EMC² documentum

Documentum
Generic Scripting Interface

This document details a generic shell scripting interface for all Documentum servers.

Dave Robertson 2005
http://forums.contology.com/

Table of Contents

1 Introduction to DCTM-GSI

Documentum is large and complex document and content management system and offers a feature rich suite of products. One of its main selling points is that it can be easily customized. Its server product contains an extensive API (+150 methods) that provides access to all areas of the EDMS.

There are a number of existing and well defined programming interfaces including Docbasic, DFC, Visual Basic, C++ and Perl (Db::Documentum).

GSI offers an interface for Unix based shell scripting. Unix shell scripts are still a large part of any large Unix system and offer an easy way automate Unix tasks.

Thus with GSI you can write Documentum API commands directly into the shell. For example, if you had a task that wanted to check if a bunch of Docbases were running you could write a script that looks like this:

```
#!/bin/sh

# Init the shell
dmAPIInit

i=0

mapID=`dmAPIGet "getdocbasemap,c"`

numDocbases=`dmAPIGet "values,c,$mapID,r_docbase_name"`

while [ "$i" -le "$numDocbases" ]
do

      docbase=`dmAPIGet "get,c,$mapID,r_docbase_name[$i]"`
      i=`expr $i + 1`
      echo $docbase

done
```

A direct interface in the Unix shell. And it does not matter which one you use. (sh csh ksh bash tcsh zsh rc es) all will work with this interface on virtually any Unix platform and that includes Windows – using Cygwin.

2 Overview of the Interface

The interface consists of the same basic commands as the Documentum API defines. All operations are exactly the same. I have added two extra ones to initalise and deinit the session within a shell.

There is no point in Documentum what each of these commands do – I will assume you as the reader already know this and are itching to give it a go.

1. dmAPIInit() : This command is normally called at the start of the shell to initiate a session and allow other API calls to be made. If its not made, nothing else will work.
2. dmAPIGet(); This command maps to the dmapiget API executive found in Docbasic etc. To use it within a shell you use the following format.

   ```
   mapID=`dmAPIGet "getdocbasemap,c"`

   or

   DocName=`dmAPIGet "get,c,$ObjID,object_name"`
   ```

   Note the way the quotes go round the command. You will need to do this if you assign the result to an environment variable – otherwise you can just use:

   ```
   dmAPIGet "getdocbasemap,c"
   ```

   You will need to use the double quote character because of the way arguments are passed to the command.
3. dmAPISet(): This is the command to set attributes. Same rules as dmAPIGet except the format would be:

   ```
   result=`dmAPISet "set,c,$ObjID,object_name" "A Name of Some Consequence"`
   ```

   Again note the quotes to establish a second argument to the command

   The command will return 0 or 1 – o for failure, 1 for success.
4. dmAPIExec(): The final API command which performs an Execute function.

   ```
   result=`dmAPISet "fetch,c,$ObjID"`

   or simply - dmAPISet "fetch,c,$ObjID"
   ```

5. dmAPIShellExit(): This is called without arguments and ensures the session is closed. If your code does a disconnect,c then that's going to work as well.

3 Example

The following is an example of a script that could be run as a simple shell to alert users if they have any queued items in their inboxes.

```
#!/bin/sh
# checker.sh
# (c) 2005 Contology Ltd

echo "\nGSI Inbox Checker Number 2\n"

dmAPIInit

DOCBASE="GEL_Dev_TA_2a"
USER=dmadmin

SESSION=`dmAPIGet "connect,$DOCBASE,$USER,XYZ"`

if [ -z "$SESSION" ]; then
      echo "Cannot connect to Docbase"
      exit 1
fi

# define SQL for counting
DQL="SELECT * FROM dmi_queue_item"
# do duery
col_id=`dmAPIGet "query,$SESSION,$DQL"`
echo Col ID:"$col_id"END

# if query successful
if [ ! -z "$col_id" ];  then

      # loop through collection and print count
      res=`dmAPIExec "next,$SESSION,$col_id"`

      while [ "$res" != "0" ]; do

            qname=`dmAPIGet "get,$SESSION,$col_id,name"`
            qitem=`dmAPIGet "get,$SESSION,$col_id,item_name"`
            qmess=`dmAPIGet "get,$SESSION,$col_id,message"`
            echo $qname $qitem $qmess
            res=`dmAPIExec "next,$SESSION,$col_id"`

      done
      res=`dmAPIExec "close,$SESSION,$col_id"`
      echo "You have $count items in your inbox.\n"
else
      echo Failed......
      dmAPIGet "getmessage,$SESSION"
fi

res=`dmAPIExec "disconnect,$SESSION"`
```