File Upload Interceptor

See this page for more examples and advanced configuration

Interceptor that is based off of MultiPartRequestWrapper, which is automatically applied for any request that includes a file. It adds the following parameters, where [File Name] is the name given to the file uploaded by the HTML form:

- [File Name] : File - the actual File
- [File Name]ContentType : String - the content type of the file
- [File Name]FileName : String - the actual name of the file uploaded (not the HTML name)

You can get access to these files by merely providing setters in your action that correspond to any of the three patterns above, such as setDocument(File document), setDocumentContentType(String contentType), etc.
See the example code section.

This interceptor will add several field errors, assuming that the action implements ValidationAware. These error messages are based on several i18n values stored in struts-messages.properties, a default i18n file processed for all i18n requests. You can override the text of these messages by providing text for the following keys:

- struts.messages.error.uploading - a general error that occurs when the file could not be uploaded
- struts.messages.error.file.too.large - occurs when the uploaded file is too large
- struts.messages.error.content.type.not.allowed - occurs when the uploaded file does not match the expected content types specified
- struts.messages.error.file.extension.not.allowed - occurs when the uploaded file does not match the expected file extensions specified

Parameters

- maximumSize (optional) - the maximum size (in bytes) that the interceptor will allow a file reference to be set on the action. Note, this is **not** related to the various properties found in struts.properties. Default to approximately 2MB.
- allowedTypes (optional) - a comma separated list of content types (ie: text/html) that the interceptor will allow a file reference to be set on the action. If none is specified allow all types to be uploaded.
- allowedExtensions (optional) - a comma separated list of file extensions (ie: .html) that the interceptor will allow a file reference to be set on the action. If none is specified allow all extensions to be uploaded.

Extending the Interceptor

You can extend this interceptor and override the acceptFile method to provide more control over which files are supported and which are not.

Examples

Example action mapping:

Error rendering macro 'code': Invalid value specified for parameter 'java.lang.NullPointerException'

```
<action name="doUpload" class="com.example.UploadAction">
    <interceptor-ref name="fileUpload"/>
    <interceptor-ref name="basicStack"/>
    <result name="success">good_result.jsp</result>
</action>
```

Notice the interceptor configuration in the preceding example.

Example JSP form tags:

Error rendering macro 'code': Invalid value specified for parameter 'java.lang.NullPointerException'

```
<s:form action="doUpload" method="post" enctype="multipart/form-data">
    <s:file name="upload" label="File"/>
    <s:submit/>
</s:form>
```

You must set the encoding to `multipart/form-data` in the form where the user selects the file to upload.

Example Action class:

Error rendering macro 'code': Invalid value specified for parameter 'java.lang.NullPointerException'

```
   package com.example;

   import java.io.File;
   import com.opensymphony.xwork2.ActionSupport;

   public UploadAction extends ActionSupport {
      private File file;
      private String contentType;
      private String filename;

      public void setUpload(File file) {
         this.file = file;
      }

      public void setUploadContentType(String contentType) {
         this.contentType = contentType;
      }

      public void setUploadFileName(String filename) {
         this.filename = filename;
      }

      public String execute() {
         //...
         return SUCCESS;
      }
}
```

Setting parameters example:

```
<interceptor-ref name="fileUpload">
  <param name="allowedTypes">
     image/png,image/gif,image/jpeg
  </param>
</interceptor-ref>
```

This part is optional and would be done in place of the `<interceptor-ref name="fileUpload"/>` line in the action mapping example above.