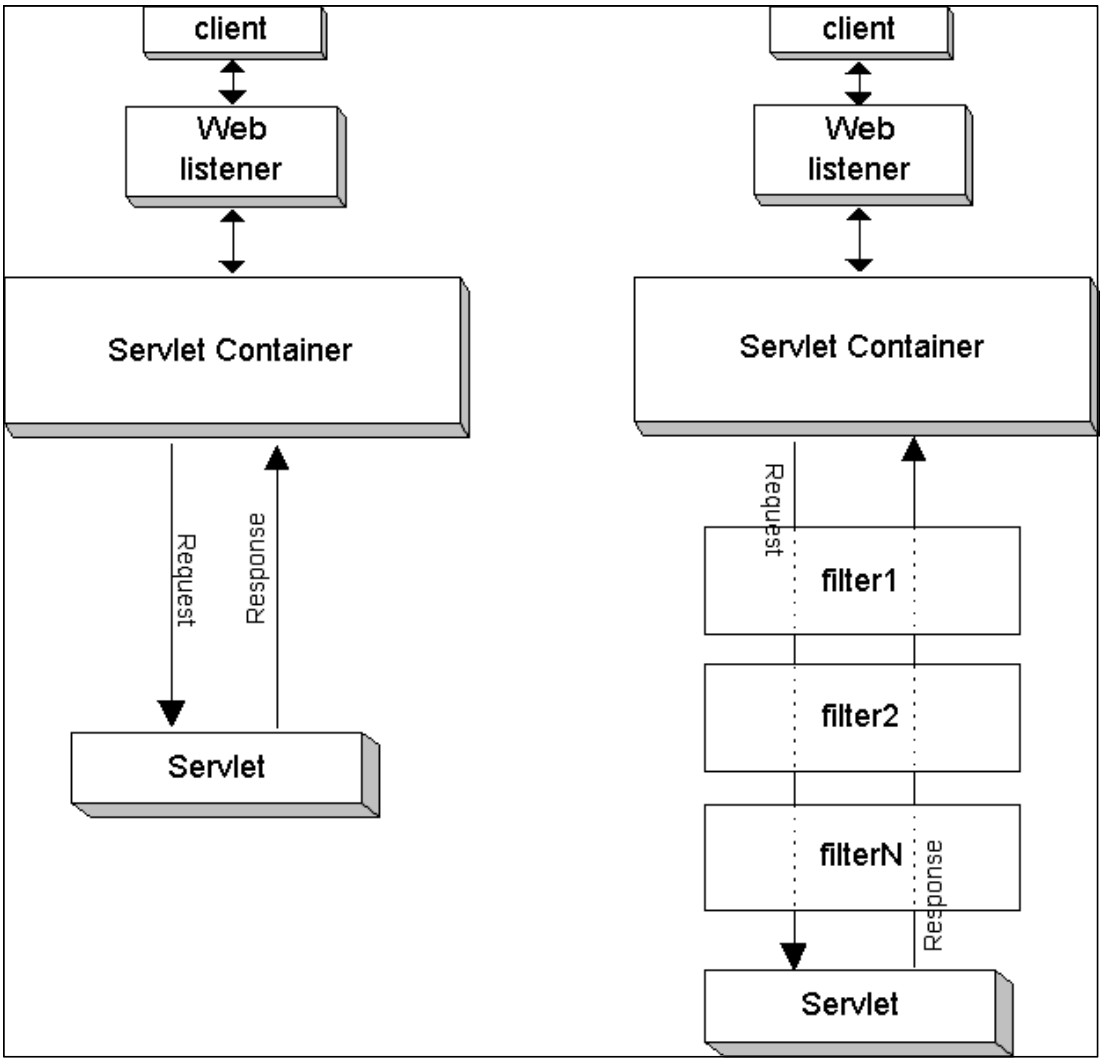
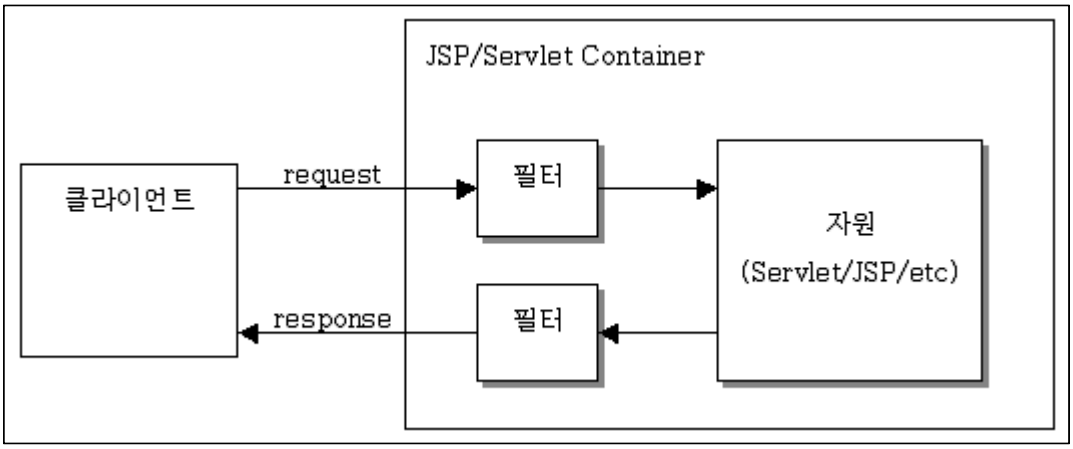
■ Filter를 이용한 post방식의 한글처리

- filter는 request와 response를 변경할 수 있는 재사용한 코드
- servlet / jsp 작동 전처리 / 후처리 가능
- servlet이나 jsp에 공통된 전후처리가 필요하다면 Filter 이용

  예) post방식의 한글처리 등
- javax.servlet.Filter 인터페이스를 구현





■ 필터 개념 예제

TestFilter.java

```
package filter;

//import 생략
public class TestFilter implements Filter {

	@Override
	public void destroy() {
		System.out.println("TestFilter destroy()");

	}

	@Override
	public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
			throws IOException, ServletException {

		System.out.println("TestFilter doFilter() start");

		chain.doFilter(request, response);

		System.out.println("TestFilter doFilter() end");

		PrintWriter out= response.getWriter();

		out.println("TestFilter가 추가한 글자들");

	}

	@Override
	public void init(FilterConfig arg0) throws ServletException {
		System.out.println("TestFilter init()");
	}

}
```

Test2Filter.java

```
package filter;

//import 생략
public class Test2Filter implements Filter {

	@Override
	public void destroy() {
		System.out.println("Test2Filter destroy()");
```

```
	}

	@Override
	public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
			throws IOException, ServletException {

		System.out.println("Test2Filter doFilter() start");

		chain.doFilter(request, response);

		System.out.println("Test2Filter doFilter() end");

		PrintWriter out= response.getWriter();

		out.println("Test2Filter가 추가한 글자들");

	}

	@Override
	public void init(FilterConfig arg0) throws ServletException {
		System.out.println("Test2Filter init()");
	}

}
```

index.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html >
<html>
<head>
<meta charset="UTF-8">
<title>index.jsp</title>
</head>
<body>
<%
	System.out.println("index.jsp");
%>
</body>
</html>
```

web.xml에 Filter를 설정

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://xmlns.jcp.org/xml/ns/javaee"
 x s i : s c h e m a L o c a t i o n = " h t t p : / / x m l n s . j c p . o r g / x m l / n s / j a v a e e
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
```

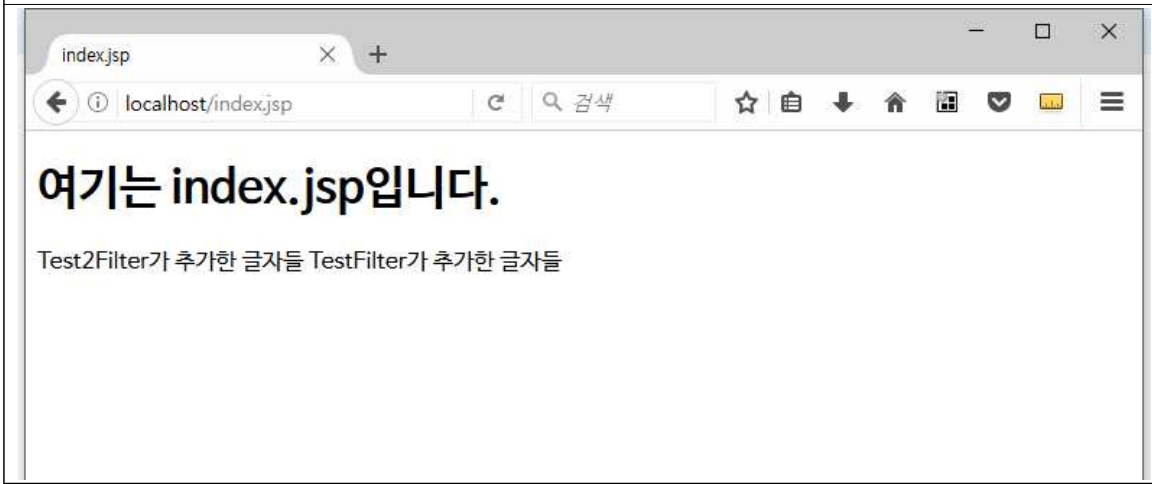
```
<filter>
	<filter-name>test</filter-name>
	<filter-class>filter.TestFilter</filter-class>
</filter>
<filter>
	<filter-name>test2</filter-name>
	<filter-class>filter.Test2Filter</filter-class>
</filter>
<filter-mapping>
	<filter-name>test</filter-name>
	<url-pattern>/*</url-pattern>
</filter-mapping>
<filter-mapping>
	<filter-name>test2</filter-name>
	<url-pattern>/*</url-pattern>
</filter-mapping>

</web-app>
```

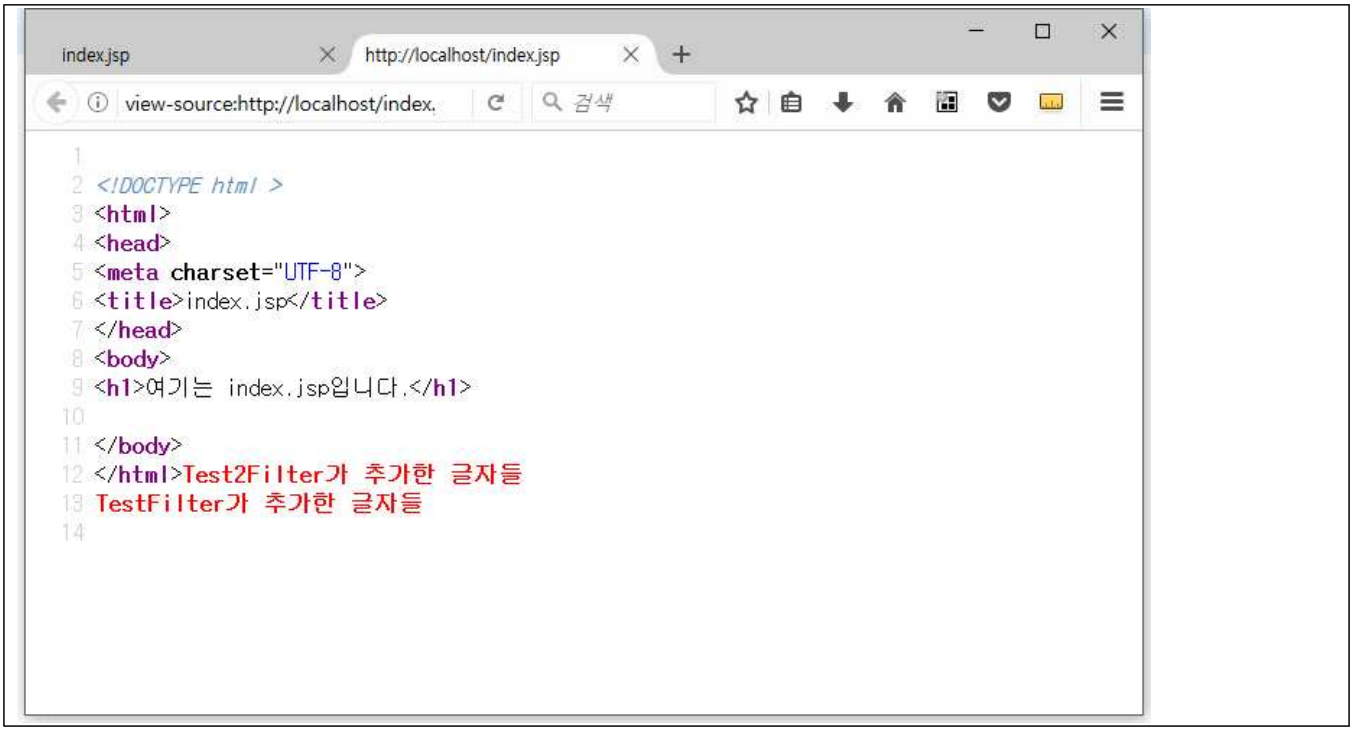
실행결과 - 콘솔

```
2월 14, 2017 11:41:06 오후 org.apache.catalina.core.
정보: Starting Servlet Engine: Apache Tomcat/8.5.11
Test2Filter init()
TestFilter init()
2월 14, 2017 11:41:06 오후 org.apache.catalina.start
```

```
TestFilter doFilter() start
Test2Filter doFilter() start
index.jsp
Test2Filter doFilter() end
TestFilter doFilter() end
```

실행결과 - 웹브라우저





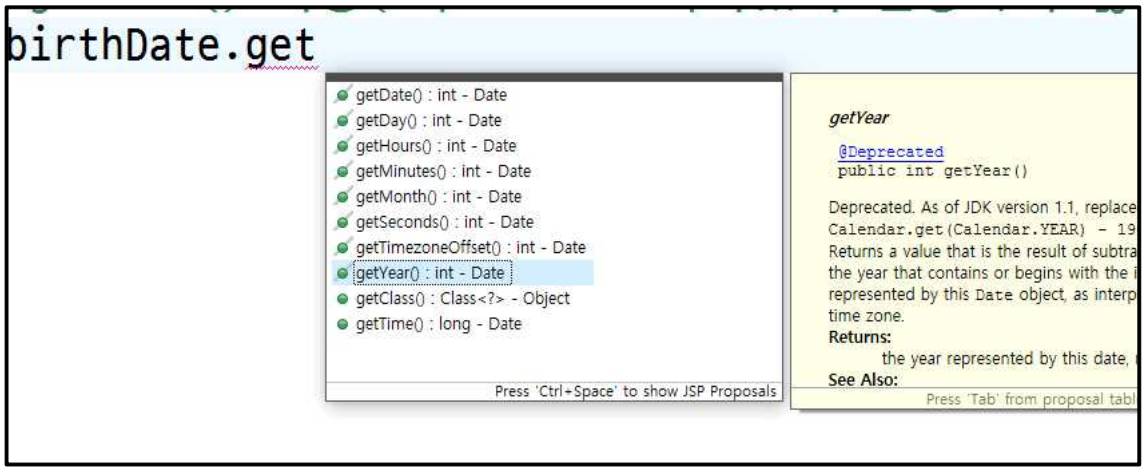
■ Filter를 이용한 post방식의 한글 처리

EncodingFilter

```
package filter;

@WebFilter("/*")
public class EncodingFilter implements Filter {

	@Override
	public void destroy() {
	}

	@Override
	public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
			throws IOException, ServletException {

		request.setCharacterEncoding("UTF-8");

		chain.doFilter(request, response);
	}

	@Override
	public void init(FilterConfig arg0) throws ServletException {
	}

}
```

■ 날짜를 년 / 월 / 일로 변경

1) Date의 getYear() 메서드등을 이용 (사용을 권장하지 않음)



2) Calendar클래스를 이용

3) SimpleDateFormat을 이용

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
Date birthDate = Date.valueOf("2001-01-06");
//1) Date의 getYear() 이용(deprecated 되어있어 권장하지 않음)
int year = birthDate.getYear();

out.println(year+1900);//밀레니엄 버그이전

//2) SimpleDateFormat이용
SimpleDateFormat sdf = new SimpleDateFormat("YYYY");
String yearStr = sdf.format(birthDate);
year = Integer.parseInt(yearStr);

out.println(year);

//3) Calendar 이용
Calendar cal = Calendar.getInstance();

cal.setTime(birthDate);

year = cal.get(Calendar.YEAR);

out.println(year);

%>
```

■ 세션을 이용한 로그인 처리

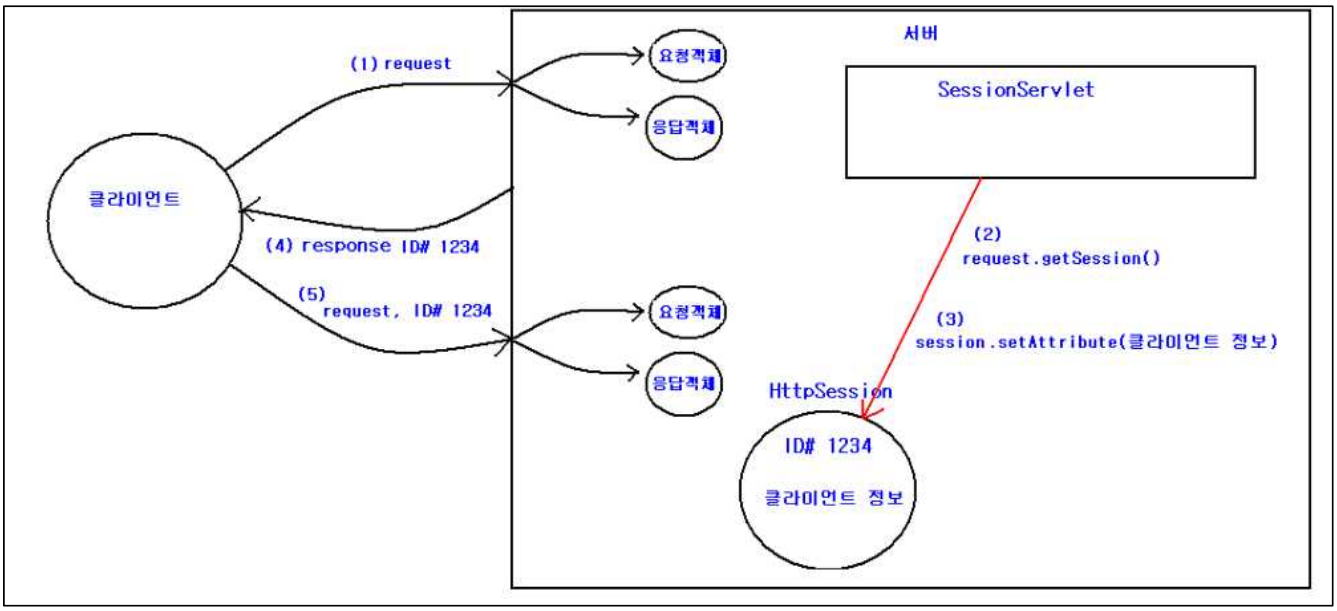
1) HttpSession 객체

- HTTP 프로토콜은 무상태(stateless) 연결 프로토콜

- 클라이언트가 서버와 연결을 맺고, 요청을 보낸 뒤, 서버가 요청을 처리한 후 응답을 보내면 클라이언트와 서버 사이의 연결은 끊어짐

- 서버에서 클라이언트의 정보를 유지하기 위하여 HttpSession 객체를 사용

- HttpSession의 동작 원리



(1) 클라이언트의 요청 접수

(2) 클라이언트의 정보를 저장할 HttpSession객체 생성, 고유한 id 부여됨

(3) HttpSession객체내에 클라이언트의 정보를 저장

(4) 고유한 id값을 응답에 넣어서 클라이언트로 보냄

(5) 클라이언트는 다음 번 요청부터 고유한 id값을 서버로 전송한다.

서버에서는 id에 해당하는 HttpSessio객체를 찾아서 요청을 처리한다.

2) HttpSession의 주요 메소드

반환형	메소드 명	설명
void	setAttribute(String name, Objec obj)	HttpSession객체에 지정된 이름으로 객체를 저장한다.
Object	getAttribute(String name)	HttpSession객체에서 지정된 이름으로 저장된 객체를 반환한다.
void	removeAttribute()	HttpSession객체에서 지정된 이름의 객체를 삭제한다.
void	setMaxInactiveInterval(int sec);	HttpSession객체의 타임아웃 시간을 지정한다.
void	invalidate()	HttpSession객체을 무효화시킨다.
String	getId()	세션 id를 반환한다.

3) 예제

index.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
        User loginUser =
        (User)session.getAttribute("loginUser");
        System.out.println(loginUser);
%>
<!DOCTYPE html>
<html lang="ko">
<head>
<meta charset="UTF-8">
<title>메인페이지</title>
</head>
<body>
        <h1>메인 페이지</h1>
        <% if(loginUser==null) { %>
        <form action="login.jsp" method="post">
                <fieldset>
                        <legend>로그인폼</legend>
                        <p>
                        <input type="text" id="id"
                        name="id"
                        placeholder="아이디" />
                        </p>
                        <p>
                        <input type="password" id="pwd" name="pwd" placeholder="비밀번호" />
                        </p>
```

```
                        <p>
                                <button>로그인</button>
                        </p>
                </fieldset>
        </form>
        <%}else { %>
        <h2><%=loginUser.getNickname() %>님 환영합니다.</h2>
        <a href="logout.jsp">로그아웃</a>
        <%} %>
</body>
</html>
```

login.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
request.setCharacterEncoding("UTF-8");

String id =
request.getParameter("id");
String password =
request.getParameter("pwd");
User user = new User();
user.setId(id);
user.setPassword(password);
User loginUser = UsersDAO.getDAO().selectLogin(user);

//세션에
session.setAttribute("loginUser", loginUser);
response.sendRedirect("index.jsp");

%>
id : <%= id%><br/>
password : <%=password%>
```

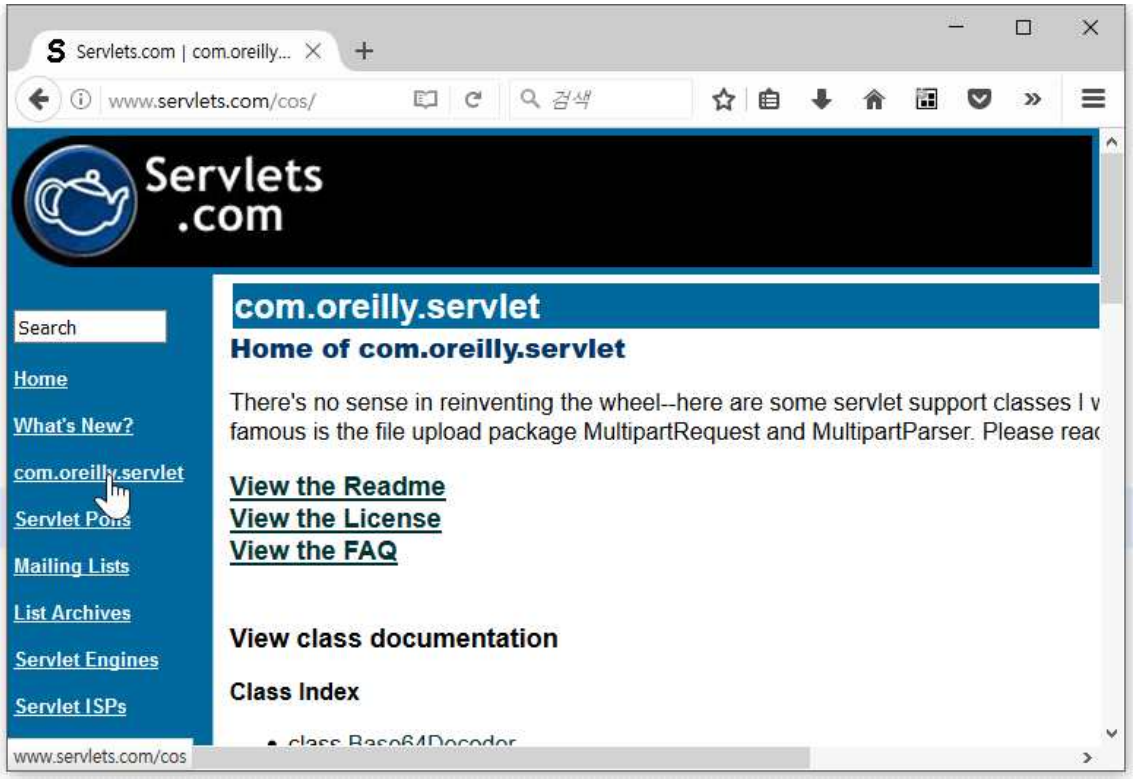
logout.jsp

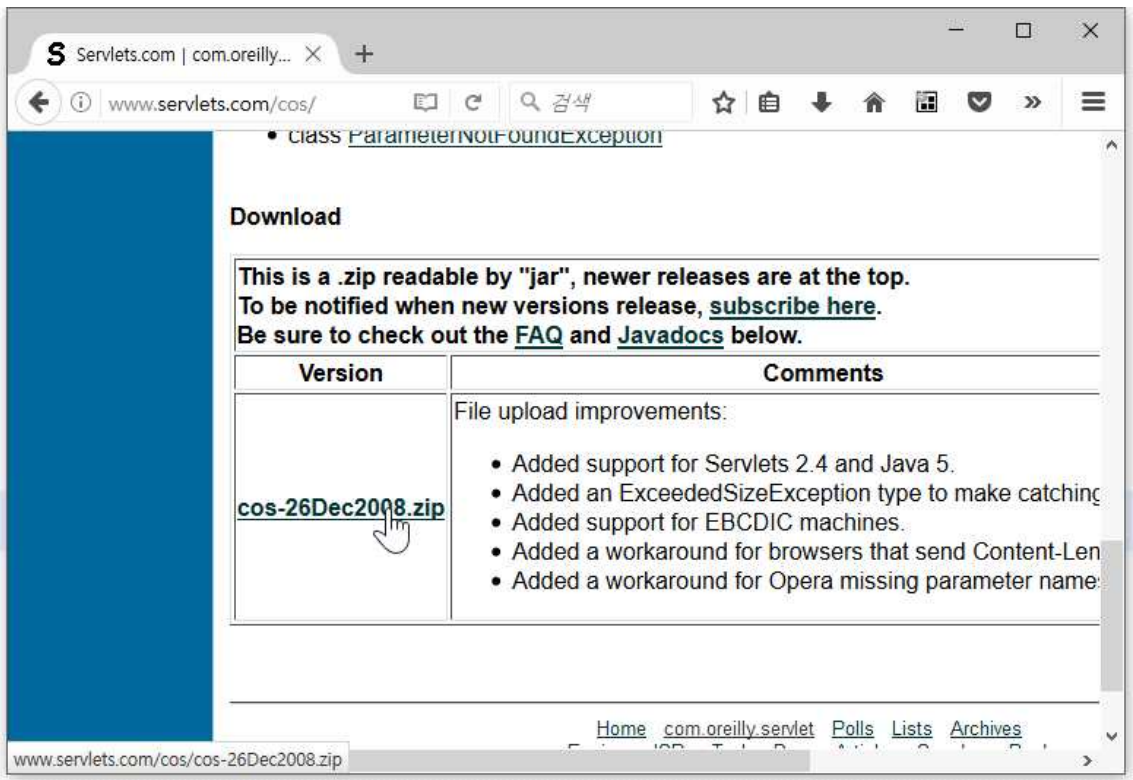
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%

//세션을 무효화
session.invalidate();

//session.removeAttribute("loginUser");

response.sendRedirect("index.jsp");
%>
```

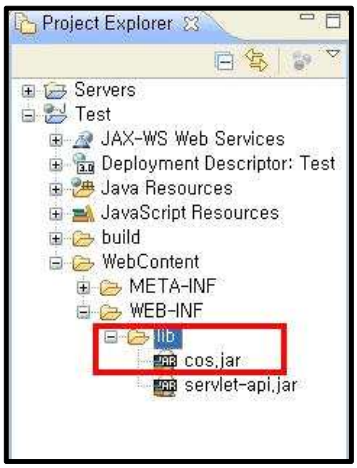
■ 파일 업로드 처리

1) cos.jar 이용

- http://www.servlets.com/cos/







■ 파일업로드를 위한 form의 enctype변경

enctype = *content-type* [CI]
This attribute specifies the content type used to submit the form to the server (when the value of method is "post"). The default value for this attribute is "application/x-www-form-urlencoded". The value "multipart/form-data" should be used in combination with the INPUT element, type="file".

타입	설명
application/x-www-form-urlencoded	기본(퍼센트 인코딩방식 사용)
multipart/form-data	파일업로드시 사용(이진데이터로 넘어감)

■ 예제

uploadForm.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>업로드폼</title>
</head>
<body>
        <h1>사진등록</h1>
        <%-- 파일을 업로드하기위해서는
                enctype="multipart/form-data'로
                method는 무조건 post
         --%>
        <form action="upload.jsp" method="post" enctype="multipart/form-data">
        <fieldset>
                <legend>사진등록폼</legend>
                <p>
```

```
                <input type="text" name="title" placeholder="사진이름"/>
                </p>
                <p>
                <input type="file" name="picture"/>
                </p>
                <p>
                <button>업로드</button>
                </p>
        </fieldset>
        </form>

</body>
</html>
```

upload.jsp

```
<%@page import="util.ResizeImageUtil"%>
<%@page import="com.oreilly.servlet.MultipartRequest"%>
<%@page import="com.oreilly.servlet.multipart.DefaultFileRenamePolicy"%>
<%@page import="com.oreilly.servlet.multipart.FileRenamePolicy"%>
<%@page import="java.io.File"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
        //multipart/form-data로 넘길때는 우리가 지금까지 했던 방식
        //request.getParameter() <-- 안됨

        //cos라이브러리 사용

        //1) 톰캣의 경로를 얻기
        String rootPath = request.getServletContext().getRealPath("/");

        //2) 실제 upload폴더 경로
        String uploadPath = rootPath+"upload"+File.separator;

        //3) 중복이름을 어떻게 처리할지 정책파일
        FileRenamePolicy renamePolicy = new DefaultFileRenamePolicy();

        //4) MultipartRequest 사용
//MultipartRequest(리퀘스트,폴더경로,업로드용량,인코딩방식,리네임폴리시);
        MultipartRequest mr =
        new MultipartRequest(request,
                                    uploadPath,
                                    1024*1024*100,
                                    "UTF-8",
                                    renamePolicy);

        //5) 넘어온 파라미터
        String title = mr.getParameter("title");
```

```
        //6) 파일이름
        String fileName = mr.getFilesystemName("picture");

        // 원본은 그대로 두고 작은 프로필 사진을 만듦

        //7) 원본소스 경로
        String original = uploadPath + fileName;

        //out.println(original);
        //8) 작은사진 경로
        String resize = rootPath+"profile"+File.separator+fileName;
        //존재하지 않음
        //out.println(resize);
        ResizeImageUtil.resize(original, resize, 200);

%>
<h1>작은사진</h1>
<img src="profile/<%=fileName %>" />
<h1>큰사진</h1>
<img src="upload/<%=fileName %>" />
<a href="uploadForm.jsp">업로드폼으로 이동</a>
```

■ ResizeImageUtil 사용법

1) 같은 사이즈 크기일 경우

```
resizeImage(원래경로,만들어질경로,크기)
```

2) 다른 사이즈 크기일 경우

```
resizeImage(원래경로,만들어질경로,너비,높이)
```

■ 예제

```
package util;



import java.awt.Image;
import java.awt.image.BufferedImage;
import java.awt.image.PixelGrabber;
import java.io.FileOutputStream;

import javax.swing.ImageIcon;
```

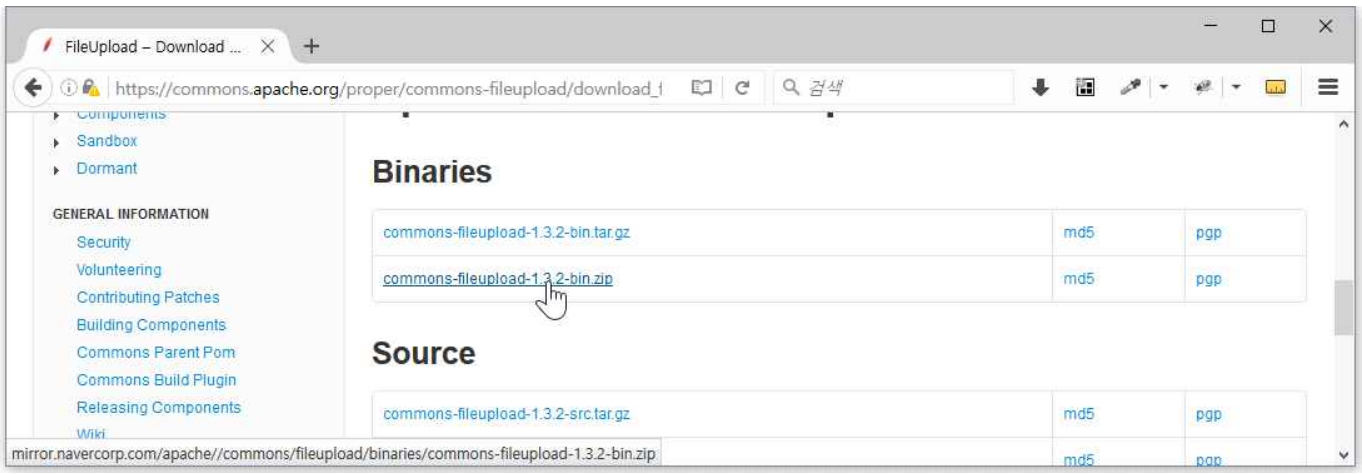
```
import com.sun.image.codec.jpeg.JPEGCodec;
import com.sun.image.codec.jpeg.JPEGEncodeParam;
import com.sun.image.codec.jpeg.JPEGImageEncoder;

public class ResizeImageUtil {

	//원본이미지를 크기에 맞게 다시 이미지를
	//만들어주는 메서드

	//1번째인자 : 원본소스경로
	//2번째인자 : 저장될경로
	//3번째인자 : width
	//4번째인자 : height
	public static boolean resize(String source, String target, int targetW, int targetH)  {

		Image imgSource = new ImageIcon(source).getImage();

	int oldW = imgSource.getWidth(null);
	int oldH = imgSource.getHeight(null);

	int sW = 0;
	int sH = 0;
	int newW = 0;
	int newH = 0;
	int cutW = 0;
	int cutH = 0;

	newW = targetW;
	newH = (targetW * oldH) / oldW;


	if(targetH>newH) {
		newW = (targetH * oldW) / oldH;
		newH = targetH;

		sW = (newW - targetW)/2;
	}else {
		sH = (newH - targetH)/2;
	}

	cutW = targetW;
	cutH = targetH;

       return process(source, target, sW,sH, newW, newH, cutW, cutH);

	}
```

```
	//메서드 오버로딩
	//인자 3개짜리
	public static boolean resize(String source,String target,int size) {

		return resize(source,target,size,size);
	}



	private static boolean process(String source, String target,int sW ,int sH,int newW, int
newH, int cutW, int cutH) {

	 Image imgSource = new ImageIcon(source).getImage();

		Image imgTarget = imgSource.getScaledInstance(newW, newH, Image.SCALE_SMOOTH);

	int pixels[] = new int[newW * newH];

	FileOutputStream fos = null;

	try {

	PixelGrabber pg = new PixelGrabber(imgTarget, sW, sH, cutW, cutH, pixels, 0, cutW);
	pg.grabPixels();

	for(int i = 0; i< newW * newH ; i++) {

		int onePixel = pixels[i];

		int alpha = (onePixel >> 24) & 0xff;

		if(alpha==0) {
			pixels[i] = -1;
		}
	}

	BufferedImage bi = new BufferedImage(cutW, cutH, BufferedImage.TYPE_INT_RGB);
	bi.setRGB(0, 0, cutW, cutH, pixels, 0, cutW);

	fos = new FileOutputStream(target);

	JPEGImageEncoder jpeg = JPEGCodec.createJPEGEncoder(fos);

	JPEGEncodeParam jep = jpeg.getDefaultJPEGEncodeParam(bi);
	jep.setQuality(1, false);
```

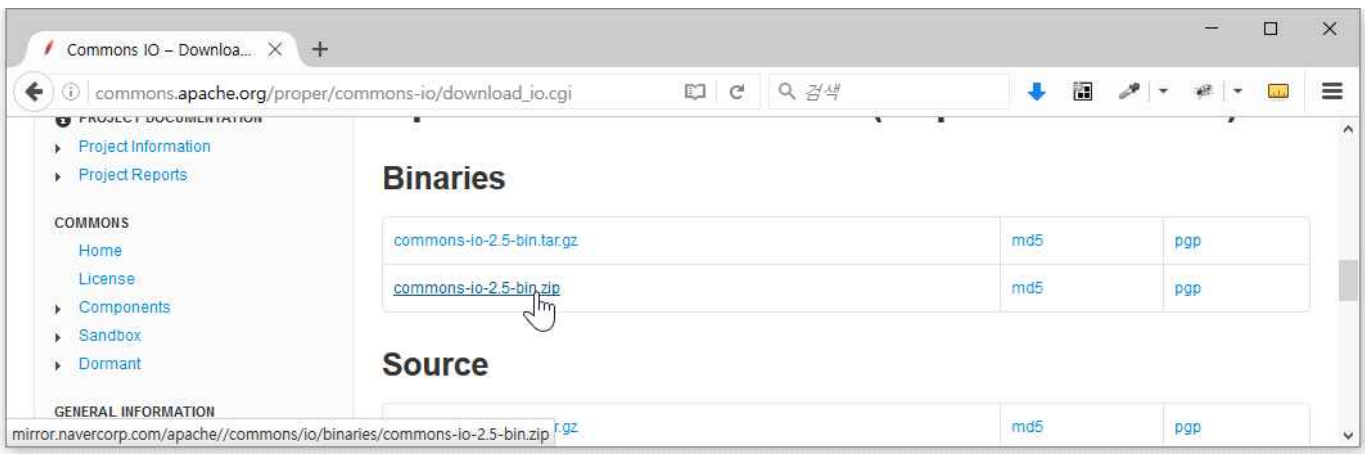
```
        jpeg.encode(bi, jep);
        return true;

        }catch(Exception e) {
                e.printStackTrace();

                return false;
        }finally {
                try {
                        fos.close();
                        } catch (Exception e2) {
                        }
        }

        }

}
```

2) common.fileupload 이용

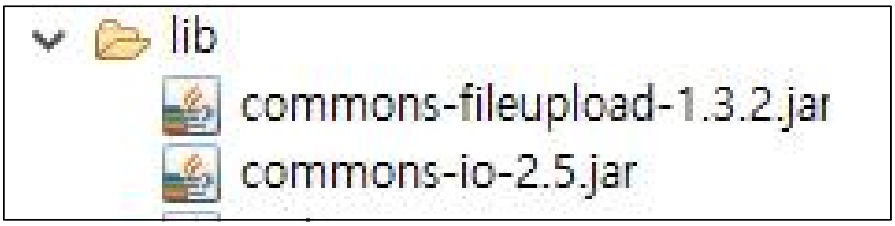
- https://commons.apache.org/proper/commons-fileupload/download_fileupload.cgi



- http://commons.apache.org/proper/commons-io/download_io.cgi



- common-io와 common-fileupload 둘 다 필요함(의존성)



2) 예제

uploadForm.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>파일 업로드예제용 폼</title>
</head>
<body>
<h1>파일업로드</h1>
<%--
        파일 업로드를 시키기 위해서는
        form의 enctype을
        'multipart/form-data'로 변경
        기본값 : application/x-www-form-urlencoded
        (글자로 넘기는 방식)

--%>
<form action="upload.jsp" enctype="multipart/form-data" method="post">
<fieldset>
        <legend>파일업로드폼</legend>
        <p>
        <label>아이디
         <input type="text" id="id" name="id"/></label>
        </p>
        <p>
        <label>닉네임
         <input type="text" id="nickname" name="nickname"/></label>
        </p>
        <p>
        <label>업로드파일
        <input type="file" name="upload" multiple="multiple" />
        </label>
        </p>
        <p>
        <button>업로드</button>
```

```
        </p>
</fieldset>
</form>
</body>
</html>
```

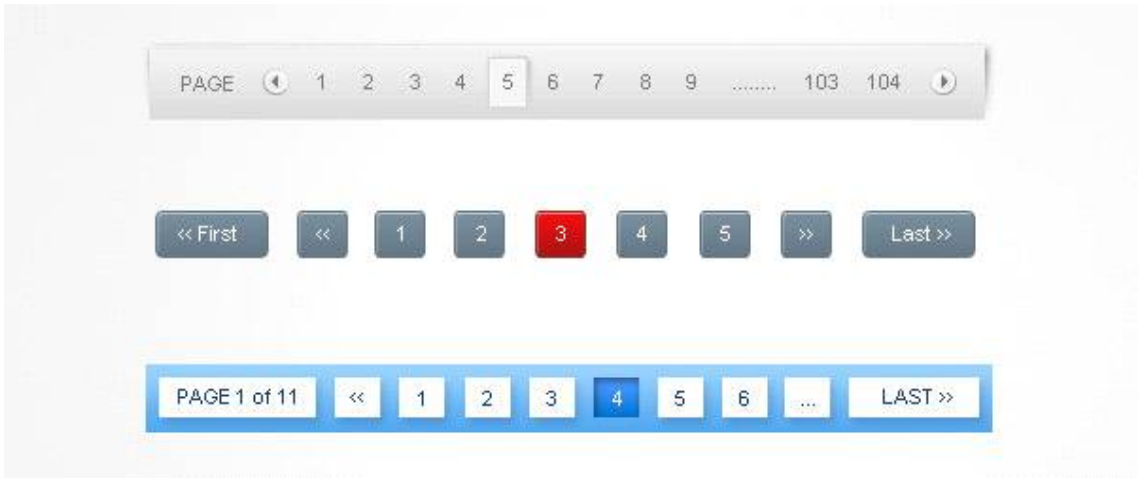
upload.jsp

```
<%@page import="java.io.IOException"%>
<%@page import="org.apache.commons.fileupload.FileItem"%>
<%@page import="org.apache.commons.fileupload.disk.DiskFileItemFactory"%>
<%@page import="org.apache.commons.fileupload.servlet.ServletFileUpload"%>
<%@page import="java.util.List"%>
<%@page import="java.io.File"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
// multipart로 전송되었는가를 체크
boolean isMultipart = ServletFileUpload.isMultipartContent(request);

File tmp = new File("e:/tmp");
if(!tmp.exists()) {
        tmp.mkdir();
}

if (isMultipart) {
        //톰켓의 전체 경로를 가져오고 upload라는 폴더를 만들고
         //tmp의 폴더의 전송된 파일을 upload 폴더로 카피 한다.
 String uploadDir = config.getServletContext().getRealPath("/upload/");

 DiskFileItemFactory factory = new DiskFileItemFactory();
 factory.setRepository(tmp);
 factory.setSizeThreshold(1 * 1024 * 1024);

 ServletFileUpload upload = new ServletFileUpload(factory);
 upload.setSizeMax(10 * 1024 * 1024);
 //최대 파일 크기(10M)

 //실제 업로드 부분(이부분에서 파일이 생성된다)
 List<FileItem> formItems = upload.parseRequest(request);

 for(FileItem fileItem : formItems) {
       //파일인지 일반 파라미터인지 확인
  if(fileItem.isFormField()){
   //넘어온 파라미터(이름:값)

   out.println("폼 파라미터이름: "+
fileItem.getFieldName()+" / 값"+fileItem.getString("utf-8")+"<br>");
```

```
  }else{
          //파일이면
          //size>0이면 업로드 성공
   if(fileItem.getSize()>0){

    String fieldName=fileItem.getFieldName();
    String fileName=fileItem.getName();
    String contentType=fileItem.getContentType();
    boolean isInMemory=fileItem.isInMemory();
    long sizeInBytes=fileItem.getSize();
    out.println("파라미터이름 : "+ fieldName +"<br/>");
    out.println("파일이름 : "+ fileName +"<br/>");
    out.println("파일컨텐츠종류(MIME) : "+ contentType +"<br/>");
    if(contentType.contains("image/")) {
        out.println("<img src='upload/"+fileName+"' alt='"+fileName+"'/><br/>");
    }
    out.println("파일크기 : "+ sizeInBytes +"<br/>");

    try{
    //실제 디렉토리에 fileName으로
     File uploadedFile=new File(uploadDir,fileName);
     fileItem.write(uploadedFile);
     fileItem.delete(); //tmp폴더의 임시파일을 제거
    }catch(IOException ex) {}
   }
  }
 }
}else{
 out.println("인코딩 타입이 multipart/form-data 가 아님.");
}
%>
```

□ 페이징 처리

■ 수많은 게시글이나 포스팅을 한 번에 모두 한 페이지에 출력할 수 없기 때문에 페이지를 나눠서 보여주는 형태



■ 페이징의 마크업

```
<div class="paginate">
<!-- 이전 페이지 이동 :  비활성 -->
<span title="이전 페이지 없음"><i class="fa fa-chevron-left"></i></span>
<!-- 이전 페이지 이동 :  활성 -->
<a href='' title="이전 페이지로"><i class="fa fa-chevron-left"></i><span class="screen_out">
이전 페이지</span></a>
<!-- 페이지들 -->
<a href="" title="1">1</a
<a href="" title="2">2</a>
<strong title="현재 3페이지">3</strong>
<a href="" title="4">4</a>
<a href="" title="5">5</a>
<!-- 다음 페이지 이동 :  활성 -->
<a href='' title="다음 페이지로"><i class="fa fa-chevron-right"></i><span class="screen_out">
다음 페이지</span></a>
<!-- 다음 페이지 이동 :  비활성 -->
<span title="다음 페이지 없음"><i class="fa fa-chevron-right"></i></span>
</div>
```

- 가운데 정렬을 위해 inline-block
- 띄어쓰기 방지를 위해 주석처리
- 현재 페이지나 이전/이후 페이지 클릭 안되게 a요소가 아님

■ 페이징의 구현

1) 게시글에서 특정 개수만 SELECT 해야함

2) 오라클에서는 rownum이라는 가상 칼럼을 이용해서 구현

(primary key인 idx나 no 등은 삭제가 되었을 수 있기 때문에 안됨)

우선,

```
SELECT no,title,contents,writer,writer_no writerNo,hit,likes,regdate
FROM board
ORDER BY regdate DESC
```

그리고 rownum 구현

```
SELECT no,title,contents,writer,writerNo,hit,likes,regdate,rownum r
FROM (SELECT no,title,contents,writer,writer_no writerNo,hit,likes,regdate
FROM board
ORDER BY regdate DESC
```

그리고 where 조건 입력

```
SELECT no,title,contents,writer,writerNo,hit,likes,regdate
FROM (SELECT no,title,contents,writer,writerNo,hit,likes,regdate,rownum r
FROM (SELECT no,title,contents,writer,writer_no writerNo,hit,likes,regdate
FROM board
ORDER BY regdate DESC))
WHERE r BETWEEN 1 AND 5
```

■ 페이징 페이지에서의 처리

1) 현재 페이지 (nowPage)

2) 한 페이지당 보여질 게시물 수 (numPage)

3) 한 블록당 보여질 페이지 수 (numBlock)

4) 현재 페이지 주소

5) 시작과 끝의 설정

6) 전체 게시물의 개수

■ 페이징 유틸 만들기

```
public class PaginateUtil {

public static String getPaginate(int pageNo,
                                 int total,
                                 int numPage,
                                 int numBlock,
                                 String url,
                                 String param) {
//현재 페이지 : pageNo
//전체 게시물수 : total
//한 페이지당 게시물수 : numPage
//한 페이지당 보여질 블록수 : numBlock
//주소 : url
//파라미터 : param

//전체 페이지수
int totalPage = (int)Math.ceil((double)total/numPage);

//System.out.println(totalPage);

//현재 블록
int nowBlock = (int)Math.ceil((double)pageNo/numBlock);


String paginate =
                "<div class='paginate'>";

if(total!=0) {

        //이전버튼
        if(pageNo<=1) {
                //비활성화
                paginate     +=     "<span     title=\"이전     페이지     없음\"><i     class=\"fa
fa-chevron-left\"></i></span>";
        }else {
                //활성화
                paginate   +=   "<a   href='"+url+"?"+param+(pageNo-1)+"'   title='이전   페이지로'><i
class='fa fa-chevron-left'></i><span class='screen_out'>이전 페이지</span></a>";

        }//if end

        for(int i = 1 ; i <= numBlock ; i++) {

                //실제 출력 페이지
```

```
                int realPage = ((nowBlock-1)*numBlock)+i;

                //현재 페이지냐? 아니냐?
                if(realPage==pageNo) {
                        //현재 페이지
                        paginate += "<strong title='현재 "+pageNo+"페이지'>"+pageNo+"</strong>";

                }else {
                        //현재 페이지가 아님
                        paginate         +=         "<a         href='"+url+"?"+param+realPage+"'
title='"+realPage+"'>"+realPage+"</a>";


                }//if ~ else end

                if(realPage==totalPage) {
                        break;
                }//if end

        }//for end(블록 만들기)


        //다음버튼
        if(pageNo >= totalPage) {
                //비활성화
                paginate+="<span        title='다음        페이지        없음'><i        class='fa
fa-chevron-right'></i></span>";
        }else {
                //활성화
                paginate+="<a    href='"+url+"?"+param+(pageNo+1)+"'    title='다음    페이지로'><i
class='fa fa-chevron-right'></i><span class='screen_out'>다음 페이지</span></a>";
        }//if end

}//if end

paginate+= "</div>";

return paginate;
}

}
```