파일 업로드

무조건 post방식

두가지 인코딩 방식에 따라 전송하는 데이터 형식이 달라진다.

파일 전송시에는 `multipart/form-data` 방식을 사용한다.

```
<form method="post" enctype="multipart/form-data">
...
</form>
```

기존에는 파일 업로드용 API가 별도로 필요했지만 서블릿 3.0부터는 라이브러리 없이 사용 가능하다.

라이브러리 없이 하는 방법

```
<form action="ex01_01.jsp" method="post" enctype="multipart/form-data">
       이름: <input type="text" name="name" value="홍길동"/>
       파일: <input type="file" name="upload"/>
       <input type="submit" />
</form>
```

```
<body>
<%
String name = request.getParameter("name");
String upload = request.getParameter("upload");
%>
전송된 이름: <%= name %> <br>
전송된 파일: <%= upload %> <br>
<hr>
${ param.name }
${ param.upload }
</body>
```

출력값

```
전송된 이름: null
전송된 파일: null
```

위파일에서 전달받아도 `null`로 출력만된다.

`requset.getParameter()`로는 `multipart/form-data`형식으로 보낸 값은 받을수 없다.

`multipart/form-data`방식으로 전송할 때 문자열, 사진 상관 없이 스트림 방식으로 전송된다.
`request.getInputStream()` 메서드가 반환한 `ServletInputStream`객체를 통해 데이터를 전달받을 수 있다.

```
<%@page import="java.io.DataInputStream"%>
<%@page import="java.util.Enumeration"%>
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%
   request.setCharacterEncoding("UTF-8");
   // String contentType = request.getContentType();
   Enumeration<String> en = request.getHeaderNames();
```

```
   out.println("> 전송받은 헤더 정보 출력<br>");
   while( en.hasMoreElements() ){
          String key = en.nextElement();
          String value = request.getHeader(key);
          out.println(key+" : "+ value + "<br>");
   }

   out.println("<br> > 전송받은 데이터 출력<br>");
   ServletInputStream sis= request.getInputStream();
   DataInputStream dis = new DataInputStream(sis);
   String line = "";
   while( ( line = dis.readLine() ) != null ){
          out.println( new String(
                        line.getBytes("ISO-8859-1"), "UTF-8" ) +"<br>" );
   }
%>
```

출력값

> 전송받은 헤더 정보 출력

```
host : localhost
connection : keep-alive
content-length : 9861
cache-control : max-age=0
origin : http://localhost
upgrade-insecure-requests : 1
content-type : multipart/form-data; boundary=----
WebKitFormBoundarykY7CNu84ndk1cpC0
user-agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36
accept :
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*
;q=0.8,application/signed-exchange;v=b3
referer : http://localhost/jspPro/days23/ex01.jsp
accept-encoding : gzip, deflate, br
accept-language : ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
cookie : JSESSIONID=1B0815C52ADC044D13BD5D88BF9A1EE2
```

> 전송받은 데이터 출력

```
------WebKitFormBoundarykY7CNu84ndk1cpC0
Content-Disposition: form-data; name="name"

홍길동
------WebKitFormBoundarykY7CNu84ndk1cpC0
Content-Disposition: form-data; name="upload"; filename="bandmember.jpg"
Content-Type: image/jpeg

����JFIF��C
```

```
%#, #&')*)-0-(0%()(��C
...
...
binary 값....
```

스트림 형식이다보니 문자열로 전달받으며 `getParameter` 등과 같은 메서드를 사용할 수 없다.

라이브러리 사용하기

전송받은 데이터를 편하게 사용하기 위해 라이브러리를 사용해야 하는데 아래 사이트에서 제공하는 라이브러리를 사용하자.

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

`cos-20.08.zip`파일을 다운받고 `cos.jar`파일을 lib폴더에 추가한다.

`cos.jar`파일에선 `MultipartRequest`객체를 제공하며 `multipart/form-data`방식으로 전달받은 데이터를 쉽게 관리할 수 있다.

`MultipartRequest` 생성과정은 다음과 같다.

```
MultipartRequest mrequest = new MultipartRequest(
  request       //MultipartRequest 를 만들기 위한 request
, saveDirectory //저장 위치 (File 객체)
, maxPostSize   //최대크기 (int)
, encoding      //인코딩 타입 ("utf-8")
, policy        //파일 정책 (FileRenamePolicy 객체)
);
```

매개변수로 5개가 들어가는데 주석과 같은 내용들이 들어간다.

`policy는` 파일을 `saveDirectory`에 업로드할 때 이미 중복된 이름이 있다면 어떻게 처리할지 결정하는 매개변수이다.

전체코드는 다음과 같다.

```
<%@page import="com.oreilly.servlet.multipart.DefaultFileRenamePolicy"%>
<%@page import="java.io.File"%>
<%@page import="com.oreilly.servlet.multipart.FileRenamePolicy"%>
<%@page import="com.oreilly.servlet.MultipartRequest"%>
```

```jsp
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%
	// 1. upload 폴더 생성이 안되어 있으면 생성
	String saveDirectory =
pageContext.getServletContext().getRealPath("/days23/upload");
	System.out.println(saveDirectory);

	File saveDir = new File(saveDirectory);
	if (!saveDir.exists())
		saveDir.mkdirs();

	// 2. 최대크기 설정
	int maxPostSize = 1024 * 1024 * 5; // 5MB  단위 byte

	//3. 인코딩 방식 설정
	String encoding = "UTF-8";

	//4. 파일정책, 파일이름 충돌시 덮어씌어짐으로 파일이름 뒤에 인덱스를 붙인다.
  //a.txt
	//a1.txt 와 같은 형식으로 저장된다.
	FileRenamePolicy policy = new DefaultFileRenamePolicy();
	MultipartRequest mrequest
	= new MultipartRequest(request //MultipartRequest 를 만들기 위한 request
			, saveDirectory //저장 위치
			, maxPostSize //최대크기
			, encoding //인코딩 타입
			, policy); //파일 정책

	String name = mrequest.getParameter("name");
	File uploadFile = mrequest.getFile("upload");
  //input type="file" 태그의 name 속성값을 이용해 파일객체를 생성
	long uploadFile_length = uploadFile.length();
	String originalFileName = mrequest.getOriginalFileName("upload"); //기존
이름
	String filesystemName = mrequest.getFilesystemName("upload"); //기존
%>

> 이름 :
<%=name%><br>
> 첨부된 파일명 :
<%=uploadFile.getName()%><br>
>>> originalFileName :
<%=originalFileName%>
<br>
>>> filesystemName :
<%=filesystemName%>
<br>
> 첨부된 파일 크기 :
<%=uploadFile_length%>
bytes
<br>
```

꼭 `type="file"` 이 아니더라도 `getParameter()`메서드로 파라미터값을 얻어올 수 있다.

출력값

```
> 이름 : 홍길동
> 첨부된 파일명 : data1.txt
>>> originalFileName : data.txt
>>> filesystemName : data1.txt
> 첨부된 파일 크기 : 9223 bytes
```

이클립스에서 사용할 경우 실제 저장 위치가 contextpath아래에 저장되는 것이 아닌 이클립스 프로젝트 전용으로 사용하는 파일 저장 경로에 저장된다.
실제위치

C:₩Class₩JSPClass₩.metadata₩.plugins₩org.eclipse.wst.server.core₩tmp0₩wtpwebapps₩jspPro₩days23₩upload

```
data.txt
data1.txt
```

contextPath 까지의 실제 파일 시스템 경로를 얻고 싶다면 request.getContextPath() 함수 사용.

`new MultipartRequest()` 생성자가 예외발생없이 끝나면 파일은 실제 시스템에 저장되어 있는 상태이고 여러개의 `input type="file"` 태그를 통해 파일을 동시에 여러 개의 파일도 저장가능하다.

단 각각의 파일을 별도의 디렉토리에 저장하는 것은 불가능... saveDirectory 에 설정한 대로만 저장 가능하다.

출처 : https://kouzie.github.io/

파일 다운로드

1) 시스템에서 평범한 html 페이지가 아닌, 어플리케이션 파일이 리턴된다고 알려주기 위해서 HpptServletResponse 를 세팅해줘야 합니다.

```
response.setContentType("application/octet-stream");
response.setHeader("Content-Disposition",
"attachment;filename=downloadfilename.csv")
```

attachment;filename= 구문을 통해서 다운받는 파일의 파일 이름을 초기화 시켜줄 수 있습니다.

2) 물리적인 위치로부터 파일을 읽어서 유저가 파일을 웹사이트로부터 다운받게 합니다.

```
File file = new File("C:\\temp\\downloadfilename.csv");
FileInputStream fileIn = new FileInputStream(file);
ServletOutputStream out = response.getOutputStream();

byte[] outputByte = new byte[4096];
//copy binary contect to output stream
while(fileIn.read(outputByte, 0, 4096) != -1)
```

```
{
       out.write(outputByte, 0, 4096);
}
fileIn.close();
out.flush();
out.close();
```

출처: https://devx.tistory.com/