RecipeUploadUsingAction

based on thread http://marc.theaimsgroup.com/?t=106917261100001&r=1&w=2 *StavrosKounis*

Problem

File upload

Solution (example)

Using action.

1. compile .java files and put them in build/webapp/WEB-INF/classes/com/mypackage

2. create a folder under cocoon samples (uploadexample) directory with dir structure:

```
uploadexample
|
+-content
|
+-style
  |
  +xsl
```

3. put the following files in the right folder

4. ask for http://...cocoon.../sample/uploadexample/

dont forget to enable upload in web.xml

sitemap.xmap

```
<?xml version="1.0"?>
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
        <map:components>
                <map:actions>
                        <map:action logger="sitemap.action.fileupload" name="file-upload-action" src="com.
mypackage.FileUploadAction"/>
                </map:actions>
        </map:components>
        <!-- ========================== Pipelines ================================== -->
        <map:pipelines>
                <map:pipeline>
                        <map:match pattern="uploadform.html">
                                <map:read mime-type="text/html" src="content/uploadform.html"/>
                        </map:match>
                        <map:match pattern="doupload">
                                <map:act type="file-upload-action">
                                        <map:parameter name="number-of-files" value="3"/>
                                        <map:parameter name="file-form-field-prefix" value="myfile"/>
                                        <!-- for physical path use
                                                <map:parameter name="upload-directory" value="C:\upload\test1"
/>
                                          -->
                                        <!-- for relative path use -->
                                        <map:parameter name="upload-directory" value="/WEB-INF/myupload/test1"/>
                                        <map:parameter name="overwrite-file" value="rename"/>
                                        <!-- action will return
                                    file-form-field-prefix#number#renamed  true / false
                                    file-form-field-prefix#number#uploaded  true / false
                                    file-form-field-prefix#number#originalFileName - as was in request
                                    file-form-field-prefix#number#physicalFilePath - full path to saved file
                                    file-form-field-prefix#number#uploadedFileName - file name if was renamed
                        -->
                                        <!-- when upload is OK run this -->
                                        <map:generate src="content/okupload.xml"/>
                                        <map:transform src="style/xsl/adduploadinfo.xsl">
                                                <!-- action puts those map parameters -->
```

```
                                        <!-- myfile1 -->
                                        <map:parameter name="myfile1renamed" value="{myfile1renamed}"/>
                                        <map:parameter name="myfile1uploaded" value="{myfile1uploaded}"
/>
                                        <map:parameter name="myfile1originalFileName" value="
{myfile1originalFileName}"/>
                                        <map:parameter name="myfile1physicalFilePath" value="
{myfile1physicalFilePath}"/>
                                        <map:parameter name="myfile1uploadedFileName" value="
{myfile1uploadedFileName}"/>
                                        <!-- myfile2 -->
                                        <map:parameter name="myfile2renamed" value="{myfile2renamed}"/>
                                        <map:parameter name="myfile2uploaded" value="{myfile2uploaded}"
/>
                                        <map:parameter name="myfile2originalFileName" value="
{myfile2originalFileName}"/>
                                        <map:parameter name="myfile2physicalFilePath" value="
{myfile2physicalFilePath}"/>
                                        <map:parameter name="myfile2uploadedFileName" value="
{myfile2uploadedFileName}"/>
                                        <!-- myfile3 -->
                                        <map:parameter name="myfile3renamed" value="{myfile3renamed}"/>
                                        <map:parameter name="myfile3uploaded" value="{myfile3uploaded}"
/>
                                        <map:parameter name="myfile3originalFileName" value="
{myfile3originalFileName}"/>
                                        <map:parameter name="myfile3physicalFilePath" value="
{myfile3physicalFilePath}"/>
                                        <map:parameter name="myfile3uploadedFileName" value="
{myfile3uploadedFileName}"/>
                                </map:transform>
                                <map:transform src="style/xsl/page2html.xsl"/>
                                <map:serialize type="html"/>
                        </map:act>
                        <!-- when action is not OK it will continue there -->
                        <map:generate src="content/notokupload.xml"/>
                        <map:transform src="style/xsl/page2html.xsl"/>
                        <map:serialize type="html"/>
                </map:match>
                <map:match pattern="">
                        <map:redirect-to uri="uploadform.html"/>
                </map:match>
        </map:pipeline>
    </map:pipelines>
</map:sitemap>
```

Content folder

content/notokupload.xml

content/okupload.xml

content/uploadform.html

notokupload.xml

```
<?xml version="1.0"?>
<page>
 <title>file upload action</title>
 <content>
  <para>upload action is NOT ok</para>
 </content>
</page>
```

okupload.xml

```
<?xml version="1.0"?>
<page>
 <title>file upload action</title>
 <content>
  <para>upload action runs ok</para>
  <uploadinfo/>
 </content>
</page>
```

uploadform.html

```
<html>
<body>
<form action="doupload" method="POST" enctype="multipart/form-data">
<input type="file" name="myfile1"><br>
<input type="file" name="myfile2"><br>
<input type="file" name="myfile3"><br>
<input type="submit">
</form>
</body>
</html>
```

Style folder

style/xsl/adduploadinfo.xsl

style/xsl/page2html.xsl

adduploadinfo.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

        <!-- myfile1 -->
        <xsl:param name="myfile1renamed"/>
        <xsl:param name="myfile1uploaded"/>
        <xsl:param name="myfile1originalFileName"/>
        <xsl:param name="myfile1physicalFilePath"/>
        <xsl:param name="myfile1uploadedFileName"/>

        <!-- myfile2 -->
        <xsl:param name="myfile2renamed"/>
        <xsl:param name="myfile2uploaded"/>
        <xsl:param name="myfile2originalFileName"/>
        <xsl:param name="myfile2physicalFilePath"/>
        <xsl:param name="myfile2uploadedFileName"/>

        <!-- myfile3 -->
        <xsl:param name="myfile3renamed"/>
        <xsl:param name="myfile3uploaded"/>
        <xsl:param name="myfile3originalFileName"/>
        <xsl:param name="myfile3physicalFilePath"/>
        <xsl:param name="myfile3uploadedFileName"/>

        <xsl:template match="*|@*|text()|node()|comment()|processing-instruction()">
                <xsl:copy>
                        <xsl:apply-templates select="*|@*|text()|node()|comment()|processing-instruction()"><
/xsl:apply-templates>
                </xsl:copy>
        </xsl:template>

        <xsl:template match="uploadinfo">
          <!-- myfile1 -->
          <FileUploadInfo>
                  <formFieldName>myfile1</formFieldName>
                  <originalFileName><xsl:value-of select="$myfile1originalFileName"/></originalFileName>
                  <physicalFilePath><xsl:value-of select="$myfile1physicalFilePath"/></physicalFilePath>
                  <renamed><xsl:value-of select="$myfile1renamed"/></renamed>
                  <uploaded><xsl:value-of select="$myfile1uploaded"/></uploaded>
                  <uploadedFileName><xsl:value-of select="$myfile1uploadedFileName"/></uploadedFileName>
          </FileUploadInfo>

          <!-- myfile2 -->
          <FileUploadInfo>
                  <formFieldName>myfile2</formFieldName>
                  <originalFileName><xsl:value-of select="$myfile2originalFileName"/></originalFileName>
                  <physicalFilePath><xsl:value-of select="$myfile2physicalFilePath"/></physicalFilePath>
                  <renamed><xsl:value-of select="$myfile2renamed"/></renamed>
                  <uploaded><xsl:value-of select="$myfile2uploaded"/></uploaded>
                  <uploadedFileName><xsl:value-of select="$myfile2uploadedFileName"/></uploadedFileName>
          </FileUploadInfo>

          <!-- myfile3 -->
          <FileUploadInfo>
                  <formFieldName>myfile3</formFieldName>
                  <originalFileName><xsl:value-of select="$myfile3originalFileName"/></originalFileName>
                  <physicalFilePath><xsl:value-of select="$myfile3physicalFilePath"/></physicalFilePath>
                  <renamed><xsl:value-of select="$myfile3renamed"/></renamed>
                  <uploaded><xsl:value-of select="$myfile3uploaded"/></uploaded>
                  <uploadedFileName><xsl:value-of select="$myfile3uploadedFileName"/></uploadedFileName>
          </FileUploadInfo>


        </xsl:template>

</xsl:stylesheet>
```

page2html.xsl

```
<?xml version="1.0"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="page">
   <html>
    <head>
     <title>
      <xsl:value-of select="title"/>
     </title>
     <link href="/styles/main.css" type="text/css" rel="stylesheet"/>
    </head>
    <body>
     <xsl:apply-templates/>
    </body>
   </html>
  </xsl:template>

  <xsl:template match="title">
   <p><xsl:apply-templates/></p>
  </xsl:template>

  <xsl:template match="content">
   <xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="para">
   <p>
    <xsl:apply-templates/>
   </p>
  </xsl:template>

  <xsl:template match="FileUploadInfo">
   <p>
   formFieldName: <b><xsl:value-of select="formFieldName"/></b><br/>
   originalFileName: <xsl:value-of select="originalFileName"/><br/>
   physicalFilePath: <xsl:value-of select="physicalFilePath"/><br/>
   renamed: <xsl:value-of select="renamed"/><br/>
   uploaded: <xsl:value-of select="uploaded"/><br/>
   uploadedFileName: <xsl:value-of select="uploadedFileName"/>
   </p>
  </xsl:template>

  <xsl:template match="@*|node()" priority="-2"><xsl:copy><xsl:apply-templates select="@*|node()"/></xsl:copy><
/xsl:template>
  <xsl:template match="text()" priority="-1"><xsl:value-of select="."/></xsl:template>

</xsl:stylesheet>
```

Actions

FileUploadAction.java

```
package com.mypackage;

import org.apache.cocoon.acting.AbstractAction;
import org.apache.cocoon.servlet.multipart.Part;
import org.apache.cocoon.environment.Context;
import org.apache.cocoon.environment.ObjectModelHelper;
import org.apache.cocoon.environment.Redirector;
import org.apache.cocoon.environment.Request;
import org.apache.cocoon.environment.SourceResolver;
import org.apache.cocoon.environment.http.HttpRequest;

import org.apache.avalon.framework.logger.Logger;
import org.apache.avalon.framework.parameters.Parameters;
import org.apache.avalon.framework.thread.ThreadSafe;
```

```
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;


/**
 * This action takes care about file uploading.
 * as input parameters will be:<br>
 * number-of-files - int - specified in the upload form<br>
 * file-form-field-prefix - name of form field
 * name of form field for processing will be file-form-field-prefix + number-of-files
 * i.e. "myfile1", ...<br>
 * upload-directory - path where the file/s will be saved<br>
 * overwrite-file - deny, allow, rename<br>
 * Action returns back to sitemap the path+name of uploaded files
 *
 * @author Roman Hrivik
 */
public class FileUploadAction extends AbstractAction
                              implements ThreadSafe
{
        /**
         * number-of-files parameter name
         */
        private static final String PARAM_NUMBER_OF_FILES = "number-of-files";
        /**
         * file-form-field-prefix parameter name
         */
        private static final String PARAM_FILE_FORM_FIELD_PREFIX = "file-form-field-prefix";
        /**
         * upload-directory parameter name
         */
        private static final String PARAM_UPLOAD_DIRECTORY = "upload-directory";
        /**
         * overwrite-file parameter name
         */
        private static final String PARAM_OVERWRITE_FILE = "overwrite-file";
        /**
         * overwrite deny
         */
        private static final String OVERWRITE_DENY = "deny";
        /**
         * overwrite allow
         */
        private static final String OVERWRITE_ALLOW = "allow";
        /**
         * overwrite rename
         */
        private static final String OVERWRITE_RENAME = "rename";
        /**
         * default form prefix parameter name for file upload
         */
        private static final String DEFAULT_PARAM_FILE_FORM_FIELD_PREFIX = "file";
        /**
         * default upload directory WEB-INF/work
         */
        private static final String DEFAULT_PARAM_UPLOAD_DIRECTORY_VALUE = "/WEB-INF/work/upload";
    /**
     * cached logger
     */
    private static Logger myLogger;
        /**
         * is debug enabled
         */
        private boolean isDebugEnabled;
```

```
 /**
  * is warn enabled
  */
 private boolean isWarnEnabled;
 /**
  * is error enabled
  */
 private boolean isErrorEnabled;

 /**
  * file-form-field-prefix - name of form field
  */
 private String parFileFormFieldPrefix;

/**
 * number-of-files - int - specified in the upload form
 */
private int parNumberOfFiles;

/**
 * upload-directory - path where the file/s will be saved
 */
private String parUploadDirectory;


/**
 * overwrite-file - deny, allow, rename
 */
private String parOverwriteFile;

/**
 * Controls the processing against some values of the
 * <code>Dictionary</code> objectModel and returns a
 * <code>Map</code> object with values used in subsequent
 * sitemap substitution patterns.
 * NOTE: This interface is designed so that implentations can be
 * <code>ThreadSafe<code>.
 * When an action is ThreadSafe, only one instance serves all requests : this
 * reduces memory usage and avoids pooling.
 * @param resolver    The <code>SourceResolver</code> in charge
 * @param objectModel The <code>Map</code> with object of the
 * calling environment which can be used
 * to select values this controller may need
 * (ie Request, Response).
 * @param source      A source <code>String</code> to the Action
 * @param parameters  The <code>Parameters</code> for this invocation
 * @param redirector
 * @return Map        The returned <code>Map</code> object with
 * sitemap substitution values which can be used
 * in subsequent elements attributes like src=
 * using a xpath like expression: src="mydir/{myval}/foo"
 * If the return value is null the processing inside
 * the <map:act> element of the sitemap will
 * be skipped.
 * @throws java.lang.Exception
 * @exception Exception Indicates something is totally wrong
 */
public Map act(
                        Redirector redirector,
                        SourceResolver resolver,
                        Map objectModel,
                        String source,
                        Parameters parameters)
                        throws Exception {


        /*
         * get logger instance
         */
        if (myLogger == null) {
             myLogger = getLogger();
```

```
                isDebugEnabled = myLogger.isDebugEnabled();
        isWarnEnabled = myLogger.isWarnEnabled();
                isErrorEnabled = myLogger.isErrorEnabled();
        }

        if (isDebugEnabled) {
                myLogger.debug("FileUploadAction start");
        }

        /*
         * fileMap keeps FileUploadInfo about uploaded files
         */
        Map fileMap = new HashMap();

        /*
         * read sitemap parameters
         */
        try {

                parNumberOfFiles = Integer.parseInt(parameters.getParameter(PARAM_NUMBER_OF_FILES, "0"));

        } catch (Exception e) {

                parNumberOfFiles = 0;
                if (isErrorEnabled) {
                        myLogger.error("NumberFormatException in parameter " + PARAM_NUMBER_OF_FILES);
                }
                /*
                 * stop processing action -> return null
                 */
                return null;
        }

        parFileFormFieldPrefix = parameters.getParameter(PARAM_FILE_FORM_FIELD_PREFIX,
DEFAULT_PARAM_FILE_FORM_FIELD_PREFIX);
        parUploadDirectory = parameters.getParameter(PARAM_UPLOAD_DIRECTORY,
DEFAULT_PARAM_UPLOAD_DIRECTORY_VALUE);
        parOverwriteFile = parameters.getParameter(PARAM_OVERWRITE_FILE, OVERWRITE_DENY);


        if (isDebugEnabled) {
                myLogger.debug("input parameters readed/default setted: "
                                                + PARAM_NUMBER_OF_FILES + "=" + parNumberOfFiles + " "
                                                + PARAM_FILE_FORM_FIELD_PREFIX + "=" + parFileFormFieldPrefix +
" "
                                                + PARAM_OVERWRITE_FILE + "=" + parOverwriteFile + " "
                                                + PARAM_UPLOAD_DIRECTORY + "=" + parUploadDirectory + " "
                                                + PARAM_FILE_FORM_FIELD_PREFIX + "=" + parFileFormFieldPrefix +
" "
                                        );
        }

        /*
         * get request
         */
    Request req = ObjectModelHelper.getRequest(objectModel);

    /*
     * get context to generate realPath
     */
    Context context = ObjectModelHelper.getContext(objectModel);

    if (req instanceof HttpRequest) {

        HttpRequest request = (HttpRequest) req;

           /*
            * create necessary directory structure
            */
           File uplDir = new File(parUploadDirectory);
       if (isDebugEnabled) {
```

```
    myLogger.debug("upload directory is: "
                   + uplDir.getAbsolutePath()
                  );
}

    if (uplDir.isAbsolute() == false) {
    // check for first relative slash
    if(parUploadDirectory.indexOf('/') != 0) {
        parUploadDirectory = "/" + parUploadDirectory;
    }

            uplDir = new File(context.getRealPath(parUploadDirectory));
    }

    if (isDebugEnabled) {
            myLogger.debug("absolute upload directory is: "
                                        + uplDir.getAbsolutePath()
                                    );
    }

    if (uplDir.exists() == false) {
            uplDir.mkdirs();
            if (isDebugEnabled) {
                    myLogger.debug("directory path does not exist creating directories: "
                                                + uplDir.getAbsolutePath()
                                                + " successfully: " + uplDir.exists()
                                               );
            }
    }


    /*
     * go thru form parameters
     * add to return Map FileUploadInfo object to get informations about
     * uploaded file
     */
     if (isDebugEnabled) {
             myLogger.debug("going thrue "+parNumberOfFiles+" form parameters ");
     }

    for (int i = 1; i <= parNumberOfFiles; i++) {
            /*
             * create file informations and put it to return map
             */
            FileUploadInfo fileInfo = new FileUploadInfo();
            fileInfo.setFormFieldName((parFileFormFieldPrefix + i));

            /*
             * get Part
             */
            Part filePart = (Part) request.get(fileInfo.getFormFieldName());
            if (filePart != null) {
                    // do upload flag
                    boolean doUpload = true;

                // strip off path (IE for Windows includes it)
                String filename = filePart.getUploadName();
                int lastSlash = filename.lastIndexOf('\\');
                filename = filename.substring(lastSlash+1);

                    // set fileinfo
                    fileInfo.setOriginalFileName(filename);
                    fileInfo.setUploadedFileName(fileInfo.getOriginalFileName());

                    // define file target
                    File target = new File(uplDir, fileInfo.getUploadedFileName());
                    // check if target exist
                    if ( target.exists()
                             && parOverwriteFile.equalsIgnoreCase(OVERWRITE_RENAME)) {
                            // rename file
                            fileInfo.setUploadedFileName("" + System.currentTimeMillis() + "" + i + "_"
```

```
+ fileInfo.getOriginalFileName());
                                fileInfo.setRenamed(true);

                                target = new File(uplDir, fileInfo.getUploadedFileName());

                        }
                        else if ( target.exists()
                                        && parOverwriteFile.equalsIgnoreCase(OVERWRITE_DENY)) {
                                // skip writing file
                                doUpload = false;
                                fileInfo.setUploaded(false);
                        }
                        // otherwise OVERWRITE_ALLOW


                        fileInfo.setPhysicalFilePath(target.getAbsolutePath());


                        if (doUpload) {
                                // when success set uploaded true otherwise = false
                                fileInfo.setUploaded( saveFile(filePart, target) );
                        }
                }




                /*
                 * put fileinfo to filemap
            * as sitemap accepts only String values - file info will be separated
                    */
           fileMap.put(fileInfo.getFormFieldName() + "renamed", Boolean.toString(fileInfo.isRenamed()));
           fileMap.put(fileInfo.getFormFieldName() + "uploaded", Boolean.toString(fileInfo.isUploaded()));
           fileMap.put(fileInfo.getFormFieldName() + "originalFileName", fileInfo.getOriginalFileName());
           fileMap.put(fileInfo.getFormFieldName() + "physicalFilePath", fileInfo.getPhysicalFilePath());
           fileMap.put(fileInfo.getFormFieldName() + "uploadedFileName", fileInfo.getUploadedFileName());




                if (isDebugEnabled) {
                        myLogger.debug("file info added to return map. "
                                                        + fileInfo.toString());
                }
           } // end for (int i = 1; i <= parNumberOfFiles; i++)
     } // end if (req instanceof MultipartHttpServletRequest)
     else
     {
         if (isWarnEnabled) {
             myLogger.warn("file upload is not enabled");
         }
         return null;
     }
    return fileMap;
   }


   /**
    * save uploaded Part to target
        * @param filePart represents a file part parsed from a http post stream.
        * @param target - target file to save
        * @return boolean true when file was saved
        */
   private boolean saveFile(Part filePart, File target)
          {

           if (isDebugEnabled) {
                   myLogger.debug("saving file to "
                                                   + target.getAbsolutePath());
           }
```

```
BufferedInputStream in = null;
BufferedOutputStream out = null;
byte [] buff = new byte[8192];
int bytesRead;

boolean flag = true;

try {

        in = new BufferedInputStream(filePart.getInputStream());
        out = new BufferedOutputStream(new FileOutputStream(target));

        // write
        while((bytesRead = in.read(buff, 0, buff.length)) != -1) {
           out.write(buff, 0, bytesRead);
        }

} catch (FileNotFoundException e) {

        // log error
   flag = false;
   if (isErrorEnabled) {
           myLogger.error("FileNotFoundException: "
                                                 + e.toString());
   }

} catch (IOException e) {

        // log error
        flag = false;
   if (isErrorEnabled) {
           myLogger.error("IOException: "
                                                 + e.toString());
   }

} catch (Exception e) {

   // log error
   flag = false;
   if (isErrorEnabled) {
           myLogger.error("Exception: "
                                                 + e.toString());
   }

}
finally {

           try {
                   if (in != null) {
                           in.close();
                   }
                   if (out != null) {
                           out.flush();
                           out.close();
                   }
           } catch (IOException e1) {
                   // log error
                   flag = false;
                   if (isErrorEnabled) {
                           myLogger.error("IOException: "
                                                                 + e1.toString());
                   }
           }
}


if (target.exists() == false) {
   flag = false;
}
```

```
                if (flag && isDebugEnabled) {
                   myLogger.debug("file saved "
                                                        + target.getAbsolutePath());
                }
                else {
                   myLogger.debug("file NOT saved "
                                                        + target.getAbsolutePath());
                }

                return flag;

          }

}
```

FileUploadInfo.java

```
package com.mypackage;

import java.io.File;

/**
 * @author Roman Hrivik
 *
 * Keeps informations about uploaded file with FileUploadAction
 *
 */
public class FileUploadInfo {

        /**
         * original field name
         */
        private String formFieldName;
        /**
         * original file name as it comes from user
         */
        private String originalFileName;
        /**
         * full physical filepath
         */
        private String physicalFilePath;
        /**
         * was file renamed with upload action
         */
        private boolean renamed;
        /**
         * was file successfully uploaded
         */
        private boolean uploaded;
        /**
         * uploaded file name - diffrent if was renamed
         */
        private String uploadedFileName;

        /**
         * @return String formFieldName
         */
        public String getFormFieldName() {
                return formFieldName;
        }

        /**
         * @param formFieldName
         */
        public void setFormFieldName(String formFieldName) {
                this.formFieldName = formFieldName;
        }
```

```
	/**
	 * @return String originalFileName
	 */
	public String getOriginalFileName() {
		return originalFileName;
	}

	/**
	 * @param originalFileName
	 */
	public void setOriginalFileName(String originalFileName) {
		this.originalFileName = originalFileName;
	}

	/**
	 * @return String physicalFilePath
	 */
	public String getPhysicalFilePath() {
		return physicalFilePath;
	}

	/**
	 * @param physicalFilePath
	 */
	public void setPhysicalFilePath(String physicalFilePath) {
		this.physicalFilePath = physicalFilePath;
	}

	/**
	 * @return boolean renamed
	 */
	public boolean isRenamed() {
		return renamed;
	}

	/**
	 * @param renamed
	 */
	public void setRenamed(boolean renamed) {
		this.renamed = renamed;
	}

	/**
	 * @return boolean uploaded
	 */
	public boolean isUploaded() {
		return uploaded;
	}

	/**
	 * @param uploaded
	 */
	public void setUploaded(boolean uploaded) {
		this.uploaded = uploaded;
	}

	/**
	 * @return String uploadedFileName
	 */
	public String getUploadedFileName() {
		return uploadedFileName;
	}

	/**
	 * @param uploadedFileName
	 */
	public void setUploadedFileName(String uploadedFileName) {
		this.uploadedFileName = uploadedFileName;
	}

	/* (non-Javadoc)
```

```
         * @see java.lang.Object#toString()
         */
        public String toString() {
                return ("FileUploadInfo: "
                                + " isUploaded=" + isUploaded()
                                + " isRenamed=" + isRenamed()
                                + " getFormFieldName=" + getFormFieldName()
                                + " getOriginalFileName=" + getOriginalFileName()
                                + " getPhysicalFilePath=" + getPhysicalFilePath()
                                );
        }

}
```

Attachment: FileUpload.zip