QF_Macro

M Andre Eckenrode

COLLABORATORS

	TITLE : QF_Macro		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	M Andre Eckenrode	July 10, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

QF_Macro

1.1 Contents

QuickFile v3.25 Macro Reference

1.2 overview

QuickFile responds to macro commands from ARexx file and inline macros, and from direct macros (individual QuickFile commands invoked from within the program, which are not sent to ARexx). The following options are available:

File Inline Single

Option Macros Macros Commands

========================== ======== ======== ========

AREXX MACRO requester Yes Yes No

DIRECT MACRO requester Yes No Yes

Function Key configuration Yes No Yes

The default filename extension for QuickFile macros, with the exception of startup macros, is ".quickfile". Startup macros are run automatically when a database is opened, and must be saved with the name "<databasename>.startup" in the database's home directory.

For the first copy of QuickFile run, the ARexx port name is 'QUICKFILE.01'. The number is incremented by one for each additional copy run. As QuickFile can open mulitiple databases, each in its own window, there should be no need to run multiple copies, unless the user wishes to run a macro for one database while working on another. The address is automatically set for macros launched from within QuickFile, so the port name need be specified only if the macro is launched from outside QuickFile (Shell, or another application), or an address change is required within the macro.

ARexx need not be available for direct macros. However, a number of the commands, especially those that are informational, have limited or no use when issued this way.

Not all QuickFile functions are available via macro commands. Contact the author if additional functions are desired.

1.3 Syntax Notes

QuickFile expects each command to consist of a command name followed by one or more parameters (where applicable), separated by spaces. Any parameter containing spaces must be enclosed in quotes. In file and inline macros, two sets of quotes are required, as ARexx removes the outer set of quotes before the command line is sent back to QuickFile. When individual commands are issued directly (via the DIRECT MACRO requester, or the function keys), only one set of quotes is required. For example, to pass the following command to QuickFile:

ReqMsg ’This is a demo message’

the ARexx code required is

ReqMsg ’"This is a demo message"’

Both ARexx and QuickFile accept either single- or double-quote marks. If parameters containing actual quote marks are required, for example to set the value of a CHARACTER field to a string that has quote marks in it, they can be included by entering a backslash followed by a three-digit ASCII character value, as in ’\034’ for double-quote marks and ’\039’ for single-quote marks. Otherwise, strings may be truncated at the first embedded quote mark encountered, depending on the order of single- and double-quote marks used. Some examples:

reqmsg ’"I can’t take this"’

>>> Invalid argument to function

reqmsg ’"I can”t take this"’ /* This will work */

>>> I can’t take this

reqmsg "’I can’t take this’" /* Order of outer quote marks reversed */

>>> I can

reqmsg ’"I can”t say ""Boo""."’

/* or */

reqmsg ’"I can”t say’ d2c(34)’Boo’d2c(34)’."’

>>> I can’t say

reqmsg ’"I can\039t say \034Boo\034."’

>>> I can’t say "Boo".

To enter parameters containing literal backslashes and digits, insert an extra backslash before them. Example:

reqmsg ’"The code required for double-quote marks is \034\\034\034."’

>>> The code required for double-quote marks is "\034".

The ’\xxx’ ASCII code method also allows one to enter any other characters as well, including tabs (’\009’), linefeeds (’\010’), etc., which may also be entered using ARexx functions, such as d2c(9), d2c(10), etc., in any case. Such characters can even be placed into fields using the PUTFIELD command, although there is not necessarily any advantage to doing so.

Extra quotes should also be placed around variable names passed to QuickFile commands as parameters if the variable values could contain spaces. For example:

pull filename

address quickfile.01 openfile ’"’filename’"’

If filename was entered as "My file" this would result in the following being (correctly) passed to quickfile:

openfile "My file"

If the two sets of quote marks had been omitted this would have caused an error.

For simplicity, spaces in field names should be avoided. While they are not illegal, the GETREC command, which uses field names to form compound variables, cannot be used with them.

Care should also be taken with variable values that might contain embedded quote marks, since the marks need to be translated to their respective '\xxx' ASCII codes prior to being passed to QuickFile commands or risk being truncated or interpreted incorrectly. The included external ARexx function macro FixQuotes.quickfile may be used to test and translate such variable values as necessary.

Note also that AmigaDOS paths and filenames can contain single-quote marks, and that QuickFile allows field, view and index names with single or double quote marks in them, and happily saves the index and view names to disk that way.

QuickFile processes macro commands regardless of case. Case is preserved in field values.

Confusion with case is a common problem with ARexx. ARexx always converts tokens to upper case unless they are quoted. A common trap is that the PULL and ARG instructions translate to upper case as they are equivalent to PARSE UPPER PULL and PARSE UPPER ARG. To preserve case, use PARSE PULL and PARSE ARG instead.

1.4 Caveats and Limitations

Modifying Index Fields in All Records

Anytime the value of a field that is used in the current index is modified in an existing record, whether via macro or manually, the record may be re-positioned within the index as soon as it is updated. If a macro is to modify the same index field in all records in the current index, there is a potential that some records are modified more than once and others not at all, due to records being re-positioned within the index upon update. Care should be taken to switch to an index that does not use the field to be modified, or to sort the records in a sequence based on other fields.

The included external function macro ReSort.quickfile may be used to test whether fields to be modified by a macro are used in the current index, and automatically re-sort records, using a field that is not being modified by the macro, when necessary. Note that if all existing database fields are being modified by a macro, there will be no unused field to re-sort with.

Using SAY and ECHO in QuickFile macros

QuickFile sets an error code of 12 (Unknown command) when the ARexx instructions SAY or ECHO are issued in macros launched from within the program if QuickFile was started from Workbench, and no output is displayed ordinarily. There are three workable remedies to this:

1) Include the following code in macros requiring an output window to open only if there is actual output:

call open('STDOUT','con:<x>/<y>/<w>/<h>/<title>/WAIT/AUTO','w')

Or, if it is desired that an output window opens whether needed or not, eliminate '/AUTO' from the line above.

2) Open any console window using OPEN(), as above, and use the ARexx functions WRITELN() and WRITECH() instead of SAY and ECHO.

3) Start QuickFile from a Shell, or from the Workbench "Execute Command..." menu item. This allows the Shell window to be used for output, or a standard ouput window to open as needed, respectively.

1.5 Command Notation Description

The following conventions are used in the command descriptions:

Command names use Initial Capitals eg GetRec;

Values in square brackets [ ] are optional;

One of the values in curly braces {A|D} is required;

Repeated values are indicated by ellipsis ...;

Keywords are shown in UPPER CASE;

<enclosed lower case words> represent user-provided values.

1.6 General Return Codes

The return codes from QuickFile have the following general meanings. See the command descriptions for details:

5 Warning or object not found.

10 Command failed

12 Unknown command

15 System error - usually memory allocation error

20 Syntax error

1.7 commands

AddSearch {AND|OR} <field> <operator> <value1> [<value2>]

Call <macro> [<arguments>]

CloseWin

ClrRec

CycleToString <cyclefield> <string#>

DelRec

DoSearch [SELECTED]

FreeIndex

FreeView

GetField <field>

GetRec <stemname>

GoTo <value1> [<value2>...]

InsRec

LoadView <view>

NewSearch <field> <operator> <value1> [<value2>]

Next [<number>]

NumRecs

OpenFile <database>

OpenWin

PutField <field> <value>

Query {CYCLE|FIELD|FILE|INDEX} <stemname> [<item>]

Refresh

Report <count> [<target> [<title>]]

ReqChoice <message1> [<message2>]

ReqField [NOCALC]

ReqMsg <message>

ReqString [<default> [<message>]]

SaveFile

SelectRec

SetFile [<database>]

SetIndex [<index>]

SetView [<view>]

ShowAll {ON|OFF}

Sort <field1> {A|D} [<field2> {A|D}...]

StringToCycle <cyclefield> <string>

UpdRec

WintoFront

1.8 addsearch

AddSearch {AND|OR} <field> <operator> <value1> [<value2>]

Adds a new search criteria to the existing criteria. Use the NEWSEARCH command to replace any existing criteria and DoSearch to start the search.

{AND|OR} Determines how multiple criteria are to be handled; whether all (AND) or any (OR) criteria are to be matched.

All other parameters are the same as for NEWSEARCH.

Result

None

Return codes

10 Unknown <field> or <operator>

20 Insufficient parameters or first parameter is not AND or OR

Example

/* select records where surname is brown or smith */

/* Use two sets of quotes around field names that have spaces in them */

newsearch '"last name"' equal 'brown'

addsearch or 'surname' equal 'smith'

dosearch

1.9 call

Call <macro> [<arguments>]

Runs the specified <macro> and passes the optional <arguments> to it. The filename extension of <macro> need not be specified if it is '.quickfile'.

If no path is specified, QuickFile looks for <macro> in the ARexx directory within the current directory (usually QuickFile's home directory). If a relative path is specified, the search is conducted relative to the current directory, not the ARexx directory. A "Macro not found" requester is displayed if <macro> cannot be located.

Note that the CALL command itself MUST be quoted if used within a file or inline macro, since ARexx has a similar command with the same name.

Result

None

Return codes

20 <macro> not specified

Examples

/* From inside a file macro */

’call demo’

’call "test macro" "Here is a line of text."’

[Configured for function keys]

call fkey f1

call ’rexx:open console.rexx’ ’con:20/30/300/150/QFConsole/auto/wait’

1.10 closewin

CloseWin

Closes the current window and database; however, the last window cannot be closed using this command. The first window in QuickFile’s internal list becomes the new current window.

Result

None

Return codes

5 An attempt was made to close the last window.

Example

closewin

1.11 clrrec

ClrRec

Clears all fields in the current record. Use this to clear all fields before setting up the values for a new record.

Result

None

Return codes

None

Example

clrrec

/* Use two sets of quotes around field names that have spaces in them */

putfield ’"last name" Brown’

putfield ’"first name" John’

putfield address ’"23 George Street"’

insrec

1.12 CycleToString

CycleToString <cyclefield> <string#>

Returns the string corresponding to <string#> for <cyclefield>. Numbering starts at 0. The string used for the current record is not changed. See also StringToCycle .

Result

String corresponding to <string#> in <cyclefield>

Return codes

5 <string#> out of range for <cyclefield>

10 <cyclefield> unknown or not a cycle field

20 Parameter missing

Example

/* Given a cycle field named "Status" with values "Active,Pending,Cancelled" */

cycletostring ’status’ 2

reqmsg result

>>> Cancelled

1.13 delrec

DelRec

Deletes the current record. The next record becomes the new current record.

Result

None

Return codes

5 There are no records in the file.

Example

getfield ’expirydate’

if result < today then delrec

1.14 dosearch

DoSearch [SELECTED]

Searches through all records in the database, or only currently selected records if the SELECTED keyword is used, and selects records matching the search criteria. The criteria are established using the NewSearch and ADDSEARCH commands.

The matching records are placed in a temporary index named SELECTED, which becomes the current index. The display is not updated until the macro completes unless the REFRESH command is issued.

Result

The number of records matched.

Return codes

None

Example

/* Use two sets of quotes around field names that have spaces in them */

newsearch '"last name"' equal 'brown'

addsearch or '"last name"' equal 'smith'

dosearch

reqmsg '"'result 'records matched"'

refresh

1.15 freeindex

FreeIndex

Discards the current temporary index and frees the memory used. Only temporary indexes, ie those named SORTED and SELECTED, may be freed.

Result

None

Return codes

10 Not a temporary index

Example

setindex selected

freeindex

1.16 freeview

FreeView

Unloads the current view from memory. The first view in the loaded views list becomes the current view. The display is redrawn. At least one view must be retained. The disk file containing the freed view is not affected.

Result

None

Return codes

5 An attempt was made to free the last view

Example

setview mylist.view

freeview

1.17 getfield

GetField <field>

Returns value of <field> in the current record. See also GetRec which obtains all fields for the current record.

Result

Value of <field> in current record

Return codes

10 Field requested does not exist

20 Field name omitted

Example

getfield ’surname’

>>> Smith

1.18 getrec

GetRec <stemname>

Returns values for all fields from the current record in compound variables in the form of <stemname>.<field>.

Note that this command cannot be used if any <field> name contains spaces; use GetField instead.

Result

<stemname>.<field> variables updated.

Return codes

20 Syntax error. stem not provided.

Example

next

getrec val

reqmsg ’"Name =’ val.firstname val.surname’"’

1.19 goto

GoTo <indexvalue1> [<indexvalue2>...]

Positions the database window on the first record with index field value(s) matching the corresponding <indexvalue(s)>, or on the next closest record. The display is not updated until the macro completes unless the REFRESH command is issued.

QuickFile does not report if the requested record was not found or not. It happily positions on the next record if there are no matches. The record must be compared with the requested key values to determine if they were found.

Result

None

Return codes

5 End of file encountered. Requested <indexvalue> higher than any record.

10 More <indexvalues> specified than index fields

Example

goto ’"Brown" "John"’

getrec val

if val.surname ~= "Brown" | val.firstname ~= "John" then reqmsg "John Brown not found"

1.20 insrec

InsRec

Inserts a new record. PutField should be used to set desired field values before this command is used. All fields not set with PUTFIELD retain their values from the current existing record, unless all fields are cleared using ClrRec first.

Result

None

Return codes

10 Error occurred. Probably duplicate key.

Example

clrrec /* omit this to use values from previous record */

putfield ’"surname" "Brown"’

putfield ’"firstname" "John"’

putfield ’"address" "23 George Street"’

insrec

1.21 loadview

LoadView <view>

Loads <view>, but does not make it the current view. The full filename including extension must be specified. A path must also be specified if the file does not reside in the current database’s home directory. Note that if a path is specified, it is also displayed in the SELECT VIEW requester with the name, and will have to be specified also when using SetView .

Result

None

Return codes

10 <view> could not be located

Example

loadview ’Product.View’

setview ’Product.View’

1.22 newsearch

NewSearch <field> <operator> <value1> [<value2>]

Discards any existing search criteria and begins specification of a new one. AddSearch is used to add additional criteria if desired, and DoSearch begins the actual search.

<operator> One of the following search operators:

LIKE

EQUAL

BETWEEN

NOTLIKE

NOTEQUAL

SOUNDS

<value1> Primary search value

<value2> Secondary search value; required for BETWEEN searches,

optional with other operators

Result

None

Return codes

10 Unknown <field> or <operator>

20 Insufficient parameters

Example

/* find all current memberships that expire in March */

newsearch ’status’ equal ’current’

addsearch and ’"expiry date"’ between ’01-mar-95’ ’31-mar-95’

dosearch

1.23 next

Next [<number>]

Moves <number> records forward or backward through the database. 1 is assumed if <number> is omitted. If the resulting position is outside the range of the database, the new position is the first or last record. <number> can be positive or negative, but must be quoted if negative.

Result

None.

Return codes

5 No more records. Positioned at first or last record

Example

numrecs

recs = result

soi = -1*result

next ’"’soi’"’ /* move to start of index */

next /* move to second record */

1.24 numrecs

NumRecs

Returns number of records in the current index. If SELECTED is the current index, this is the number of selected records.

Result

Number of records in the current index

Return codes

10 No file open

Example

SetIndex ’name’

numrecs

reqmsg ’"’result ’records in index \034Name\034"’

SetIndex ’selected’

numrecs

reqmsg ’"’result ’selected records"’

1.25 openfile

OpenFile <database>

Opens file <database> in the current window. Closes the currently open database, if any. To open additional databases in separate windows, the OPENWIN command must first be issued.

Result

None

Return codes

10 <database> could not be opened. Any current database is still

closed.

20 <database> not specified

Example

/* Use two sets of quotes around file names that have spaces in them */

openfile ’"ram disk:Membership"’

1.26 openwin

OpenWin

Opens a new empty window and makes it the current window. Until a database is opened in the new window, the only commands that can be successfully issued are OpenFile , CloseWin and SetFile .

Result

None

Return codes

10 The window could not be opened

Example

pull filename

openwin

/* Use two sets of quotes in case file names have spaces in them */

openfile ’"’filename’"’

1.27 putfield

PutField <field> <value>

Sets the contents of <field> to <value> for the current record or a new record. The change for an existing record does not take effect until the UPDREC command is issued. A new record is inserted using the INSREC command after all desired fields have been set.

<value> must be consistent with <field>'s type, and must be enclosed two set of quotes if it contains spaces. For cycle fields, <value> should be a string matching one of the values defined for <field>. Matching is not case sensitive. Note that INTEGER and TIME fields may be set to certain out-of-range values without error using this command. See the field type descriptions.

Result

None

Return codes

5 <value> is inconsistent with <field>'s type, or out of range for a

numeric <field>

10 <field> does not exist

20 Either <field> or <value> were omitted

Example

next

newdate = '31-Mar-1996'

putfield '"expiry date"' newdate

updrec

1.28 query

Query {CYCLE|FIELD|FILE|INDEX} <stemname> [<item>]

Obtains details about fields, CYCLE fields, indexes, and opened databases.

<stemname> A stem variable to receive the details

<item> Optional name of a field, database or index, or required

name of a CYCLE field, to be queried. If omitted (except for

CYCLE), a summary of all fields, databases or indexes is

returned.

Result

CYCLE with <item>

<stemname>.0 Number of values for cycle field <item>

<stemname>.1 First value

. . .

<stemname>.# Last value (# = <stemname>.0)

FIELD without <item>

<stemname>.0 Number of fields

<stemname>.1 Name of first field

. . .

<stemname>.# Name of last field (# = <stemname>.0)

FIELD with <item>

<stemname>.type Field type, plus "CALC" if a calculated field

<stemname>.length Maximum length

<stemname>.calc Calculation expression if a calculated field

FILE without <item>

<stemname>.0 Number of databases and/or windows open. Empty windows

opened with the OPENWIN command are counted.

<stemname>.1 First opened database or window (empty string if no

database)

. . .

<stemname>.# Last opened database or window (# = <stemname>.0)

FILE with <item>

<stemname>.path The path used to open the named database. This will

have a trailing slash (if applicable) if the file was

opened using the OPENFILE command, but not if it was

opened by selecting OPEN or LOAD from the FILE menu.

INDEX without <item>

<stemname>.0 Number of indexes including any temporary indexes

<stemname>.1 Name of first index

. . .

<stemname>.# Name of last index (# = <stemname>.0)

INDEX with <item>

<stemname>.0 Number of fields used in the named index

<stemname>.1 Name of first index field

. . .

<stemname>.# Name of last index field (# = <stemname>.0)

Return codes

5 <item> not found for keyword used (FIELD, FILE or INDEX)

10 CYCLE keyword used, but <item> not a CYCLE field

20 <stemname> not specified, or CYCLE keyword used but <item> not

specified.

Example

query file db

msg = ”

do i = 1 to db.0 /* displays all opened database names */

msg = msg’\010’db.i

end

reqmsg '"'msg'"'

query file abook 'AddressBook' /* This file was opened from the menu */

reqmsg '"'abook.path'"'

>>> Examples/AddressBook

openwin

openfile 'Work:QuickFile/Examples/Tapes/Tapes'

query file tapes 'Tapes'

reqmsg '"'tapes.path'"'

>>> Work:QuickFile/Examples/Tapes/

NOTE: If the <stemname> to be used is already an assigned variable by itself, it must be quoted in uppercase. Example:

reqfield

fld = result

query field fld '"'fld'"'

reqmsg '"'fld.type fld.length fld.calc'"'

>>> FLD.TYPE FLD.LENGTH FLD.CALC

query field 'FLD' '"'fld'"'

reqmsg '"'fld.type fld.length fld.calc'"'

>>> Char CALC 25 12 City|c","+State|c","+Country+PostCode

query cycle cyc 'season'

reqmsg '"'cyc.0 cyc.2'"'

>>> 4 Summer

1.29 refresh

Refresh

Redraws the screen display. This is only required in macros if it is desired that the display be updated with any changes that have taken place prior to the macro completing.

Result

None

Return codes

None

Example

refresh

1.30 report

Report <count> [{PRINTER|SCREEN} [<title>]]

Prints the report defined for the current view, if any. Writes <count> records starting with the current record. The parameters are positional so <title> cannot be specified without PRINTER or SCREEN.

<count> Number of records to process. Specify '-1' (with the

quotes) for all records.

{PRINTER|SCREEN} The desired report destination.

<title> Up to 50 characters to be used instead of the title

defined for the report in the PRINT requester. Use two

sets of quotes if the new title contains spaces.

Note that the report definition for the current view is not changed by the parameters used with this command.

Result

None

Return codes

10 No report has been defined for the current view.

15 Could not allocate memory.

Example

/* Use two sets of quotes around title strings that have spaces in them */

report '-1' printer '"Report Title over-ride"'

report 10 screen

1.31 reqchoice

ReqChoice <message1> [<message2>]

Displays a requester with up to two lines of text and OK and CANCEL buttons. As each line can contain linefeeds, only one line need be used. Two set of quotes must be used for each line if both lines are specified, or if either line contains spaces.

Result

None

Return codes

0 User selected OK.

5 User selected CANCEL.

20 No message text specified

Example

reqchoice '"Delete requested.\010Are you sure?"'

if rc = 0 then delrec

else reqmsg '"Delete cancelled"'

1.32 reqfield

ReqField [NOCALC]

Displays a requester including the names of the fields of the current base. If the user selects a field and then the OK button, the name of the selected field is returned. Double-clicking on a field is equivalent to selecting it and the OK button.

If the NOCALC keyword is used, any existing calculated fields are omitted from the list.

Result

The selected field name.

Return codes

0 User selected OK

5 User selected CANCEL

20 An error occured (the requester was not opened)

Example

reqfield

if rc = 0 then do

fieldname = result

getfield '"'fieldname'"'

reqmsg '"Field' fieldname '=' result'"'

end

1.33 reqmsg

ReqMsg <message>

Displays <message> in a requester with an OK button. The message may contain linefeeds, and must be enclosed in two set of quote marks if it contains spaces.

Result

None

Return codes

20 No message specified

Example

ReqMsg '"This message is from an ARexx script"'

1.34 reqstring

ReqString [<default> [<message>]]

Displays a requester with a string gadget and an optional <default> string in it, an optional <message> above the string gadget, and OK and CANCEL buttons.

<default> A default string to appear in the string gadget. Specify '""' if

specifying <message> and string gadget is to be left empty.

<message> A message to be displayed above the string gadget, which may

contain linefeeds.

Result

The string entered, if OK was selected

Return codes

0 User selected OK

5 User selected CANCEL

Example

reqstring '"France" "Please enter the country"' if rc = 0 then do

putfield 'country' '"'result'"'

updrec

end

1.35 savefile

SaveFile

Saves changes to the database, if any, to disk.

Result

None

Return codes

None

Example

savefile

1.36 selectrec

SelectRec

Adds the current record to the SELECTED index, unless SELECTED is the current index. Otherwise, removes the record from SELECTED. The ShowAll or SETINDEX commands may be used to switch to or from the SELECTED index. The display is not updated until the macro completes unless the REFRESH command is used.

Result

None

Return codes

10 No record to select (empty window, or no records in database)

Example

SelectRec

1.37 setfile

SetFile [<database>]

Makes <database> (which must have been previously opened) the current database, to which further macro commands are to be directed. If <database> is not specified, returns the name of the current database (filename only; no path included), or an empty string if the current window has no database opened in it.

Note that this command does not bring <database>'s window to the front. See WinToFront .

Result

Name of database in current window if issued without <database>,

otherwise none.

Return code

5 <database> not found

Example

setfile ’Images’

setfile

reqmsg ’"The current database is \034’result’\034."’

1.38 setindex

SetIndex [<index>]

Makes <index> the current index, or returns the name of the current index if <index> not specified. <index> may be SORTED or SELECTED, in addition to any of the permanent indexes defined for the current database.

Result

Name of current index if <index> not specified, otherwise none

Return codes

5 <index> not found

Example

setindex ’sorted’

setindex

reqmsg ’"The current index is \034’result’\034."’

1.39 setview

SetView [<view>]

Makes <view> (which must have been previousoly loaded) the current view and updates the display as necessary, or returns the name of the current view if <view> not specified. <view> must include the filename extension, if any. Views may be loaded by using the LOADVIEW command. Note that if a path was specified for <view> when it was loaded, the path must also be specified with this command.

Result

Name of current view if <view> not specified, otherwise none.

Return codes

5 <view> not found

Example

setview ’namelist.view’

loadview ’RAM:New.View’

setview ’RAM:New.View’

setview

reqmsg ’"The current view is \034’result’\034."’

1.40 showall

ShowAll {ON|OFF}

Toggles between the current permanent index and the SELECTED index. The display is not updated until macro completes unless the REFRESH command is used.

ON Show all records

OFF Show selected records only

Result

None

Return codes

20 Parameter missing

Example

showall on

1.41 sort

Sort <field1> {A|D} [<field2> {A|D} ...]

Sorts all records in the current index in the specified sequence, places them in a temporary index named SORTED and makes it the current index.

<field1> The field to sort by first, etc.

A Ascending order

D Descending order

Both <field> and order must be specified.

Result

None

Return codes

10 Any <field> no found

20 No parameters specified, or not an even number of parameters

Example

sort ’country’ a ’surname’ a ’firstname’ a

1.42 StringToCycle

StringToCycle <cyclefield> <string>

Returns the string number corresponding to <string> for <cyclefield>. The numbering starts at 0. The string used for the current record is not changed. See also CycleToString .

Result

Cycle field string number.

Return codes

5 <string> not found for <cyclefield>

10 <cyclefield> not found or not a cycle field

20 Parameter missing

Examples

/* Cyclefield name = 'Status'; Expression = 'Active,Pending,Cancelled' */

stringtocycle 'status' 'cancelled'

say result

>>> 2

1.43 updrec

UpdRec

Updates an existing record in which one or more field values have been modified using the PUTFIELD command. The updated record is not necessarily written to disk at this time. To insert a new record, use the INSREC command.

Result

None

Return codes

10 Internal error occurred. Possibly a duplicate key occurred.

Example

putfield 'ExpiryDate' '31-Mar-95'

updrec

1.44 wintofront

WinToFront

Brings the current database's window to the front of the screen.

Result

None

Return codes

None

Example

wintofront

1.45 Included Macros

Console Open an interactive ARexx console window

DelRecs Delete all records in the current index

Demo Demonstrate QuickFile's ARexx capabilities

FieldLengths Display the maximum used length of all fields

FindReplace Replace one substring with another in all fields

FKey Execute database-specific macros via function keys

GetSet Set one field to the value of another in all records

SetCycle Set a cycle field to its next/first/selected value

SetField Set/append/prepend a field to/with a constant value

in all records

External function macros, used by some of the macros above:

FixQuotes Translate quote marks in strings to '\xxx' ASCII codes

GetViewWin Get size/position of current view window, and font size

ProgressWin Open a progress window for lengthy operations

ReSort Re-sort records if any fields to be modified are used in

the current index