Managing Uploaded Files with FileMgr

Posted At : January 28, 2009 8:30 AM | Posted By : Steve
Related Categories: ColdFusion, com.sebtools

Years ago, I used to put uploaded files any random place in my sites. This turned out to be a pain when I wanted to deploy a new version of a site. If I was using FTP, I couldn't just upload a set of root folders because some folders used for uploads might be within them and I wouldn't want to overwrite files that had been uploaded on the live site. If I was using subversion, then I would have to select "ignore" on each upload location individually. Even comparisons in **BeyondCompare** were a little more tedious.

Basically, the problem is that uploaded files are really data. It just doesn't make sense to store that data directly in the database for performance reasons. Even so, I don't want to treat them the same as code files that I can safely deploy from the staging/development site.So, I decided to make one folder for uploads and use subfolders for different kinds of uploads (that is, for different sets of data). This worked well, but I didn't want any given program in a site to be forced to know about the structure of that site. I just wanted it to know what folder it was using - not the full path location of that folder. I also didn't want to have to worry about creating that folder as part of deployment of new code - in my mind my code should create what it needs (see **ActiveSchema in DataMgr**).

FileMgr was created to solve these challenges.

The init() method of FileMgr takes one argument: UploadPath. This holds the full server path for where to store file uploads.

If I want a component to ensure the existence of a folder, I call the makeFolder() method.

Example code to initialize FileMgr:

```
<cfset Application.FileMgr = CreateObject("component","FileMgr").init("C:\inetpub\wwwroot\files\")>
```

So, if I have a CMS and I want it to put images in a "cms-images" folder (C:\inetpub\wwwroot\files\cms-images\ using the above example), I might use the following code:

```
<cffunction name="init" access="public">
<cfargument name="FileMgr" type="any" required="yes">

<cfset variables.FileMgr = arguments.FileMgr>

<cfset variables.FileMgr.makeFolder("cms-images")>

<cfreturn This>
</cffunction>
```

Uploading a file to the folder then uses the uploadFile() method with the following arguments:

- FieldName (required): The name of the form field holding the uploaded file
- Folder (optional): The folder in which to put the file
- NameConflict (optional): The value of the NameConflict attribute in **cffile**

The method returns a CFFILE structure.

So, uploading a file might use the following code:

```
<cfset sImage = variables.FileMgr.uploadFile("Image","cms-images")>
```

Reading or deleting a file uses the readFile() and deleteFile() method respectively. Both have the same arguments:

- FileName (required): The name of the file
- Folder (optional): The folder in which the file is located

Code to delete a file might look like this:

```
<cfset variables.FileMgr.deleteFile("myimage.jpg","cms-images")>
```

All of this allows the component itself to not know anything about where the files are stored, other than the name of the folder being used by that component.

FileMgr is open source and free for any use.