O

P

Q

R

S

T

U

W

`SOUNDPREFS.PRELOAD`
: Boolean

5.22 The CONSOLEPREFS object

This object features no fields, it contains a single line of text: the console output window specification.

5.23 The FILEPREFS object

Available fields are:

`FILEPREFS.TRANSFERPROTOCOLNAME`
: Text

`FILEPREFS.TRANSLATIONFILENAME`
: Text

`FILEPREFS.MACROFILENAME`
: Text

`FILEPREFS.CURSORFILENAME`
: Text

`FILEPREFS.FASTMACROFILENAME`
: Text

`CURSORKEYPREFS.CONTROL.DOWNTEXT`
: Text

`CURSORKEYPREFS.CONTROL.LEFTTEXT`
: Text

5.18 The FASTMACROPREFS object

`FASTMACROPREFS.COUNT`
: Numeric

  The number of fast macros available, entries range from `FASTMACROPREFS.0...` to `FASTMACROPREFS.n-1...`

`FASTMACROPREFS.n.NAME`
: Text

`FASTMACROPREFS.n.CODE`
: Text

5.19 The HOTKEYPREFS object

Available fields are:

`HOTKEYPREFS.TERMSCREENTOFRONTTEXT`
: Text

`HOTKEYPREFS.BUFFERSCREENTOFRONTTEXT`
: Text

`HOTKEYPREFS.SKIPDIALENTRYTEXT`
: Text

`HOTKEYPREFS.ABORTAREXX`
: Text

`HOTKEYPREFS.COMMODITYPRIORITY`
: Numeric

`HOTKEYPREFS.HOTKEYSENABLED`
: Boolean

`TRANSFERPREFS.BINARYUPLOADTYPE`
: `XPR`, `PROGRAM` or `DEFAULT`

`TRANSFERPREFS.BINARYUPLOADSIGNATURE`
: Text

`TRANSFERPREFS.BINARYDOWNLOADSIGNATURE`
: Text

`TRANSFERPREFS.BINARYDOWNLOADTYPE`
: `XPR`, `PROGRAM` or `DEFAULT`

`TRANSFERPREFS.BINARYDOWNLOADSIGNATURE`
: Text

`TRANSFERPREFS.BINARYDOWNLOADSIGNATURE`
: Text

5.14 The PROTOCOLPREFS object

This object features no fields, it contains a single line of text: the XPR protocol options.

5.15 The TRANSLATIONPREFS object

Indices referring to the ascii codes range from 0 to 255, available fields are:

`TRANSLATIONPREFS.n.SEND`
: Text

`TRANSLATIONPREFS.n.RECEIVE`
: Text

5.16 The FUNCTIONKEYPREFS object

Key indices range from 1 to 10 (representing F1 through F10), available fields are:

`FUNCTIONKEYPREFS.n`
: Text

`FUNCTIONKEYPREFS.SHIFT.n`
: Text

`TRANSFERPREFS.ERRORNOTIFYWHEN`
: `NEVER ALWAYS START END`

`TRANSFERPREFS.ASCIIUPLOADPROTOCOL`
: Text

`TRANSFERPREFS.ASCIIDOWNLOADPROTOCOL`
: Text

`TRANSFERPREFS.QUIETTRANSFER`
: Boolean

`TRANSFERPREFS.MANGLEFILENAMES`
: Boolean

`TRANSFERPREFS.LINEDELAY`
: Numeric

`TRANSFERPREFS.CHARDELAY`
: Numeric

`TRANSFERPREFS.LINEPROMPTTEXT`
: Text

`TRANSFERPREFS.TEXTPACING`
: `DIRECT ECHO ANYECHO PROMPT DELAY KEYBOARD`

`TRANSFERPREFS.SENDTIMEOUT`
: Numeric

`TRANSFERPREFS.STRIPBIT8`
: Boolean

`TRANSFERPREFS.IGNOREDATAPASTTERMINATOR`
: Boolean

`TRANSFERPREFS.TERMINATORCHAR`
: Numeric

`TRANSFERPREFS.TEXTUPLOADPROTOCOL`
: Text

`TRANSFERPREFS.TEXTDOWNLOADPROTOCOL`
: Text

`TRANSFERPREFS.BINARYUPLOADPROTOCOL`
: Text

`TRANSFERPREFS.BINARYDOWNLOADPROTOCOL`
: Text

`TRANSFERPREFS.OVERRIDEPATH`
: Boolean

`TRANSFERPREFS.SETARCHIVEDBIT`
: Boolean

`COMMANDPREFS.LOGOFFMACROTEXT`
Text

`COMMANDPREFS.UPLOADMACROTEXT`
Text

`COMMANDPREFS.DOWNLOADMACROTEXT`
Text

5.11 The MISCPREFS object

Available fields are:

`MISCPREFS.PRIORITY`
Numeric

`MISCPREFS.BACKUPCONFIG`
Boolean

`MISCPREFS.OPENFASTMACROPANEL`
Boolean

`MISCPREFS.RELEASEDEVICE`
Boolean

`MISCPREFS.CREATEICONS`
Boolean

`MISCPREFS.SIMPLEIO`
Boolean

`MISCPREFS.PROTECTIVEMODE`
Boolean

`MISCPREFS.IOBUFFERSIZE`
Numeric

`MISCPREFS.ALERTMODE`
`NONE BELL SCREEN BOTH`

`MISCPREFS.REQUESTERMODE`
`IGNORE CENTRE RELATIVE`

`MISCPREFS.REQUESTERLEFT`
Numeric

`MISCPREFS.REQUESTERTOP`
Numeric

`MISCPREFS.REQUESTERWIDTH`
Numeric

`CLIPBOARDPREFS.UNIT`
    Numeric

`CLIPBOARDPREFS.CONVERTLF`
    Wahrheitswert

`CLIPBOARDPREFS.LINEDELAY`
    Numeric

    Paste line delay in 1/100 seconds.

`CLIPBOARDPREFS.CHARDELAY`
    Numeric

    Paste character delay in 1/100 seconds.

`CLIPBOARDPREFS.LINEPROMPTTEXT`
    Text

`CLIPBOARDPREFS.SENDTIMEOUT`
    Numeric

    Timeout in 1/100 seconds.

`CLIPBOARDPREFS.TEXTPACING`
    `DIRECT ECHO ANYECHO PROMPT DELAY KEYBOARD`

`CLIPBOARDPREFS.INSERTPREFIXTEXT`
    Text

`CLIPBOARDPREFS.INSERTSUFFIXTEXT`
    Text

5.9 The CAPTUREPREFS object

Available fields are:

`CAPTUREPREFS.LOGACTIONS`
    Boolean

`CAPTUREPREFS.LOGFILENAME`
    Text

`CAPTUREPREFS.LOGCALLS`
    Boolean

`CAPTUREPREFS.CALLLOGFILENAME`
    Text

`CAPTUREPREFS.MAXBUFFERSIZE`
    Numeric

`TERMINALPREFS.SENDCRMODE`
: `IGNORE CR CRLF`

`TERMINALPREFS.SENDLFMODE`
: `IGNORE LF LFCR`

`TERMINALPREFS.RECEIVECRMODE`
: `IGNORE CR CRLF`

`TERMINALPREFS.RECEIVELFMODE`
: `IGNORE LF LFCR`

`TERMINALPREFS.NUMCOLUMNS`
: Numeric

`TERMINALPREFS.NUMLINES`
: Numeric

`TERMINALPREFS.KEYMAPNAME`
: Text

`TERMINALPREFS.EMULATIONNAME`
: Text

`TERMINALPREFS.FONTNAME`
: Text

`TERMINALPREFS.FONTSIZE`
: Numeric

`TERMINALPREFS.USETERMINALPROCESS`
: Boolean

5.7 The EMULATIONPREFS object

Available fields are:

`EMULATIONPREFS.CURSORMODE`
: `STANDARD APPLICATION`

`EMULATIONPREFS.NUMERICMODE`
: `STANDARD APPLICATION`

`EMULATIONPREFS.CURSORWRAP`
: Boolean

`EMULATIONPREFS.LINEWRAP`
: Boolean

`EMULATIONPREFS.INSERTMODE`
: Boolean

`MODEMPREFS.DIALRETRIES`
: Numeric

`MODEMPREFS.DIALTIMEOUT`
: Numeric

  The dial timeout in seconds

`MODEMPREFS.VERBOSEDIALING`
: Boolean

`MODEMPREFS.CONNECTAUTOBAUD`
: Boolean

`MODEMPREFS.HANGUPDROPSDTR`
: Boolean

`MODEMPREFS.REDIALAFTERHANGUP`
: Boolean

`MODEMPREFS.NOCARRIERISBUSY`
: Boolean

`MODEMPREFS.CONNECTLIMIT`
: Numeric

  Time limit in minutes.

`MODEMPREFS.CONNECTLIMITMACRO`
: Text

`MODEMPREFS.TIMETOCONNECT`
: Numeric

`MODEMPREFS.INTERDIALDELAY`
: Numeric

5.5 The SCREENPREFS object

Available fields are:

`SCREENPREFS.COLOURMODE`
: `TWO FOUR EIGHT SIXTEEN`

`SCREENPREFS.FONTNAME`
: Text

`SCREENPREFS.FONTSIZE`
: Numeric

`SERIALPREFS.DEVICENAME`
: Text

`SERIALPREFS.UNIT`
: Numeric

`SERIALPREFS.BITSPERCHAR`
: Numeric

  The number of bits per transferred char. This can be either seven or eight.

`SERIALPREFS.PARITYMODE`
: `NONE` `EVEN` `ODD` `MARK` `SPACE`.

`SERIALPREFS.STOPBITS`
: Numeric

  The number of stop bits to be used. This can be either 0 or 1.

`SERIALPREFS.HANDSHAKINGMODE`
: `NONE` `RTSCTS` `RTSCTSDSR`

`SERIALPREFS.DUPLEXMODE`
: `HALF` `FULL`

`SERIALPREFS.INTERNALXONXOFF`
: Boolean

`SERIALPREFS.HIGHSPEED`
: Boolean

`SERIALPREFS.SHARED`
: Boolean

`SERIALPREFS.STRIPBIT8`
: Boolean

`SERIALPREFS.CARRIERCHECK`
: Boolean

`SERIALPREFS.PASSXONXOFFTHROUGH`
: Boolean

`SERIALPREFS.DIRECTCONNECTION`
: Boolean

`SERIALPREFS.QUANTUM`
: Numeric

`SERIALPREFS.USEOWNDEVUNIT`
: Boolean

`SERIALPREFS.OWNDEVUNITREQUEST`
: `RELEASE` `RELEASERETRY` `IGNORE`

`TERM.SESSION.SESSIONSTART`
: Text

  Time and date when the 'term' program was started.

`TERM.SESSION.BYTESSENT`
: Numeric

`TERM.SESSION.BYTESRECEIVED`
: Numeric

`TERM.SESSION.CONNECTMESSAGE`
: Text

  The message issued by the modem when the connection was established.

`TERM.SESSION.BBSNAME`
: Text

`TERM.SESSION.BBSNUMBER`
: Text

`TERM.SESSION.BBSCOMMENT`
: Text

`TERM.SESSION.USERNAME`
: Text

`TERM.SESSION.ONLINEMINUTES`
: Numeric

  The number of minutes the program is currently connected to a BBS.

`TERM.SESSION.ONLINECOST`
: Numeric

  The cost of the connection to the BBS.

`TERM.AREXX`
: Text

  The name of the ARexx host port the program is communicating with.

`TERM.LASTERROR`
: Numeric

  The code corresponding to the error the last command has caused.

`TERM.TERMINAL.ROWS`
: Numeric

  The number of available terminal screen rows.

`TERM.TERMINAL.COLUMNS`
: Numeric

  The number of available terminal screen columns.

The main program window. Supports the `Open`, `Close`, `Activate`, `Front` and `Back` commands.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Open all available windows. */

WINDOW buffer review packet fastmacros status main OPEN
```

`Format:`

WAIT [Noecho] [[Text] <Text>]

`Template:`

NOECHO/S,TEXT/F

`Purpose:`

Waits for a certain sequence of characters to be received from the serial line.

`Specifications:`

Wait for text to be received from the serial line. If no text to wait for is provided wait for either item of the wait list to appear. The `Noecho` parameter suppresses terminal output. *Note that text comparison does not consider the case of characters (in respect to the ECMA Latin 1 character set).* As 'term' has control over the incoming data stream before and after the `WAIT` command is executed, it may 'eat' and process data the `WAIT` command ought to receive. In order to avoid this effect you can use the `PROCESSIO` command (see Section 4.29 [PROCESSIO], page 30). For example, at the beginning of a program you could tell 'term' to leave the incoming data stream alone with the `PROCESSIO OFF` command, then invoke the `WAIT` command as needed, and eventually when your program exits allow 'term' to process incoming data with the `PROCESSIO ON` command.

`Result:`

Returns the string found.

`Warning:`

If timeout has elapsed before any matching text was received.

`Example:`

```
/* Enable command results. */

OPTIONS RESULTS

/* Set the read timeout. */

TIMEOUT SEC 30

/* Wait for a single line of text. */

WAIT 'some text'

/* Clear the wait list, add a few items. */

CLEAR wait
ADDITEM TO wait NAME 'foo'
ADDITEM TO wait NAME 'bar'

/* Wait for the text to appear. */

WAIT
```

Result:

-

Warning:

-

Example:

```
/* Set the serial line stop bits. */

STOPBITS 1
```

4.56 The TEXTBUFFER command

Format:

TEXTBUFFER [Lock | Unlock]

Template:

LOCK/S,UNLOCK/S

Purpose:

Locks or unlocks the text buffer contents.

Specifications:

Locks or unlocks the text buffer contents, similar to the effect of the corresponding main menu entry.

Result:

-

Warning:

-

Example:

```
/* Lock the text buffer. */

TEXTBUFFER LOCK
```

4.57 The TIMEOUT command

Format:

TIMEOUT [[Sec | Seconds] <Number>] [Off]

Template:

SEC=SECONDS/N,OFF/S

Files whose names do not include a fully qualified path name are expected to reside in the default upload directory as specified in the main program paths settings.

`Result:`

-

`Warning:`

If user cancels file selection.

`Example:`

```
/* Prompt for files to be uploaded. */

SENDFILE

/* Send a single file. */

SENDFILE 'c:list'

/* Clear the upload list, add a single file name. */

CLEAR upload
ADDITEM TO upload NAME 'c:dir'

/* Transfer the file. */

SENDFILE
```

4.53 The SETATTR command

`Format:`

SETATTR [Object] <Name> [Field] <Name> [Stem <Name>] [Var <Name>]

`Template:`

OBJECT/A,FIELD,STEM/K,VAR

`Purpose:`

Sets a certain application attribute.

`Specifications:`

Sets a certain application attribute, retrieves the information from the supplied stem or simple variable. For a list of valid attributes, see the section entitled Section 5.23 [Attributes], page 74.

`Result:`

-

`Warning:`

-

```
/* Output the contents of the download list. */

SELECTITEM FROM download TOP

DO WHILE rc = 0
   SAY result
   SELECTITEM FROM download NEXT
END
```

4.50 The SEND command

Format:

SEND [Noecho] [Local] [Literal] [Byte <ASCII code>] [Text] <Text>

Template:

NOECHO/S,LOCAL/S,LITERAL/S,BYTE/K/N,TEXT/A/F

Purpose:

Sends the provided text to the serial line, executes embedded command sequences.

Specifications:

Sends the provided text to the serial line, executes embedded command sequences (see main program documentation). To send a single byte, use the `Byte` parameter. The `Noecho` parameter will suppress terminal output. The `Local` parameter will cause the text to be output only locally in the terminal window, it will not be sent across the serial line. The `Literal` parameter keeps ‘term’ from interpreting any special characters, such as `\\r`, in the text to send and just transmits the text you passed in.

Result:

-

Warning:

-

Example:

```
/* Send some text to the serial line. */

SEND 'This is some text.\r\n'

/* Send a single byte (a null) to the serial line. */

SEND BYTE 0

/* Execute an embedded command (send a break signal). */

SEND '\x'
```

```
/* Launch the 'term' command shell. */

RX CONSOLE ASYNC 'term:cmdshell.term'
```

4.47 The SAVE command

`Format:`

SAVE [From] <Translations | Functionkeys | Cursorkeys | Fastmacros | Hotkeys | Speech | Sound | Buffer | Configuration | Phone | Screentext | Screenimage>

`Template:`

FROM/A

`Purpose:`

Saves data to a disk file.

`Specifications:`

Saves data to a disk file, will prompt for a file name to save to. See Section 4.48 [SAVEAS], page 44 command for more information.

`Result:`

-

`Warning:`

If user cancels save operation.

`Example:`

```
/* Save the terminal screen contents to an
 * IFF-ILBM file.
 */

SAVE FROM screenimage
```

4.48 The SAVEAS command

`Format:`

SAVEAS [Name <File name>] [From] <Translations | Functionkeys | Cursorkeys | Fastmacros | Hotkeys | Speech | Sound | Buffer | Configuration | Phone | Screentext | Screenimage>

`Template:`

NAME/K,FROM/A

`Purpose:`

Saves data to a disk file.

RESETSTYLES

`Template:`

,

`Purpose:`

Resets the text rendering styles to defaults.

`Specifications:`

Resets the text rendering styles to defaults, turning off inverse video, boldface, italics, etc. modes.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Reset the text rendering styles. */

RESETSTYLES
```

4.44 The RESETTEXT command

`Format:`

RESETTEXT

`Template:`

,

`Purpose:`

Reset the terminal text to defaults.

`Specifications:`

Reset the terminal text to defaults, this includes switching back from graphics text or G1 mode.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Reset the terminal text. */

RESETTEXT
```

Warning:

If user selected negative choice.

Example:

```
/* Enable command results. */

OPTIONS RESULTS

/* Request a response. */

REQUESTRESPONSE PROMPT 'Are you indecisive?' ...
...OPTIONS '"Yes|Don't know|No"'

/* Look at the response. */

IF rc ~= 0 THEN
   SAY 'Not indecisive'
ELSE DO
   IF result = 0 THEN
      SAY 'Indecisive'
   ELSE
      SAY 'Probably indecisive'
END
```

4.41 The REQUESTSTRING command

Format:

REQUESTSTRING [Secret] [Default <String>] [Prompt <Text>]

Template:

SECRET/S,DEFAULT/K,PROMPT/K/F

Purpose:

Requests a string from the user.

Specifications:

Requests a string from the user, will display the provided prompt text or a default text and present the provided default string, so user can simply hit return to accept the defaults.

If the `Secret` parameter is provided, will not display the characters typed.

Result:

The text the user entered.

Warning:

If user cancelled the requester.

Example:

```
IF rc ~= 0 THEN
   SAY 'no files were selected'
ELSE DO
   SAY 'files selected:' names.count

   DO i = 0 TO names.count - 1
      SAY names.i
   END
END
```

4.38 The REQUESTNOTIFY command

`Format:`

REQUESTNOTIFY [Title <Title text>] [Prompt] <Prompt text>

`Template:`

TITLE/K,PROMPT/A/F

`Purpose:`

Notify the user with a message.

`Specifications:`

Opens a requester to notify the user, the prompt text can include line feed characters (`'0A'X`), the user will be able to answer the requester by clicking on a `Continue` button.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Notify the user. */

REQUESTNOTIFY TITLE '"Important information"' ...
...PROMPT 'This message is important'
```

4.39 The REQUESTNUMBER command

`Format:`

REQUESTNUMBER [Default <Default number>] [Prompt <Prompt text>]

`Template:`

DEFAULT/K/N,PROMPT/K/F

Example:

```
/* Redial the list. */

REDIAL

/* Successful? */

IF rc ~= 0 THEN SAY 'dialing list is empty'
```

4.36 The REMITEM command

Format:
: REMITEM [From] <Upload | Download | Dial | Wait> [Name <Name or wildcard>]

Template:
: FROM/A,NAME/K/F

Purpose:
: Removes one or more items from a list.

Specifications:
: Removes one or more items from a list. If no `Name` parameter is given will remove the currently selected list item (selectable using the Section 4.49 [SELECTITEM], page 45 command). The `Name` parameter can be a proper name or a wildcard pattern.

  Note: Cannot remove named items from the dial list.

Result:
: -

Warning:
: If no list item would match the name pattern.

Example:

```
/* Remove the currently selected item from the wait list. */

REMITEM FROM wait

/* Remove all items from the wait list which end with 'z'. */

REMITEM FROM wait NAME '#?z'
```

4.37 The REQUESTFILE command

Format:

Warning:

If command was cancelled, no input was made or, if the CR parameter is used, the timeout elapsed.

Example:

```
/* Enable command results. */

OPTIONS RESULTS

/* Set the read timeout to five seconds. */

TIMEOUT 5

/* Read seven bytes. */

READ NUM 7

/* Output the result. */

IF rc ~= 0 THEN
   SAY 'no data was read'
ELSE
   SAY result

/* Turn the timeout off. */

TIMEOUT OFF

/* Prompt for input. */

READ CR PROMPT 'Enter a line of text:'

/* Output the result. */

IF rc ~= 0 THEN
   SAY 'no input was made'
ELSE
   SAY result
```

4.34 The RECEIVEFILE command

Format:

RECEIVEFILE [Mode <ASCII|Text|Binary>] [Name <File name>]

Template:

MODE/K,NAME/K

4.31 The PUTCLIP command

`Format:`

PUTCLIP [Unit <Unit>] [TEXT] <Text to store>

`Template:`

UNIT/K/N,TEXT/A/F

`Purpose:`

Stores text in the clipboard.

`Specifications:`

Stores the provided text in the clipboard. Will store it in the given clipboard unit if the `Unit` parameter is given. Will use the unit number configured in the program settings otherwise.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Store a short string in the clipboard. Note: can
 * only be up to 65,536 characters long.
 */

PUTCLIP 'hello sailor!'
```

4.32 The QUIT command

`Format:`

QUIT [Force]

`Template:`

FORCE/S

`Purpose:`

Terminates the application.

`Specifications:`

Terminates program execution, will ask for a confirmation to leave unless the `Force` parameter is used. *Caution: this command may fail if there are still output windows open on the 'term' screen.*

`Result:`

-

`Purpose:`

Prints the contents of the screen, the clipboard, the textbuffer or one of the lists.

`Specifications:`

Outputs the contents of the screen, the clipboard, the textbuffer or one of the lists (see Section 4.2 [ADDITEM], page 11 command) to a file or the printer. If the `To` parameter is omitted, will output the data to the printer. The parameters `Serial` through `Attr` control what will be printed:

`Screentext, Clipboard, Buffer, Wait list`

Options have no effect on the output.

`Dial list`

Responds to the `Serial`, `Modem`, `Screen`, `Terminal`, `User` and `Comment` parameters. The printout will contain information on the corresponding settings.

`Upload list, Download list`

Responds to the `Comment`, `Size`, `Date` and `Attr` parameters. The printout will contain information on the corresponding file attributes. *Note: if any of these parameters are given, only the base file names will be printed along with the corresponding information.*

`Result:`

-

`Warning:`

If user cancelled print operation.

`Example:`

```
/* Clear the dialing list, then add the entire phone book to it. */

CLEAR dial
additem to dial phone #?

/* Send the contents of the dial list to a disk file. */

PRINT FROM dial TO 'ram:phonebook' SERIAL MODEM SCREEN...
...TERMINAL USER COMMENT
```

4.29 The PROCESSIO command

`Format:`

PROCESSIO <On | Off>

`Template:`

ON/S,OFF/S

```
/* Open a different device driver. */

OPENDEVICE DEVICE 'duart.device' UNIT 5
```

4.25 The OPENREQUESTER command

`Format:`

OPENREQUESTER [REQUESTER] <Serial | Modem | Screen | Terminal | Emulation | Clipboard | Capture | Commands | Misc | Path | Transfer | Translations | Functionkeys | Cursorkeys | Fastmacros | Hotkeys | Speech | Sound | Phone>

`Template:`

REQUESTER/A

`Purpose:`

Opens a requester window.

`Specifications:`

Opens a requester window. Only a single window can be open at a time. The window opened can be closed using the Section 4.11 [CLOSEREQUESTER], page 17 command.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Open the phonebook window. */

OPENREQUESTER phone
```

4.26 The PARITY command

`Format:`

PARITY [Even | Odd | None | Mark | Space]

`Template:`

EVEN/S,ODD/S,NONE/S,MARK/S,SPACE/S

`Purpose:`

Sets the serial line parity mode.

`Specifications:`

Sets the serial line parity mode.

`Format:`

HELP [[Command] <Name>] [Prompt]

`Template:`

COMMAND,PROMPT/S

`Purpose:`

Returns the template of a command or invokes the online help system.

`Specifications:`

This command will either return the template associated with a 'term' ARexx command specified using the `Command` parameter or invoke the AmigaGuide(tm) help system.

`Result:`

Command template if requested.

`Warning:`

-

`Example:`

```
/* Enable command results. */

OPTIONS RESULTS

/* Query the template associated with the 'selectitem' command. */

HELP selectitem

/* Output the result. */

SAY result

/* Invoke the online help. */

HELP PROMPT
```

4.23 The OPEN command

`Format:`

OPEN [Name <File name>] [TO] <Translations | Functionkeys | Cursorkeys | Fastmacros | Hotkeys | Speech | Sc

`Template:`

NAME/K,TO/A

`Purpose:`

Reads data from a disk file.

4.19 The GETCLIP command

`Format:`

GETCLIP [Unit <Number>]

`Template:`

UNIT/K/N

`Purpose:`

Retrieves the contents of the clipboard.

`Specifications:`

Obtains the contents of the clipboard and returns it in the `result` variable. Will optionally read from the given clipboard unit, but uses the unit number selected in the application settings by default. *Note that a string returned can be up to 65,536 characters long!*

`Result:`

Contents of the clipboard if it contains any text.

`Warning:`

If clipboard does not contain any text.

`Example:`

```
/* Enable command results. */

OPTIONS RESULTS

/* Get the primary clipboard contents. */

GETCLIP

/* Output the contents. */

IF rc ~= 0 THEN
   SAY 'clipboard does not contain any text'
ELSE
   SAY result
```

4.20 The GOONLINE command

`Format:`

GOONLINE

`Template:`

,

4.17 The FAULT command

`Format:`

FAULT [Code] <Error code>

`Template:`

CODE/A/N

`Purpose:`

Returns the descriptive text associated with an error code as returned by 'term'.

`Specifications:`

'term' will return error codes in the `term.lasterror` variable. In order to get the descriptive text associated with an error code, use this command. All internal Rexx and AmigaDOS errors codes are supported as well as the error codes special to 'term'.

`Result:`

The error description associated with the error code.

`Warning:`

-

`Example:`

```
/* Enable command results. */

OPTIONS RESULTS

/* Get the text associated with error #1001. */

FAULT 1001

/* Output the result. */

SAY result
```

4.18 The GETATTR command

`Format:`

GETATTR [Object] <Name> [Field] <Name> [Stem <Name>] [Var <Name>]

`Template:`

OBJECT/A,FIELD,STEM/K,VAR/K

`Purpose:`

Obtains information on an application attribute.

`Purpose:`

Dials the provided phone number. If no phone number was given, will dial the numbers and phone book entries stored in the dial list.

`Specifications:`

This command will build a dialing list from the available sources and pass it to the dialing function which is to schedule the dialing process and perform any login-actions. Available sources are the `Num` parameter which will cause the command to dial only this single number or the dial list whose contents will be used if the `Num` parameter is omitted.

If the `WAIT` parameter is used the command will wait until a connection is made. If the parameter is not use this command will return as soon as the dialing process has been initiated.

`Result:`

-

`Warning:`

If no connection was to be made.

`Example:`

```
/* Dial a single phone number. */

DIAL 424242

/* Wait a bit and abort the dialing process. */

DELAY 5
CLOSEREQUESTER

/* Clear the dialing list, then add all the phonebook entries
 * to the list.
 */

CLEAR FROM dial
ADDITEM TO dial PHONE #?

/* Dial the dial list. */

DIAL WAIT

/* Did we get a connection? */

IF RC == 0 THEN SEND TEXT "Ack!\r"
```

4.15 The DUPLEX command

`Format:`

`Template:`

,

`Purpose:`

Closes the currently open requester window

`Specifications:`

Will close any currently open requester window, such as the dialing window, the phone book, the serial settings window, etc. Will not close windows such as the file transfer window or the text/numeric input windows.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Close the currently open requester window,
 * whatever it may be.
 */

CLOSEREQUESTER
```

4.12 The DEACTIVATE command

`Format:`

DEACTIVATE

`Template:`

,

`Purpose:`

Iconifies the program.

`Specifications:`

Puts the application to sleep. Requires Workbench to be running, so an AppIcon can be put on the Workbench backdrop. This command will be ignored if the application has already been put to sleep. To wake the application up, use the Section 4.1 [ACTIVATE], page 10 command.

`Result:`

-

`Warning:`

-

`Example:`

```
CLEAR FROM buffer

/* If no confirmation was given, clear it by force. */

IF rc ~= 0 THEN CLEAR FROM buffer FORCE
```

4.8 The CLEARSCREEN command

`Format:`

CLEARSCREEN

`Template:`

,

`Purpose:`

Clears the terminal screen

`Specifications:`

Clears the terminal screen and positions the cursor in the top left corner.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Clear the terminal screen. */

CLEARSCREEN
```

4.9 The CLOSE command

`Format:`

CLOSE [From] <Printer|File|All>

`Template:`

FROM/A

`Purpose:`

Terminates file and/or printer capture.

`Specifications:`

Terminates a capture process as started with the Section 4.6 [CAPTURE], page 14 command. Will terminate printer capture, file capture or both.

`Format:`

CALLMENU [Title] <Title text or wildcard pattern>

`Template:`

TITLE/A/F

`Purpose:`

Invokes the function associated with a menu item.

`Specifications:`

Calls a pull-down menu function just as if the user had selected it using the mouse. The `Title` parameter can be any valid menu item name or a wildcard pattern. In the latter case, only the first menu item to match the pattern will be called.

`Result:`

-

`Warning:`

If no matching menu item was to be found.

`Example:`

```
/* Invoke the 'About...' menu item. */

CALLMENU abou#?
```

4.6 The CAPTURE command

`Format:`

CAPTURE [To] <Printer|File> [Name <File name>]

`Template:`

TO/A,NAME/K

`Purpose:`

Starts a file or printer capture.

`Specifications:`

If a capture is not already in progress will open a capture file or start capturing incoming terminal text to the printer. If the `File` argument is given and the `Name` parameter is omitted, will prompt for the name of a file to capture to.

If to capture to a given file, will append the captured text. If user is to select a file to capture to, will ask whether to append the text to the file or to overwrite it.

`Result:`

-

`Phone numbers`
: These are passed using the `Name` parameter.

List item can be inserted before or after the currently selected list item (see Section 4.49 [SELECTITEM], page 45 command). The default is to insert them after the currently selected list item.

`Result:`

-

`Warning:`

-

`Example:`

```
/* Enable result codes. */

OPTIONS RESULTS

/* Get a file name from the user. */

REQUESTFILE TITLE '"Select a file to upload"'

/* Add the file name to the upload list. */

IF rc = 0 THEN ADDITEM TO upload NAME result

/* Add phonebook entry #2 to the dial list. */

ADDITEM TO dial PHONE 2

/* Add all phonebook entries whose names start
 * with an `a' to the dial list.
 */

ADDITEM TO dial PHONE a#?

/* Add a plain phone number to the dial list. */

ADDITEM TO dial NAME 424242
```

4.3 The BAUD command

`Format:`

BAUD [Rate] <Baud rate in bits per second>

`Template:`

RATE/A/N

`, (Comma)`
Indicates that the command takes no parameters.

`Purpose:`
Briefly describes what the command will do.

`Specifications:`
Describes the command and its possible uses in more detail.

`Result:`
The type of the command result code if any.

`Warning:`
If the command can return with a warning and when.

`Example:`
An example code fragment to illustrate how to use the command. Commands and keywords are given in upper case, the names of variables and command arguments are given in lower case. Where a single command line would not fit into a single line on the screen, an ellipsis ('...') is meant to join the broken line.

4.1 The ACTIVATE command

`Format:`
ACTIVATE

`Template:`
,

`Purpose:`
De-iconifies the program, brings the main window to the front and makes it active.

`Specifications:`
The program can be put to sleep using the Section 4.12 [DEACTIVATE], page 18 command, to bring it back to proper operation, use the `ACTIVATE` command. If this command is invoked while the program is not asleep, it will cause the main window to be brought to the front and activated.

`Result:`
-

`Warning:`
-

`Example:`
```
/* This is how the main programm can be (re-)activated: */

ACTIVATE
```

Most commands will return results or error codes on failure. To enable result codes, one has to use the `options results` command. The results returned by commands will be placed in the `result` variable:

```
/* We assume that the script will address the host it was invoked from.
 *
 * Enable command results.
 */

OPTIONS RESULTS

/* Request a string from the user. */

REQUESTSTRING DEFAULT 'anything' PROMPT 'Enter anything'

/* Did the user cancel the requester? */

IF rc ~= 0 THEN
   SAY 'user cancelled requester'
ELSE
   SAY result /* Output the result . */
```

Failure codes will always be returned in the `rc` variable (see previous example).

In case of failure (variable `rc` >= 10), 'term' will leave an error code in the `term.lasterror` variable:

```
/* Enable command results. */

OPTIONS RESULTS

/* Produce an error by not supplying any arguments. */

STOPBITS

/* Display the error code. */

SAY term.lasterror
```

Rexx tries to tokenize any command parameters, this process involves promoting them to all upper case letters and checking for illegal characters. This feature inhibits the use of the : (colon) and blank space characters in parameter names unless the corresponding arguments are enclosed in quotes. To make things even more complicated, the parser will not always accept parameters to contain blank spaces. If a command template accepts the entire command line (such as `TEXT/K/F`) a parameter can include any number of blank spaces. A command template to accept just a single parameter (such as `TEXT/K`) requires double quotes if blank spaces are included. Text such as `tea or coffee?` thus becomes `'"tea or coffee?"'`.

The program ‘term’ and the data received/sent by it must not be used for the following purposes:

1. The construction, development, production or testing of weapons or weapon systems of any kind.
2. The construction, development, production or use of plants/installations which include the processing of radioactive/fissionable material.
3. The training of persons to deal with the abovesaid actions.

Listen to your conscience.