

JSP Board Paging

JSP Board Paging

Paging

Paging Algorithm

자료의 수가 많거나 한 화면에
다 보여줄 수 없는 경우
페이지를 분할하여
해당 내용을 보여주는 기법으로

통상적으로 Query문을 통하여 구현하지만
전체의 내용을 JavaScript로 분할하거나
java Object를 이용하여 분할하기도 한다.

페이징을 처리하는 방식은 매우 다양한다.

데이터베이스 입력

```
mysql> desc page;
```

Field	Type	Null	Key	Default	Extra
num	int	NO	PRI	NULL	auto_increment
title	text	YES		NULL	
content	text	YES		NULL	

```
3 rows in set (0.00 sec)
```

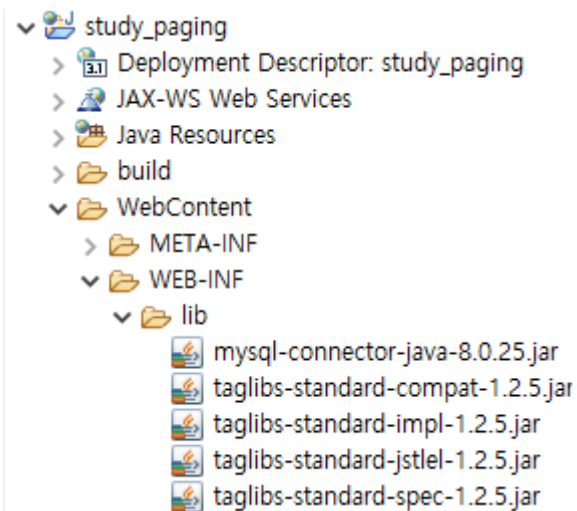
```
mysql> select * from page;
```

num	title	content
1	1번째 제목	1번째 내용
2	2번째 제목	2번째 내용
3	3번째 제목	3번째 내용
4	4번째 제목	4번째 내용
5	5번째 제목	5번째 내용
6	6번째 제목	6번째 내용
7	7번째 제목	7번째 내용
8	8번째 제목	8번째 내용
9	9번째 제목	9번째 내용
10	10번째 제목	10번째 내용
11	11번째 제목	11번째 내용
12	12번째 제목	12번째 내용
13	13번째 제목	13번째 내용
14	14번째 제목	14번째 내용
15	15번째 제목	15번째 내용
16	16번째 제목	16번째 내용
17	17번째 제목	17번째 내용
18	18번째 제목	18번째 내용
19	19번째 제목	19번째 내용
20	20번째 제목	20번째 내용
21	21번째 제목	21번째 내용
22	22번째 제목	22번째 내용
23	23번째 제목	23번째 내용
24	24번째 제목	24번째 내용
25	25번째 제목	25번째 내용
26	26번째 제목	26번째 내용
27	27번째 제목	27번째 내용
28	28번째 제목	28번째 내용
29	29번째 제목	29번째 내용
30	30번째 제목	30번째 내용
31	31번째 제목	31번째 내용
32	32번째 제목	32번째 내용
33	33번째 제목	33번째 내용
34	34번째 제목	34번째 내용

```
34 rows in set (0.00 sec)
```

자료의 양이 많을 수록 테스트가 편리하다.

개발 세팅



```
index.jsp
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
4 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
5 <!DOCTYPE html>
6 <html>
7   <head>
8     <meta charset="utf-8">
9     <title>자료 목록</title>
10  </head>
11  <body>
12    <form action="ReadPage" method="get">
13      <input type="hidden" name="currentPage" value="1">
14      한 페이지에 보여줄 수를 골라주세요
15      <select name="recordsPerPage">
16        <option value="5">5</option>
17        <option value="10" selected>10</option>
18        <option value="15">15</option>
19      </select>
20      <input type="submit" value="보기">
21    </form>
22  </body>
23 </html>
```

JDBC와 taglib들을 저장해준뒤 index.jsp를 작성

Paging 처리의 개념

SQL limit의 활용

`select * from table limit n1, n2;`
해당 자료를 보여주되
n1번부터 시작하여 n2개를 보여준다.

```
mysql> select * from page limit 0, 10;
```

num	title	content
1	1번째제목	1번 내용
2	2번째제목	2번 내용
3	3번째제목	3번 내용
4	4번째제목	4번 내용
5	5번째제목	5번 내용
6	6번째제목	6번 내용
7	7번째제목	7번 내용
8	8번째제목	8번 내용
9	9번째제목	9번 내용
10	10번째제목	10번 내용

10 rows in set (0.00 sec)

```
mysql> select * from page limit 10, 10;
```

num	title	content
11	11번째제목	11번 내용
12	12번째제목	12번 내용
13	13번째제목	13번 내용
14	14번째제목	14번 내용
15	15번째제목	15번 내용
16	16번째제목	16번 내용
17	17번째제목	17번 내용
18	18번째제목	18번 내용
19	19번째제목	19번 내용
20	20번째제목	20번 내용

10 rows in set (0.00 sec)

```
mysql> select * from page limit 20, 10;
```

num	title	content
21	21번째제목	21번 내용
22	22번째제목	22번 내용
23	23번째제목	23번 내용
24	24번째제목	24번 내용
25	25번째제목	25번 내용
26	26번째제목	26번 내용
27	27번째제목	27번 내용
28	28번째제목	28번 내용
29	29번째제목	29번 내용
30	30번째제목	30번 내용

10 rows in set (0.00 sec)

```
mysql> select * from page limit 30, 10;
```

num	title	content
31	31번째제목	31번 내용
32	32번째제목	32번 내용
33	33번째제목	33번 내용
34	34번째제목	34번 내용

4 rows in set (0.00 sec)

n1과 n2값을 변경하여 페이징 구성이 가능

Paging 처리의 개념

Current Page 변수

현재 확인중인 페이지를 나타낸다.

확인중인 페이지는

n1으로 부터 시작하여 n2만큼의 자료를 보여준다.

현재 페이지에 보여주는 자료의 수는 n2개이며
다음 페이지에서 시작되는 자료는 n1+n2+1부터이다.

P1 : 0~10 (n1=0, n2=10)

P2 : 11~20 (n1=10, n2=10)

P3 : 21~30 (n1=20, n2=10)

P4 : 31~34 (n1=30, n2=10)

다음 페이지의 n1을 결정하는 것은
현재 페이지의 n1과 n2이다.

auto_increment의 index와 limit의 index가 다름

```
mysql> select * from page limit 10, 10;
```

num	title	content
11	11번째제목	11번 내용
12	12번째제목	12번 내용
13	13번째제목	13번 내용
14	14번째제목	14번 내용
15	15번째제목	15번 내용
16	16번째제목	16번 내용
17	17번째제목	17번 내용
18	18번째제목	18번 내용
19	19번째제목	19번 내용
20	20번째제목	20번 내용

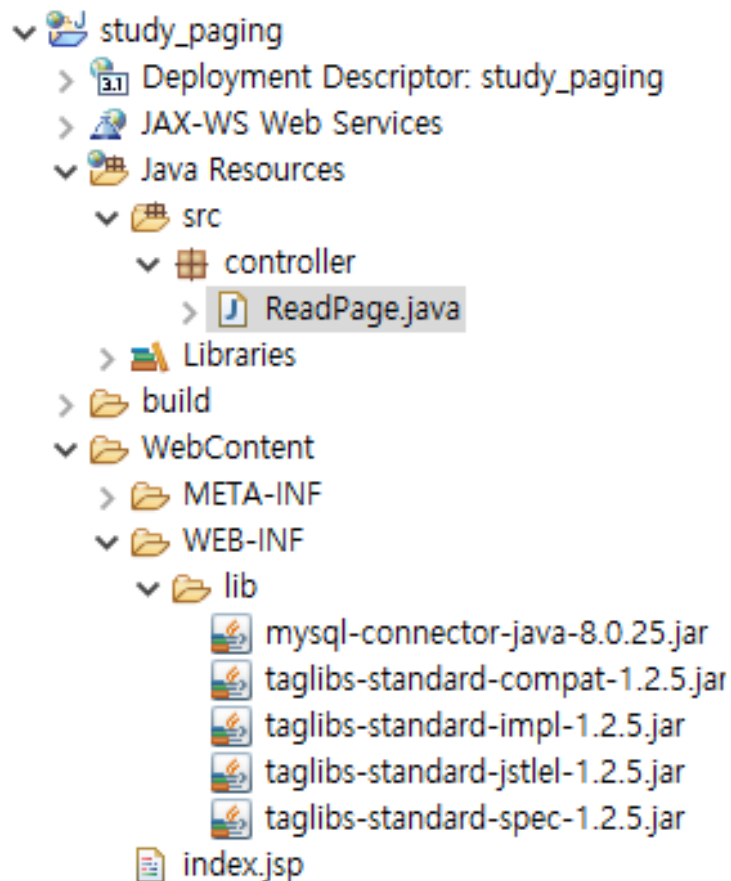
10 rows in set (0.00 sec)

```
mysql> select * from page limit 20, 5;
```

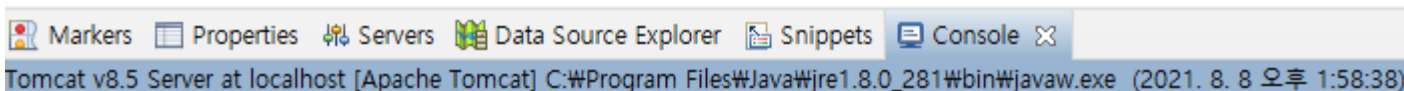
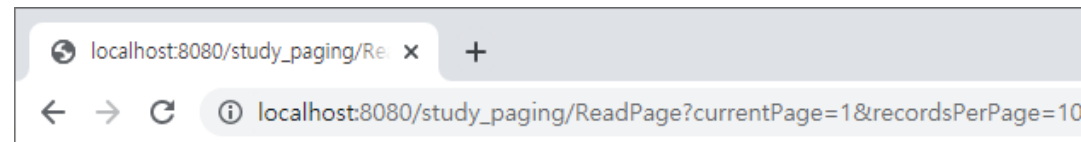
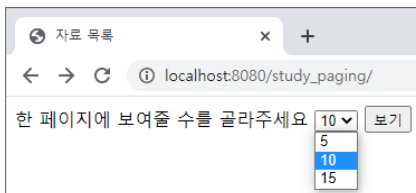
num	title	content
21	21번째제목	21번 내용
22	22번째제목	22번 내용
23	23번째제목	23번 내용
24	24번째제목	24번 내용
25	25번째제목	25번 내용

5 rows in set (0.00 sec)

서블릿 제작

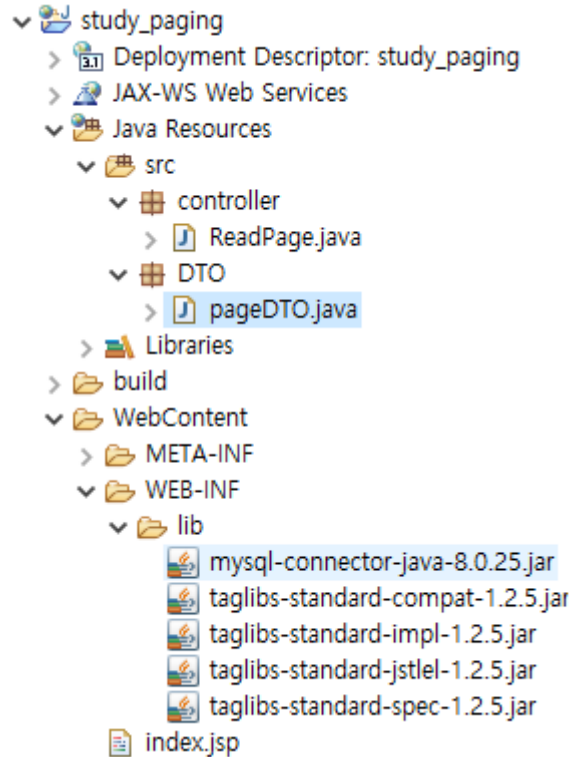


```
1 package controller;
2
3 import java.io.IOException;
4
5
6
7
8
9
10 /**
11  * Servlet implementation class ReadPage
12  */
13 @WebServlet("/ReadPage")
14 public class ReadPage extends HttpServlet {
15     private static final long serialVersionUID = 1L;
16     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
17         String currentPage=request.getParameter("currentPage");
18         String recordsPerPage=request.getParameter("recordsPerPage");
19         System.out.println("현재 페이지 : "+currentPage+", 페이지 당 데이터 : "+recordsPerPage);
20     }
21     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
22     }
23 }
24 }
```



get을 통하여 페이지징을 구현할 패러미터 전달

DTO 제작

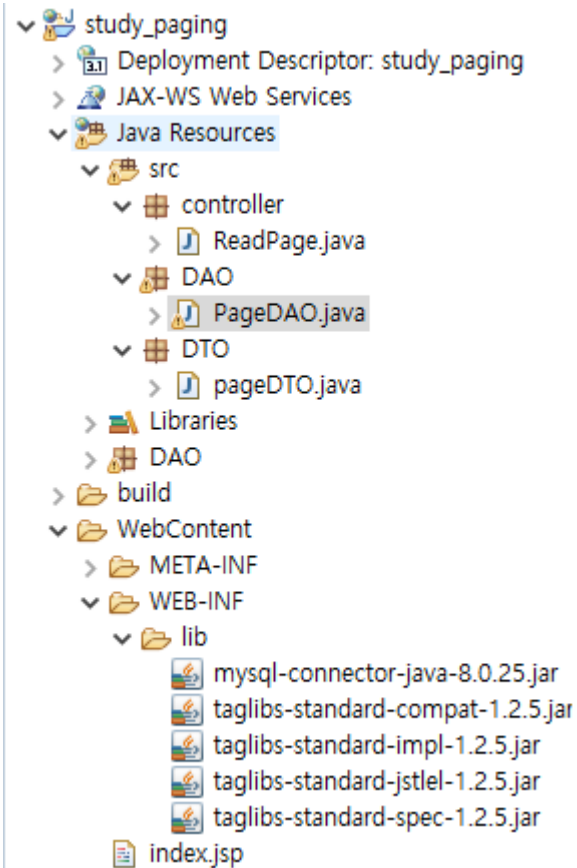


```
1 package DTO;
2
3 public class pageDTO {
4     private int num;
5     private String title;
6     private String content;
7     public int getNum() {
8         return num;
9     }
10    public void setNum(int num) {
11        this.num = num;
12    }
13    public String getTitle() {
14        return title;
15    }
16    public void setTitle(String title) {
17        this.title = title;
18    }
19    public String getContent() {
20        return content;
21    }
22    public void setContent(String content) {
23        this.content = content;
24    }
25 }
```

```
mysql> desc page;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| num   | int  | NO   | PRI | NULL    | auto_increment |
| title | text | YES  |     | NULL    |                 |
| content | text | YES  |     | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

데이터를 보관하기 위한 클래스 제작

DAO 제작



```
PageDAO.java
1 package DAO;
2
3 import java.sql.*;
4 import java.util.*;
5
6 public class PageDAO {
7     private PageDAO() {
8
9     }
10
11     private static PageDAO instance=new PageDAO();
12
13     public static PageDAO getInstance() {
14         return instance;
15     }
16
17     //커넥션 기능 처리
18     public Connection getConnection() throws Exception{
19         Connection conn=null;
20         String url="jdbc:mysql://127.0.0.1:3306/test";
21         String id="root";
22         String pass="iotiot";
23
24         Class.forName("com.mysql.cj.jdbc.Driver");
25         conn=DriverManager.getConnection(url, id, pass);
26         return conn;
27     }
28 }
```

데이터 접근을 위한 DAO 제작

조회기능 제작

```
30 //리스트 출력을 위한 정보처리
31 public List<pageDTO> findList() {
32     List<pageDTO> List=new ArrayList<pageDTO>();
33     String sql="select * from page";
34     Connection conn=null;
35     PreparedStatement pstmt=null;
36     ResultSet rs=null;
37
38     try {
39         conn=getConnection();
40         pstmt=conn.prepareStatement(sql);
41         rs=pstmt.executeQuery();
42         while(rs.next()) {
43             pageDTO p=new pageDTO();
44             p.setNum(rs.getInt("num"));
45             p.setTitle(rs.getString("title"));
46             p.setContent(rs.getString("content"));
47             List.add(p);
48         }
49     }catch(Exception e) {
50         System.out.println("접속중 오류발생 : "+e);
51     }finally {
52         try {
53             if(rs!=null) {rs.close();}
54             if(pstmt!=null) {pstmt.close();}
55             if(conn!=null) {conn.close();}
56         }catch(Exception e){|
57             System.out.println("접속 종료 중 오류발생");
58         }
59     }
60     return List;
61 }
```

DAO 내부에 제작

servlet 처리

```
ReadPage.java
1 package controller;
2
3 import java.io.IOException;
4 import java.util.List;
5
6 import DTO.pageDTO;
7 import DAO.PageDAO;
8
9 import javax.servlet.*;
10 import javax.servlet.annotation.*;
11 import javax.servlet.http.*;
12
13 @WebServlet("/ReadPage")
14 public class ReadPage extends HttpServlet {
15     private static final long serialVersionUID = 1L;
16     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
17         String currentPage=request.getParameter("currentPage");
18         String recordsPerPage=request.getParameter("recordsPerPage");
19         System.out.println("현재 페이지 : "+currentPage+", 페이지 당 데이터 : "+recordsPerPage);
20
21         PageDAO BDO=PageDAO.getInstance();
22
23         List<pageDTO> data=BDO.findList();
24         request.setAttribute("data", data);
25
26         request.setAttribute("currentPage", currentPage);
27         request.setAttribute("recordsPerPage", recordsPerPage);
28
29         RequestDispatcher dispatcher=request.getRequestDispatcher("list.jsp");
30         dispatcher.forward(request, response);
31     }
32     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
33
34     }
35 }
```

만들어진 DAO의 doGet을 변경

리스트 제작

```
list.jsp
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
4 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
5 <!DOCTYPE html>
6 <html>
7   <head>
8     <meta charset="utf-8">
9     <title>도시 리스트</title>
10    <link rel="stylesheet" href="page.css"/>
11  </head>
12  <body>
13    <h1>
14      currentPage : ${currentPage}<br>
15      recordsPerPage : ${recordsPerPage}<br>
16    </h1>
17    <table>
18      <tr>
19        <th>num</th>
20        <th>name</th>
21        <th>population</th>
22      </tr>
23      <c:forEach items="${data}" var="data">
24        <tr>
25          <td>${data.getNum()}</td>
26          <td>${data.getTitle()}</td>
27          <td>${data.getContent()}</td>
28        </tr>
29      </c:forEach>
30    </table>
31  </body>
32 </html>
```

```
page.css
1 @charset "utf-8";
2
3 ul li{
4   display:inline-block;
5 }
```

currentPage : 1
recordsPerPage : 10

num	name	population
1	1번째목	1번 내용
2	2번째목	2번 내용
3	3번째목	3번 내용
4	4번째목	4번 내용
5	5번째목	5번 내용
6	6번째목	6번 내용
7	7번째목	7번 내용
8	8번째목	8번 내용
9	9번째목	9번 내용
10	10번째목	10번 내용
11	11번째목	11번 내용
12	12번째목	12번 내용
13	13번째목	13번 내용
14	14번째목	14번 내용
15	15번째목	15번 내용
16	16번째목	16번 내용
17	17번째목	17번 내용
18	18번째목	18번 내용
19	19번째목	19번 내용
20	20번째목	20번 내용
21	21번째목	21번 내용
22	22번째목	22번 내용
23	23번째목	23번 내용
24	24번째목	24번 내용
25	25번째목	25번 내용
26	26번째목	26번 내용
27	27번째목	27번 내용
28	28번째목	28번 내용
29	29번째목	29번 내용
30	30번째목	30번 내용
31	31번째목	31번 내용
32	32번째목	32번 내용
33	33번째목	33번 내용
34	34번째목	34번 내용

간략한 css까지 제작

페이지 알고리즘 제작

```
30 //리스트 출력을 위한 정보처리
31 //현재 페이지가 몇 페이지인지, 한 페이지 당 보여줄 자료의 양을 매개변수로 전달
32 public List<pageDTO> findList(int currentPage, int recordsPerPage) {
33     List<pageDTO> list=new ArrayList<pageDTO>();
34     int start=currentPage*recordsPerPage-recordsPerPage;
35     //1페이지가 1~9
36     //2페이지가 10~19
37     //3페이지가 20~29
38     //인 경우 현재 페이지 값(1)*한 페이지 당 자료 수(10)을 진행한 뒤 페이지 당 자료 수를 빼주면 시작값을 찾아낼 수 있다.
39     String sql="select * from page limit ?, ?";//쿼리문 수정
40     Connection conn=null;
41     PreparedStatement pstmt=null;
42     ResultSet rs=null;
43
44     try {
45         conn=getConnection();
46         pstmt=conn.prepareStatement(sql);
47         pstmt.setInt(1, start);
48         pstmt.setInt(2, recordsPerPage);
49         rs=pstmt.executeQuery();
50         while(rs.next()) {
51             pageDTO p=new pageDTO();
52             p.setNum(rs.getInt("num"));
53             p.setTitle(rs.getString("title"));
54             p.setContent(rs.getString("content"));
55             list.add(p);
56         }
57     }catch(Exception e) {
58         System.out.println("접속중 오류발생 : "+e);
59     }finally {
60         try {
61             if(rs!=null) {rs.close();}
62             if(pstmt!=null) {pstmt.close();}
63             if(conn!=null) {conn.close();}
64         }catch(Exception e){
65             System.out.println("접속 종료 중 오류발생");
66         }
67     }
68     return list;
69 }
```

DAO.findList() 부분을 수정

페이지 알고리즘 제작

```
3 import java.io.IOException;
4 import java.util.List;
5
6 import DTO.pageDTO;
7 import DAO.PageDAO;
8
9 import javax.servlet.*;
10 import javax.servlet.annotation.*;
11 import javax.servlet.http.*;
12
13 @WebServlet("/ReadPage")
14 public class ReadPage extends HttpServlet {
15     private static final long serialVersionUID = 1L;
16     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
17         int currentPage=Integer.parseInt(request.getParameter("currentPage")); //계산이 가능하도록 int로 변경
18         int recordsPerPage=Integer.parseInt(request.getParameter("recordsPerPage"));
19         System.out.println("현재 페이지 : "+currentPage+", 페이지 당 데이터 : "+recordsPerPage);
20
21         PageDAO BDO=PageDAO.getInstance();
22
23         List<pageDTO> data=BDO.findList(currentPage, recordsPerPage); //메서드에게 해당 내용을 전달
24         request.setAttribute("data", data);
25
26         request.setAttribute("currentPage", currentPage);
27         request.setAttribute("recordsPerPage", recordsPerPage);
28
29         RequestDispatcher dispatcher=request.getRequestDispatcher("list.jsp");
30         dispatcher.forward(request, response);
31     }
32     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
33
34     }
35 }
```

currentPage : 1
recordsPerPage : 10

num	name	population
1	1번제목	1번 내용
2	2번제목	2번 내용
3	3번제목	3번 내용
4	4번제목	4번 내용
5	5번제목	5번 내용
6	6번제목	6번 내용
7	7번제목	7번 내용
8	8번제목	8번 내용
9	9번제목	9번 내용
10	10번제목	10번 내용

currentPage : 1
recordsPerPage : 5

num	name	population
1	1번제목	1번 내용
2	2번제목	2번 내용
3	3번제목	3번 내용
4	4번제목	4번 내용
5	5번제목	5번 내용

ReadPage.doGet 부분을 수정

페이지 개수 처리

```
70 //전체 자료의 갯수를 가져오는 메서드
71 public int getNumberOfRows() {
72     String sql="select count(num) from page";
73     int numOfRows=0;//몇개의 페이지가 존재하는지 확인하는 부분
74     Connection conn=null;
75     PreparedStatement pstmt=null;
76     ResultSet rs=null;
77
78     try {
79         conn=getConnection();
80         pstmt=conn.prepareStatement(sql);
81         rs=pstmt.executeQuery();
82         rs.next();
83         numOfRows=Integer.parseInt(rs.getString(1));
84     }catch(Exception e) {
85         System.out.println("접속중 오류발생");
86     }finally {
87         try {
88             if(rs!=null) {rs.close();}
89             if(pstmt!=null) {pstmt.close();}
90             if(conn!=null) {conn.close();}
91         }catch(Exception e){
92             System.out.println("접속 종료 중 오류발생");
93         }
94     }
95     return numOfRows;
96 }
```

```
13 @WebServlet("/ReadPage")
14 public class ReadPage extends HttpServlet {
15     private static final long serialVersionUID = 1L;
16     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
17         int currentPage=Integer.parseInt(request.getParameter("currentPage"));
18         int recordsPerPage=Integer.parseInt(request.getParameter("recordsPerPage"));
19         System.out.println("현재 페이지 : "+currentPage+", 페이지 당 데이터 : "+recordsPerPage);
20
21         PageDAO BDO=PageDAO.getInstance();
22
23         List<pageDTO> data=BDO.findList(currentPage, recordsPerPage);
24         request.setAttribute("data", data);
25         int row=BDO.getNumberOfRows(); //페이지 수를 가져오는 메서드를 실행
26         int nOfPages=row/recordsPerPage; //전체자료수/페이지당자료수=페이지의 수
27         //34/10인 경우 나머지가 4가 남는다, 이 경우 하나의 별도 페이지를 구성해야 하므로 1개의 페이지를 추가로 늘려준다.
28         if(nOfPages%recordsPerPage>0) {
29             nOfPages++;
30         }
31         request.setAttribute("nOfPages", nOfPages); //총 몇페이지인지 데이터를 request에 저장
32         request.setAttribute("currentPage", currentPage);
33         request.setAttribute("recordsPerPage", recordsPerPage);
34
35         RequestDispatcher dispatcher=request.getRequestDispatcher("list.jsp");
36         dispatcher.forward(request, response);
37     }
38     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
39
40     }
41 }
```

전체 자료 수를 페이지 당 자료수로 나누어준다.
DAO 내부에 메서드를 작성한 뒤 servlet에서 처리한다.

페이지 표시

```
list.jsp
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
4 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
5 <!DOCTYPE html>
6 <html>
7   <head>
8     <meta charset="utf-8">
9     <title>도시 리스트</title>
10    <link rel="stylesheet" href="page.css"/>
11  </head>
12  <body>
13    <h1>
14      nOfPages : ${nOfPages}</br>
15      currentPage : ${currentPage}</br>
16      recordsPerPage : ${recordsPerPage}</br>
17    </h1>
18    <table>
19      <tr>
20        <th>num</th>
21        <th>name</th>
22        <th>population</th>
23      </tr>
24      <c:forEach items="${data }" var="data">
25        <tr>
26          <td>${data.getNum() }</td>
27          <td>${data.getTitle() }</td>
28          <td>${data.getContent() }</td>
29        </tr>
30      </c:forEach>
31    </table>
32
33    <ul>
34      <c:forEach begin="1" end="${nOfPages}" var="i">
35        <li>${i }</li>
36      </c:forEach>
37    </ul>
38  </body>
39 </html>
```

localhost:8080/study_paging/

한 페이지에 보여줄 수를 골라주세요 10 보기

localhost:8080/study_paging/

한 페이지에 보여줄 수를 골라주세요 5 보기

nOfPages : 4
currentPage : 1
recordsPerPage : 10

num	name	population
1	1번째목	1번 내용
2	2번째목	2번 내용
3	3번째목	3번 내용
4	4번째목	4번 내용
5	5번째목	5번 내용
6	6번째목	6번 내용
7	7번째목	7번 내용
8	8번째목	8번 내용
9	9번째목	9번 내용
10	10번째목	10번 내용

1 2 3 4

nOfPages : 7
currentPage : 1
recordsPerPage : 5

num	name	population
1	1번째목	1번 내용
2	2번째목	2번 내용
3	3번째목	3번 내용
4	4번째목	4번 내용
5	5번째목	5번 내용

1 2 3 4 5 6 7

list.jsp에서 데이터를 통해 페이지를 표시

페이지 링크처리

```
<ul>
  <c:forEach begin="1" end="${nOfPages}" var="i">
    <c:choose>
      <c:when test="${currentPage eq i}">
        <li><a>${i }(현재)</a></li>
      </c:when>
      <c:otherwise>
        <li><a href="ReadPage?recordsPerPage=${recordsPerPage }&currentPage=${i}">${i }</a></li>
      </c:otherwise>
    </c:choose>
  </c:forEach>
</ul>
```

nOfPages : 7
currentPage : 1
recordsPerPage : 5

num name population

1	1번제목	1번 내용
2	2번제목	2번 내용
3	3번제목	3번 내용
4	4번제목	4번 내용
5	5번제목	5번 내용

1(현재) 2 3 4 5 6 7

nOfPages : 7
currentPage : 4
recordsPerPage : 5

num name population

16	16번제목	16번 내용
17	17번제목	17번 내용
18	18번제목	18번 내용
19	19번제목	19번 내용
20	20번제목	20번 내용

1 2 3 4(현재) 5 6 7

nOfPages : 7
currentPage : 7
recordsPerPage : 5

num name population

31	31번제목	31번 내용
32	32번제목	32번 내용
33	33번제목	33번 내용
34	34번제목	34번 내용

1 2 3 4 5 6 7(현재)

servlet에 currentPage값을 전달하여 이동

부가기능 처리

```
<ul>
  <c:if test="${currentPage != 1 }">
    <li><a href="ReadPage?recordsPerPage=${recordsPerPage }&currentPage=${currentPage-1}">이전페이지</a></li>
  </c:if>
  <c:forEach begin="1" end="${nOfPages}" var="i">
    <c:choose>
      <c:when test="${currentPage eq i }">
        <li><a href="#">${i }(현재)</a></li>
      </c:when>
      <c:otherwise>
        <li><a href="ReadPage?recordsPerPage=${recordsPerPage }&currentPage=${i}">${i }</a></li>
      </c:otherwise>
    </c:choose>
  </c:forEach>
  <c:if test="${currentPage lt nOfPages }">
    <li><a href="ReadPage?recordsPerPage=${recordsPerPage }&currentPage=${currentPage+1}">다음페이지</a></li>
  </c:if>
</ul>
```

nOfPages : 7
currentPage : 1
recordsPerPage : 5

num	name	population
1	1번째목	1번 내용
2	2번째목	2번 내용
3	3번째목	3번 내용
4	4번째목	4번 내용
5	5번째목	5번 내용

1(현재) 2 3 4 5 6 7 다음페이지

nOfPages : 7
currentPage : 4
recordsPerPage : 5

num	name	population
16	16번째목	16번 내용
17	17번째목	17번 내용
18	18번째목	18번 내용
19	19번째목	19번 내용
20	20번째목	20번 내용

이전페이지 1 2 3 4(현재) 5 6 7 다음페이지

nOfPages : 7
currentPage : 7
recordsPerPage : 5

num	name	population
31	31번째목	31번 내용
32	32번째목	32번 내용
33	33번째목	33번 내용
34	34번째목	34번 내용

이전페이지 1 2 3 4 5 6 7(현재)

첫 페이지와 마지막인 경우는 없음