next()) {
			x = rs.getInt(1); // 1번째 인덱스의 값을 가져와서 x에 넣어라 , rs.get자료형 (필드명 or 인덱스번호)
			}
		}catch(Exception e) {
			System.out.println("getArticleCount() method error:"+e);
		}finally {
			pool.freeConnection(con,pstmt,rs);
		}
		return x;

	}
```

페이징 처리를 위해서 '전체 레코드 수(게시물 수)'를 구해오는 method
> select count(*) from board;

2) 페이지 개념 (10page → 1 block)

: 레코드가(게시물이) 10 개 → 1 페이지 형성 → 10 개 페이지 → Block

(1) 이전블럭(prev)

1~10 21~30

이전 10 개 11 12 13 14 15 16 17 18 19 20 다음 10 개

: 최소 11 페이지 이상이 되어야 **지금 보고 있는 11~20** 페이지가 나옴 (레코드 수 : 최소 100 개이상)

(2) 현재블럭

: 1 2 3 4 **[5]** 6 7 8 9 10

(3) 다음블럭

: 1 2 3 4 5 6 7 8 9 10 ::: 다음 10 개 ➔ 레코드수가 100 개 이상이면 다음블럭 링크가 나옴

▶ 페이징을 위한 변수들

1. **int nowPage :** 현재 보고 있는 페이지(클릭해서)=>레코드번호
2. **int nowBlock :** 현재블럭, 현재 페이지가 포함된 전체 페이지들
3. **int numPerPage** : 10; : 페이지당 보여주는 레코드수를 지정
4. **int pagePerBlock** = 10; : 블럭당 보여주는 페이지수를 지정
5. **총페이지수 = 총레코드수 / 페이지당 보여주는 레코드수**

122/10=12.2 ➔ 12.2 + 1 = 13.2 ➔ (int)13.2=13

12페이지 10개 + 2개 → 1 페이지를 더 생성->총 13페이지

122/10

```
int totalPage =(int)Math.ceil((double)totalRecord / numPerPage);
```

무조건 올림메서드

6. **총블럭수 = 총페이지수 /블럭당 보여주는 페이지수**

```
int totalBlock =(int)Math.ceil((double)totalPage / pagePerBlock);
```

13/10=1.3

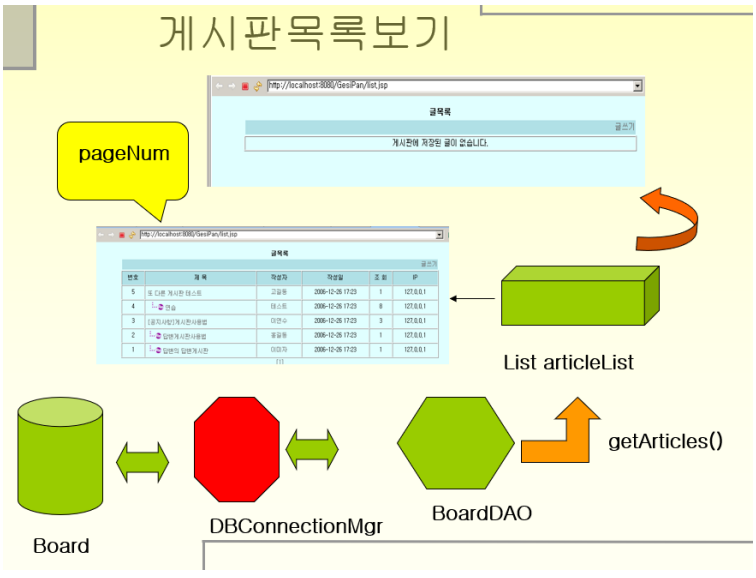
7. 페이지당 맨 처음에 보여주는 게시물 번호

현재페이지*페이지당 보여주는 레코드수

int beginPerPage = nowPage * numPerPage;

ex) 1*10=10,9,8,7,6

2*10=20,19,18,17,16,,



list.jsp (: 게시판 리스트(글 목록) 를 표시하는 main 페이지, 페이징처리도 필요)

```
<%!
        int pageSize = 1; // numPerPage(페이지당 보여주는 게시물 수 )
        int blockSize = 2; // pagePerBlaock(블럭당 보여주는 페이지 수)
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm");
%>
<%
        // 페이징 처리에 해당하는 환경설정  > 게시판을 처음 들어가면, 무조건 1페이지
        String pageNum = request.getParameter("pageNum");
        if(pageNum == null){
                pageNum="1";   // default(무조건 1페이지는 선택하지 않아도 보여줘야함)
        }
        int currentPage = Integer.parseInt(pageNum);
        int startRow = (currentPage-1)*pageSize+1;
        int endRow = currentPage*pageSize;
        int count =0;
        int number =0;
        List articleList = null;
        BoardDAO dbPro = new BoardDAO();
        count = dbPro.getArticleCount();
        System.out.println("현재 레코드수(count):"+count);

        if(count>0){
                articleList = dbPro.getArticles(startRow, pageSize);
        }
        number = count-(currentPage-1)*pageSize;
        System.out.println("페이지별 number:"+number);
%>
<html>
<body bgcolor="#e0ffff">
        <b>글목록(전체 글:<%=count%>)
        </b>
        <table width="700">
                <tr>
                        <td align="right" bgcolor="#b0e0e6"><a href="writeForm.jsp">글쓰기</a></td>
                </tr>
        </table>
        <!-- 데이터의 유무 -->
        <%
                if(count==0){
        %>
        <table border="1" width="700" cellpadding="0" cellspacing="0" align="center">
                <tr>
                        <td align="center">게시판에 저장된 글이 없습니다.</td>
                </tr>
        </table>
```

pageSize : 페이지 당 보여주는 '게시물' 수
blockSize : 블록당 보여주는 '페이지' 수

pageNum = parameter로 넘겨받은 현재페이지 번호
currentPage : 현재 페이지
startRow : 현재 페이지의 시작 게시물 번호
> (currentPage-1) * pageSize + 1;
endRow : 현재 페이지의 마지막 게시물 번호
> currentPage * pageSize;
count : 총 게시물 수 : DAO의 getArticleCount() 로 구함
number : beginPerPage : 페이지별로 시작하는 맨 처음 나오는 게시물 번호
articleList : 화면에 출력할 레코드들을 저장할 변수 (List 객체)

getArticles → 글 목록을 띄우는 method
레코드의 시작번호, 한번에 불러들일 레코드 개수
(startRow, pageSize) 를 매개변수로

DB접속 → select 쿼리문 → 찾아온 데이터(rs)를 articleList 객체에 담음

글이 없으면

```
<%}else{ %>
```
글이 있으면

```
        <table border="1" width="700" cellpadding="0" cellspacing="0" align="center">
            <tr height="30" bgcolor="#b0e0e6">
                <td align="center" width="50">번 호</td>
                <td align="center" width="250">제 목</td>
                <td align="center" width="100">작성자</td>
                <td align="center" width="150">작성일</td>
                <td align="center" width="50">조 회</td>
                <td align="center" width="100">IP</td>
            </tr>
            <!-- 실질적인 출력부분 -->
            <%
                for(int i=0;i<articleList.size();i++){
                    BoardDTO article = (BoardDTO)articleList.get(i); // vecList.elementAt(i)

            %>
            <tr height="30">
                <td align="center" width="50"><%=number-- %></td>
                <td width="250">
                    <!-- 답변글인 경우 먼저 답변이미지를 부착 -->
                    <%
                        int wid=0; // 공백계산용 변수
                        if(article.getRe_level() >0){
                            wid=7*(article.getRe_level());
                    %>
                    <img src="images/level.gif" width="<%=wid %>" height="16">
                    <img src="images/re.gif">
                    <% } else { %>
                    <img src="images/level.gif" width="<%=wid %>" height="16">
                    <% } %>
                    <!-- 글 상세보기 num(게시물번호), pageNum(페이지번호) -->
                    <a href="content.jsp?num=<%=article.getNum() %>&pageNum=<%=currentPage %>">
                        <%=article.getSubject() %>
                    </a>
                    <%
                        if(article.getReadcount()>=20){ %>
                            <img src="images/hot.gif" border="0" height="16">
                        }
                    <% } %>
                </td>
                <td align="center" width="100"><a
                    href="mailto:<%=article.getEmail()%>"><%=article.getWriter() %></a></td>
                <td align="center" width="150"><%=sdf.format(article.getReg_date())%></td>
                <td align="center" width="50"><%=article.getReadcount() %></td>
                <td align="center" width="100"><%=article.getIp() %></td>
            </tr>
            <% }//for %>
        </table>
        <%   }//else %>
```

for문으로
articleList 객체안에 데이터가
있는 만큼 반복해서 출력

답변글인 경우
즉. re_level이 0보다 크면
re_level*7 만큼 들여쓰기

게시물 링크

게시물 제목에다가

조회수가 20 이상이면
Hot 이미지 부착

게시물 레코드의 데이터를 표시
email, writer, reg_date,readcount,ip

```
<div clas="paging">
        <%
```

페이징 처리

```
        if(count>0){
                //1. 총 페이지수를 먼저 구한다
                int pageCount = count / pageSize + (count%pageSize==0?0:1);
                //2. 시작 페이지 => 블럭 당 페이지수를 계산 -> 10 (10의 배수, 3->3의 배수, 6->6의배수)
                int startPage = 0;
                if(currentPage%blockSize!=0){ //1~9, 11~19, 21~29
                        startPage = currentPage/blockSize*blockSize+1;
                }else{
                        startPage = ((currentPage/blockSize)-1)*blockSize+1;
                }
                int endPage = startPage+blockSize-1;
                System.out.println("startPage:"+startPage+",endPage:"+endPage);
```

시작페이지

if 조건이 지금 페이지를 블록크기로 나눈 나머지가 0이 아니래
if(currentPage%blockSize != 0)
자 현재페이지가 5야 블록크기는 10이고
그럼 당연히 나머지는 **5**겠지? 0이 아니라고 true로 넘어가
그러면 startPage = 5/10*10+1 ➔ 1 , 즉, 시작페이지가 1이야
그럼 else를 볼까
현재페이지가 20이고, 블록크기가 10이야 그럼 나누면 딱 떨어지겠찌, **0**으로
그때는 startpPage = ((20/10)-1)*10)+1 ➔ 11이야
즉, 현재페이지가 20인 블록의 시작페이지가 11이란 소리야

현재 블록에서의 마지막페이지가 총 페이지수보다 크다면, 총페이지 수를 엔드페이지로 넣어.
현재 블록이 11~20 인데, 실제 게시물이 들어있는 마지막 페이지가 15야 즉, 총 페이지수가 15란 소리지.
그럼 블록에 표시된 16~20는 그냥 쓸모없는 칸이잖아?
그러니까 그걸 총 페이지수로 바꿔준단 소리지

```
                if(endPage > pageCount) endPage=pageCount;
```

마지막 페이지

```
                //3. 블럭별로 구분해서 링크를 걸어 출력
                // 1) 이전블럭 -> 11 > 10, 4>3
                if(startPage > blockSize) { %>
                        <a href="list.jsp?pageNum=<%=startPage-blockSize %>">[이전]</a>
                <% }
                // 2) 현재블럭 ([1],[2],[3],[4],[5],,,,)
                for(int i=startPage; i<=endPage;i++){%>
                        <a href="list.jsp?pageNum=<%=i %>">[<%=i %>]</a>
                <%}
                // 3) 다음블럭
                if(endPage < pageCount) { %>
                        <a href="list.jsp?pageNum=<%=startPage+blockSize %>">[다음]</a>
                <%
                        }
        %>
        </div>
</body>
</html>
```

[이전] 블록 링크 ([다음]도 같은 원리
지금 시작페이지가 블록 크키보다 크대
현재 블록이 11~20 이야 그럼 시작페이지는 11 이겠지? 블록크기는 10이야 그럼 범위보다 크잖아?
그래서 시작페이지에서 블록크기를 빼
그러면 11-10 ➔ 1 이지, 즉 "이전"버튼을 누르면 시작페이지가 1인 블록으로 간단 소리야

내가 보고있는 블록이 21~30 도 마찬가지야
현재 시작이 21이야 거기서 블록크기 10을 빼면 11이지 , 이전버튼을 누르면 11~20 블록으로 간단소리야

[현재블록] [1] [2] [3] [4] [5] ,,,,[10]
여긴 간단해
for 문으로 출력해주는거야
현재 블록의 startPage 부터 endPage 까지 출력해주는데, 거기에다가 pageNum을 파라미터로 넘겨주는 링크를 걸어주는거야

시작페이지가 11, 엔드페이지가 20이면
11부터 20까지
11 12 13 14 15 16 17 18 19 20 이 차례대로 출력되는거지 거기에 [ ] 만 붙여준거야

BoardDAOjsp (: 글 목록)

```
// 2. 글 목록 보기에 대한 method 구현 -> 레코드가  한 개 이상 -> 페이지당 10개식 끊어주는 기능
// (1) 레코드의 시작번호  (2) 한 번에불러올 레코드의 갯수(10,15,30~)
        public List<BoardDTO> getArticles(int start,int end) {
        List<BoardDTO> articleList = null; // ArrayList<BoardDTO> articleList = null; 도 같음

        try {
                con = pool.getConnection();
                sql="select * from board order by ref desc,re_step asc limit ?,?"; // limit ?,? 몇개의 범위?
                pstmt = con.prepareStatement(sql);
                pstmt.setInt(1, start-1); // mysql은 순번이 내부적으로 0부터 시작)
                pstmt.setInt(2, end);
                rs = pstmt.executeQuery();
                // 페이징 처리 는 기본적으로 누적의 개념을 도입한다.
                // 기존의 레코드 외에 추가된 레코드를 첨부해서 모두 보여주기 위해서는 누적(do~while)을 사용한다.
                if(rs.next()) { // 만약 레코드가 1개라도 존재한다면
                        articleList = new ArrayList(end); // end 가 10이라면 10개가 저장 가능한 데이터 공간이 만들어짐.
                        do {
                                BoardDTO article = makeArticleFromResult();
                                // 레코드를 찾을 때마다, articleList에 담아야함
                                articleList.add(article);
                        }while(rs.next()); // 자료가 있는 동안!
                }
        }catch(Exception e) {
                System.out.println("getArticles() method error:"+e);
        }finally {
                pool.freeConnection(con,pstmt,rs);
        }
        return articleList;
        }

        ,,,
        // 중복된 내용의 레코드를 담을 수 있는 method를 따로 만들어서 호출 -> 공개할 필요가 없다
        private BoardDTO makeArticleFromResult() throws Exception {
                BoardDTO article = new BoardDTO();
                article = new BoardDTO(); // end 가 10이라면 10개가 저장 가능한 데이터 공간이 만들어짐.
                article.setNum(rs.getInt("num"));
                // 문자열값
                article.setWriter(rs.getString("writer"));
                article.setSubject(rs.getString("subject"));
                article.setEmail(rs.getString("email"));
                article.setPasswd(rs.getString("passwd"));
                article.setReg_date(rs.getTimestamp("reg_date")); // 오늘 날짜 값
                // 정수값(조회수,답변에 대한 필드)
                article.setReadcount(rs.getInt("readcount")); // default =0
                article.setRef(rs.getInt("ref")); // 그루번호
                article.setRe_step(rs.getInt("re_step")); // 답변글의 순서
                article.setRe_level(rs.getInt("re_level")); // 들여쓰기
                article.setContent(rs.getString("content")); // 글내용
                article.setIp(rs.getString("ip")); // 글쓴이의 ip
                return article;
        }
```

DB연결 및 SQL문 결합

rs 객체로부터 필드들을 읽어와서 article 객체에 담고 그 article 객체를 return 하는 method

BoardDAOjsp (: 글 쓰기)

```
//3.글쓰기 및 답변글까지 구현
	public void insertArticle(BoardDTO article) {   //1.article -> 신규 or 답변? -> re_ref 로 확인
	int num = article.getNum(); // 이게 0 이면 신규글  /  !=0 답변글   // 구분이 목적인 변수
	int ref = article.getRef();
	int re_step = article.getRe_step();
	int re_level= article.getRe_level();
	int number = 0; // 데이터를 넣어줄 때 필요로 하는 게시물 번호 // 새로!! 넣을 목적의 변수

	try {
		con = pool.getConnection();
		sql="select max(num) from board"; // 최대값 +1 -> 실제 저장할 게시물 번호를 생성
		pstmt = con.prepareStatement(sql);
		rs = pstmt.executeQuery();
		if(rs.next()) { // 현재 테이블에서 데이터가 존재한다면
			number = rs.getInt(1)+1;
		}else { // 맨처음에 레코드가 없다면
			number=1;
		}
		if(num!=0) {   //-> 답변글
			// 같은 그룹번호를 가지고 있으면서 나(새로 추가된 답변글)보다 re_step값이 큰 답변글을 찾아서
			// re_step값을 하나 증가시킨다
			sql="update board set re_step = re_step+1 where ref=? and re_step > ?";
			pstmt = con.prepareStatement(sql);
			pstmt.setInt(1, article.getRef());
			pstmt.setInt(2, re_step);
			int update=pstmt.executeUpdate();
			re_step+=1;
			re_level+=1;
		}else { // 신규글
			ref=number; // 그룹번호가 단독으로 사용될 때에는 num과 같이 구분자로 사용하기 때문
			re_step=0;
			re_level=0;
		}
		sql="insert into board(writer,email,subject,passwd,reg_date,ref,re_step,re_level,content,ip)values(?,?,?,?,?,?,?,?,?,?)";
		pstmt = con.prepareStatement(sql); // 웹에 입력, 저장 -> setter Method 호출
			pstmt.setString(1, article.getWriter());
			pstmt.setString(2, article.getEmail());
			pstmt.setString(3, article.getSubject());
			pstmt.setString(4, article.getPasswd());
			//wirtePro.jsp(날짜는 어떻게 저장할것인가?)
			pstmt.setTimestamp(5, article.getReg_date()); //
			pstmt.setInt(6, ref); // number pstmt.setInt(6,article.getRef()   --> X 아님!
			pstmt.setInt(7, re_step); //0
			pstmt.setInt(8, re_level); //0
			pstmt.setString(9, article.getContent());
			pstmt.setString(10, article.getIp()); // request.getRemoteAddr();
			int insert = pstmt.executeUpdate();
		}catch(Exception e) {
			System.out.println("insertArticle method오류 :"+e);
		}finally{

		}
	}
```

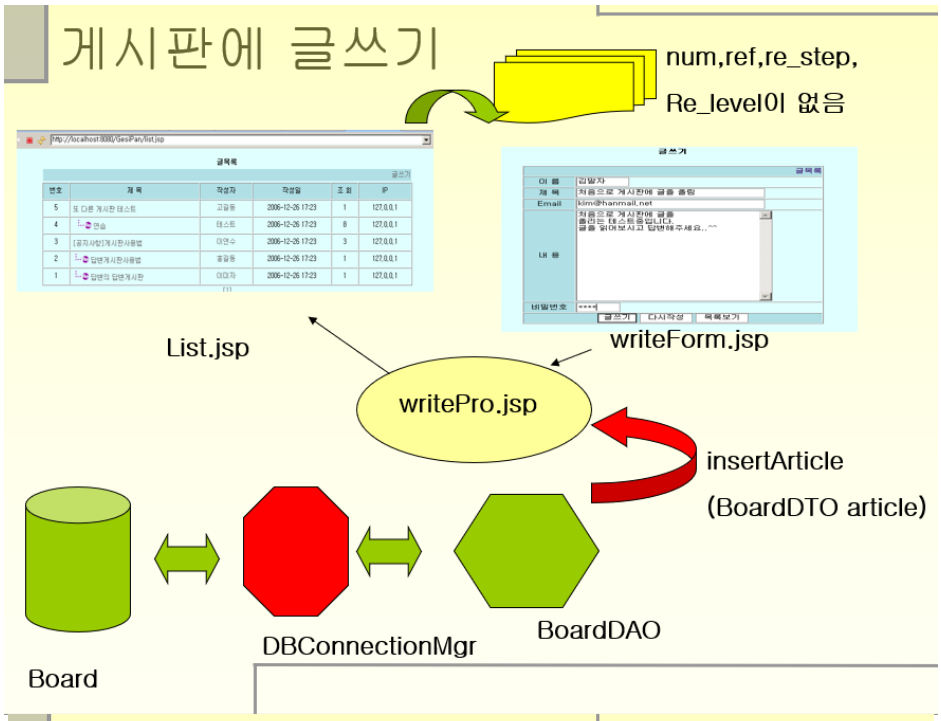
DB연결 및 SQL문 결합 1
: 먼저 마지막 게시물 번호를 가져온다!

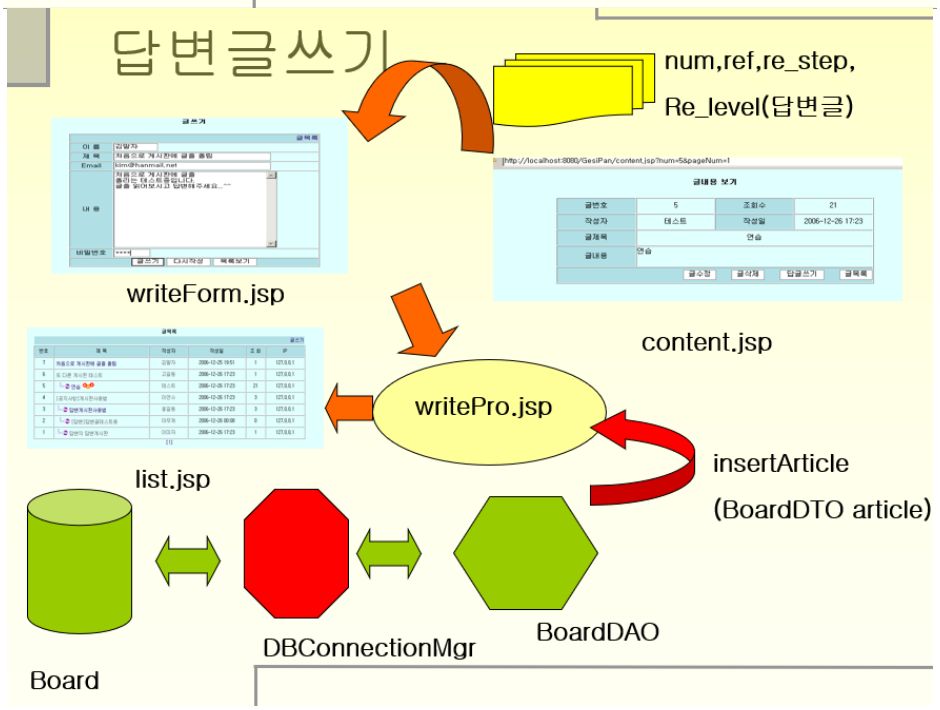
이게 답변글이면

ref
re_step
re_level
설정
답변글이던 신규글이던
둘다 글쓰기

이게 신규글이면

article로부터 DB로
값을 insert 해줌





남은 부분은 **글 상세보기, 글 수정, 글 삭제임**

그 부분들도 거의 거기서 거기

jsp 로 화면구현 과 java 구현부를 만들어주고,

DAO에 method 구현임