

# **CFAxess User Guide**

by WiseTREND, Inc.

# ***WISETREND***

**Converging knowledge from digital content and print.**

This page intentionally left blank.

## Table of Contents

Use your PDF viewer bookmarks feature for clickable links.

Introduction.....	1	LDir.....	19
Changes and Improvements .....	1	Let .....	19
Licensing .....	1	Lineage.....	20
Command Console .....	1	Lock.....	21
Command Syntax.....	2	Login.....	21
Commands .....	3	Logout .....	22
AddMember .....	3	LOpen.....	22
CD .....	3	Mkdir .....	22
CLS .....	4	Move.....	22
Copy.....	4	NewVersion .....	23
CreateGroup.....	5	Open.....	23
CreateUser .....	5	Pause .....	23
Delete.....	6	Prompt.....	23
DeleteGroup .....	7	Put.....	24
DeleteUser .....	7	RemoveMember .....	25
Dir .....	8	RemoveProps.....	25
Disown .....	9	Rename .....	25
Echo.....	9	Replicate.....	26
Exec .....	10	ResumeOnError.....	26
Exit .....	10	Search .....	26
Get .....	10	SetAccess.....	27
GetHandle .....	12	SetParam.....	30
GetParam .....	12	SetProps.....	30
GetProps .....	15	SwitchTo.....	31
GetSchema .....	16	Tree .....	31
Goto .....	17	Unlock .....	33
Help.....	17	UpdateSchema .....	33
History .....	18	WriteLog .....	33
If-then .....	18	Customer Support.....	34
LCD .....	19		

## Introduction

CFAxess is our product to replace DSAxess which was part of the DocuShare SDK that was created to work with their former DocuShare Windows Client. CFAxess has been and will continue to be developed by WiseTREND to address this issue. It's a compatible replacement for DSAxess. All the most commonly used features are addressed in version 1.0. CFAxess commands are the same as those documented in DSAxess, so you can use the DocuShare Windows Client "DocuShare SDK.chm" help file and sample scripts to learn and support CFAxess. You can refer to that along with our documentation. CFAxess will allow you to upgrade to new platforms and continue running your scripts or explore using them for the first time.

CFAxess is primarily based on the WebDAV interface initially popularized by DocuShare and in use with many content platforms today. Xerox, Netscape, Microsoft, IBM, and others in the IETF collaborated to create the WebDAV standard. It extends the http interface to perform library services for checking out documents, locking them, and checking in new versions as well as associating metadata with them. There are many additional functions supporting the overall library services found in document management systems that can be found in CFAxess. Saving commands as scripts allows you to incorporate additional functionality with DocuShare resulting in streamlined processes and automated tasks.

The current version as of this update is CFAxess 1.5.9.

## Changes and Improvements

We've updated many of the capabilities to include enhancements, such as ...

- CFAxess also supports custom document objects and their properties.
- Support for custom collection objects.
- Ability to log in using LDAP or Active Directory accounts.
- Support for SSL and TL3 security models.
- Works alongside DocuShare Drive.
- Easy to install and includes documentation that is called via Help requests.
- Avoids your having to rewrite DSAxess scripts in .Net or Java or other languages.
- Accepts variables passed from batch files and scripts.
- Commands and scripts run faster than in DSAxess.

Track additional changes and improvements on our [CFAxess page](#).

## Licensing

You can evaluate CFAxess for 30 days and develop scripts to run. To license CFAxess, send the software code presented to [ocrsupport@wisetrend.com](mailto:ocrsupport@wisetrend.com) along with a purchase order or call with a credit card. We can then respond with a liberation key for use on your PC or server. This is not licensed per corporate entity, so call if you want multiple licenses. Once licensed, you are eligible for updates while under a WiseTREND support & maintenance agreement.

## Command Console

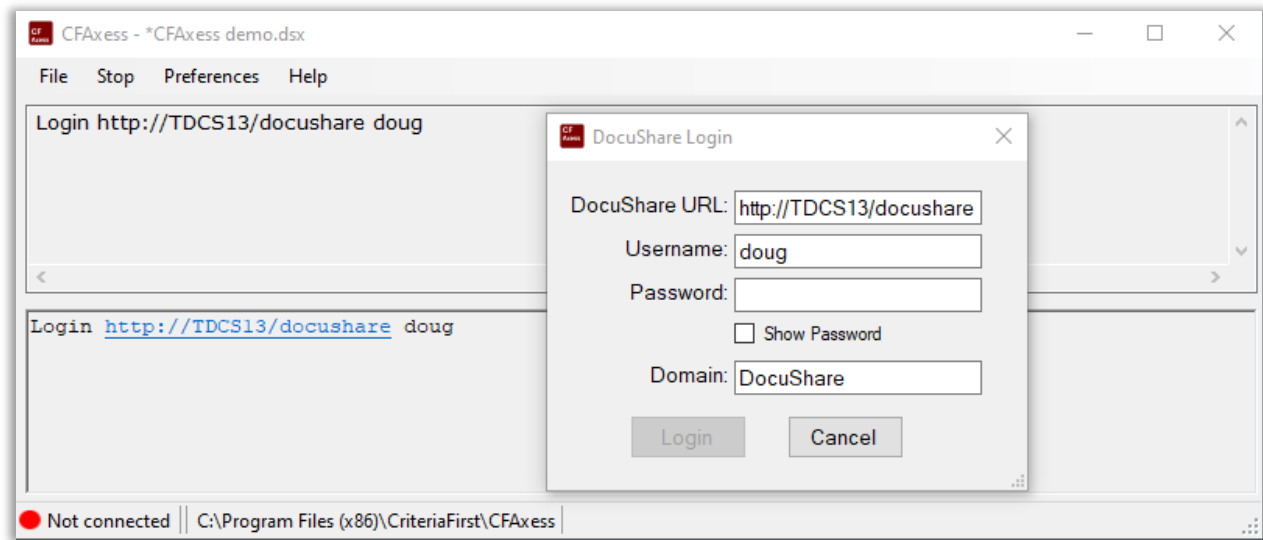
The CFAxess console is not like DSAxess which immediately runs the command after you press enter. CFAxess doesn't do anything until you click "Run", then it runs the entire script in the top pane and displays the results in the bottom pane. The top pane has a white background when it's ready to alter while the lower pane will remain gray. When a script or command is being processed the top pane will also be gray indicating the need to wait before making changes. If you highlight only the line(s) of code you want to test and click Run or use the Alt+R shortcut it will run only that or those lines. This lets you do things like test if-then statements without having to save it as a script and execute it. You can have more than one window open so you can examine your script in one and perform additional development and testing in the other. Commands are not case sensitive. Use the File menu to save your scripts with

the default extension of ".dsx". You can reopen it in CFAxess or call CFAxess and the script from another program in the following manner. A Windows batch file would use the call command.

Call "[drive]\Program Files (x86)\WiseTREND\CFAxess\CFAxess.exe" [scriptname].dsx

The command to call or run CFAxess will vary in different programming or scripting environments.

Following is an example of beginning a new session where you specify which DocuShare server to connect to and then issue a login command. If you leave out either your name or password you will see the DocuShare Login dialog appear for you to fill out before the remaining lines are processed. For scripts that should run in the background you should enter both the username and password in the login line.



Once logged in the remainder of the commands would show in the lower pane that you've changed into Collection-10 and then issued a command to view its contents with the Dir command. Commands, such as the examples at the bottom for addressing the help document, can be commented out from running by preceding them with the # character. You can execute one or more lines without running an entire script by highlighting them and clicking on Run (hotkey: Alt+r).

## Command Syntax

These notations are used in this guide:

- Items within square brackets [] are optional.
- Items within braces <> are placeholders, described by the contents.
- <handle> is a placeholder for a DocuShare object handle such as Collection-10, File-123, MailMessage-567.
- A DocuShare handle is composed of 2 or 3 components:
  - an object class, such as Collection, Document, or MailMessage
  - an object index, which is a positive integer, separated from the object class by a dash "-"
  - an optional version index, which is a positive integer, separated from the object index by a forward slash "/"
- Items separated by vertical bars | are "either-or" options.
- When the -quiet or -q argument is added to a command it will suppress the listing normally presented as output while still gathering the associated variable(s).
- When a triple dot ellipse (...) appears in a command string it means that segment can be repeated, as in the case where assigning metadata values to properties occurs.

There's an implied ^ and \$ wrapped around the regex string. Those anchor the match to the start and end of the title. So the regex has to match the entire title, not just a fragment of it.

To suppress echoing of a script command, start a command with an @ character. The statement, @Echo ON, executes the Echo command but does not print the command line in the console window. The @echo suppression may be used with any statement.

Additionally, the -quiet option on a command suppresses all result output for that command.

For example:

```
Dir Collection-123 # echoes the "dir" command, followed by the "dir" results
@Dir Collection-123 # outputs the "dir" results without first echoing the command
Dir -quiet Collection-123 # echoes the "dir" command only; "dir" results are suppressed
@Dir -quiet Collection-123 # neither echoes the command, nor outputs the "dir" results
```

## Commands

These commands are in alphabetical order, so it helps to think in terms of logging onto DocuShare and performing functions somewhat common to those you'd use in a Windows command window. You'll change and make directories, move or copy files, search for files, rename or delete directories or files, etc. Of course, DocuShare does much more that you'll be able to perform. All commands and parameters are case insensitive except for references to DocuShare objects.

To receive help on a command, use any of the following syntax:

```
<command> -help
<command> -h
<command> -?
help <command>
Put -h
Help Put
```

The CFAxess User Guide is available in the CFAxess installation directory and will appear in your preferred PDF viewer advanced to your command in the documentation.

## AddMember

**AddMember <GroupHandle> <UserOrGroupHandle> [<UserOrGroupHandle> ...]**

This allows you to add one or more members to a group in the DocuShare domain.

```
AddMember Group-28 User-46 User-34 User-57
```

Refer to CreateGroup to create a new group for the members and RemoveMember to remove Users from groups, when appropriate.

### *Environment results*

1. Sets DSHandle to the modified group's handle index (e.g. 123)
2. Sets DSOBJECT to the modified group's handle (e.g. Group-123)
3. Sets DSResultCount to the number of added/removed members (e.g. 2)
4. Sets DSResult-\* to the added/removed members (e.g. DSResult-1="User-238", DSResult-2="User-31")
5. Does not modify DSItemCount, DSItem-\*, DSSearchHitCount, DSSearchHit-\*

## CD

**CD <Title>**

**CD -regex <TitleRegex>**

Changes the current directory (collection) to the collection. You can use either the title (which can accept \* and ? wildcards), or may use a regular expression.

## **CD <CollHandle>**

Changes the current directory (collection) to the collection specified by CollHandle.

Valid parameters formats are:

```
cd collection-10
cd c10
cd 10
cd Sales
cd "Sales Reports"
```

### *Environment results*

1. Sets DSHandle to the selected collection's handle index (e.g. 12)
2. Sets DSOBJECT to the selected collection's handle (e.g. Collection-12)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## **CLS**

The CLS command clears the CFAxess console screen.

## **Copy**

**Copy <Handle> <DestCollectionHandle>**

**Copy -regex <TitleRegex> <DestCollectionHandle>**

The Copy command copies a DocuShare object (ObjectHandle) to a specified collection (of CollHandle).

Copy sets DSHandle to the handle number of the destination collection.

Valid command samples are:

- copy file-13 collection-10
- copy f13 c10
- copy collection-15 c10

Copy adds an object to another container, typically a collection, but does not replicate the contents. If a file or collection deletion occurs to any directory to which the file or collection was copied, the deletion occurs for all collections containing the deleted source.

If you wish to replicate the contents of a file or collection to another collection, refer to the Replicate command (when available).

If you wish to move a file or a collection, and remove any reference to the file or collection from the source collection, use the Move command.

If you wish to remove a file or collection from a collection without deleting it from all parent collections, refer to the Disown command.

### *Environment results*

1. Sets DSHandle to the destination container's handle index (e.g. 12)
2. Sets DSOBJECT to the destination container's handle (e.g. Collection-12)
3. Sets DSResultCount to the number of copied objects (e.g. 2)
4. Sets DSResult-\* to the copied objects (e.g. DSResult-1="File-123", DSResult-2="File-456")
5. Does not modify DSItemCount, DSItem-\*, DSSearchHitCount, DSSearchHit-\*

## CreateGroup

**CreateGroup <GroupTitle> [<PropName>=<PropValue> ...]**

You can create a group in the embedded DocuShare domain and assign one or more properties such as Summary, Description, and Keywords. You cannot directly create or delete groups on an LDAP server. If your DocuShare server is connected to an LDAP system, such as Microsoft's Active Directory, then you would create your groups there and go to the DocuShare Admin Home and Add them to DocuShare. There is also a Convert from DocuShare to LDAP for users and groups, but that's not how most administrators choose to add users or groups. Refer to the DocuShare Administrator's Guide for more information.

CreateGroup "Project Managers" "summary=Thought leaders"

### *Environment results*

1. Sets DSHandle to the created/deleted group's handle index (e.g. 12)
2. Sets DSOBJECT to the created/deleted group's handle (e.g. Group-12)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## CreateUser

**CreateUser <Username> <Password> first\_name=<name> last\_name=<name>  
userLevel=<1|2|3|4|5> [email=<email address>] [isActive=<1|0>]**

You'll generally want to make them active right away. 1 = Yes and 0 = No.

Their userLevel pertains to the index value of the capability you want to assign to them. These are the designations:

- |                  |  |
|------------------|--|
| 1 = Read-Only    | DocuShare secure read-only user                                      |
| 2 = dsLimited    | Not implemented.   |
| 3 = imageManager | Not implemented.   |
| 4 = dsFull       | DocuShare Read/Write/Manage contributing user                        |
| 5 = CPX          | Advanced DocuShare user able to create workspaces and content rules. |

createuser brubble Wilm@b00b5 first\_name=Barney last\_name=Rubble  
email=brubble@wisetrend.com isActive=1 userLevel=4

You can determine or use their user handle using echo or other commands, such as ...

echo User-%DSHandle%  
echo %DSObject%

There is no designation for anonymous guest users for public or unsecured internal content. Once you've created a user you cannot change their designation using SetProps for the user. Go to the DocuShare Users area in the browser to make such changes.

### *Environment results*

1. Sets DSHandle to the created/deleted user's handle index (e.g. 32)
2. Sets DSOBJECT to the created/deleted user's handle (e.g. User-32)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*



## Delete

**Delete <Handle>**

**Delete <TitleMask>**

**Delete -regex <TitleRegex>**

The Delete command deletes a DocuShare object, either by handle, by title (with \* and ? wildcards accepted), or by title regex.

DSAxess supported all DocuShare standard and custom object types.

If you're using this command in an unmonitored script, you should enter "prompt off" on a previous line to avoid it requesting permission to complete the deletion.

CFAxess also supports the following additional objects types and can accept both \* and ? as wildcards against object titles (not filenames) for the current collection:

```
Delete "Re: Quote 2022022 from BillsFromUs"
```

```
Delete "Re: Quote *"
```

```
Delete Collection-856
```

```
Delete Collection="2015"
```

Delete sets DSHandle to the handle number of the deleted object.

To Delete a collection by the collection handle number, add the Collection- prefix: for example, on a directory search that returns:

```
COLL10: Directory 1
```

```
COLL20: Directory 2
```

To delete a collection whose handle is 10, enter:

```
delete Collection-10 or Delete c10
```

To delete a file by the file handle, add the File- prefix; for example, on a directory search that returns:

```
FILE1: Document1
```

```
FILE2: Document2
```

To delete Document1 you must type in:

Delete File-1 or Delete f1

You can specify a complete filename followed with a name=specifier to cause a deletion by name. You can also specify a file mask using a wildcard, such as name=\*.txt, to delete from the current collection all files which are found by the mask.

For mask deletions, you are prompted for confirmation. To disable prompting, execute prompt off beforehand. Only an asterisk (\*) is a valid wildcard; you cannot use the question mark (?). You must use the format, \*.[ext], where [ext] is the file extension for files that you want to delete. By specifying name=\*.\*, you are deleting all files (excluding sub-collections) that are in the current collection. Files are deleted by filename, not by title. The command records the transaction result to the system registry. The registry key is [DSAxess\Console] and is located in [HKCU\Software\Xerox\DocuShare Client]. The transacted handle is stored as Handle. The output handle is the handle of the last deleted item.

If you delete a file or collection, it will be deleted from all collections that may be parents to the file or collection you wish to delete. To remove a file or collection from a single parent collection, not all parent collections, refer to the command Disown

### *Enhancements*

Previously, Delete was only invoked by typing the complete Delete keyword. This behavior was modified to allow use of the abbreviated DEL keyword as well.

Note: -q may be used as well.

Valid wild cards are:

- \*.\*—deletes all
- \*.xxx—deletes files with extension name xxx
- xxx\*—deletes objects of all types with names starting with xxx

The conventional methods of specifying a target object are preserved. So, these examples are still valid:

```
del File-123
del f123
del Collection-456
del c456
del name=*.htm
del name=test123.docx
```

### *Environment results*

1. Sets DSHandle to the first deleted object's handle index (e.g. 123)
2. Sets DSOBJECT to the first deleted object's handle (e.g. File-123)
3. Sets DSResultCount to the number of deleted objects (e.g. 2)
4. Sets DSResult-\* to the deleted objects (e.g. DSResult-1="File-123", DSResult-2="File-456")
5. Does not modify DSItemCount, DSItem-\*, DSSearchHitCount, DSSearchHit-\*

## **DeleteGroup**

### **DeleteGroup <GroupHandle>**

Deleting a group has no effect on its member users other than to eliminate a group common to them. This affects only groups in the DocuShare domain and not those imported from an LDAP server. Refer to CreateGroup, when appropriate.

```
DeleteGroup Group-28
```

### *Environment results*

1. Sets DSHandle to the created/deleted group's handle index (e.g. 12)
2. Sets DSOBJECT to the created/deleted group's handle (e.g. Group-12)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## **DeleteUser**

### **DeleteUser <UserHandle>**

Be sure to consider using

```
SetProps <UserHandle> isActive=0
```

instead of DeleteUser, because DeleteUser will delete all their contributed content. You probably won't want to do that.

```
deleteuser User-456
```

### *Environment results*

1. Sets DSHandle to the created/deleted user's handle index (e.g. 32)

2. Sets DSOBJECT to the created/deleted user's handle (e.g. User-32)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## Dir

**Dir [<-long>] [<SearchText>]**

**Dir [<-long>] -regex <TitleRegex>**

The Dir command prints the current directory. If SearchText is specified, items that contain the specified text is printed. By default, the title, filename, size and lock status are printed. To get more information, including the summary, use the -long switch.

If SearchText contains a file extension wild card, Dir will output items whose filenames match the specified file extension.

Dir does not update the DSHandle console variable.

Dir updates the DSItem and DSItemCount script variables. To obtain the handle of an item, use variable %DSItem-\*% where asterisk (\*) is the index number of the item within the obtained directory. The following example obtains the directory of Collection-10, and downloads all files in the directory to the temporary folder given by system variable %temp%:

```
@ECHO OFF
LOGIN
CD Collection-10
DIR
ECHO Found items=%DSItemCount%
ECHO Non-collection items follow:
LET ItemNum=1
:StartLoop
IF %ItemNum% > %DSItemCount% THEN
GOTO EndLoop
IF %DSItem-%ItemNum%% doesnotcontain File THEN
GOTO BumpIndex
GETPROPS %DSItem-%ItemNum%%
GET %handle% %temp%\%document%
:BumpIndex
LET ItemNum=ItemNum+1
GOTO StartLoop
:EndLoop
```

The example used a nested variable %DSItem-%ItemNum%% to enumerate the handles of all items. The example also used variables set by GetProps, %handle% and %document% to supply the file handle, and file name to the command Get. Refer to GetProps for more information on the property variables updated by this command.

### *Environment results*

1. Sets DSHandle to the viewed container's handle index (e.g. 12)
2. Sets DSOBJECT to the viewed container's handle (e.g. Collection-12)
3. Sets DSItemCount to the number of viewed children in the container (e.g. 4)
4. Sets DSItem-\* to the viewed children (e.g. DSItem-1="File-123", DSItem-2="File-456", DSItem-3="Collection-13", DSItem-4="MailMessage-92")
5. Does not modify DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## Disown

**Disown <ObjectHandle>**  
**Disown -regex <TitleRegex>**

The Disown command removes a DocuShare object of ObjectHandle from a parent collection.

Disown sets DSHandle to the handle number of one of the remaining parents (collections).

To disown a collection by the collection handle number you must add the collection- prefix. For instance on a directory search that returns;

COLL10: Directory 1

COLL20: Directory 2

To disown Directory 1, enter either:

disown collection-10

disown c10

To disown a file by the file handle you must add the File- prefix. For instance on a directory search that returns:

FILE1: Document1

FILE2: Document2

To disown Document1, enter either:

disown file-1

disown f1

If you disown a file or collection, it only removes it from the collection in which the Disown command was executed. The document may still be accessed from other collections that are parents of the disowned file or collection. To delete a file or collection from all parent collections, refer to the Delete command.

### *Environment results*

1. Sets DSHandle to the first disowned object's handle index (e.g. 123)
2. Sets DSOBJECT to the first disowned object's handle (e.g. File-123)
3. Sets DSResultCount to the number of disowned objects (e.g. 2)
4. Sets DSResult-\* to the disowned objects (e.g. DSResult-1="File-123", DSResult-2="File-456")
5. Does not modify DSItemCount, DSItem-\*, DSSearchHitCount, DSSearchHit-\*

## Echo

### **[@]Echo [on|off]**

The Echo command echoes script commands (on or off) during script execution. Echo without an argument prints the current Echo setting.

If the supplied argument is not on or off, Echo prints the literal text of the argument.

Echo This is a test

The Echo statement prints This is a test in the console window regardless of the current Echo setting. It is possible to include one or more variables in the output text. Following execution of a Put statement, you can print the handle of the uploaded file by issuing this command:

Echo File handle is File-%DSHandle%

If the handle number generated by Put was 123, %DSHandle% is expanded to 123 resulting in an output of the text File handle is File-123.

To suppress echoing of a script command, start a command with an @ character. The statement, @Echo ON, executes the Echo command but does not print the command line in the console window. The @echo suppression may be used with any statement.

## Exec

**Exec [-async] [-suppressed] <command> [<arg> ...]**

The Exec command causes CFAxess to execute the given command with the supplied arguments, which are optional.

The -async option causes CFAxess to proceed immediately to the next script command; without -async, CFAxess waits for the executed command to finish.

The -suppressed option causes CFAxess to launch the command without displaying its main window.

## Exit

The Exit command causes CFAxess to logout and close the current session.

## Get

**Get <CollHandle> [DestPath]**

**Get <MailMessageHandle> [DestPath]**

**Get [-lock] <FileHandle[/VersionHandle]> [DestPath]**

**Get [-lock] -regex <TitleRegex> [DestPath]**

The Get command downloads a collection, mailmessage, file, or file version to your PC. A new directory or file is created using the name, FilePath, to receive the item. If you download a collection, all files and sub-collections are downloaded. The command records the transaction result to DSHandle. The output handle is the handle of the downloaded file.

When downloading a collection, Get updates script variables DSResult and DSResultCount. Variable DSResultCount contains the number of downloaded items. Variables DSResult-\* where \* range from 1 to DSResultCount, holds the handle of a downloaded item.

The -lock option will place a lock on a file that is downloaded. Do not use a -lock option when executing a Get command on collections or file versions.

[FilePath] must be place in quotes if the path contains spaces.

See Downloading Documents for examples on downloading documents.

Version option

The -version option has been added to the Get command. When enabled, the option causes the command to download or copy all versions of files that the command processes.

To download all file versions, the following syntax is used:

**Get -versions [file or collection handle] [destination file system path]**

The short form, -v may be used instead of -versions.

If the handle parameter specifies a file, Get retrieves the file's properties, creates a subfolder using the filename, stores the properties in an XML file in the subfolder, and downloads the file versions into the subfolder. Because the subfolder's name is automatically obtained from DocuShare, the destination path

parameter should only contain the path that specifies an existing folder (see the section below on the -name option). Get creates the subfolder in the specified parent folder. A downloaded file version receives a filename prefixed with a file handle and version number form, File-123\_10, where the file handle is File-123 and the version number is 10. The filename part that follows the prefix is the filename of the version file in DocuShare. The file properties are stored in an XML file named after the file handle with a .xml file extension.

The following example downloads the properties and versions of File-123 to the temp file directory:

```
Get -v File-123 c:\temp
```

If File-123 had the name, Business proposal.doc, a subfolder called Business proposal.doc is created in the C:\temp folder, and the XML properties file and all of the version files of File-123 are retrieved and stored in the new subfolder.

If the handle parameter specifies a collection, Get downloads the preferred file version only. It does not retrieve the file properties.

The Get command in DSAXess 3.0 and earlier assumed that a download path specified in a Get command line ended with a filename or folder name. Get assigned this name to the downloaded file or collection. DSAXess 3.1 no longer made that assumption. Instead, it regards a specified download path as the path to the folder into which the file or collection is downloaded. If you need the previous version's behavior, issue the following command:

```
Setparam getname 0
```

If you wish to revert to the 3.1 behavior, set the getname control parameter to 1. Also, if you want to temporarily switch to the 3.1 behavior, the new -name option can be used as part of a Get command line. The option instructs CFAxess to retrieve the file or collection name from DocuShare and appends the name to the path you supplied.

The -lock option cannot be used with the -versions option. If used, the option causes an invalid argument error.

When downloading a Mailmessage you specify the handle of the object and the filepath to where it should be saved. If the Mailmessage is stored in DocuShare with attachments, those attachments will be downloaded and embedded in the Outlook .msg file. When opening the file in Outlook or a compatible viewer you will find the attachments shown in a table at the top of the message. The formatting of the message is determined by the manner in which DocuShare provides which is well-formatted and pleasing to the eye but not the same as it would have appeared when sent from the original email sender. DocuShare will show thumbnail placeholders in the bottom of the message, but those placeholders will not show the thumbnail when exported using the Get command. We think this will not be a problem and would be difficult to change.

```
Get <ObjectHandle> [filepath]
```

```
Get Mailmessage-123 c:\temp
```

There is no Put counterpart to Get for the purpose of uploading .msg files as a DocuShare Mailmessage. Attempting to do this would result in a generic document with an .msg file extension. Instead, use DocuShare Drive and the Outlook Add-on or the DocuShare Email Agent to send email to be saved as well-formed Mailmessages. The body of the .msg file is saved in DocuShare as a record in the database and the attachments are saved as renditions linked to the record.

#### *Environment results*

1. Sets DSHandle to the first retrieved object's handle index (e.g. 123)
2. Sets DSOBJECT to the first retrieved object's handle (e.g. File-123)
3. Sets DSResultCount to the number of retrieved objects (e.g. 2)
4. Sets DSResult-\* to the retrieved objects (e.g. DSResult-1="File-123", DSResult-2="File-456")
5. Does not modify DSItemCount, DSItem-\*, DSSearchHitCount, DSSearchHit-\*

## GetHandle

**GetHandle** <Handle>

**GetHandle** <File-n/[m]>

The GetHandle command finds a DocuShare object handle (Version-x) from a File version handle (File-n/m). A Version-x handle can be used to retrieve object properties for a file version or set properties of a file version.

Example 1: List properties of a file version

```
GetHandle File-80/1
```

```
GetProps %DSObject%
```

Example 2: Set comment property of a file version

```
GetHandle File-80/1
```

```
SetProps %DSObject% "comment=Very first version"
```

*Environment results*

1. Sets DSHandle to the resolved object's handle index (e.g. 789)
2. Sets DSObject to the resolved object's handle (e.g. Version-789)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## GetParam

**GetParam** <name>

The GetParam command retrieves the value of various parameters for a DSAccess console session.

The command updates script variable DSParam with the value of the queried parameter for referencing within a script. The script below sets DSParam to the server address and prints the address to the console window.

```
@ECHO OFF
```

```
GETPARAM server
```

```
ECHO DocuShare URL: %DSParam%
```

The parameters are grouped and listed below. The queried name and found value are output to the console and also stored under a registry key of CFAXESS for external referencing.

The queried name and found value outputs to the console and also stored under the CFAXESS registry key for external referencing. The registry key is [DsAccess\Console] is located in [HKCU\Software\Xerox\DocuShare Client]. The name and value are stored as **ParamName**.

Some of the parameters are writable. For a listing of writable parameters see Setparam.

### Access parameters

Parameter	Description
server	This parameter lists the http connection for the currently logged in server.
version	This parameter returns the DocuShare version of the currently logged in server.

### User parameters

Parameter	Description
-----------	-------------

userid	This parameter returns the user's id number for the logged in DocuShare server.
username	This parameter returns the user's name for the logged in DocuShare server.
auth	This parameter returns the user's authentication token string for the logged in DocuShare server.

#### Socket parameters

Parameter	Description
txpacket	This parameter returns the transmission packet size (measured in bytes).
txtimeout	This parameter returns the transmission timeout (measured in milliseconds).
ipaddr	This parameter returns the IP address for the logged in DocuShare server.
hostname	This parameter returns the hostname for the logged in DocuShare server.
port	This parameter returns the port address of the client that is communicating with the logged in DocuShare server.

#### Transaction parameters

Parameter	Description
handle	This parameter returns the handle of the last command that returns a value to the handle. This parameter is equivalent to the console variable %DSHandle%. Refer to individual commands that return a handle value.
handlelist	This parameter returns the list of result handles for the items downloaded or uploaded by a multi-file command. The first entry in the list is the item count; subsequent entries are the item handles. The contents of this parameter are equivalent to script variables DDSResultCount and DSResult-* that are automatically updated by the commands Get, Put, and Replicate when the operated source item is in regards to a collection.
error	This parameter returns the error code for the last error encountered. This parameter is equivalent to console variable %DSError%. A zero value is returned if an error is not encountered in the current session.
status	This parameter returns the status code for the last command. This parameter is equivalent to console variable %DSStatus%. A zero value indicates a successful command execution.

#### Server information

Server Information	Description
--------------------	-------------



xml	This parameter returns a value of one, if the current session with the logged in DocuShare server is XML enabled; otherwise, a zero is returned.
serverreadonly	This parameter returns a value of one, if the current session with the logged in DocuShare server is a read-only connection; otherwise, a zero is returned.
capability	<p>This read-only integer parameter returns a bit-OR value containing the following server information:</p> <ul style="list-style-type: none"> <li>• Server supports new version comment—bit 0</li> <li>• Server supports collection upload—bit 1</li> <li>• Server uses secure HTTP protocol—bit 3</li> <li>• Server supports XML—bit 4</li> <li>• Server is in read-only mode—bit 5</li> <li>• Server uses NTLM challenge/response protocol for user authentication—bit 6</li> </ul>
version	This read-only parameter returns the major and minor version number of a DocuShare server last contacted.
time	This read-only parameter returns the time of the last server transaction reported by the web server.
localtime	This read-only parameter returns the time of the last server transaction reported by the client machine.
timeoffset	This read-only parameter returns the time difference between time and local time.
pctime	This parameter returns the current time reported by the client machine.

#### Transfer mode

Transfer Mode	Description
downloadclashprompt	This parameter returns a value of one (1), if the Get command prompts the user when a name clash occurs during download. A zero value return signifies that a prompt will not appear if the Put command causes a name clash.
uploadclashprompt	This parameter returns a value of one (1), if the Put command prompts the user when a name clash occurs during uploading of a file or collection. A zero value return signifies that a prompt will not appear if the Get command causes a name clash.
uploadasnewdoc	<p>This parameter returns a value of zero (0), if a Put command treats all uploaded files as new versions in those instances where a file with the same name already exists in the target collection. A zero value return signifies that all uploads are treated as new files.</p> <p>If setparam uploadclashprompt is set to 0 the Put command will do whatever uploadasnewdoc directs, but only on name clashes. If there's no name clash, then the put is always a new document as there was no existing document to apply a</p>

	new version to.
--	-----------------

## Directory cache control

**cacheexpireminutes**—this parameter returns the expiration time in minutes for the current session.

### Environment results

1. Sets DSParam to the parameter's value
2. Sets ParamName to the parameter's name (this is a DSAccess registry entry)
3. Sets ParamValue to the parameter's value (this is a DSAccess registry entry)
4. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## GetProps

### GetProps <Handle> [PropertyName]

#### GetProps [-all] <Handle>

The GetProps command finds properties of a DocuShare object (ObjectHandle). Only the standard properties are queried. GetProps supports all standard and custom object types and their properties.

GetProps sets DSHandle to the handle number of the parent (collection) of the queried object. When

The following list of standard properties are associated with a file:

title, summary, description, keywords, handle, owner.title, owner.handle, create\_date, modified\_date, modified\_by, parent.title, parent.handle, document, content\_type, author, max\_versions, locked\_by, size.

The following list of standard properties are associated with a collection:

title, summary, description, keywords, handle, owner.title, owner.handle, create\_date, modified\_date, modified\_by, parent.title, parent.handle, logo, bg\_image, sort\_order, n\_items.

GetProps saves the values of all queried property script variables. A script variable for a queried property is constructed by enclosing the property name with % signs. For example, the owner name property can be obtained by not referencing variable %owner.title%.

#### Retrieving specified properties

To specify properties to be retrieved, enter the properties after the object handle specification:

GETPROPS <handle> [prop1 prop2 ... propN]

The command retrieves the values of prop1, prop2, ..., propN where prop1, prop2, ..., propN are the names of valid properties of an object specified by handle. Properties should be separated by spaces.

GETPROPS File-123 title summary owner author

#### Retrieving all properties

The option instructs the command to retrieve all core, custom and instance properties of a DocuShare object. The command achieves this by first requesting for <propname> and then issuing a <propfind> for the retrieved property names.

GETPROPS -all <handle>

#### Retrieving file version properties

The command can be used to retrieve properties of a file version that is specified by a file handle and a version number.

GETPROPS File-n/m

Where File-n is the file handle and m is the ordinal version number of a file version.

If you know the Version handle of a version object, use the version handle.

GETPROPS Version-n <prop list>

If no property name is given, the command retrieves the following properties:

- MIME type of content data
- owner of object
- creation time
- modification time
- comment text

#### **Retrieving link information** (not yet implemented)

GetProps can be used to find out if an object is linked to another. Specify the name of a link specifier following the object handle. The specifier must conform to the format link:

<link-type>.<linkage-direction>

Where link-type is the name of a link type that is pre-defined in the link schemata of the server, and linkage-direction is either source or destination.

This command example finds out what object is linked to File-116.

GetProps File-116 link.jobTicket.destination

link.jobTicket.destination=abc.xjt (File-119)

#### **Retrieving permissions information**

The property "acl" is a special synthetic property. Retrieving it displays the permissions for the object, such as a collection or document. This example is followed by the results.

```
GetProps File-48609 acl
acl=
"Russell Kent"(User-238): Reader Writer Manager readlinked readobject readhistory writelinked
writeobject manage
"Bob Dylan"(User-47): Reader readlinked readobject
"George Harrison"(User-35): Reader readlinked readobject
"Jimi Hendrix"(User-46): Reader readlinked readobject
"Content Administrators"(Group-2): Reader Writer Manager readlinked readobject readhistory
writelinked writeobject manage
```

See SetAccess for more information.

#### *Environment results*

1. Sets DSHandle to the queried/modified object's handle index (e.g. 123)
2. Sets DSOBJECT to the queried/modified object's handle (e.g. File-123)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## **GetSchema**

### **GetSchema [-link] [<className>]**

The GetSchema command retrieves and prints server schemata. This command can retrieve either class or link schemata. The command works for DocuShare Release 5 or greater.

The only available option is the -link option. The -link option causes the command to retrieve link schemata instead of class schemata. A class should not be specified if this option is used.

Example 1: GetSchema output

To display all items in the schemata, enter the command without an argument.

```
GETSCHEMA
SchemaVersion: 1126592331345
Class-0.Name: File
Class-0.Label: Document
File.Prop-1: abstract
```

```
File.abstract.IsRequired: 0
File.Prop-2: add_as_draft
...
File.Prop-30: summary
File.summary.IsRequired: 0
Class-1.Name: SavedQuery
Class-1.Label: SavedQuery
SavedQuery.Prop-1: children
SavedQuery.children.IsRequired: 0
SavedQuery.Prop-2: Class Label
SavedQuery.Class Label.IsRequired: 0
...
```

#### Example 2: GetSchema [class] output

To limit the display to the schema data of a particular object class, add the name of the class. To get schemata for the MailMessage class, enter:

```
GETSCHEMA MailMessage
SchemaVersion: 1126592331345
Class-0.Name: MailMessage
Class-0.Label: MailMessage
MailMessage.Prop-1: body
MailMessage.body.Label: Message
MailMessage.body.Type: 2
...
```

#### Example 3: link option

This is an example to retrieve link schemata; enter:

```
GETSCHEMA -link
```

## Goto

### Goto <Label>

The Goto command passes control to a subsequent line containing the specified label prefixed with a colon (:). Labels are not case sensitive.

This example passes control to the line marked by label :NEXTJOB.

```
Goto NEXTJOB
```

```
...
:NextJob
...
```

## Help

### Help [<command>]

**<command> [-help] | [-h] | [-?]**

The Help command opens the included PDF documentation included in the installation directory for CFAxess. It will generally appear in your default browser and take you to the specific command in the documentation. Some browsers may not fully function in this manner. There are several methods of calling for Help shown in the following examples:

```
Help GetProps
GetProps -help
GetProps -h
GetProps -?
```

You can also check for the latest version of the [CFAxess User Guide.pdf](#) on our website.

## History

### History <FileHandle>

The History command lists the versions of the specified file by FileHandle.

History sets DSHandle to the handle number for the queried file.

## If-then

### If <condition> Then

The If..Then command executes the statement in the next script line if the specified condition is met. If the condition is not satisfied, the next line is skipped. A condition is specified by one of the following comparison statements:

A = B	A equals B
A <> B	A does not equal B
A < B	A is less than B
A > B	A is greater than B
A <= B	A is less than or equal to B
A >= B	A is equal to or greater than B
A contains B	A contains the string of B
A does not contain B	A does not contain the string of B

Note: A, B, or both may contain a variable (console, script or environment variable). No other comparison formats are supported. Also, the operands and operator must be separated by white space. For example, specifying A=B generates a syntax error.

A line containing an If..Then statement must not be a compound statement. It must terminate without any subordinate statement. Therefore, the following statements are valid.

```
If %DSStatus% <> 0 Then  
Goto DOJOB1
```

The following compound statement is not valid.

```
If %DSStatus% <> 0 Then Goto DOJOB1
```

The script interpreter ignores the subordinate Goto statement.

The If..Then command executes the next occurring command only, if the condition is satisfied. Execution resumes at the following line if the condition is not met. Use an If..Then statement with a Goto statement to execute a condition dependent, multi-statement block.

An iterative loop may be written by combining statements If..Then, Let, and Goto. The following example queries Collection-10 through Collection-13 using a loop structure.

```
Let CollHandle=10  
:StartLoop  
GetProps Collection-%CollHandle%  
If %CollHandle% = 13 Then  
Goto EndLoop  
Let CollHandle=%CollHandle%+1  
Goto StartLoop  
:EndLoop
```

## LCD

### LCD <DirPath>

The LCD command changes the current directory to the new directory given by LocalDirPath in your machine.

## LDir

### LDir

The LDir command lists the contents of the current local directory set by the LCD command.

## Let

### Let <Script Variable>=<Value>

The Let command assigns a number or text value to a script variable. An assigned text value is enclosed with double quotation marks. Let supports signed numbers, such as -123, -VariableA, or +VariableB. Variables are not case sensitive. Nested variables are supported, such as %A%B%. For example, the variable, %DSItem-%ItemIndex%, yields a proper handle provided that %ItemIndex% is a user-defined integer variable and its value falls within 1 through %DSItemCount%.

If the right hand side of the assignment describes an arithmetic operation, Let evaluates it and assigns the numeric result to the variable of the left hand side. Let only supports addition (+), subtraction (-), multiplication (\*), and division (/). Let supports signed numbers, such as -123, -VariableA, or +VariableB. Valid example assignments:

```
Let TextVar1 = "This is a text variable"
Let TextVar2 = "File handle is File-%DSHandle%"
Let NumVar1 = 1
Let NumVar1 = NumVar1 + 1
Let NumVar2 = NumVar1 * 1.5
Let NumVar3 = NumVar1 - NumVar2
```

Although the interpreter does not raise an error, text and number variables should not be mixed.

A script variable is addressed as %VARIABLE% where VARIABLE is the name of a valid variable. A script variable can be used in a command argument or as part of a conditional statement in an If...Then statement. The following statements create script variables and assign a pathname and title to them.

```
Let FilePath = "c:\my documents\proposal.doc"
Let FileTitle = "Business proposal"
Let FileCollection = "Collection-13"
```

To upload the above file, this script can be used.

```
CD %FileCollection%
Put %FilePath% title=%FileTitle%
```

The value assigned to a script variable can contain the following:

- one or more valid script variables
- Windows system environment variables
- CFAxess console variables

The following script variables are pre-defined and automatically updated when certain commands are executed.

Pre-defined Variables	Description
DSHandle	Transaction handle number.
DSStatus	Command execution Status code.
DSError	Command execution Error code.
DSCommand	Executed command.
DSParam	Parameter value obtained from GetParam.
DSItemCount	Item count from Dir, Lineage.
DSItem-*	Item handles from Dir, Lineage.
DSSearchHitCount	Item count from Search
DSSearchHit-*	Item handles from Search.
DSResultCount	Item count from Get, Put, Replicate.
DSResult-*	Item handles from Get, Put, Replicate.

Note: Asterisk (\*) represents a placeholder for a numeric handle number.

## Lineage

### Lineage [<ObjectHandle>]

The Lineage command prints the lineage for the current collection or the one specified by [ObjectHandle]. This allows the user to list the path back to the root collection for any object contained in a collection including the collection itself. Lineage is useful when you want to know where something exists in DocuShare or you want to export an object, such as a document or mailmessage, and retain its file path.

Lineage Collection-740

Ancestral Hierarchy of Collection-740 (<http://TDCS13:8080/docushare/>)

Collection-740: Testimonium

Collection-67: Active Accounts

Collection-66: Ad Vantage

Collection-10: DocuShare Business Center

echo %DSPath%

DocuShare Business Center\Ad Vantage\Active Accounts\Testimonium

Lineage File-16200

Ancestral Hierarchy of File-16200 (<http://TDCS13:8080/docushare/>)

File-16200: Using DocuShare document routing to update a Word document.doc

Collection-740: Testimonium

Collection-67: Active Accounts

Collection-66: Ad Vantage

Collection-10: DocuShare Business Center

echo %DSPath%

DocuShare Business Center\Ad Vantage\Active Accounts\Testimonium\Using DocuShare  
document routing to update a Word document.doc

#### *Environment results*

1. Lineage sets DSHandle to the handle number of the current or specified collection.
2. Lineage sets DSOBJECT as the object type and handle for the current or specified collection.
3. Lineage updates script variables DSItem-Count and DSItem-\* that are used to iterate the ancestor collections in a script.
4. DSItem-Count indicates how many levels deep the collection is from the root including the target collection.  
DSItem-\* indexes the order from the specified object as 1 and counts up to the root collection. So, if the specified object is a collection the target ObjectHandle is DSItem-1, the parent is DSItem-2, the grandparent is DSItem-3, etc.. If the specified object is a file the target ObjectHandle is DSItem-1, the collection containing the file would show as the parent collection DSItem-2, the grandparent is DSItem-3, etc..
5. Lineage sets DSPath as a string to indicate the Windows-compliant path. For example a target collection could show as ...  
Reports\Top Secret\CIA\Phone tapping  
... while a target file could show as ...  
Reports\Top Secret\CIA\Phone tapping\Dave Smith's ESG profile.docx

## Lock

### **Lock <FileHandle>**

The Lock command locks a file. The command records the transaction result to the system registry. The registry key is [DsAxess\Console] is located in [HKCU\Software\Xerox\DocuShare Client]. The transacted handle is stored as DSHandle.

Lock sets DSHandle to the locked file.

#### *Environment results*

1. Sets DSHandle to the locked/unlocked object's handle index (e.g. 123)
2. Sets DSOBJECT to the locked/unlocked object's handle (e.g. File-123)
3. Does not modify DSItem-Count, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## Login

### **Login [<server URL>] [<username>] [<password>] [<domain>]**

The Login command logs you into a DocuShare session. Username must be specified first. If any other than the <server URL> credential parameter is not specified, a dialog runs to let you specify them. The <server URL> can appear anywhere. The <username>, <password>, and <domain> must appear in that order, if they appear at all. The command records the transaction result to the system registry. The [DsAxess\Console] registry key is located in [HKCU\Software\Xerox\DocuShare Client]. The transacted handle is stored as DSHandle.

To login using the internal DocuShare domain ...

Login <username> <password>

To specify the server when logging using the internal DocuShare domain ...

Login <server URL> <username> <password>

To login using an external network domain ...

Login <username> <password> <domain>



You can specify both the server and the domain when logging in. Login sets DSHandle to the handle number for the authenticated user. CFAxess will remember the server you'd last logged into. See Command Console on page 2 for examples of logging into DocuShare.

#### *Enhancements*

1. Ability to specify the server and domain in the command line.
2. Successful login reveals information about the server you've connected to.

#### *Environment results*

1. Sets DSHandle to the logged-in user's handle index (e.g. 31)
2. Sets DSOBJECT to the logged-in user's handle (e.g. User-31)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## Logout

### **Logout**

The Logout command logs you out from the current server session and clears the internally cached password.

#### *Environment results*

1. Does not modify DSHandle, DSOBJECT, DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## LOpen

### **LOpen <path>**

The LOpen command opens a file system object in the client machine.

The command may readily be used to open a file downloaded from DocuShare.

LOpen "c:\users\doug\documents\products\CFAxess\Documentation\CFAxess User Guide.docx"

## MkDir

### **MkDir [-type=<objectClass>] <Title> [propName=value ...]**

The Mkdir command makes a new directory (collection). The name of the new directory is TitleText. The command records the transaction result to the system registry. The [DsAxess\Console] registry key is located in [HKCU\Software\Xerox\DocuShare Client]. The transacted handle is stored as DSHandle. The output handle is the handle of the created collection.

Mkdir sets script variable DSHandle to the handle number of the created collection.

#### *Environment results*

1. Sets DSHandle to the created collection's handle index (e.g. 12)
2. Sets DSOBJECT to the created collection's handle (e.g. Collection-12)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## Move

### **Move <object-to-move-handle> <destination-handle>**

The Move command changes the location of an object from one container to another, generally in reference to files and collections.

Move File-456 Collection-123

*Environment results*

1. Sets DSHandle to the destination container's handle index (e.g. 12)
2. Sets DSOBJECT to the destination container's handle (e.g. Collection-12)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## NewVersion

### **NewVersion <FilePath> <FileHandle> ["summary=CommentText"]**

The NewVersion command uploads a local file given by FilePath as a new version of the DocuShare file specified by FileHandle. You can add version comments by giving a summary option. The command records the transaction result to the system registry. The CFaxess registry values are located under the key "HKCU\Software\Xerox\DocuShare Client\DsAxess\Console". The transacted handle is stored as DSHandle.

Newversion sets script variable DSHandle to the file handle of the updated file.

*Environment results*

1. Sets DSHandle to the document's handle index (e.g. 123)
2. Sets DSOBJECT to the document's handle (e.g. File-123)
3. Sets DSResultCount to 1 (the number of created versions)
4. Sets DSResult-\* to the document's handle (e.g. File-123)
5. Does not modify DSItemCount, DSItem-\*, DSSearchHitCount, DSSearchHit-\*
6. Note that neither the created Version's handle nor version index are returned.

## Open

### **Open <Handle> | [-regex <Title>]**

The Open command without parameters will open the current collection in use. To open a file, enter:

Open f<handle number>

Open f123

## Pause

### **Pause <seconds>**

The Pause command provides a convenient pause function without having to return to the calling program or script to provide that. A dialog will appear and remain until Resume is clicked to continue the script. Clicking Cancel will exit the script and return to the command console or the calling script or the Windows command prompt if run from the command line. Otherwise, it will pause for the number of seconds entered. It generally should not be used within a script unless a previous Put command needs time to index documents before a subsequent Search command is used on them.

Pause 30

## Prompt

### **Prompt [on/off]**

The Prompt command toggles on or off prompts for confirmation of deleting, uploading, or downloading files.

## Put

### Put [-type=<objectname>] <FilePath> ["prop1=value1" ...]

The Put command uploads to the current DocuShare collection one or more local files specified by the file mask LocalFilePath. A wildcard (such as c:\temp\\*.txt) can be used to Put more than one file. Wildcard characters are "\*" and "?".

By specifying a local directory, you can copy the directory and its contents in one step. Optional parameters for setting properties are applicable to a non-wildcard specification. You can use any of these combinations: "title=(display name)", "name=(filename)", "summary=", "description=", "keywords", "author=", and "object\_type=(MIME type)".

Put Scan001789.pdf "title=Invoice 0098734 from ABC Co.pdf" "summary=Payment due 06/06/2002"

The command records the transaction result to the system registry. The [DsAxess\Console] registry key is located in [HKCU\Software\Xerox\DocuShare Client]. The transacted handle is stored as DSHandle. The output handle is the handle of the (last) uploaded file.

The Put command can be modified by setting Uploadasnewdoc and Uploadclashprompt settings. Modifying these settings only affect the uploading of wild card or directories.

Put sets script variable DSHandle to the file handle for the created file. Put also updates script variables DSResultCount and DSResult-\* if Put uploads a directory.

#### *Enhancements*

CFAxess has added the ability to upload files as custom document objects. You can specify this with the -type=<objectname> argument. The DocuShare objectname is case sensitive.

Put -type=invoice Scan001789.pdf "title=Invoice 0098734 from ABC Co.pdf"

Two optional command line switches are added to the PUT command to support silencing a confirmation prompt and uploading a compound document.

#### Disabling Confirmation Prompt

When uploading multiple files, such as files that match a file name pattern, the -quiet option may be used to disable the confirmation prompt.

Put [file name pattern]

Put [file name pattern]

To upload all files that appear in folder c:\doc without prompting, the following command can be used:

Put -quiet c:\documents\\*.docx

### **Name Clash detection**

The CFAxess parameter, **uploadclashprompt**, is used to control the detection of a pre-existing Document object that has the same title as the file that is being uploaded. If an object with the same title is found, CFAxess prompts for a confirmation to replace the object. In versions 3.0 and 2.x of DSAxess, the detection was limited to bulk upload operations, such as uploading by a wild card and uploading of a directory. Version 3.1 of DSAxess can use the **uploadclashprompt** detection in a single upload operation as well. To enable it, set the new **extend\_uploadclashprompt** parameter to 1 using the **Setparam** command. By default, the parameter is set to 0, and the single upload detection mode is off. In this manner, a file is always uploaded as a new Document object.

#### *Enhancements*

Example 1: Uploading mail

PUT -mail <msg-path>

Use the mail switch to save an MS Outlook mail item (a file with the extension, MSG). The mail item is stored as a MailMessage object on DocuShare (not as a file object). You must have Outlook Client installed on your workstation for this feature to work.

msg-path is the pathname of a msg file. If msg-path is a filename without a directory specification, the command locates the file in the current directory.

You can use a wildcard (\* or ?) to upload multiple mail items. You are prompted to confirm the uploading of each item. The confirmation prompt can be turned off using the quiet switch.

This command quietly uploads all mail items with a RE subject line that are in the directory c:\mymail.

```
lcd c:\mymail
```

```
put -mail -quiet RE*.msg
```

#### *Environment results*

1. Sets DSHandle to the last created/modified object's handle index (e.g. 123)
2. Sets DSOBJECT to the last created/modified object's handle (e.g. File-123)
3. Sets DSResultCount to the number of created/modified objects (e.g. 2)
4. Sets DSResult-\* to the created/modified objects (e.g. DSResult-1="File-123", DSResult-2="File-456")
5. Does not modify DSItemCount, DSItem-\*, DSSearchHitCount, DSSearchHit-\*
6. Note that this is unlike "Get", which sets DSHandle/DSObject to the first retrieved object. Need to confirm this is the correct behavior.

## RemoveMember

**RemoveMember <groupHandle> <userOrGroupHandle> [<userOrGroupHandle> ...]**

The RemoveMember command removes one or more DocuShare users from a group. If you have one or more groups nested within another group you can remove them, too.

```
RemoveMember Group-29 User-15 User-38 User-22
```

To remove users and a sub-group at the same time from the first group you can use the command in this manner without regard for the order following the parent group being affected.

```
RemoveMember Group-29 User-15 User-38 Group-14 User-25
```

Refer also to AddMember, when useful.

## RemoveProps

**RemoveProps <Handle> <UserHandle | GroupHandle> [<UserHandle | GroupHandle> ...]**

The RemoveProps command removes all metadata from all properties of an object.

```
RemoveProps -quiet Document-987
```

## Rename

**Rename <Handle> "New Title Text"**

The Rename command renames a DocuShare object (ObjectHandle). The command records the transaction result to the system registry. The CFAXess registry values are located under the key "[HKCU\Software\Xerox\DocuShare Client\DsAxess\Console". The transacted handle is stored as DSHandle.

#### *Enhancements*

CFAXess has been extended so that you can rename custom objects in the same manner.

```
Rename invoice-2345 "Invoice 039884A from Indelible, Inc.pdf"
```

Rename sets script variable DSHandle to the handle number for the renamed object.

#### *Environment results*

1. Sets DSHandle to the renamed object's handle index (e.g. 12)
2. Sets DSOBJECT to the renamed object's handle (e.g. Collection-12)

3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## Replicate

**Replicate [-versions] <CollHandle> [<DestColl> [<DestServer>]]**

**Replicate [-versions] <DocHandle> [<DestColl> [<DestServer>]]**

The Replicate command copies a collection or document to another DocuShare server. If a collection handle is the source object, then the collection and all descendant collection and document objects are copied to the destination server.

The DestColl is the collection handle on the destination server where the collection or document should be copied to.

The DestServer is the destination server. It must first be connected to using a Login command. The DestServer is specified by name similar to how the SwitchTo command uses it. The SwitchTo -list command can be used to see the current server login sessions and the names for them.

When replicated, some properties are set by the destination server, most notably the creation date, the last modified date, and the object ACL (inherits from containing destination collection).

### Versions option

The -versions option has been added to the Replicate command. When enabled, the option causes the command to copy all versions of files that the command processes.

The short form, -v may be used instead of -versions.

## ResumeOnError

**ResumeOnError [on|off]**

ResumeOnError controls resumption of the script execution after CFAxess detects an error condition.

The statuscheck switch should be off. Otherwise, if a non-zero status code is generated, the script will quit without executing the rest of the script. Even with the statuscheck off, CFAxess aborts executing a script if it encounters an error.

## Search

**Search <"term1=value1" ...> [output=FilePath]**

The Search command searches DocuShare objects meeting criteria given by a combination of these terms:

max\_hits=(defaults to 100)

scope=(any collection handle or 'all'; defaults to current collection)

entity=File (see below) (defaults to all)

contains=not (to negate criteria)

created\_from=-10, created\_to=0 (find items created since 10 minutes ago)

modified\_from=-60, modified\_to=-30 (find items modified 30 to 60 min ago)

title=(title of item)

file=(file name)

object\_type=(MIME type)

owner=(owner of item)

summary=, description=, keywords=, author=, modified\_by= (only one of these may be specified)

The entity term can be set to one of the following object types: Collection, File, Calendar, BulletinBoard, User and Group.

The property identifier, title, can be omitted for a search by title. Thus, to search items whose titles include proposal, you can issue the command:

Search proposal

The command records the transaction result to the system registry. The [DsAcess\Console] registry key is located in [HKCU\Software\Xerox\DocuShare Client]. The transacted handle is stored as Handle. The output handle value is the number of found items.

Search sets script variable DSHandle to the number of found items. Search also updates script variables DSSearchHitCount and DSSearchHit-\*. Variable DSSearchHitCount contains the number of found items. Variable DSSearchHit-\* contains the handles for the found objects and uses them to enumerate the found items in a script.

The following script uses the Search command to find MS Word documents whose titles contain the phrase proposal, and archives them in the PC folder specified by system variable %temp%:

```
@echo off
let SearchTitle="proposal"
login
cd Collection-10
search title=%SearchTitle% entity=File object_type=application/msword
let ItemIndex=1
:StartLoop
if %ItemIndex% > %DSSearchHitCount% then
goto EndLoop
let %ItemHandle%="%DSSearchHit-%ItemIndex%"
if ItemHandle doesnotcontain File then
goto BumpIndex
getprops %ItemHandle%
get %ItemHandle% %temp%\%document%
:BumpIndex
let ItemIndex=ItemIndex+1
goto StartLoop
:EndLoop
```

#### *Environment results*

1. Sets DSHandle to the number of found hits (e.g. 7)
2. Sets DSSearchHitCount to the number of found hits (e.g. 7)
3. Sets DSSearchHit-\* to the search hits' handles (e.g. DSSearchHit-1="File-123", DSSearchHit-2="Collection-12", ...)
4. Does not modify DSOBJECT, DSItemCount, DSItem-\*, DSResultCount, DSResult-\*

## **SetAccess**

**SetAccess <ObjectHandle> [+|-]=readProps:<principalHandle>]**  
**[+|-]=readHistory:<principalHandle>] [+|-]=readContent:<principalHandle>]**  
**[+|-]=writeProps:<principalHandle>] [+|-]=writeContent:<principalHandle>]**  
**[+|-]=Manage:<principalHandle>] [Purge:<principalHandle>]**

**SetAccess <ObjectHandle> [+reader:<principalHandles>] [+writer:<principalHandles>]**  
**[+searcher:<principalHandles>] [+manage:<principalHandles>] [purge:<principalHandles>]**

The SetAccess command allows you to add or remove specific levels of permissions to an object. This can be very effective when intending to set specific levels of access for users and/or groups. The Purge argument is to remove a principal from the Access Control List (ACL) as if you remove all the grants for a

principal, they will still display in the ACL. The Manage option allows a user or members of a group to change access levels for others using the DocuShare web interface.




Note: SetAccess requires DocuShare 7.5 or higher.

The operators +, -, and = are for adding, removing, and overriding previous settings, respectively. + and - affect user permissions individually, but = replaces all user and group settings for the object so that a command stated as ...

SetAccess Collection-10 =readContent:User







... sets readContent exclusively to that user or group. Other users and groups with readContent access will have that setting removed. Following are common commands you can use showing their results.

Our starting point appears as:

Access List	User/Group	Read Properties	Read Content	Read History	Write Properties	Write Content	Manage
	 Crews, Doug (doug) CPX	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	 Content Administrators	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	 Document Management Systems	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>






Add users and a group in addition to setting their access rights.

setaccess File-48650 +ReadContent:User-557,User-558,Group-14 +ReadProps:User-557,User-558,Group-14

Access List	User/Group	Read Properties	Read Content	Read History	Write Properties	Write Content	Manage
	 Crews, Doug (doug) CPX	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	 Content Administrators	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	 Document Management Systems	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	 Flintstone, Fred (fflintstone) DS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	 Rock and Roll Hall of Fame	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	 Rubble, Barney (brubble) DS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Remove the Document Management Systems group.

setaccess File-48650 purge:Group-13

Access List	User/Group	Read Properties	Read Content	Read History	Write Properties	Write Content	Manage
	 Crews, Doug (doug) CPX	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	 Content Administrators	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
	 Flintstone, Fred (fflintstone) DS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	 Rock and Roll Hall of Fame	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	 Rubble, Barney (brubble) DS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

You can also view this using the following GetProps command noting the results that follow:

```
getprops File-48650 acl
acl=
"Fred Flintstone"(User-558): ReadContent ReadProps
"Barney Rubble"(User-557): ReadContent ReadProps
"Rock and Roll Hall of Fame"(Group-14): ReadContent ReadProps
"Content Administrators"(Group-2): ReadContent ReadProps ReadHistory WriteContent WriteProps
Manage
"Doug Crews"(User-31): ReadContent ReadProps ReadHistory WriteContent WriteProps Manage
```

This last user is the owner of this file and has full access.

When using SetAccess you should be cognizant of whether your DocuShare server uses 3 Level or 6 Level Permissions. This can be determined by logging in as the DocuShare Admin and looking in Admin Home > Site Configuration > Permissions. To change from 3 Level to 6 Level Permissions only requires that you check the box and click Apply. The transformation typically takes about a minute, but it may be best to do this when server usage is low. You cannot change back to 3 Level Permissions. You must use the proper syntax for each to avoid receiving an error message.

For 6-level ACLs:

```
setaccess <ObjectHandle> [+readProps:<principalHandles>] [+readHistory:<principalHandles>]
[+readContent:<principalHandles>] [+writeProps:<principalHandles>] [+writeContent:<principalHandles>]
[+manage:<principalHandles>] [purge:<principalHandles>]
```

For 3-level ACLs the command is restricted to the following:

```
setaccess <ObjectHandle> [+reader:<principalHandles>] [+writer:<principalHandles>]
[+searcher:<principalHandles>] [+manage:<principalHandles>] [purge:<principalHandles>]
```

```
logon admin
let doc = Document-48609
getprops %doc% acl
echo %doc%
setaccess %doc% +readobject:User-216
setaccess o/odoc% -readobject:User-216
let theGang = "User-43, User-47, user-35, user-46"
setaccess %doc% +readobject: %theGang% +readlinked:%theGang%
getprops %doc% acl
setaccess %doc% -readobject:Group-2,User-238 -readlinked:Group-2,User-238
getprops %doc% acl
setaccess %doc% purge:%theGang% purge:User-216
getprops %doc% acl
```



### Enhancements

This command did not exist in DSAccess.

## SetParam

### SetParam <name> <value>

The Setparam command assigns a new value to the named parameter. The modified names and their valid values are:

- server=(URL), auth=(cookie),
- xml=(0/1), txpacket=(128..0x10000), txtimeout=(1..300)
- downloadclashprompt=(0/1), uploadclashprompt=(0/1), uploadasnewdoc=(0/1)
- cacheexpireminutes=(0 for disabling directory cache)

The table lists parameters of GetParam/Setparam, their descriptions, and the units of measurement used by numeric parameters:

Parameters	Units/Type	Description
txpacket	bytes	Transmission buffer size
txtimeout	milliseconds	Transmission timeout
cacheexpireminutes	minutes	Gateway cache expiration time
server	--	Current DocuShare server home URL
auth	--	DocuShare session handle
ipaddr	--	Web server IP
hostname	--	Web server hostname
port	--	Web server port number
xml	boolean	Server supports XML API
serverreadyonly	boolean	Server in maintenance mode

### Environment results

1. Sets DSParam to the parameter's value
2. Sets ParamName to the parameter's name (this is a DSAccess registry entry)
3. Sets ParamValue to the parameter's value (this is a DSAccess registry entry)
4. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## SetProps

### SetProps <ObjectHandle> ["prop1=value1" ...]

The SetProps command modifies the specified properties of a DocuShare object (ObjectHandle). To change the the summary text to Meeting agenda, specify summary=Meeting agenda. Refer to GetProps for the list of available properties.

SetProps set script variable DSHandle to the handle number of the modified object. Use quotes when your metadata includes spaces.

SETPROPS File-48698 "title=Meeting Agenda for 8/23/2022" author=Doug keywords=BYOB

#### *Enhancements*

#### **File version properties**

The SetProps command can be used to modify properties of a file version that is specified by a file handle and a version number.

SETPROPS File-n/m <prop1=value1 ...>

Where File-n is the file handle, and m is the ordinal version number of a file version.

If you know the Version handle of a version object, use the version handle.

SETPROPS Version-n <prop-value list>

Use the property name, comment, to update the comment text of a version.

SETPROPS f123/3 "comment=Change in workflow process."

#### *Environment results*

1. Sets DSHandle to the queried/modified object's handle index (e.g. 123)
2. Sets DSOBJECT to the queried/modified object's handle (e.g. File-123)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## SwitchTo

#### **SwitchTo <servername> [<username>]**

SwitchTo allows you to have more than one DocuShare server referenced at the same time to make it easier to log into them and quickly address one from another. This can be useful for moving or copying content one DocuShare server to another. DocuShare Federation is not required, but accommodated. We will have the Replicate command added later to help facilitate copying content in batch mode.

To use SwitchTo you merely have to log into multiple DocuShare servers. CFaxess stores an alias for them for ease of switching. Here's an example for using three accounts among two DocuShare servers: the first one for the DocuShare admin, the second for another user with sufficient access permissions, and a third one for another user accessing a server connected to an LDAP or Active Directory server.

```
login https://finance.corpoffice.com/docushare admin password
login andy password https://finance.corpoffice.com/docushare
login linda password corpdomain https://sales.corpoffice.com/docushare
switchto Finance admin # Specify the user since there are 2 finance users logged in.
switchto Finance andy # Second finance logged in user.
switchto Sales # You can omit the user since there is only 1 sales user logged in.
```

#### *Environment results*

1. Sets DSHandle to the session's logged-in user's handle index (e.g. 32)
2. Sets DSOBJECT to the session's logged-in user's handle (e.g. User-32)
3. Does not modify DSItemCount, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## Tree

#### **Tree [<handle>] [-all] [-flat] [-depth=n] [max\_hits=n] [output=FilePath]**

The Tree command prints the current directory, including all subdirectories.

If `-flat` is specified, the output displays the hierarchical structure. If `-all` is specified, only collections are displayed. The depth level is infinite, unless explicitly specified. If `FilePath` is specified, `Tree` outputs the directory information to the specified file.

`Tree` sets script variable `DSHandle` to the handle number for the current collection.

### Optional line switches

The `Tree` command has additional optional command line switches that can be used.

- Optional switch `-flat` displays the names of found objects flush to the left edge of the display window. Do not perform the indenting based on an object's nest level.
- Optional switch `-all` lists objects of all types including Document and other non-Collection types. If the switch is not specified, the `Tree` command lists Collection objects only.

Note: The `-all` option does not support DocuShare workspaces.

The `-all` option directs the `Tree` command to use the DocuShare HTTP/XML PROPFIND command to generate a query request to DocuShare. This allows objects of all types to be reported. If the option is not used, the `Tree` command uses the DocuShare HTTP/XML SEARCH command, and queries only for objects of the Collection type that appear in and under the current Collection. The SEARCH-based Collection query can, by default, display up to 1,000 items. To increase the display limit, use the `max_hits` property. For example, the command, `Tree max_hits=4000`, lists up to 4,000 Collection names.

The display limitation of 1,000 items per query does not apply if the `-all` option is used. Since the display buffer used by `DSAccess` is limited to a maximum display of approximately 2,000 lines, all queried items may not be printed in practice. You may want to use the `-output` option to direct a large list to a file.

- Optional switch `-depth=n`—Limits the `Tree` query to the `n`th nest level. The switch works only when `-all` is specified.
- Optional switch `-output=filename`—Use this option to save the queried tree in a file. The filename can be a fully qualified pathname or a filename relative to the current directory previously set by the `LCD` command (refer to `LCD`).

### Enhancements

The enhancement enables listing of the versions and content elements of a docushare document object or enables listing of file attachments linked to a DocuShare mail message object.

The added functionality relies on the `<document_tree>` and `<mail_message_tree>` properties supported by DocuShare Release 4 and greater.

Note: The `Tree` command using `<object handle>` does not support Wikis and Weblogs.

To list the versions and content elements of a document, enter:

```
TREE <File-n>
```

To list the attachments of a mail message, enter:

```
TREE <MailMessage-n>
```

To print the versions and content elements of a file in a tree format, enter:

```
TREE File-31
Document Tree of File-31
- A4Pptest.rdo (File-31/1)
+-- A4Pptest.rdo (Rendition-46)
+-- A4Pptest.rdo (ContElem-1)
+-- 00000001.tif (ContElem-2)
+-- 00000002.!! (ContElem-3)
+-- LOCK (ContElem-4)
+-- totalsz (ContElem-5)
```

In the example, the file has one version with one rendition and five content elements.

This command lists the files attached to a mail message. The results follow.

```
tree MailMessage-225
Mail Message Tree of MailMessage-225
- AR8900_Test_Case_2.msg (Handle=File-2357, Type=NativeMessage)
```

- Attach-1\_DS\_4.0.1\_Locales\_Update.msg (Handle=File-2359, Type=File)
- Attach-2\_to\_DS\_4.0.1\_Locales\_Update.msg (Handle=File-2358, Type=File)
- Attach-3\_DS\_4.0.1\_Locales\_Update.msg (Handle=File-2360, Type=File)

The mail message above has three file attachments and one native-message attachment.

#### *Environment results*

1. Does not modify DSHandle, DSOBJECT, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## Unlock

### **Unlock <FileHandle>**

The Unlock command unlocks a file. The command records the transaction result to the system registry. The [DsAcess\Console] registry key is located in [HKCU\Software\Xerox\DocuShare Client]. The transacted handle is stored as DSHandle.

Unlock sets DSHandle to the unlocked file.

#### *Environment results*

1. Sets DSHandle to the locked/unlocked object's handle index (e.g. 123)
2. Sets DSOBJECT to the locked/unlocked object's handle (e.g. File-123)
3. Does not modify DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## UpdateSchema

The Updateschema command retrieves the schema from a DocuShare server and caches it for later use by GetProps and Replicate. To invoke the command:

Log into a server containing the schema to be updated.

Enter the name of the Updateschema command without any argument (no optional switches defined).

## WriteLog

### **Writelog <FilePath> [-truncate | -trunc | -erase]**

The Writelog command writes console text to a local file given by FilePath. If the directory contains spaces, then the FilePath must be in quotes. By default, it appends an existing file but you can tell it to erase the content of the previously used file by specifying -truncate or one of the aliases in the -trunc or -erase options. This is useful when permissions have been set whereas overwriting the file would change said permissions. Using the CLS clear screen command either in interactive or script mode can also affect the contents of the writelog output file.

The directory for the file path must be created; if the directory does not exist, DSAccess will terminate with a notice or warning.

#### *Environment results*

1. Does not modify DSHandle, DSOBJECT, DSItem-\*, DSResultCount, DSResult-\*, DSSearchHitCount, DSSearchHit-\*

## **Customer Support**

WiseTREND's Customer Support for CFAxess is available Monday through Friday from 8:00 AM to 6:00 PM CST via email to [ocrsupport@wisetrend.com](mailto:ocrsupport@wisetrend.com)

We welcome suggestions for improvement.