JSP - File Uploading

In this chapter, we will discuss File Uploading in JSP. A JSP can be used with an HTML form tag to allow users to upload files to the server. An uploaded file can be a text file or a binary or an image file or just any document.

Creating a File Upload Form

Let us now understand how to create a file upload form. The following HTML code creates an uploader form. Following are the important points to be noted down –

- The form **method** attribute should be set to **POST** method and GET method can not be used.
- The form **enctype** attribute should be set to **multipart/form-data**.
- The form **action** attribute should be set to a JSP file which would handle file uploading at backend server. Following example is using **uploadFile.jsp** program file to upload file.
- To upload a single file you should use a single **<input .../>** tag with attribute **type = "file"**. To allow multiple files uploading, include more than one input tag with different values for the name attribute. The browser associates a Browse button with each of them.

```
<html>
   <head>
      <title>File Uploading Form</title>
   </head>

   <body>
      <h3>File Upload:</h3>
      Select a file to upload: <br />
      <form action = "UploadServlet" method = "post"
         enctype = "multipart/form-data">
         <input type = "file" name = "file" size = "50" />
         <br />
         <input type = "submit" value = "Upload File" />
      </form>
   </body>

</html>
```

This will display the following result. You can now select a file from the local PC and when the user clicks "Upload File", the form gets submitted along with the selected file –

File Upload –

Select a file to upload –

Choisir un fichier Aucun fichier choisi

Upload File

NOTE – Above form is just dummy form and would not work, you should try above code at your machine to make it work.

Writing Backend JSP Script

Let us now define a location where the uploaded files will be stored. You can hard code this in your program or this directory name can also be added using an external configuration such as a **context-param** element in web.xml as follows –

```
<web-app>
....
<context-param>
   <description>Location to store uploaded file</description>
   <param-name>file-upload</param-name>
   <param-value>
      c:\apache-tomcat-5.5.29\webapps\data\
   </param-value>
</context-param>
....
</web-app>
```

Following is the source code for **UploadFile.jsp**. This can handle uploading of multiple files at a time. Let us now consider the following before proceeding with the uploading of files.

- The following example depends on **FileUpload**; make sure you have the latest version of **commons-fileupload.x.x.jar** file in your classpath. You can download it from https://commons.apache.org/fileupload/ .

- FileUpload depends on Commons IO; make sure you have the latest version of **commons-io-x.x.jar** file in your classpath. You can download it from https://commons.apache.org/io/ .
- While testing the following example, you should upload a file which is of less size than *maxFileSize* otherwise the file will not be uploaded.
- Make sure you have created directories **c:\temp** and **c:\apache-tomcat5.5.29\webapps\data** well in advance.

```
<%@ page import = "java.io.*,java.util.*, javax.servlet.*" %>
<%@ page import = "javax.servlet.http.*" %>
<%@ page import = "org.apache.commons.fileupload.*" %>
<%@ page import = "org.apache.commons.fileupload.disk.*" %>
<%@ page import = "org.apache.commons.fileupload.servlet.*" %>
<%@ page import = "org.apache.commons.io.output.*" %>

<%
   File file ;
   int maxFileSize = 5000 * 1024;
   int maxMemSize = 5000 * 1024;
   ServletContext context = pageContext.getServletContext();
   String filePath = context.getInitParameter("file-upload");

   // Verify the content type
   String contentType = request.getContentType();

   if ((contentType.indexOf("multipart/form-data") >= 0)) {
      DiskFileItemFactory factory = new DiskFileItemFactory();
      // maximum size that will be stored in memory
      factory.setSizeThreshold(maxMemSize);

      // Location to save data that is larger than maxMemSize.
      factory.setRepository(new File("c:\\temp"));

      // Create a new file upload handler
      ServletFileUpload upload = new ServletFileUpload(factory);

      // maximum file size to be uploaded.
      upload.setSizeMax( maxFileSize );

      try {
         // Parse the request to get file items.
         List fileItems = upload.parseRequest(request);

         // Process the uploaded file items
         Iterator i = fileItems.iterator();
```

```
            out.println("<html>");
            out.println("<head>");
            out.println("<title>JSP File upload</title>");
            out.println("</head>");
            out.println("<body>");

            while ( i.hasNext () ) {
               FileItem fi = (FileItem)i.next();
               if ( !fi.isFormField () ) {
                  // Get the uploaded file parameters
                  String fieldName = fi.getFieldName();
                  String fileName = fi.getName();
                  boolean isInMemory = fi.isInMemory();
                  long sizeInBytes = fi.getSize();

                  // Write the file
                  if( fileName.lastIndexOf("\\") >= 0 ) {
                     file = new File( filePath +
                     fileName.substring( fileName.lastIndexOf("\\"))) ;
                  } else {
                     file = new File( filePath +
                     fileName.substring(fileName.lastIndexOf("\\")+1)) ;
                  }
                  fi.write( file ) ;
                  out.println("Uploaded Filename: " + filePath +
                  fileName + "<br>");
               }
            }
            out.println("</body>");
            out.println("</html>");
         } catch(Exception ex) {
            System.out.println(ex);
         }
      } else {
         out.println("<html>");
         out.println("<head>");
         out.println("<title>Servlet upload</title>");
         out.println("</head>");
         out.println("<body>");
         out.println("<p>No file uploaded</p>");
         out.println("</body>");
         out.println("</html>");
      }
   %>
```

Now try to upload files using the HTML form which you created above. When you try **http://localhost:8080/UploadFile.htm**, it will display the following result. This will help you upload

any file from your local machine.

File Upload –

Select a file to upload –

Choisir un fichier Aucun fichier choisi

Upload File

If your JSP script works fine, your file should be uploaded in **c:\apache-tomcat5.5.29\webapps\data** directory.