IBM eServer™

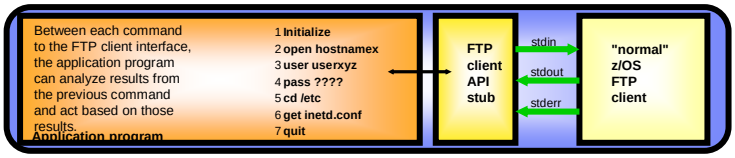
Applications: FTP Client API in REXX

ebusiness on demand software

FTP client API in REXX

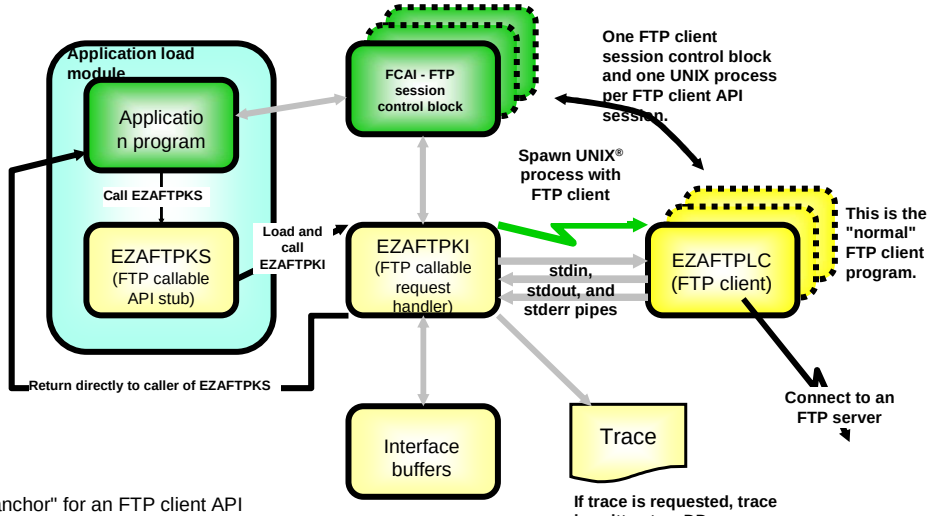
- **z/OS® V1R6 provided the initial callable FTP client programming interface**
  - This initial version was provided as a callable program interface to be used from Assembler, Cobol, or PL/I application programs
- **z/OS V1R7 extended that FTP client programming interface with a C library interface**
  - To be used by C or C++ application programmers
- **z/OS V1R8 further extends the FTP client programming interface support by providing a REXX API**
  - To be used by REXX application programmers
  - TSO Interactive or batch, ISPF, USS Shell
  - REXX programs can be compiled for optimum performance

```
/* Create FTP client control information    */
if ftp('create','fcai.', TRACEID) < 0 then do
   Say 'Unable to create the FCAI'
   exit
end
/* Enable trace                             */
if ftp('fcai.', 'set_trace', 'ON') < 0 then do
   call ftp_error 'fcai.'
end
/* Open a connection                        */
if ftp('fcai.', 'init', OPENSTRING, VAR1, VAR2) then do
   call ftp_error 'fcai.'
end
/* Send USER command                        */
if ftp('fcai.', 'scmd', USER_COMMAND, 'W') < 0 then do
   call ftp_error 'fcai.'
end
/* Send password                            */
if ftp('fcai.', 'scmd', PASS_COMMAND, 'W') < 0 then do
   call ftp_error 'fcai.'
end
```



Signifantly improved automation capabilities for file transfer operations that are initiated on z/OS

Structure of the FTP client API implementation



The "anchor" for an FTP client API "session" is the FCAI control block. In the REXX version of the FTP client API, the FCAI is represented by a REXX stem variable as the first parameter on all calls.

REXX FTP client API request format

- **REXX FTP Client API request format:**

```
result = ftpapi(stem, request_type, parm1, parm2, ...)
```

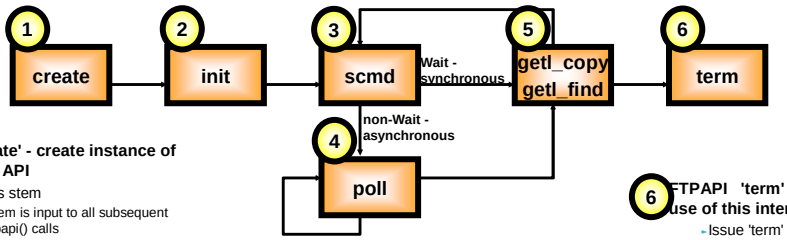
- **'STEM' - name of the REXX stem**
  - refers to a specific instance of the FTP client environment.
- **request_type - the API request that is being made**
  - 'init', 'scmd', 'term', and so on
- **parm1, parm2, ...**
  - Depends on the REXX FTP API request.
- **REXX FTP Client API return code format:**
  - <0: An error ocurred
  - 0: OK
  - >0: OK, and the return code value gives additional information.

```
rc = ftpapi('fcai.', 'scmd', 'DIR /tmp/*', 'N')
rc = ftpapi('fcai.', 'term')
```

REXX FTP client API request_type summary

- **create** - Creates a new FTP client API stem variable
- **init** - Initializes an FTP client API environment
- **scmd** - Send a subcommand to the FTP client API environment
- **poll** - Complete and retrieve the results from a prior SCMD request
- **getl_find** - Find and return a line of output that was returned from the FTP client by an INIT, SCMD or POLL request
- **getl_copy** - Find and return all lines of output that was returned from the FTP client by an INIT, SCMD or POLL request
- **term** - The user program issues the term request to terminate this instance of interface use
- **set_trace** - Enables or disables the tracing of subsequent FTP Client API calls
- **set_request_timer** - Sets the amount of time that the FTP client API waits for an init, scmd or term request to complete
- **get_fcai_map** - Gets the contents of the FCAI_Map structure

REXX FTP client API program structure



1 ftpapi 'create' - create instance of FTP Client API
- creates stem
  - stem is input to all subsequent ftpapi() calls
- creates predefined REXX variables
  - for application's use

2 ftpapi 'init' - initialize FTP Client API environment
- ftp client command line parameters can be passed
- up to nine environment variables can be passed

3 ftpapi 'scmd' - send a subcommand to the FTP client
- wait or nowait mode

4 ftpapi 'poll' - complete and retrieve results of prior 'scmd'
- returns FCAI_RESULT_INPROGRESS until 'scmd' completes

5 FTPAPI GETL_COPY - Find and return lines of output from the FTP client

Filter input by specifying type of output
- M Message from the client.
- R Reply from the server.
- L List data from a DIR or LS subcommand.
- T Trace output from debug or dump.
- A Any type of output.

Output is in a stem you specify
- lines.0 number lines found
- lines.id.i - type of line (M, R, L, T)
- lines.i - a line of output

5 ftpapi 'getl_find' - Find and return a single line of output from the FTP client
- Like ftpapi 'getl_copy'
  - returns one line
- Specify which line you want:
  - first, last, or next line

6 FTPAPI 'term' - Terminate use of this interface
- Issue 'term' for every 'init'

Sample application

- **REXX sample: hlq.SEZAINST(EZAFTPIR)**
  - This sample:
    - issues REXX ftpapi() calls to
      - login to ftp server
      - issue a dir subcommand in nonblocking mode
      - parse the output and find the largest file in the file list
      - logoff ftp

The Shipped Sample Application

NOTES

```
/* REXX */
/*********************************************************************/
/*                                                                   */
/* Part Name:          EZAFTPRIR.REX                                 */
/*                                                                   */
/* Descriptive Name:   REXX FTP Client API Sample                    */
/*                                                                   */
/* Copyright:    Licensed Materials - Property of IBM                */
/*               5694-A01                                            */
/*               (C) Copyright IBM Corp. 2006                        */
/*                                                                   */
/* Status:             CSV1R8                                        */
/*                                                                   */
/*                                                                   */
/* Function:           Sample FTP Client API for REXX language       */
/*                                                                   */
/*                     This sample does a directory list of the      */
/*                     home directory.  For a z/OS FTP server, this  */
/*                     may be the the user's home directory in HFS   */
/*                     or the user high level qualifier, depending   */
/*                     on whether the server FTP.DATA has            */
/*                     STARTDIRECTORY set to HFS or MVS™.            */
/*                                                                   */
/*                     It then does a directory search of /tmp and   */
/*                     finds the largest file in /tmp.               */
/*                                                                   */
/*********************************************************************/
```

The Shipped Sample Application (cont.)

NOTES

```
/*********************************************************************/
/* Setup the constants for the FTPAPI invocations                    */
/*********************************************************************/
TRACEID      = 'PAZ'
USER_COMMAND = 'user user1'
PASS_COMMAND = 'pass tcpsup'
DIR_COMMAND  = 'dir *'
TMP_COMMAND  = 'DIR /tmp/*'
OPENSTRING   = '-w 300 127.0.0.1 21 '
ENVVAR1      = '_CEE_DMPTARG=/tmp'
ENVVAR2      = '_BPX_JOBNAME=MYJOB'


/*********************************************************************/
/* Initialize the FCAI stem.                                         */
/*********************************************************************/
if ftpapi('fcai.', 'create', TRACEID) < 0 then do
   Say 'Unable to create the FCAI'
   exit -1
end
```

The Shipped Sample Application (cont.)

NOTES

```
/*********************************************************************/
/* Turn on tracing of the Client API tracing.  This is a different   */
/* trace from the REXX FTP Client API trace and is always written    */
/* to SYSOUT.                                                        */
/*********************************************************************/
if ftpapi('fcai.', 'set_trace', 'ON') < 0 then do
   call ftp_error 'fcai.'
end
/*********************************************************************/
/* Initialize the FTP client environment and open a connection       */
/* to the FTP server using the OPENSTRING.  Two environment          */
/* variables are also provided to the FTP Client API to be used      */
/* when intitializing its environment.                               */
/*********************************************************************/
if ftpapi('fcai.', 'init', OPENSTRING, ENVVAR1, ENVVAR2) < 0 then do
   call ftp_error 'fcai.'
end

/*********************************************************************/
/* Sign on                                                           */
/* Enter the userid.                                                 */
/*********************************************************************/
if ftpapi('fcai.', 'scmd', USER_COMMAND, 'W') < 0 then do
   call ftp_error 'fcai.'
end
```

The Shipped Sample Application (cont.)

NOTES

```
/*********************************************************************/
/* If the FTP server prompts for a password, provide one.  In some   */
/* instances, an FTP server may not prompt for a password (e.g.,     */
/* when an anonymous user logs on to some FTP servers).              */
/*********************************************************************/
if fcai.FCAI_Result =  FCAI_RESULT_PROMPTPASS then do
   if ftpapi('fcai.', 'scmd', PASS_COMMAND, 'W') < 0 then do
      call ftp_error 'fcai.'
   end
end

/*********************************************************************/
/* List directory entries                                            */
/* Enter a subcommand DIR_COMMAND to retrieve a listing of all files */
/*    in the directory.  The REXX program requests the FTP client    */
/*    API wait for the subcommand to complete before returning.      */
/*********************************************************************/
if ftpapi('fcai.', 'scmd', DIR_COMMAND, 'W') < 0 then do
   call ftp_error 'fcai.'
end
```

The Shipped Sample Application (cont.)

NOTES

```
/*********************************************************************/
/* Fetch the lines returned by the DIR subcommand.                   */
/*********************************************************************/
if ftpapi('fcai.', 'getl_copy', 'lines.', 'L') < 0 then do
   call ftp_error 'fcai.'
end
/*********************************************************************/
/* Display the results of the output.  lines.0 contains the total    */
/* number of lines returned, while lines.1...lines.n contains the    */
/* output for each individual line.                                  */
/*********************************************************************/
say 'Directory output is:'
do i=1 to lines.0
   say '  'lines.i
end
/*********************************************************************/
/* Find the largest file in /tmp                                     */
/* Do a DIR of /tmp in non-wait mode, find the largest file in the   */
/*    directory.  The FTP client will return immediately, even if    */
/*    the subcommand has not completed.  The POLL request is then    */
/*    used to determine when the subcommand has completed.           */
/*********************************************************************/
if ftpapi('fcai.', 'scmd', TMP_COMMAND, 'N') < 0 then do
   call ftp_error 'fcai.'
end
```

The Shipped Sample Application (cont.)

NOTES

```
/*********************************************************************/
/* Poll to see when the DIR is done.                                 */
/*                                                                   */
/* If running under USS/OMVS, make sure that syscalls are not        */
/* enabled prior to making the first CREATE request.  Enabling       */
/* syscalls makes the USS/OMVS environment variables no longer       */
/* available to the REXX program, so the REXX FTP Client API would   */
/* be unable to read the FTP_REXX_TRACE_FILE environment variable    */
/* (if provided) once syscalls are enabled.                          */
/*********************************************************************/
call syscalls "ON"                /* Enable SYSCALLS                 */
if ftpapi('fcai.', 'poll') < 0 then do
   call ftp_error 'fcai.'
end
do while fcai.fcai_result = FCAI_RESULT_INPROGRESS
   address syscall 'sleep 20'
   if ftpapi('fcai.', 'poll') < 0 then do
      call ftp_error 'fcai.'
   end
end
```

The Shipped Sample Application (cont.)

NOTES

```
/*********************************************************************/
/* find the largest file                                             */
/*    Each line of output of a DIR command appears as:               */
/* drwxrwxrwx cnt userid groupid size date file                      */
/*********************************************************************/
largest_file = ""
largest_file_size = 0
if ftpapi('fcai.', 'getl_find', 'lines.', 'L', 'F') < 0  then do
   call ftp_error 'fcai.'
end
/*********************************************************************/
/* Continue to search until there are no more lines.  When there     */
/* is no matching line, the REXX FTP Client API returns a result     */
/* of FCAI_RESULT_NOMATCH and sets lines.0 to 0.                     */
/*********************************************************************/
do until lines.0 = 0
   parse var lines.1 dirinfo count userid groupid size date file .
   if size > largest_file_size then do
      largest_file = lines.1
      largest_file_size = size
   end
   if ftpapi('fcai.', 'getl_find', 'lines.', 'L', 'N') < 0 then do
      call ftp_error 'fcai.'
   end
end
```

The Shipped Sample Application (cont.)

NOTES

```
if largest_file <> ""  then do
   say 'Characteristics of the largest file are'
   say '  'largest_file
end
else do
   say 'No files found'
end
/*********************************************************************/
/* Enter the QUIT subcommand and  terminate the connection.          */
/* It's better to enter the QUIT subcommand before entering the      */
/*    ftpapi('term') command.   In this way, any error traces can be */
/*    displayed.                                                     */
/*********************************************************************/
if ftpapi('fcai.', 'scmd', 'QUIT', 'W') < 0 then do
   call ftp_error 'fcai.'
end
/*********************************************************************/
/* Enter the TERM request to close this instance of the FTP client   */
/* API.                                                              */
/*********************************************************************/
if ftpapi('fcai.', 'term') < 0 then do
   Say "Unexpected error on ftpapi('term')"
   exit -1
end
exit
```

The Shipped Sample Application (cont.)

NOTES

```
/*********************************************************************/
/* Generic error handling routine for the FTP interface.  This       */
/* should only be called with a valid FCAI.                          */
/*********************************************************************/
ftp_error: arg stem
   say 'FTP Client API Error:'
   say '    Result     =' value(stem'FCAI_Result')
   say '    Status     =' value(stem'FCAI_Status')
   say '    IE         =' value(stem'FCAI_IE')
   say '    CEC        =' value(stem'FCAI_CEC')
   say '    ReturnCode =' value(stem'FCAI_ReturnCode')
   say '    ReasonCode =' value(stem'FCAI_ReasonCode')

   rc = ftpapi('fcai.', 'term')
   exit -1

   return
```

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM MVS z/OS

UNIX is a registered trademark of The Open Group in the United States and other countries.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.