

JSP

1. 파일 업로드(2가지 방법)
2. 이미지 게시판

## 파일 업로드

- 파일 업로드는 클라이언트에서 웹 서버로 파일을 전하는 것을 말한다

1. 서블릿 버전 3.0부터 기본제공되는 Part API를 사용한다.
2. 서블릿 버전 3.0미만은 MultiPart API를 사용한다.

1.Part API를 사용하도록 하겠습니다.



## 첨부파일과 일반 form 데이터의 차이점

브라우저에서 <form>을 통해 전송하는 일반적인 데이터는 단순한 문자열이므로 Servlet상에서 request.getParameter()메서드를 통해 구할 수 있지만 파일의 경우는 조금 특별한 방법을 통해서만 얻을 수 있습니다.

첨부파일은 단순한 문자열이 아니고 0과 1로 이루어진 이진수 형태의 데이터이기에 브라우저에서 전송시에 **multipart/form-data**라는 형식을 이용해 전송해야 합니다.

## Part API 주요 메서드

메서드	설명
<code>getInputStream()</code>	Part에 대한 InputStream을 리턴합니다. 직접 데이터를 추출할때 사용합니다.
<code>getContentType()</code>	Content-Type을 리턴합니다.
<code>getName()</code>	파라미터명을 리턴합니다.
<code>getSubmittedFileName()</code>	업로드한 파일명을 리턴합니다. servlet 3.1부터 사용 가능합니다.
<code>getSize();</code>	파일의 크기를 byte단위로 리턴합니다.
<code>write(String fileName)</code>	임시저장되어 있는 파일 데이터를 복사하여 fileName에 지정한 경로로 저장합니다. 임시저장 되어있는 파일데이터가 메모리상에 있든 디스크에 있든 신경쓰지 않아도 됩니다.
<code>delete()</code>	임시저장된 파일 데이터를 제거합니다. HTTP요청이 처리되고 나면 자동으로 제거되지만 그 전에 메모리나 디스크 자원을 아끼고 싶다면 수동으로 제거할 수 있습니다.
<code>getHeader(String name)</code>	Part로부터 지정한 name헤더값을 리턴합니다.

## Part API 설정 속성의 의미

### <multipart-config>

multipart/form-data 로 전송된 데이터에 대해 Part API로 처리할 수 있도록 하는 설정입니다.

### <location>

업로드한 파일이 임시적으로 저장될 경로를 지정합니다.

### <max-file-size>

업로드 가능한 최대 size를 byte단위로 지정합니다. -1인 경우 제한을 두지 않습니다.

request.getParts()호출시 파일 크기가 이 값을 넘는 경우 IllegalStateException가 발생합니다.

### <max-request-size>

전체적인 multipart 데이터 최대 size를 byte단위로 지정합니다. -1인 경우 제한을 두지 않습니다.

### <file-size-threshold>

임시파일로 저장 여부를 결정할 데이터 크기를 byte로 지정합니다. 이 값을 넘지 않으면 업로드된 데이터를 메모리상에 가지고 있지만 이 값을 넘는 경우 <location>으로 지정한 경로에 임시파일로 저장합니다. 메모리상에 저장된 파일 데이터는 언젠가 제거되겠지만 크기가 큰 파일을 메모리상에 적재하게 되면 서버에 부하를 줄 수 있으므로 적당한 크기를 지정해 곧바로 임시파일로 저장하는 것이 좋습니다.

## 실습

### 1. 파일 업로드 폼 만들기

```
<form action="../../UploadServlet" method="post" enctype="multipart/form-data">
    글쓴이:<input type="text" name="name"><br>
    제목:<input type="text" name="title"><br>
    파일선택:<input type="file" name="fileName"><br>
    <input type="submit" value="전송"><br>
</form>
```

### 2. 서블릿이나 web.xml에 Part설정하기

#### 방법1

```
@MultipartConfig(
    location = "D:\\test\\upload",
    maxFileSize = -1,
    maxRequestSize = -1,
    fileSizeThreshold = 1024)
@WebServlet("/UploadServlet")
public class UploadServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    public UploadServlet() {
        super();
    }
}
```

#### 방법2

```
<servlet>
    <servlet-name>apple</servlet-name>
    <servlet-class>실제경로</servlet-class>
    <multipart-config>
        <location>D:\test\upload</location>
        <max-file-size>-1</max-file-size>
        <max-request-size>-1</max-request-size>
        <file-size-threshold>1024</file-size-threshold>
    </multipart-config>
</servlet>

<servlet-mapping>
    <servlet-name>apple</servlet-name>
    <url-pattern>맵핑패턴</url-pattern>
</servlet-mapping>
```

## 실습

### 3. 파일 업로드 코드작성하기

```
try {
    Collection<Part> parts = request.getParts();
    String realFileName = null;

    for(Part part : parts) {

        if(part.getHeader("Content-Disposition").contains("filename=") ) { //전달된 코드가 파일업로드 형식이라면

            realFileName = part.getSubmittedFileName(); //업로드된 파일명을 받는다

            if(part.getSize() > 0) {
                part.write( " D:\\test\\upload\\" + realFileName) //헤딩 경로에 업로드시킨다
                part.delete();
            }

        }

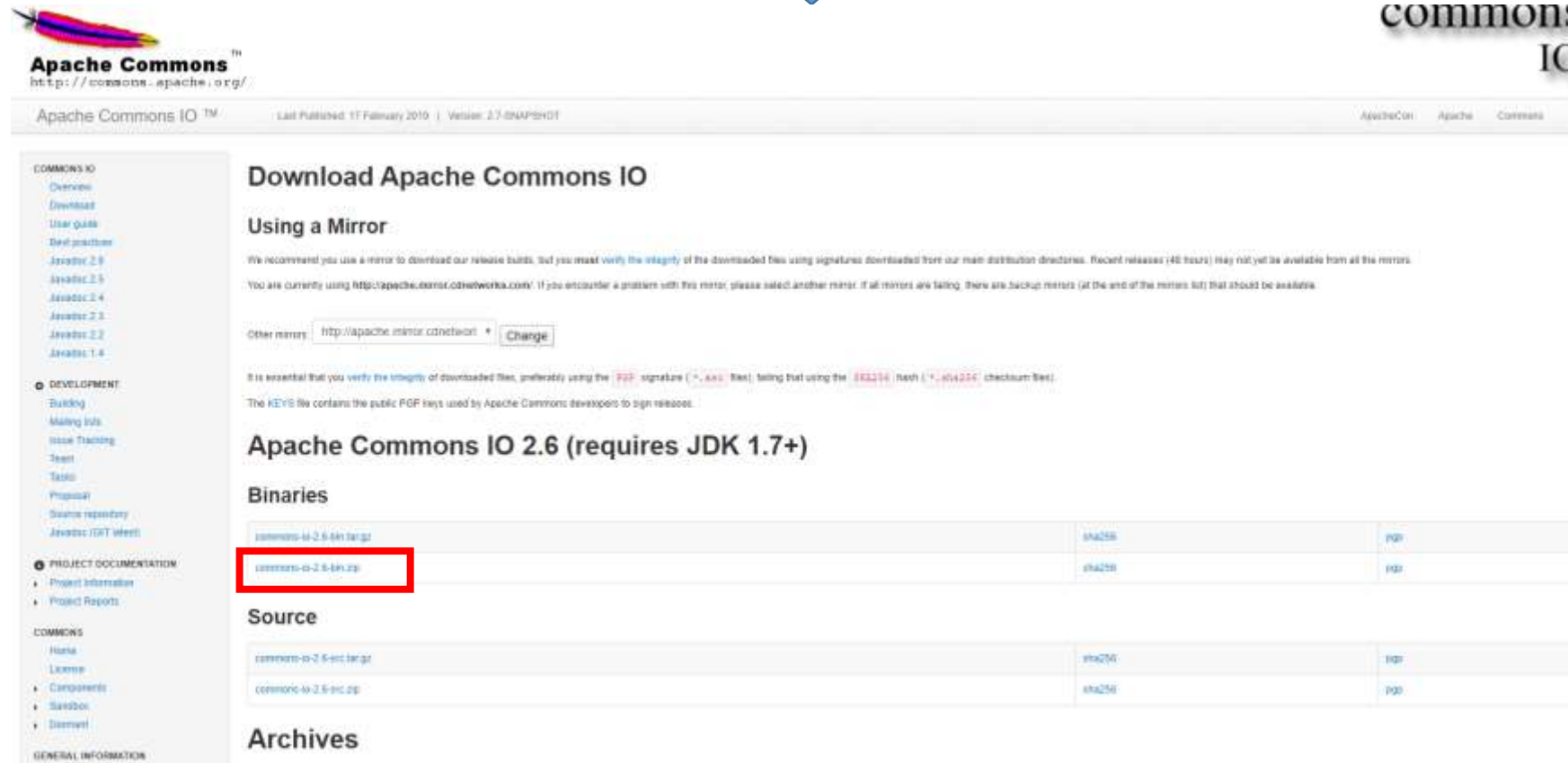
    }
    //끝!!!!
} catch (Exception e) {
    e.printStackTrace();
}
```

## 서블릿 에서 파일데이터 전송처리를 하기위한 API



**commons-io-2.6.jar**

[http://commons.apache.org/proper/commons-io/download\\_io.cgi](http://commons.apache.org/proper/commons-io/download_io.cgi)



The screenshot shows the Apache Commons IO download page. The page title is "Download Apache Commons IO". It includes a sidebar with navigation links for Commons IO, Development, Project Documentation, Commons, and General Information. The main content area is titled "Using a Mirror" and provides instructions on how to download the software. It includes a table of mirrors and a table of binaries. The binary table has a red box around the "commons-io-2.6-bin.zip" entry. Below the binaries table is a "Source" section with a table of source files. At the bottom is an "Archives" section.

**Download Apache Commons IO**

**Using a Mirror**

We recommend you use a mirror to download our release builds, but you must [verify the integrity](#) of the downloaded files using signatures downloaded from our main distribution directories. Recent releases (48 hours) may not yet be available from all the mirrors.

You are currently using <http://apache.mirror.com/networks.com/>. If you encounter a problem with this mirror, please select another mirror. If all mirrors are failing, there are backup mirrors (at the end of the mirrors list) that should be available.

Other mirrors:  [Change](#)

It is essential that you [verify the integrity](#) of downloaded files, preferably using the [PGP](#) signature ( [\\*.asc](#) files), testing that using the [SHA256](#) (hash) ( [\\*.sha256](#) checksum files).

The [KEYS](#) file contains the public PGP keys used by Apache Commons developers to sign releases.

**Apache Commons IO 2.6 (requires JDK 1.7+)**

**Binaries**

File	SHA256	PGP
<a href="#">commons-io-2.6-bin.tar.gz</a>	<a href="#">sha256</a>	<a href="#">pgp</a>
<a href="#">commons-io-2.6-bin.zip</a>	<a href="#">sha256</a>	<a href="#">pgp</a>

**Source**

File	SHA256	PGP
<a href="#">commons-io-2.6-src.tar.gz</a>	<a href="#">sha256</a>	<a href="#">pgp</a>
<a href="#">commons-io-2.6-src.zip</a>	<a href="#">sha256</a>	<a href="#">pgp</a>

**Archives**

## 서블릿에서 JSP로의 파일데이터 전송

```
response.setContentType("image");  
byte[] image = IOUtils.toByteArray(new FileInputStream(파일경로));  
response.getOutputStream().write(image);
```



## 파일 업로드

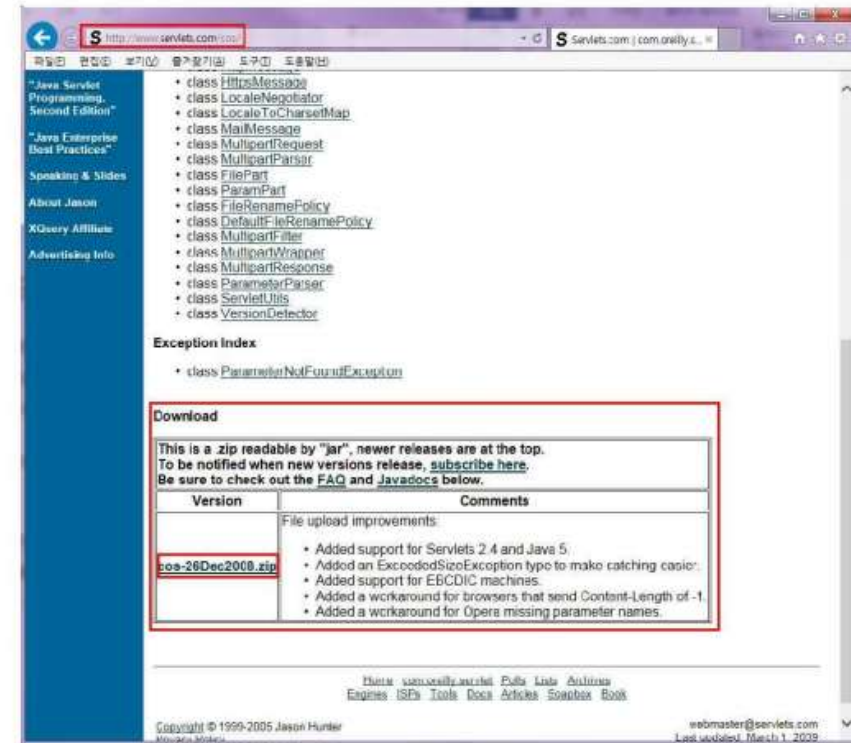
- 파일 업로드는 클라이언트에서 웹 서버로 파일을 전하는 것을 말한다

1. 서블릿 버전 3.0부터 기본제공되는 Part API를 사용한다.
2. 서블릿 버전 3.0미만은 MultiPart API를 사용한다.

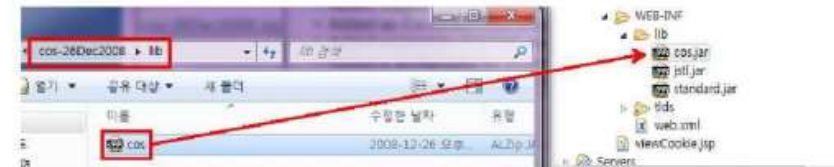
## 2.MultiPart를 사용하는 경우



- <http://www.servlets.com/cos> 접속
- Download에서 cos-26Dec2008.zip 파일 다운



- 압축해제 후 cos-26Dec2008\lib 폴더의 cos.jar파일을 복사하여 WEB-INF\lib폴더에 추가



파일전송을 위한 폼 태그 형식

```
<form name="이름" method="post" enctype="multipart/form-data">  
    <input type="file" name="이름">  
</form>
```

form데이터의 형식으로 **enctype="multipart/form-data"** 가 입력되어야 합니다  
type은 file로 지정합니다

### MultipartRequest 클래스

- MultipartRequest 클래스는 파일 업로드 기능을 담당하는 클래스 입니다.
- cos.jar 파일이 해당 클래스를 제공합니다

메서드명	설명
getParameterNames()	폼에서 전송된 파라미터의 이름을 Enumeration 타입으로 리턴한다.
getParameterValues()	폼에서 전송된 파라미터들을 배열로 받아온다.
getParameter()	객체에 있는 해당 파라미터의 값을 가져온다.
getFileNames()	파일 여러개 업로드할 경우 값들을 Enumeration 타입으로 리턴한다.
getFileSystemName()	서버에 실제로 업로드된 파일의 이름을 의미한다.
getOriginalFileName()	클라이언트가 업로드한 파일의 원본 이름을 의미한다.
getContentType()	업로드 파일의 콘텐츠 타입을 얻을 때 사용한다.
getFile()	서버에 업로드된 파일의 정보를 객체로 얻어낼 때 사용한다.

getParameter는 Request객체의 getParameter기능과 같습니다.  
단, 중복으로 사용할 수 없습니다

## 파일 업로드 실습해보기

WebContent/fileUpload/upload 폴더 생성

파일을 저장하기 위한 디렉터리와 파일의 최대 크기 정의

```
String savePath = "/fileUpload/upload"; //다운받는 경로 설정  
int uploadFileSizeLimit = 5*1024*1024; //파일 최대크기 5M로 제한  
String encType="UTF-8" //char인코딩 방식
```

서버 상의 실제 경로 설정

```
ServletContext context = getServletContext();  
String uploadFilePath = context.getRealPath(savePath);
```

MultipartRequest 클래스 객체 생성

```
MultipartRequest multi = new MultipartRequest(  
    request,          //request 객체  
    uploadFilePath,  //서버 상의 실제 데이터  
    uploadFileSizeLimit, //최대 업로드 파일크기  
    encType,         //인코딩 타입  
    new DefaultFileRenamePolicy()  
);
```

**MultipartRequest** 객체가 생성되는 순간 파일 업로드가 진행되며 생성자는 5가지를 받습니다.



## 파일 업로드 실습해보기

업로드 된 파일의 이름 얻기

```
String filename = multi.getFilesystemName("uploadFile");  
// "uploadFile" 이름은 input 태그의 name과 동일한 이름을 사용한다.
```

실제 업로드 된 파일이 저장될 위치

```
이클립스에서 사용하는 workspace안에  
.metadata\plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps\fileUpload\upload
```

### 알아둬야 할 것

실제 파일은 톰캣 내부의 WAS환경의 프로젝트에 저장됩니다.

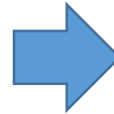
우리가 이클립스에 사용하는 프로젝트는 로컬환경이고  
실제 운영되는 서버는 톰캣 내부에서 실행되게 됩니다.

## 파일 업로드 예제

```
/WebContent/fileUpload/upload_process.jsp
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import="com.oreilly.servlet.MultipartRequest" %>
<%@ page import="com.oreilly.servlet.multipart.DefaultFileRenamePolicy" %>
<html>
<head><title>업로드 처리</title></head>
<body>
<%
    request.setCharacterEncoding("UTF-8");
    String savePath = "/fileUpload/upload";
    int uploadFileSizeLimit = 5 * 1024 * 1024;
    String encType = "UTF-8";

    ServletContext context = getServletContext();
    String uploadFilePath = context.getRealPath(savePath);
    System.out.println("서버상의 실제 디렉터리");
    System.out.println(uploadFilePath);
    try{
        MultipartRequest multi = new MultipartRequest(
            request,
            uploadFilePath,
            uploadFileSizeLimit,
            encType,
            new DefaultFileRenamePolicy());
        String fileName = multi.getFilesystemName("uploadFile");
        if(fileName == null){
            System.out.print("파일 업로드 실패");
        }
        else{
            out.println("<br> 글쓴이 : " + multi.getParameter("name"));
            out.println("<br> 제 &nbsp;목 : " + multi.getParameter("title"));
            out.println("<br> 파일명 : " + fileName);
        }
    }catch(Exception e){
        System.out.print("예외 발생 : " + e);
        e.printStackTrace();
    }
%>
</body>
</html>
```



글쓴이:   
 제 목:   
 파일 선택:

글쓴이 : test  
제 목 : 1  
파일명 : 심리게임만드는중.txt

File Explorer Path: tmp0 > wtpwebapps > JSP\_TEST > freUpload > upload

File List:

이름	수정
심리게임만드는중	2011...

## 다중 파일 업로드 예제

```
<%
request.setCharacterEncoding("utf-8");
//파일 업로드 경로
String uploadFilePath = "C:\\Users\\Park\\Desktop\\jsp\\workspace\\MyWeb\\WebContent\\upload";
//파일 사이즈 선언
int fileSizeLimit = 10 * 1024 * 1024;
//인코딩타입 선언
String encoding = "UTF-8";

try{
    MultipartRequest multi = new MultipartRequest(
        request,
        uploadFilePath,
        fileSizeLimit,
        encoding,
        new DefaultFileRenamePolicy()
    );

    //업로드된 파일명 얻기

    Enumeration files = multi.getFileNames();
    while(files.hasMoreElements()) { //다음 파일이 있다면,

        String file = (String)files.nextElement(); //업로드된 파일 name속성 얻기
        String file_name = multi.getFilesystemName(file); //업로드된 파일명 얻기
        String file_content = multi.getContentType(file); //업로드된 파일타입 얻기

        System.out.println(file);
        System.out.println(file_name);
        System.out.println(file_content);
    }
} catch(Exception e) {
    e.printStackTrace();
}

%>
```