

# **JSP File Upload**

## **JSP File Upload**

# File Upload

## File Upload

클라이언트에서 서버로  
파일을 전송하는 기법으로  
파일의 전송은 ftp를 사용하지만

http를 이용해야 할 경우  
라이브러리를 이용하여  
jsp로 구현한다.

**파일업로드를 구현하는 방식은 다양하다.**

# cos.jar

Com. Oreilly. Servlet

가장 대중적으로 사용되고 있는 라이브러리

파일을 업로드 하고  
다운로드 하는 기능을  
도화주고 있다.

**라이브러리를 통하여 보다 편하게 구현 가능**

# cos.jar 다운받기

http://www.servlets.com/cos

**Servlets.com**  
com.oreilly.servlet  
Home of com.oreilly.servlet

There's no sense in reinventing the wheel--here are some servlet support classes I wrote that you can use. Most famous is the file upload package MultipartRequest and MultipartParser. Please read the license before use.

[View the README](#)  
[View the License](#)  
[View the FAQ](#)

**View class documentation**

**Class Index**

- class Base64Decoder
- class Base64Encoder
- class CacheHttpServlet
- class DaemonHttpServlet
- class RemoteDaemonHttpServlet (implements java.rmi.Remote)
- class RemoteHttpServlet (implements java.rmi.Remote)
- class HttpMessage
- class HttpsMessage
- class LocaleNegotiator
- class LocaleToCharsetMap
- class MailMessage
- class MultipartRequest
- class MultipartParser
- class FilePart
- class ParamPart
- class FileRenamePolicy
- class DefaultFileRenamePolicy
- class MultipartFilter
- class MultipartWrapper
- class MultipartResponse
- class ParameterParser
- class ServletUtils
- class VersionDetector

**Exception Index**

- class ParameterNotFoundException

**Download**

This is a .zip readable by "jar", newer releases are at the top.  
To be notified when new versions release, subscribe here.  
Be sure to check out the FAQ and Javadocs below.

Version	Comments
cos-20.08.zip	File upload improvements: <ul style="list-style-type: none"><li>Added support for Servlets 2.4 and Java 5.</li><li>Added an ExceededSizeException type to make catching easier.</li><li>Added support for EBCDIC machines.</li><li>Added a workaround for browsers that send Content-Length of -1.</li><li>Added a workaround for Opera missing parameter names.</li></ul>

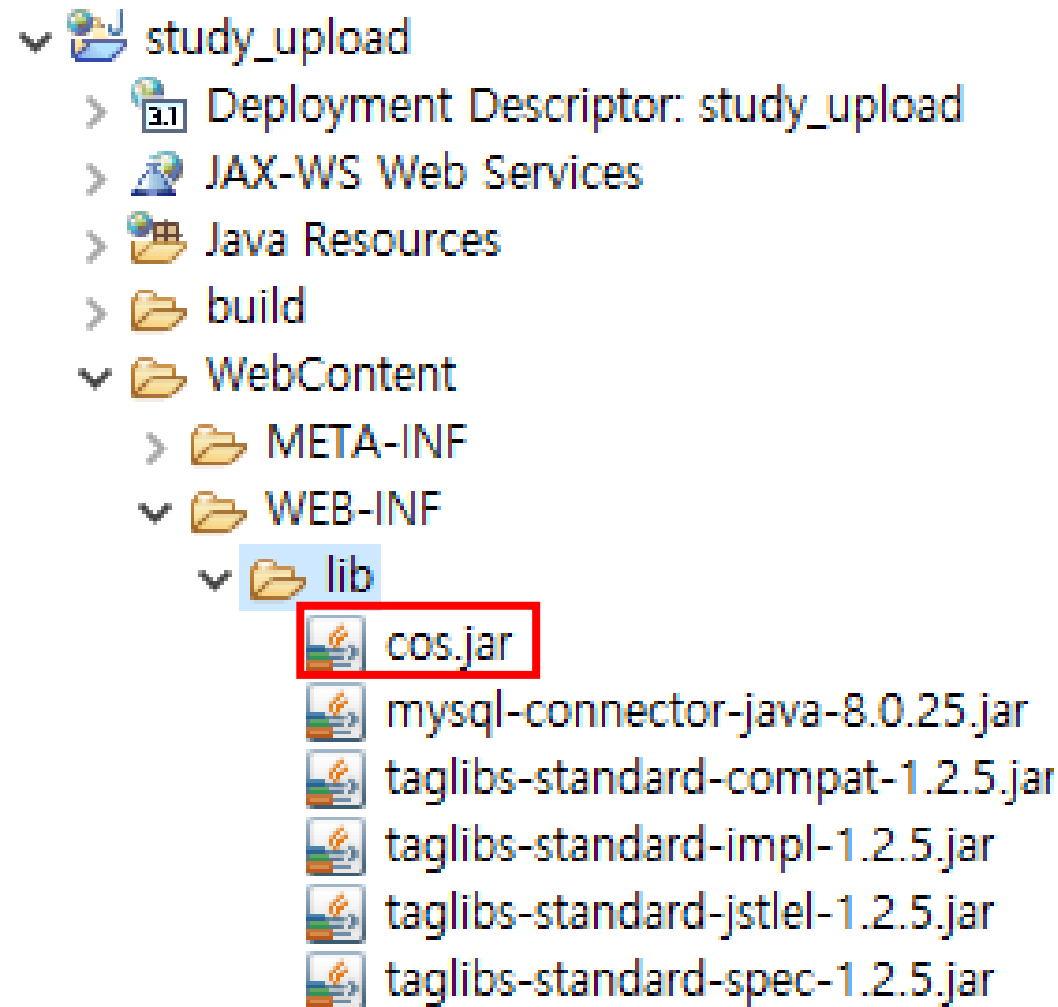
이름	수정된 날짜	유형	크기
doc	2008-12-27 오전 8:07	파일 폴더	
lib	2021-08-08 오후 7:42	파일 폴더	
src	2021-08-08 오후 7:42	파일 폴더	
license.txt	2008-12-27 오전 8:16	텍스트 문서	4KB
readme.txt	2008-12-27 오전 8:14	텍스트 문서	2KB

이름	수정된 날짜	유형	크기
cos.jar	2021-08-08 오후 7:42	Executable Jar File	56KB

## 압축을 풀어 파일을 확인, cos.jar

# 프로젝트 제작 및 라이브러리 설정

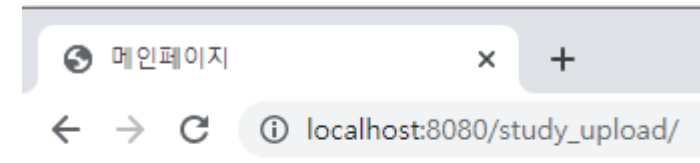


자주 사용하는 라이브러리는 따로 보관하면 편리하다

# form 제작

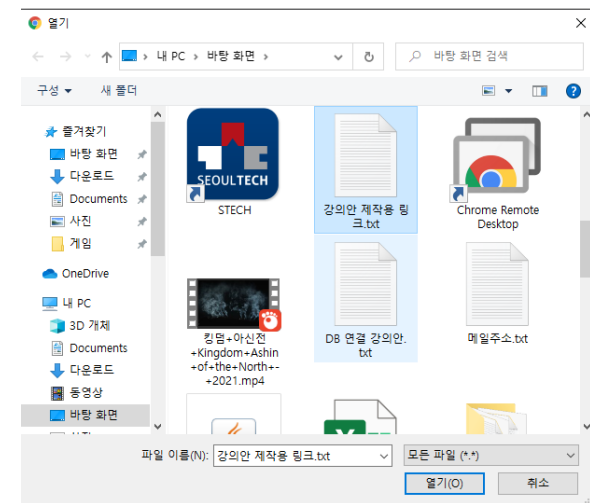
index.jsp 제작.

```
index.jsp
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>메인페이지</title>
8   </head>
9   <body>
10    <p>파일입력용 form</p>
11    <form name="form" method="post" enctype="multipart/form-data">
12      <input type="file" name="file_name">
13    </form>
14  </body>
15 </html>
```



파일입력용 form

파일 선택 선택된 파일 없음

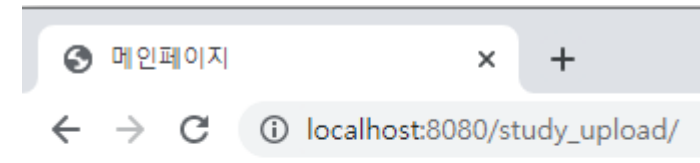


파일 선택 버튼을 눌러 업로드 창을 켤 수 있음

# form 제작

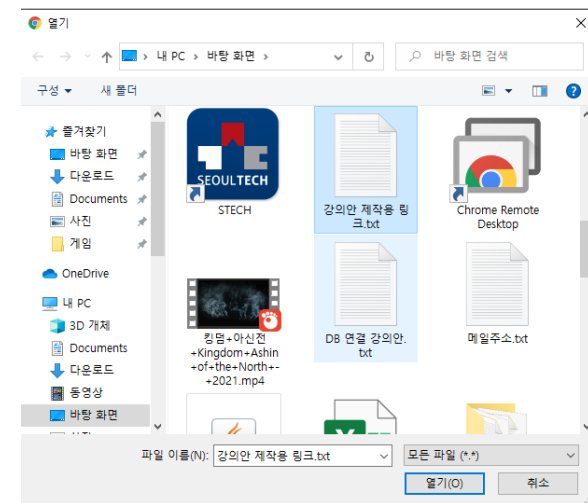
index.jsp 제작.

```
index.jsp
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>메인페이지</title>
8   </head>
9   <body>
10    <p>파일입력용 form</p>
11    <form name="form" method="post" enctype="multipart/form-data">
12      <input type="file" name="file_name">
13    </form>
14  </body>
15 </html>
```



파일입력용 form

파일 선택 선택된 파일 없음



파일 선택 버튼을 눌러 업로드 창을 켤 수 있음

# 데이터베이스 제작

```
mysql> desc product;
```

Field	Type	Null	Key	Default	Extra
code	int	NO	PRI	NULL	auto_increment
name	text	YES		NULL	
price	int	YES		NULL	
pictureurl	text	YES		NULL	
description	text	YES		NULL	

```
5 rows in set (0.01 sec)
```

code : primary key

name : 상품명

price : 가격

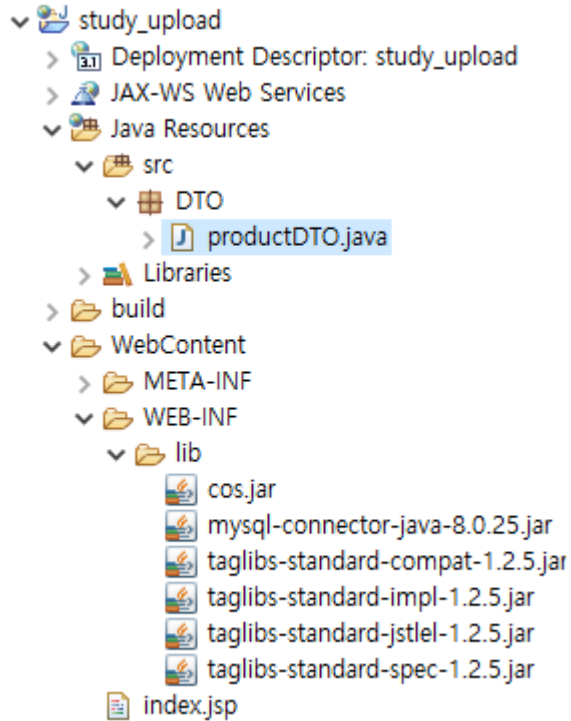
pictureurl : 사진 경로

description : 설명

**파일이 저장된 경로를 저장하여 html에 반영한다.**



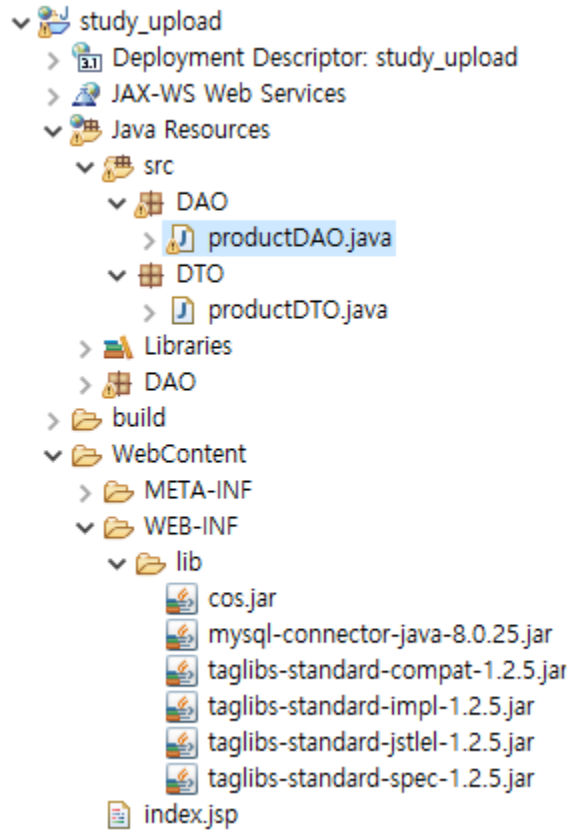
# 데이터 오브젝트 설정



```
productDTO.java
1 package DTO;
2
3 public class productDTO {
4     private int code;
5     private String name;
6     private int price;
7     private String pictureurl;
8     private String description;
9     public int getCode() {
10         return code;
11     }
12     public void setCode(int code) {
13         this.code = code;
14     }
15     public String getName() {
16         return name;
17     }
18     public void setName(String name) {
19         this.name = name;
20     }
21     public int getPrice() {
22         return price;
23     }
24     public void setPrice(int price) {
25         this.price = price;
26     }
27     public String getPictureurl() {
28         return pictureurl;
29     }
30     public void setPictureurl(String pictureurl) {
31         this.pictureurl = pictureurl;
32     }
33     public String getDescription() {
34         return description;
35     }
36     public void setDescription(String description) {
37         this.description = description;
38     }
39 }
```

파일이 저장된 경로를 저장하여 html에 반영한다.

# DAO 제작



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

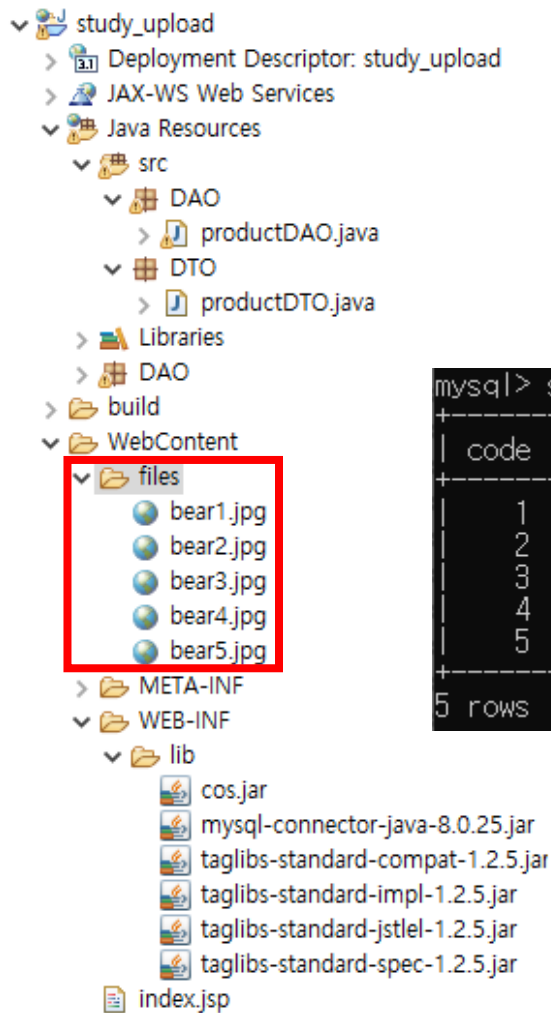
파일이 저장된 경로를 저장하여 html에 반영한다.

# DAO close메서드

```
26 //리소스 해제
27 public static void close(Connection conn, Statement stmt, ResultSet rs) {
28     try {
29         rs.close();
30         stmt.close();
31         conn.close();
32     } catch (Exception e) {
33         System.out.println("연결 해제중 오류발생 : "+e);
34     }
35 }
36 public static void close(Connection conn, Statement stmt) {
37     try {
38         stmt.close();
39         conn.close();
40     } catch (Exception e) {
41         System.out.println("연결 해제중 오류발생 : "+e);
42     }
43 }
```

객체를 통하지 않기 위해 static으로 제작

# 데이터베이스 입력



```
mysql> select * from product;
```

code	name	price	pictureurl	description
1	반달곰	10000	/files/bear1.jpg	반달가슴곰입니다. 아시아볼곰이기도 합니다.
2	회색곰	10000	/files/bear2.jpg	회색곰입니다. 볼곰이라고 불립니다.
3	북극곰	10000	/files/bear3.jpg	북극곰입니다. 폴라를 좋아합니다.
4	팬더	10000	/files/bear4.jpg	팬더입니다. 대나무를 좋아합니다.
5	안경곰	10000	/files/bear5.jpg	안경곰입니다. 체구가 작고 귀엽습니다.

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

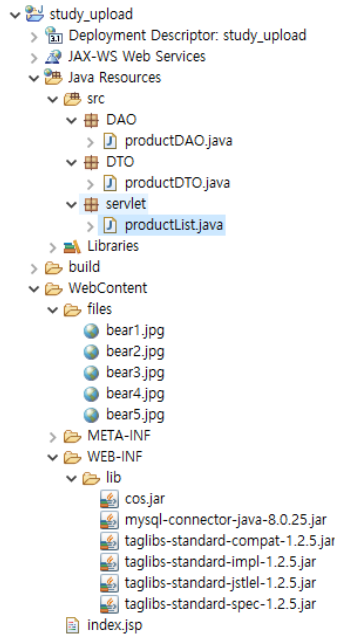
**테스트를 위해 데이터베이스를 입력해 둔다.  
화면으로 출력할 파일들도 미리 지정해 둔다.**

# DAO select 제작

```
45 //DTO를 import해야함을 잊지 말자
46 public List<productDTO> selectAllProducts(){
47     List<productDTO> list=new ArrayList<productDTO>();
48     String sql="select * from product order by code desc";
49     Connection conn=null;
50     PreparedStatement pstmt=null;
51     ResultSet rs=null;
52     try {
53         conn=getConnection();
54         pstmt=conn.prepareStatement(sql);
55         rs=pstmt.executeQuery();
56         while(rs.next()) {
57             productDTO p=new productDTO();
58             p.setCode(rs.getInt("code"));
59             p.setName(rs.getString("name"));
60             p.setPrice(rs.getInt("price"));
61             p.setPictureurl(rs.getString("pictureurl"));
62             p.setDescription(rs.getString("description"));
63             list.add(p);
64         }
65     }catch(Exception e) {
66         System.out.println("접속 중 오류발생 : "+e);
67     }finally{
68         productDAO.close(conn, pstmt, rs);
69     }
70     return list;
71 }
```

객체를 통하지 않기 위해 static으로 제작

# 메인 구성 및 서블릿 제작



```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>메인페이지</title>
8   </head>
9   <body>
10    <a href="productList">제품목록 보기</a>
11  </body>
12 </html>
```

```
1 package servlet;
2
3 import java.io.IOException;
4 import javax.servlet.*;
5 import javax.servlet.annotation.*;
6 import javax.servlet.http.*;
7
8 import DAO.productDAO;
9 import DTO.productDTO;
10 import java.util.*;
11
12 @WebServlet("/productList")
13 public class productList extends HttpServlet {
14     private static final long serialVersionUID = 1L;
15
16     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
17         productDAO dao=productDAO.getInstance();
18         List<productDTO> productList=dao.selectAllProducts();
19         request.setAttribute("productlist", productList);
20
21         RequestDispatcher dispatcher=request.getRequestDispatcher("/productList.jsp");
22         dispatcher.forward(request, response);
23     }
24 }
```

서블릿의 호출경로와 jsp의 명칭에 주의한다.

# 리스트 페이지 제작

```
productList.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>상품목록 페이지</title>
10  </head>
11  <body>
12    <div id="wrap">
13      <h1>상품 리스트</h1>
14      <table>
15        <tr>
16          <td colspan="5">상품등록</td>
17        </tr>
18        <tr>
19          <th>번호</th>
20          <th>이름</th>
21          <th>가격</th>
22          <th>수정</th>
23          <th>삭제</th>
24          <th>이미지</th>
25        </tr>
26        <c:forEach items="${productlist}" var="product">
27          <tr>
28            <td>${product.getCode()}</td>
29            <td>${product.getName()}</td>
30            <td>${product.getPrice()}원</td>
31            <td>수정</td>
32            <td>삭제</td>
33            <td></td>
34          </tr>
35        </c:forEach>
36      </table>
37    </div>
38  </body>
39 </html>
```

## 상품 리스트

상품등록

번호 이름 가격 수정 삭제

이미지

5 안경곰 10000원 수정 삭제



4 팬더곰 10000원 수정 삭제



3 북극곰 10000원 수정 삭제



© doopedia.co.kr

# 지정 경로 내의 이미지가 표현되는 것을 확인

# CSS 처리

```
1 @charset "utf-8";
2
3 #wrap{
4     width:900px;
5     margin:0 auto;
6 }
7 h1{
8     color:green;
9 }
10 table{
11     width:100%;
12     border-collapse:collapse;
13     font-size:15px;
14     line-height:25px;
15 }
16 table td, th{
17     border:1px solid black;
18 }
19 th{
20     background-color:yellowgreen;
21 }
22 img{
23     width:200px;
24     height:300px;
25 }
26 a{
27     text-decoration:none;
28     color:black;
29 }
30 a:hover{
31     text-decoration:underline;
32     color:green;
33 }
```

```
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>상품목록 페이지</title>
10    <link rel="stylesheet" href="style.css">
11  </head>
12  <body>
13    <div id="wrap">
14      <h1>상품 리스트</h1>
15      <table>
16        <tr>
17          <td colspan="5" style="text-align:right;">상품등록</td>
18        </tr>
19        <tr>
20          <th>번호</th>
21          <th>이름</th>
22          <th>가격</th>
23          <th>수정</th>
24          <th>삭제</th>
25        </tr>
26        <c:forEach items="${productlist}" var="product">
27          <tr>
28            <td>${product.getCode()} </td>
29            <td>${product.getName()} </td>
30            <td>${product.getPrice()} 원</td>
31            <td>수정</td>
32            <td>삭제</td>
33          </tr>
34        </c:forEach>
35      </table>
36    </div>
37  </body>
38 </html>
```

상품 리스트

상품등록				
번호	이름	가격	수정	삭제
5	안경공	10000원	수정	삭제
4	팬더공	10000원	수정	삭제
3	북극공	10000원	수정	삭제
2	회색공	10000원	수정	삭제
1	반달공	10000원	수정	삭제

## 이미지를 제거하여 리스트만 출력



# 상품등록

```
<td colspan="5" style="text-align:right;">
    <a href="productWrite">상품등록</a>
</td>
```

```
1 package servlet;
2
3 import java.io.IOException;
4
5 import javax.servlet.*;
6 import javax.servlet.annotation.*;
7 import javax.servlet.http.*;
8
9 @WebServlet("/productWrite")
10 public class productWrite extends HttpServlet {
11     private static final long serialVersionUID = 1L;
12
13     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
14         RequestDispatcher dispatcher=request.getRequestDispatcher("/productWrite.jsp");
15         dispatcher.forward(request, response);
16     }
17     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
18     }
19 }
20 }
```

- study\_upload
  - Deployment Descriptor: study\_upload
  - JAX-WS Web Services
  - Java Resources
    - src
      - DAO
        - productDAO.java
      - DTO
        - productDTO.java
      - servlet
        - productList.java
        - productWrite.java
    - Libraries

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2     pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5     <head>
6         <meta charset="utf-8">
7         <title>상품등록</title>
8         <link rel="stylesheet" href="style.css">
9     </head>
10    <body>
11        <div id="wrap">
12            <h1>상품 등록 - 관리자</h1>
13            <form method="post" enctype="multipart/form-data" name="frm">
14                <table>
15                    <tr>
16                        <th>상품명</th>
17                        <td><input type="text" name="name"></td>
18                    </tr>
19                    <tr>
20                        <th>가격</th>
21                        <td><input type="text" name="price">원</td>
22                    </tr>
23                    <tr>
24                        <th>사진</th>
25                        <td><input type="file" name="pictureurl"></td>
26                    </tr>
27                    <tr>
28                        <th>설명</th>
29                        <td><textarea cols="80" rows="10" name="description"></textarea></td>
30                    </tr>
31                </table>
32                <br>
33                <input type="submit" value="등록" onclick="return check()">
34                <input type="reset" value="다시작성">
35                <input type="button" value="목록보기" onclick="location.href='productList'">
36            </form>
37        </div>
38    </body>
39 </html>
```

## 상품 리스트

상품등록				
번호	이름	가격	수정	삭제
5	안경곰	10000원	수정	삭제
4	팬더곰	10000원	수정	삭제
3	북극곰	10000원	수정	삭제
2	회색곰	10000원	수정	삭제
1	반달곰	10000원	수정	삭제

# 서블릿과 해당 페이지 제작

# 상품등록 DAO

//상품 등록

```
public void insertProduct(productDTO p) {  
    String sql="insert into product (name, price, pictureurl, description) values (?, ?, ?, ?)";  
    Connection conn=null;  
    PreparedStatement pstmt=null;  
    try {  
        conn=getConnection();  
        pstmt=conn.prepareStatement(sql);  
        pstmt.setString(1, p.getName());  
        pstmt.setInt(2, p.getPrice());  
        pstmt.setString(3, p.getPictureurl());  
        pstmt.setString(4, p.getDescription());  
        pstmt.executeUpdate();  
    }catch(Exception e) {  
        System.out.println("데이터 입력 중 오류 발생 : "+e);  
    }finally {  
        productDAO.close(conn, pstmt);  
    }  
}
```

---

서블릿과 해당 페이지 제작

# productWrite 서블릿 제작

```
import com.oreilly.servlet.*;
import com.oreilly.servlet.multipart.*;
import DTO.productDTO;
import DAO.productDAO;

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    request.setCharacterEncoding("utf-8");
    ServletContext context=getServletContext();
    String path=context.getRealPath("/files");
    System.out.println(path); //파일이 저장되고 있는 경로확인
    String encType="utf-8";
    int sizeLimit=20*1024*1024; //20mb 제한

    //request이긴 한데 파일을 함께 받아들 수 있는 request타입
    MultipartRequest multi=new MultipartRequest(request, path, sizeLimit, encType, new DefaultFileRenamePolicy());
    String name=multi.getParameter("name");
    int price=Integer.parseInt(multi.getParameter("price"));
    String description=multi.getParameter("description");
    String pictureurl=multi.getFilesystemName("pictureurl");

    productDTO DTO=new productDTO();
    DTO.setName(name);
    DTO.setPrice(price);
    DTO.setDescription(description);
    DTO.setPictureurl("/files/"+pictureurl);

    productDAO DAO=productDAO.getInstance();
    DAO.insertProduct(DTO);

    response.sendRedirect("productList");
}
```

## doPost 부분에 제작 import에 주의

# productWrite.jsp 유효성

```
<script>
function check(){
    if(document.frm.name.value.length==0){
        alert("상품명을 써주세요.");
        frm.name.focus();
        return false;
    }
    if(document.frm.price.value.length==0){
        alert("가격을 써주세요");
        frm.price.focus();
        return false;
    }
    if(isNaN(document.frm.price.value)){
        alert("숫자를 입력해야 합니다.");
        frm.price.focus();
        return false;
    }
    return true;
}
</script>
```

localhost:8080 내용:  
상품명을 써주세요.

확인

localhost:8080 내용:  
가격을 써주세요

확인

localhost:8080 내용:  
숫자를 입력해야 합니다.

확인

## 잘못된 내용이 입력되지 않도록 주의

# 제품 수정 구현

```
<c:forEach items="${productlist }" var="product">
  <tr>
    <td>${product.getCode() }</td>
    <td>${product.getName() }</td>
    <td>${product.getPrice() }원</td>
    <td><a href="productUpdate?code=${product.getCode() }">상품 수정</a></td>
    <td>삭제</td>
  </tr>
</c:forEach>
```

a 65.28 × 20	삭제
상품 수정	삭제

```
▼ <tr>
  <td>1</td>
  <td>반달곰</td>
  <td>10000원</td>
  ...
  ▼ <td> == $0
    <a href="productUpdate?code=1">상품 수정
    </a>
```

productList.jsp의 상품 수정부분 구현

# 제품 조회 구현

//상품 조회 기능

```
public productDTO selectProductByCode(String Code) {
    String sql="select * from product where code=?";
    productDTO DTO=null;
    Connection conn=null;
    PreparedStatement pstmt=null;
    ResultSet rs=null;
    try {
        conn=getConnection();
        pstmt=conn.prepareStatement(sql);
        pstmt.setString(1, Code);
        rs=pstmt.executeQuery();
        if(rs.next()) {
            DTO=new productDTO();
            DTO.setCode(rs.getInt("code"));
            DTO.setName(rs.getString("name"));
            DTO.setPrice(rs.getInt("price"));
            DTO.setPictureurl(rs.getString("pictureurl"));
            DTO.setDescription(rs.getString("description"));
        }
    }catch(Exception e) {
        System.out.println("조회 중 오류 발생");
    }finally {
        productDAO.close(conn, pstmt, rs);
    }
    return DTO;
}
```

## productDAO 내부에 구현

# 제품 조회/수정 서블릿 (productUpdate.class)

```
1 package servlet;
2
3 import java.io.*;
4 import javax.servlet.*;
5 import javax.servlet.annotation.*;
6 import javax.servlet.http.*;
7
8 import com.oreilly.servlet.*;
9 import com.oreilly.servlet.multipart.*;
10 import DAO.productDAO;
11 import DTO.productDTO;
12
13 @WebServlet("/productUpdate")
14 public class productUpdate extends HttpServlet {
15     private static final long serialVersionUID = 1L;
16
17     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
18         String code=request.getParameter("code");
19         productDAO DAO=productDAO.getInstance();
20         productDTO DTO=DAO.selectProductByCode(code);
21
22         request.setAttribute("product", DTO);
23         RequestDispatcher dispatcher=request.getRequestDispatcher("productUpdate.jsp");
24         dispatcher.forward(request, response);
25     }
26
27     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
28
29     }
30 }
```

**doGet과 doPost를 구별하여 제작한다.**

# 제품 조회 화면

```
productUpdate.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>상품 관리</title>
10    <link rel="stylesheet" href="style.css">
11  </head>
12  <body>
13    <div id="wrap">
14      <h1>상품 수정</h1>
15      <form method="post" enctype="multipart/form-data" name="frm" action="productUpdate">
16        <input type="hidden" name="code" value="${product.getCode()}">
17        <input type="hidden" name="nonmakeImg" value="${product.getPictureurl()}">
18        <table>
19          <tr>
20            <td>
21              <input type="reset" value="다시작성">
22              <input type="button" value="목록" onclick="location.href='productList'">
23            </td>
24          </tr>
25        </table>
26      </form>
27    </div>
28  </body>
29  <script>
30  </script>
31 </html>
```


```
<table>
  <tr>
    <td>
      <c:choose>
        <c:when test="${product.getPictureurl()=='/files/null'}">
          이미지가 없습니다.
        </c:when>
        <c:otherwise>
          
        </c:otherwise>
      </c:choose>
    </td>
    <td>
      <table>
        <tr>
          <th>상품명</th>
          <td><input type="text" name="name" value="${product.getName()}"></td>
        </tr>
        <tr>
          <th>가격</th>
          <td><input type="text" name="price" value="${product.getPrice()}"></td>
        </tr>
        <tr>
          <th>사진</th>
          <td>
            <input type="file" name="pictureurl"><br>
            (이미지를 변경하시려면 파일을 선택해 주세요)
          </td>
        </tr>
        <tr>
          <th>설명</th>
          <td><textarea cols="90" rows="10" name="description">${product.getDescription()}</textarea></td>
        </tr>
      </table>
    </td>
  </tr>
</table>
```

패러미터의 name간의 연결에 주의한다.



# 제품 조회 테스트

## 상품 수정

	상품명	<input type="text" value="안경곰"/>
	가격	<input type="text" value="10000"/>
	사진	<input type="button" value="파일 선택"/> 선택된 파일 없음 (이미지를 변경하시려면 파일을 선택해 주세요)
	설명	<div>안경곰 입니다. 체구가 작고 귀엽습니다.</div>

## 상품 수정

이미지가 없습니다.	상품명	<input type="text" value="테스트곰"/>
	가격	<input type="text" value="9999"/>
	사진	<input type="button" value="파일 선택"/> 선택된 파일 없음 (이미지를 변경하시려면 파일을 선택해 주세요)
	설명	<div>할인판매 안합니다</div>

```
<script>
    function check(){
        if(document.frm.name.value.length==0){
            alert("상품명을 써주세요.");
            frm.name.focus();
            return false;
        }
        if(document.frm.price.value.length==0){
            alert("가격을 써주세요");
            frm.price.focus();
            return false;
        }
        if(isNaN(document.frm.price.value)){
            alert("숫자를 입력해야 합니다.");
            frm.price.focus();
            return false;
        }
        return true;
    }
</script>
```

유효성 체크를 구현해 준다.

# DAO 수정기능 구현

//상품 수정 기능

```
public void updateProduct(productDTO DTO) {  
    String sql="update product set name=?, price=?, pictureurl=?, description=? where code=?";  
    Connection conn=null;  
    PreparedStatement pstmt=null;  
    try {  
        conn=getConnection();  
        pstmt=conn.prepareStatement(sql);  
        pstmt.setString(1, DTO.getName());  
        pstmt.setInt(2, DTO.getPrice());  
        pstmt.setString(3, DTO.getPictureurl());  
        pstmt.setString(4, DTO.getDescription());  
        pstmt.setInt(5, DTO.getCode());  
        pstmt.executeUpdate();  
    }catch(Exception e) {  
        System.out.println("수정 중 오류 발생 : "+e);  
    }finally {  
        productDAO.close(conn, pstmt);  
    }  
}
```

이후 서블릿에서 doPost()를 구현한다.

# productUpdate 서블릿의 doPost 수정

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    request.setCharacterEncoding("utf-8");
    ServletContext context=getServletContext();
    String path=context.getRealPath("files");
    System.out.println("업로드 경로 확인 : "+path);
    String encType="utf-8";
    int sizeLimit=20*1024*1024;
    MultipartRequest multi=new MultipartRequest(request, path, sizeLimit, encType, new DefaultFileRenamePolicy());

    String code=multi.getParameter("code");
    String name=multi.getParameter("name");
    int price=Integer.parseInt(multi.getParameter("price"));
    String description=multi.getParameter("description");
    String pictureurl=multi.getFilesystemName("pictureurl");
    if(pictureurl==null) {
        pictureurl=multi.getParameter("nomakeImg");
    }

    productDTO DTO=new productDTO();
    DTO.setCode(Integer.parseInt(code));
    DTO.setName(name);
    DTO.setPrice(price);
    DTO.setDescription(description);
    DTO.setPictureurl("/files/"+pictureurl);
    productDAO DAO=productDAO.getInstance();
    DAO.updateProduct(DTO);

    response.sendRedirect("productList");
}
```

실재 경로를 확인해서 업로드를 체크할 수 있다.

# 테스트 확인

## 상품 리스트

상품등록				
번호	이름	가격	수정	삭제
6	수정 이전 데이터	12345원	상품 수정	삭제
5	안경곰	10000원	상품 수정	삭제
4	팬더곰	10000원	상품 수정	삭제
3	북극곰	10000원	상품 수정	삭제
2	회색곰	10000원	상품 수정	삭제
1	반달곰	10000원	상품 수정	삭제

## 상품 수정

이미지가 없습니다.

상품명

수정 이전 데이터

가격

12345

사진

파일 선택

선택된 파일 없음

(이미지를 변경하시려면 파일을 선택해 주세요)

설명

수정되지 않은 데이터 입니다.

수정


다시작성

목록

## 상품 리스트

상품등록				
번호	이름	가격	수정	삭제
6	수정된 데이터입니다.	12345원	상품 수정	삭제
5	안경곰	10000원	상품 수정	삭제
4	팬더곰	10000원	상품 수정	삭제
3	북극곰	10000원	상품 수정	삭제
2	회색곰	10000원	상품 수정	삭제
1	반달곰	10000원	상품 수정	삭제

## 상품 수정



상품명

수정된 데이터입니다.

가격

12345

사진

파일 선택

선택된 파일 없음

(이미지를 변경하시려면 파일을 선택해 주세요)

설명

수정된 데이터입니다.

수정

다시작성

목록

수정이 잘 되고 있는지 확인

# 데이터 삭제

```
<c:forEach items="${productlist}" var="product">
  <tr>
    <td>${product.getCode()} </td>
    <td>${product.getName()} </td>
    <td>${product.getPrice()} 원</td>
    <td><a href="productUpdate?code=${product.getCode()}">상품 수정</a></td>
    <td><a href="productDelete?code=${product.getCode()}">상품 삭제</a></td>
  </tr>
</c:forEach>
```

삭제
<a href="#">상품 삭제</a>
<a href="#">상품 삭제</a>
<a href="#">상품 삭제</a>
<a href="#">상품 삭제</a>
<a href="#">상품 삭제</a>
<a href="#">상품 삭제</a>

```
<h1>상품 리스트</h1>
<table>
  <tbody>
    <tr>...</tr>
    <tr>...</tr>
    <tr>
      <td>6</td>
      <td>수정된 데이터입니다.</td>
      <td>12345원</td>
      <td>...</td>
      <td>
        <a href="productDelete?code=6">상품 삭제</a>
      </td>
    </tr>
  </tbody>
</table>
```

## productList.jsp 내부의 내용 변경

# 데이터 삭제 서블릿 제작

\*productDelete.java

```
1 package servlet;
2
3 import java.io.*;
4 import javax.servlet.*;
5 import javax.servlet.annotation.*;
6 import javax.servlet.http.*;
7
8 import DTO.productDTO;
9 import DAO.productDAO;
10
11 @WebServlet("/productDelete")
12 public class productDelete extends HttpServlet {
13     private static final long serialVersionUID = 1L;
14
15     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
16         String code=request.getParameter("code");
17         productDAO DAO=productDAO.getInstance();
18         productDTO DTO=DAO.selectProductByCode(code);
19
20         request.setAttribute("product", DTO);
21         RequestDispatcher dispatcher=request.getRequestDispatcher("productDelete.jsp");
22         dispatcher.forward(request, response);
23     }
24
25     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
26
27     }
28 }
```

정보를 확인한 후 삭제할 수 있도록 구현

# 데이터 삭제 화면 제작

```
<body>
  <div id="wrap">
    <h1>상품 삭제</h1>
    <form method="post" name="frm" action="productDelete">
      <input type="hidden" name="code" value="${product.getCode()} ">
      <input type="hidden" name="nonmakeImg" value="${product.getPictureurl()} ">
      <table>
        <tr>
          <td>
            <c:choose>
              <c:when test="${product.getPictureurl()=='/files/null' }">
                이미지가 없습니다.
              </c:when>
              <c:otherwise>
                
              </c:otherwise>
            </c:choose>
          </td>
          <td>
            <table>
              <tr>
                <th>상품명</th>
                <td>${product.getName()} </td>
              </tr>
              <tr>
                <th>가격</th>
                <td>${product.getPrice()} </td>
              </tr>
              <tr>
                <th>설명</th>
                <td><div style="height:220px; width:100%">${product.getDescription()} </div></td>
              </tr>
            </table>
          </td>
        </tr>
      </table>
      <br>
      <input type="submit" value="삭제">
      <input type="button" value="목록" onclick="location.href='productList'">
    </form>
  </div>
</body>
```

## productDelete.jsp 수정과 유사하게 작성

# DAO에 삭제기능 구현

//상품 삭제 기능

```
public void deleteProduct(String code) {  
    String sql="delete from product where code=?";  
    Connection conn=null;  
    PreparedStatement pstmt=null;  
    try {  
        conn=getConnection();  
        pstmt=conn.prepareStatement(sql);  
        pstmt.setString(1, code);  
        pstmt.executeUpdate();  
    }catch(Exception e) {  
        System.out.println("삭제 중 오류 발생 : "+e);  
    }finally {  
        productDAO.close(conn, pstmt);  
    }  
}
```

**productDelete.jsp 수정과 유사하게 작성**



# 데이터 삭제 서블릿 doPost

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    String code=request.getParameter("code");
    productDAO DAO=productDAO.getInstance();
    DAO.deleteProduct(code);

    response.sendRedirect("productList");
}
```


## 상품 리스트

상품등록				
번호	이름	가격	수정	삭제
6	수정된 데이터입니다.	12345원	상품 수정	상품 삭제
5	안경곰	10000원	상품 수정	상품 삭제
4	팬더곰	10000원	상품 수정	상품 삭제
3	북극곰	10000원	상품 수정	상품 삭제
2	회색곰	10000원	상품 수정	상품 삭제
1	반달곰	10000원	상품 수정	상품 삭제

## 상품 리스트

상품등록				
번호	이름	가격	수정	삭제
5	안경곰	10000원	상품 수정	상품 삭제
4	팬더곰	10000원	상품 수정	상품 삭제
3	북극곰	10000원	상품 수정	상품 삭제
2	회색곰	10000원	상품 수정	상품 삭제
1	반달곰	10000원	상품 수정	상품 삭제

## 상품 삭제



상품명	수정된 데이터입니다.
가격	12345
설명	수정된 데이터입니다.

[삭제](#) [목록](#)

```
mysql> select * from product;
+----+-----+-----+-----+-----+
| code | name  | price | pictureurl | description |
+----+-----+-----+-----+-----+
| 1    | 반달곰 | 10000 | /files/bear1.jpg | 반달가슴곰입니다. 아시아볼곰이기도 합니다. |
| 2    | 회색곰 | 10000 | /files/bear2.jpg | 회색곰입니다. 볼곰이라고 불립니다. |
| 3    | 북극곰 | 10000 | /files/bear3.jpg | 북극곰입니다. 골라를 좋아합니다. |
| 4    | 팬더곰 | 10000 | /files/bear4.jpg | 팬더곰입니다. 대나무를 좋아합니다. |
| 5    | 안경곰 | 10000 | /files/bear5.jpg | 안경곰입니다. 체구가 작고 귀엽습니다. |
+----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```