FileUploadsWithCocoon

This was written at a time between 2.0 and some critical refactoring in April 2003. I'm reworking the document to reflect reality in each of the releases. See FileUploadsWithCocoon2.1 or FileUploadsWithCocoon --GeoffHoward

---

To begin with, it may be startling for some to learn that file uploads are handled automatically by Cocoon in some (possibly default) configurations from **any** request. What this means is that a form constructed to post a file to any Cocoon page will result in that file being saved automatically to disk. Without this knowledge, the file upload sample shipping with Cocoon (here in most installations) seems confusingly simple to some - the code in the xsp does not upload the file (nor does any action as some may suspect) but merely lists information about it..

Overview

When the Cocoon servlet receives a request, org.apache.cocoon.components.request.MultipartRequestFactoryImpl (configurable) parses out the multipart data which comprises the file data, saves it to disk (configurable) in the upload directory (configurable) and places an org.apache.cocoon.components.request.multipart.FilePart object in the request using the name of the multipart file parameter (the name of the <input type="file" ...> from the html). This object provides access methods for later Cocoon steps to deal with the file. It handles multiple files by saving them all to disk and placing multiple objects in the request.

FilePart is abstract, but has two concrete subclasses: FilePartFile and FilePartArray which are the actual objects placed in the request as described above, depending on configuration options to be described below. FilePartFile is a file already saved to disk (as above) and provides an additional method getFile(), not defined in FilePart, which returns the java.io.File object for the file on disk. FilePartArray holds the contents of the uploaded file in a byte array in memory which will not persist after the request is serviced unless otherwise acted upon.

Configuration

Unlike most configuration in Cocoon, all the options mentioned above are set in web.xml:

To configure the request factory, and therefore the handling of Multipart requests, use init-param "request-factory". Options are (classes are all in the org.apache.cocoon.components.request package):

Class name	Description
MultipartRequestFactoryImpl	This is the default factory.
MaybeUploadRequestFactoryImpl	You can opt in for this factory if maybeupload.jar is present. (what does that get you? MaybeUploadRequestFactory
SimpleRequestFactoryImpl	This factory does not allow uploads.

The last option will cause Cocoon to ignore all file uploads (a good idea for live sites not interested in file uploads).

In addition, the following init-param options are available in web.xml#1:

parameter name	default value (may vary by dist)	Description (see web.xml comments for more)
upload-directory	$TOMCAT_HOME\work\standalone\localhost\cocoon\cocoon-files\upload-dir in most installations	where Cocoon should put uploaded files.
autosave-uploads	true (false in some dists?)	Causes all files to be saved to upload-dir
overwrite-uploads	rename	what to do with name conflicts with existing file. Acceptable values are **deny**, **allow**, **rename** (default, but may still have race condition with concurrent requests)
upload-max-size	10000000 (10 Mb)	maximum allowed size of the upload.

Examples See FileUploadWithAction for an action which retrieves a FilePartFile object by name from the request (requires autosave-uploads=true) for further processing (i.e., move to different directory, save to Blob in database, etc.)

: Most of these options are available in 2.0.x and 2.1 alike - however, some (autosave-uploads and overwrite-uploads) were introduced after the release of 2.0.3, and are therefore only available in 2.0.4 and of course, 2.1. Additionally, some parameters while available were ignored in 2.0.3 due to a bug.

Attachment: database-actions.pdf

Attachment: telefonos.csv