

ARexx

Troels Walsted Nilsen

COLLABORATORS

	<i>TITLE :</i> ARexx		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Troels Walsted Nilsen	August 24, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	ARexx	1
1.1	Overview on using ARexx with THOR, by Troels Walsted Hansen and Petter Nilsen	1
1.2	Author information and credits	1
1.3	Introduction/THOR and ARexx	2
1.4	Instructions/Command execution	3
1.5	Instructions/Command table	4
1.6	Instructions/Included scripts	5
1.7	Included scripts/AddUser.thor	7
1.8	Included scripts/BuildList.thor	7
1.9	Included scripts/CEDDownload.thor	7
1.10	Included scripts/MultiUpload.thor	7
1.11	Included scripts/NCommPoll.thor	8
1.12	Included scripts/NCommPollCurrent.thor	8
1.13	Included scripts/PurgeEvents.thor	8
1.14	Included scripts/SaveRange.thor	9
1.15	Included scripts/UploadFromDopus.thor	9
1.16	Included scripts/UUDecode.thor	9
1.17	Included scripts/UUEncode.thor	9
1.18	Included scripts/CfgSortMail.thor	10
1.19	Included scripts/ForwardMsg.thor	10
1.20	Included scripts/KeepMsg.thor	10
1.21	Included scripts/MIMEUnpack.thor	10
1.22	Included scripts/PGPAddToKeyRing.thor	10
1.23	Included scripts/PGPDecrypt.thor	11
1.24	Included scripts/PGPViewPubKeys.thor	11
1.25	Included scripts/QuotePercent.thor	11
1.26	Included scripts/RePostEN.thor	11
1.27	Included scripts/StealTagline.thor	12
1.28	Included scripts/ChessMaster3000.thor	12
1.29	Included scripts/TrapDoorPoll.thor	13

1.30	Included scripts/UnMime.thor	14
1.31	Included scripts/ZapFiles.thor	14
1.32	Included scripts/ASCIIPic.fse	14
1.33	Included scripts/Figlet.fse	14
1.34	Included scripts/FindFile.fse	15
1.35	Included scripts/forward.fse	15
1.36	Included scripts/Insult.fse	15
1.37	Included scripts/PreviewMsg.fse	15
1.38	Included scripts/Quote.fse	15
1.39	Included scripts/UnQuote.fse	16
1.40	Included scripts/UUEncode.fse	16
1.41	Included scripts/requote.fse	16
1.42	Included scripts/Style.fse	17
1.43	Included scripts/AddAmiNetFileList.br	17
1.44	Included scripts/AddFileList.br	17
1.45	Included scripts/CheckDataBase.br	17
1.46	Included scripts/CheckUserDataBase.br	18
1.47	Included scripts/ClearFileDataBase.br	18
1.48	Included scripts/Filedescr2Comment.br	18
1.49	Included scripts/JoinConfs.br	18
1.50	Included scripts/PackBBS.br	19
1.51	Included scripts/PackData.br	19
1.52	Included scripts/Reply2Enter.br	19
1.53	Included scripts/SetLastRead.br	19
1.54	Command Reference/FSE	19
1.55	FSE/General/Hide	20
1.56	FSE/General/Quit	21
1.57	FSE/General/Refresh	21
1.58	FSE/General/Reveal	22
1.59	FSE/Information/CurrentChar	22
1.60	FSE/Information/FSEScreenName	23
1.61	FSE/Information/GetFSERevision	23
1.62	FSE/Information/GetFSEVersion	24
1.63	FSE/Information/GetLine	25
1.64	FSE/Information/IsFirstCall	25
1.65	FSE/Information/LineLength	26
1.66	FSE/Information/MaxMSGLength	26
1.67	FSE/Information/MSGFileName	27
1.68	FSE/Information/MSGLength	27

1.69 FSE/Information/Recipient	28
1.70 FSE/Information/RecipientAddr	28
1.71 FSE/Information/Sender	29
1.72 FSE/Information/SenderAddr	30
1.73 FSE/Information/SenderDate	30
1.74 FSE/Information/Conference	31
1.75 FSE/Information/XPos	31
1.76 FSE/Information/YPos	32
1.77 FSE/Information/AutoIndent	32
1.78 FSE/Information/OverWrite	33
1.79 FSE/SpecificCommands/BackSpace	33
1.80 FSE/SpecificCommands/BegOfFile	34
1.81 FSE/SpecificCommands/BegOfLine	34
1.82 FSE/SpecificCommands/DeleteLines	35
1.83 FSE/SpecificCommands/EndOfFile	35
1.84 FSE/SpecificCommands/EndOfLine	36
1.85 FSE/SpecificCommands/IncludeFile	36
1.86 FSE/SpecificCommands/InsertBufferLines	37
1.87 FSE/SpecificCommands/InsertInput	37
1.88 FSE/SpecificCommands/InsertLines	38
1.89 FSE/SpecificCommands/InsertString	38
1.90 FSE/SpecificCommands/InsertTag	39
1.91 FSE/SpecificCommands/NewLine	39
1.92 FSE/SpecificCommands/PasteClip	40
1.93 FSE/SpecificCommands/PlayMacro	40
1.94 FSE/SpecificCommands/SaveFile	41
1.95 FSE/SpecificCommands/SetPos	42
1.96 FSE/Requesters/RequestNotify	42
1.97 FSE/Requesters/RequestString	43
1.98 FSE/General/NOP	44

Chapter 1

ARexx

1.1 Overview on using ARexx with THOR, by Troels Walsted Hansen and Petter Nilsen

This is the documentation for the ARexx implementation in THOR, the command host BBSREAD and in THOR's internal editor, the FSE.

Introduction and instructions

ARexx.guide	Author information and credits
THOR and ARexx	A brief introduction to ARexx with THOR
Command execution	How to execute THOR/BBSREAD ARexx commands
Commands	The command table form
Included scripts	Documentation for the included scripts

ARexx Command Reference

THOR commands	ARexx reference for THOR
BBSREAD commands	ARexx reference for the BBSREAD command host
FSE commands	ARexx reference for the Internal Editor

BBSREAD ARexx definitions ARexx definitions for BBSREAD

1.2 Author information and credits

About ARexx.guide

This AmigaGuide document was created by Troels Walsted Hansen with layout inspiration taken from termRexx.guide by Olaf Barthel and some text taken from Petter Nilsen's example scripts. Modified for THOR 2.0 by Petter Nilsen. Some text was also taken from Marius' original documentation for the FSE (the internal editor in THOR).

Contact me, Troels, through either of these:
Ultima Thule BBS: +47-77-(681999/613205/639830)
Internet: troelsh@powertech.no
FidoNet: 2:212/8.39
SnailMail: Troels Walsted Hansen

O.L. Aunesgate 15
9009 Tromsø
(NORWAY)

For information on how to contact Petter Nilsen, see the THOR.guide.

The ARexx scripts included with THOR have been made by:

- Troels Walsted Hansen
- Eirik Synnes
- Eivind Nordseth
- Petter Nilsen
- Magne Østlyngen
- Kjell Irgens
- Marius Mortensen

1.3 Introduction/THOR and ARexx

THOR and ARexx

For those of you who haven't programmed ARexx before, please don't expect this document to explain all of the details to you. This is only meant as a reference guide for looking up the template and usage of the ARexx commands in THOR, BBSREAD and in the FSE.

The section entitled `Command_execution` gives a brief introduction on how to write and run ARexx commands. For more information refer to the Release 2 Users Manual 'Using the System Software'.

By default THOR opens an ARexx host by the name of THOR.01. If more than a one THOR is running on your machine, the hostname will be named according to the order in which the THORs were run. The first will be named THOR.01, the second one THOR.02, the third one THOR.03, etc. The name of the host is displayed in the About window in THOR.

The above is also valid for the internal editor, the FSE, where the hostnames are named THOR_FSE.01, THOR_FSE.02 etc. In general, all that is said about the THOR commands in this guide is also true for the FSE and BBSREAD commands.

The BBSREAD command host library will always have the name BBSREAD when it is running. THOR do not need to be running for BBSREAD to work! Use this code in all ARexx scripts for starting the BBSREAD command host library:

```
if ~show('p', 'BBSREAD') then do
  address command
    "run >nil: 'GetEnv THOR/THORPath'bin/LoadBBSRead"
    "WaitForPort BBSREAD"
end
```

Then you can just address BBSREAD the usual way with `"address("BBSREAD")"` whenever you want to use a command.

For several advanced example scripts check out the included scripts in your <Your_THOR_Dir>/Rexx/ drawer.

1.4 Instructions/Command execution

Command execution

In order to invoke any command supported by THOR and BBSREAD, one usually has to address the host explicitly:

```
/* Address the THOR host. */

ADDRESS 'THOR.01'

/* Invoke the THORTOFRONT command. */

THORTOFRONT
```

However, if an ARexx program is invoked directly by the THOR program or from the the Internal Editor, the program will by default address the THOR (or Internal Editor) it was invoked by.

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 program will address the host it was invoked from.
 *
 * Enable command results.
 */

OPTIONS RESULTS

/* Request a string from the user. */

REQUESTSTRING TITLE '"Stringrequest"' BT '"_Ok|_Cancel"' ID ...
'"Initial string"' MAXCHARS 100

/* 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).

When BBSREAD or THOR return an error code of 30 in rc, the stem variable THOR.LASTERROR (or BBSREAD.LASTERROR in BBSREAD's case) will contain a message explaining what went wrong.

```
/* REQUESTNOTIFY returns 0 on success and 30 on failure, in
   which case THOR.LASTERROR will contain a description of why
   the function returned an error code.
 */
```



```

REQUESTNOTIFY TEXT '"Simple requester"' BT '"_Ok|_Cancel"'
if(rc ~= 0) then
do
    say THOR.LASTERROR
    exit
end

```

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. 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 following command will fail to get the configuration for the
 * BBS 'Ultima Thule' as the BBS name is treated as two single names:
 */

```

```
GETBBSCONFIG Ultima Thule
```

```

/* The next line will still fail to get the configuration for
 * 'Ultima Thule' as the ARexx parser will split the argument into
 * two parameters.
 */

```

```
GETBBSCONFIG 'Ultima Thule'
```

```
/* Here is how to do it correctly: */
```

```
GETBBSCONFIG '"Ultima Thule"'
```

```
/* Another correct way of doing it: */
```

```

bbsname = 'Ultima Thule'
GETBBSCONFIG '"'bbsname'"'

```

1.5 Instructions/Command table

Command table

The commands supported by THOR, BBSREAD and the FSE are listed in a standard man-file table format with the following form:

NAME

The name of the function and a short description of what it does.

SYNOPSIS

The command and the template, similar to the command templates employed by AmigaDOS Shell commands. Possible templates are:

<Parameter>/A

The parameter must always be included in order to get accepted.

<Option>/K

The option's keyword must be given.

<Option>/S

This option works as a switch. If this option keyword is included the switch is on, else it is off.

<Option>/N

A numeric parameter is expected.

<Text>/F

The text must be the final parameter on the command line.

, (Comma)

Indicates that the command takes no parameters.

FUNCTION

Describes what the command will do.

INPUTS

Describes the command, its possible uses and the parameters in more detail.

RESULT

The type of the command result code if any.

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.

NOTES

Various notes about the command.

SEE ALSO

Link to other commands that is related to this command.

1.6 Instructions/Included scripts

Included scripts

This is the documentation for the ARexx scripts included with THOR 2.0.

All of the THOR scripts have one thing in common and that is their startup procedure. They first determine whether they were started from inside THOR (via the "ARexx Scripts" meny) and if so, they will address that THOR. Alternatively the scripts will try to address

the first THOR port it can find.

NOTE:

Scripts that end with .br is meant to be run from shell and will only use the BBSREAD command host interface.

Scripts that end with .thor is (in most cases) meant to be run from within THOR and will use the THOR ARexx port and the BBSREAD command host interface whenever needed.

Scripts that end with .fse is meant to be run from the Internal Editor (FSE) in THOR and will only use the FSE ARexx port.

Those scripts that require external libraries or other programs, like CED or DirOpus, will check for these at startup and complain if they are not found.

Below here you will find detailed information on each of the scripts. If something is missing, you would like new features or in case you have found a bug in one the scripts, please contact the author.

THOR Scripts

AddUser.thor	PGPViewPubKeys.thor
BuildList.thor	PurgeEvents.thor
CEDDownload.thor	QuotePercent.thor
CfgSortMail.thor	RePostEN.thor
ChessMaster3000.thor	SaveRange.thor
ForwardMsg.thor	SortMail.thor
KeepMsg.thor	StealTagline.thor
MIMEUnpack.thor	TrapDoorPoll.thor
MultiUpload.thor	UnMime.thor
NCommPoll.thor	UploadFromDopus.thor
NCommPollCurrent.thor	UUDecode.thor
PGPAddToKeyRing.thor	UUEncode.thor
PGPDecrypt.thor	ZapFiles.thor

FSE Scripts

Reqoute.fse	Unquote.fse
ASCIIPic.fse	UUEncode.fse
Figlet.fse	Style.fse
FindFile.fse	and these:
Forward.fse	· Bold.fse
Insult.fse	· ~Underline.fse
PreviewMsg.fse	· Italic.fse
Quote.fse	· Reverse.fse

BBSREAD Scripts

AddAmiNetFileList.br	JoinConfs.br
AddFileList.br	PackBBS.br
CheckDataBase.br	PackData.br
CheckUserDataBase.br	Reply2Enter.br
ClearFileDataBase.br	SetLastRead.br
Fileddescr2Comment.br	

1.7 Included scripts/AddUser.thor

AddUser.thor

This script will let you add the sender of the current message to the user database in THOR. Both name, address, alias and a comment can be entered.

1.8 Included scripts/BuildList.thor

BuildList.thor

This script will let you build a file containing a list of BBSes and conferences for use with MultiUpload.thor. First choose or enter the name of the file you want to hold the list. Then pick BBS names and conference names until you are finished. Pressing escape or clicking on the close gadget in the conference list window will leave the conference field empty, useful for MultiUpload lists where you don't want to make the upload private to a conference.

1.9 Included scripts/CEDDownload.thor

CEDDownload.thor

This script is very simple. Place the cursor on a filename in CED (CygnusEditor) and start the script. THOR will request a BBS name and a download event will be created with the filename from CED. Useful for browsing through a filelist in CED and selecting files to download.

A permanent BBS name can be specified in the script if you don't want the BBS name requester to pop up for every filename.

For details on how to make CED execute CEDDownload.thor on the press of a function key, check out page 94 in your CED v3.5 manual.

1.10 Included scripts/MultiUpload.thor

MultiUpload.thor

This script will let you choose and enter data for a file to be

uploaded. This script utilize lists made by BuildList.thor. Creating an upload event requires that the System Type of the BBS you choose, supports uploading.

MultiUpload will search for a file named <filename_without_extension>.readme, say THOR20.readme for THOR20.lha. This file will be used for the detailed description of the file and also if the line "Short: Short_descr_of_file" is found in the readme file, MultiUpload will use "Short_descr_of_file" instead of requesting a description through a string requester.

1.11 Included scripts/NCommPoll.thor

NCommPoll.thor

This script was made to make it simple to dial one or many BBSes from inside THOR. The script should be easily adapted for other communication programs than NComm if you so desire.

The script will open a list of BBSes and will let you multiselect the ones you want to call.

After this, the script will try to tell NComm to dial the selected BBSes. If NComm isn't already started the script will attempt to do so for you. This requires that NComm: is assigned to the directory containing the program.

To use this script, the BBSs configured in THOR must have the same name as those in your NComm phonebook.

1.12 Included scripts/NCommPollCurrent.thor

NCommPollCurrent.thor

This ARexx script will automatically dial with NComm the currently hi-lighted BBS (in the Startup Window) or the currently open BBS.

If NComm isn't already started the script will attempt to do so for you. This requires that NComm: is assigned to the directory containing the program.

To use this script, the BBS configured in THOR must have the same name as in your NComm phonebook.

1.13 Included scripts/PurgeEvents.thor

PurgeEvents.thor

This script will delete all events at the current BBS (so be careful when using this script!).

1.14 Included scripts/SaveRange.thor

SaveRange.thor

A script for saving a range of messages to disk from the database. The script will request a start- and endnumber and a directory. The messages will be saved as ConfName.MsgNumber.

1.15 Included scripts/UploadFromDopus.thor

UploadFromDopus.thor

This script will simply go through the files selected in Dopus one by one, requesting data whenever it needs it. This may be BBS name, description etc. which is needed to create an upload event. Creating an upload event requires that the BBS Type of the BBS you choose, supports uploading.

This script will utilise <filename>.readme files in way similar to MultiUpload.thor. If no .readme file is around, the script will attempt to obtain a short description from the filecomment of the file.

Making a button in Dopus that executes this script should be trivial.

1.16 Included scripts/UUDecode.thor

UUDecode.thor

This script will save the current message in THOR and run uuOut/UUFast/UUxT (see the script itself) on it, uudecoding files to your download directory.

The ending file will get a file comment like this:

From <name> in <conf> on <bbsname>

1.17 Included scripts/UUEncode.thor

UUEncode.thor

After you choose a file to uuencode this script will give you many different options. The script may archive the file using LhA before uuencoding and you may choose between putting the file into the clipboard or posting the file as a message.

1.18 Included scripts/CfgSortMail.thor

CfgSortMail.thor

This is the configuration script for SortMail.thor that will let you configure parameters for the SortMail-script.

1.19 Included scripts/ForwardMsg.thor

ForwardMsg.thor

This script is used for forwarding the current message to another user on the same system or on some other system in the THOR database. The user to forward the message to must exist in the user database on the destination system. The userdatabase will be searched to find matching users for the data given in the string requester in the script.

This script also contains nice, commented code for searching the user database from ARexx and processing the matches.

1.20 Included scripts/KeepMsg.thor

KeepMsg.thor

This script will copy the current message to a 'Keep' conference and set the KEEP flag on the message in that conference. By setting the KEEP flag, the message can't be purge from the conference as long as the KEEP flag is set.

If the destination conference doesn't exist it will be created.

1.21 Included scripts/MIMEUnpack.thor

MIMEUnpack.thor

This script will unpack the MIME encoded file contained in the message currently displayed in THOR and place the file in your configured download directory with the subject of the message as filename

'munpack' (v1.2 or higher) is assumed to be in the search path.

1.22 Included scripts/PGPAddToKeyRing.thor

PGPAddToKeyRing.thor

This script will add a PGP key to the public keyring. The PGP key to add is assumed to be in the current message in THOR.

PGP must be correctly installed for this script to work and the AmigaOS command "search" must be in your search path.

1.23 Included scripts/PGPDecrypt.thor

PGPDecrypt.thor

This script is used to decrypt PGP signed or encrypted messages. When executed, the current message in THOR will be fed to PGP. PGP will ask you for a password and the decrypted message will be put into the message listview where it can be viewed and replied.

1.24 Included scripts/PGPViewPubKeys.thor

PGPViewPubKeys.thor

This script will let you view the contents of your public PGP keyring.

PGP must be correctly installed for this script to work.

1.25 Included scripts/QuotePercent.thor

QuotePercent.thor

This script calculates the percentage of quoted characters in a message, and put up a requester showing the percentage.

The script is a port to THOR from a script originally written by Kent Hansen for the Fidonet-reader Spot.

1.26 Included scripts/RePostEN.thor

RePostEN.thor

This script is for reposting messages from a System/BBS to conferences on another. The reposted message will get a header that will look something like this:

```
Message crossposted from: News&Mail:comp.sys.amiga.applications
Originally written by: David Griffiths <dgriff@unixg.ubc.ca>
Subject: Where is AmigaBlueWave (AmyBW) author? Can't register!!!
```


1.27 Included scripts/StealTagline.thor

StealTagline.thor

This script will let you steal taglines from the currently active message by opening a multiselect listview where the tagline to steal can be selected. The tagline will be appended to the global tagfile or the tagfile configured for the active conference/system (if any).

1.28 Included scripts/ChessMaster3000.thor

ChessMaster3000.thor

This is a script for playing a game of chess against another THOR-user through messages on a System.

Instructions for use

You need to have the main THOR window open, because the script uses the message listview for displaying the chess board. If you start the script and the current message doesn't contain a game underway, you have the option of starting a new one. Starting a new game will make you the white player and thus let you have the first move. You may choose any BBS to post the message on, any receiver, any subject, etc.

Making your move is done via a string requester asking for coordinates. Entering e.g. 'blc3' will move your white knight from B1 to C3. The game **does** **not** check the validity of your moves under the rules of chess. It merely checks whether you're really moving one of your own pieces to a valid square.

You may also click the 'Special moves' button (leaving the string gadget empty) and be prompted with three different choices. Exchanging of pawns is probably the most common one and you may use it when one of your pawns have reached the other end of the board. It may be exchanged into either of the other pieces, except for the king of course.

To castle is the second option. This involves moving your king two steps the left (long castling) or to the right (short castling). The squares between the king and either of your castles must be open, and if it is, the castle will jump over the king and stand next to him.

The last option is the rarest (at least I had never heard of it before), it let's you beat an opponent pawn 'en passant' (in passing). One of your pawns must've been guarding the square in front of an opponent pawn, an opponent pawn which has not moved earlier in the game. Not having moved earlier gives the pawn the opportunity to walk past you by taking it's two-step first move. If it does this you may beat it, without moving your own pawn, when it is your turn. This must happen immediately after the opponent pawn jumped past your guarding pawn.

Winning the game is accomplished by actually beating the opponent's

king. This is not usually done in chess, but this script has no facilities for detecting a checkmate.

After you've made your move, you will be prompted with the possibility to edit the message. This allows you to relay haughty comments to your opponent, but you must be careful not to edit below the '***ChessMaster [...]' line! Doing so might make the script unable to parse the message when your opponent is making his/hers move. Anything above the indicated line will be ignored and discarded by the script.

Other info

The chess pieces are represented as follows:

K = King	B = Bishop	C = Castle
Q = Queen	N = Knight	P = Pawn

A square with a '^'-character below it is *black*.

Black chess pieces are indicated with styletags for reverse video. If you haven't switched on Styletags and ANSI in THOR's Visual Options, all you'll see is a pair of '#'-characters next to the black pieces, ruining the game.

Author

Troels Walsted Hansen <troelsh@powertech.no>. Contact me through e-mail or on Ultima Thule for bugreports or requests for features.

History

New in v1.05:

- fixed names of the chess pieces
- ~userdatabase lookup

New in v1.10:

- implemented Pawning and Castling
- corrected starting positions
- ~switched coordinates, to conform to chess standards

New in v1.20:

- ~enabled editing of file
- implemented Passant
- ~wrote the docs

1.29 Included scripts/TrapDoorPoll.thor

TrapDoorPoll.thor

This is a script that automates TrapDoor polling of a Systems selected from a listview. The script assumes that you have configured one main TrapDoor.cfg and several small cfg files for each of your bosses. You will need to edit a few lines in this script to suit your environment.

1.30 Included scripts/UnMime.thor

UnMime.thor

This script to convert the current mime quoted-printable message readable text and show the converted message in the message listview, ready to be replied to. Assumes charset=iso-8859-1.

Usually, the use of this script is not needed since the SOUP and UUCP import modules for THOR will do this automatically when importing messages. However, in some cases, an incorrect MIME header is generated by buggy mail- and newsreaders, and this is where this script can handy. You can easily identify a message containing "mime quoted-printable" by the various "=E7=E8" etc. characters.

Known bugs: Not as speedy as it should have been. :-)

1.31 Included scripts/ZapFiles.thor

ZapFiles.thor

This script will create sysop script commands for zapping (deleting) selected files on the remote BBS system using the files selected in the filedatabase window.

The usage is pretty easy, select the files in the file database window, then start this script.

Will only work on ABBS/MBBS (or compatible) systems.

1.32 Included scripts/ASCIIPic.fse

ASCIIPic.fse

This script will convert a picture to ASCII and include it in the FSE using 'ilbm2ascii' v1.4 by Tobias Ferber.

The default path to your picture directory can be edited at the top of the script.

1.33 Included scripts/Figlet.fse

Figlet.fse

This script will ask you for a string and a figlet font, and then, using the figlet program, write whatever string you entered, in the font you chose, into the editor.

The default path to your font and figlet directory can be edited at the top of the script.

1.34 Included scripts/FindFile.fse

FindFile.fse

An ARexx script that finds files matching a search pattern on a chosen system and inserts the file information into the FSE you are currently using.

A search for "pic" might come up with this output:

```
(Gfx/Misc)
ShowJPEG16.lha  950503  39467  0 V1.3 of the JPEG-viewer for the Picass

(Util/Misc)
BootPaintPic.lha 950503  8342  2 Excellent Boot Picture-WB3.1 Paint Log
```

1.35 Included scripts/forward.fse

Forward.fse

A small script showing how to make a custom forward string, like this:

Petter Nilsen wrote 17-Jul-94 the following:

1.36 Included scripts/Insult.fse

Insult.fse

This script will insert a random insult into the FSE when invoked. Do not use it on someone who can't take it :-)

The insult might look something like this:

You goatish fly-bitten moldwarp!

1.37 Included scripts/PreviewMsg.fse

PreviewMsg.fse

This script will show you (in THOR's messagetext listview) what your message will look like when read by others in THOR. This is nice for "testing" Style Tags etc.

1.38 Included scripts/Quote.fse

Quote.fse

A small script that will simply quote all non-empty lines with the '>' prefix.

1.39 Included scripts/UnQuote.fse

UnQuote.fse

Reduce the layer of quote chars (">") in a block of text to just one level.

1.40 Included scripts/UUEncode.fse

UUEncode.fse

An ARexx script that uuencodes a file and imports it into the FSE you are currently using. Unless the file is already archived this script will optionally do it for you (using LhA).

Utilises LhA by Stefan Boberg and either of the following:

- ~UUFast v1.25 by Jørn Halonen
- ~uuIn v1.03 by Nicolas Dade
- ~UUxT v3.0 by Asher Feldman

1.41 Included scripts/requote.fse

Requote.fse

This script was made by the author of the FSE, Marius Mortensen. It is useful when you want to break the current line and reflow and requote the text below it.

Example:

```
>>>First line of quoting.
>>Second line of quoting.
>Third line of quoting. #Something completely different that you
>wish to answer seperately below your answer to the text "Third
>line of quoting".
```

Executing Requote.fse at cursor position # will make the text appear like this:

```
>>>First line of quoting.
>>Second line of quoting.
>Third line of quoting.
```

#

```
>Something completely different that you wish to answer  
>seperately below your answer to the text "Third line of  
>qouting".
```

With the cursor now placed at the # ready for typing.

1.42 Included scripts/Style.fse

Style.fse

This script is the central 'hub' for four other scripts. Namely Bold.fse, Underline.fse, Italic.fse and Reverse.fse. When either of these four scripts are invoked they will call upon the main script with a single character as an argument. This character is '*' for Bold.fse, '_' for Underline.fse, '/' for Italic.fse and '#' for Reverse.fse. These characters are recognized as Style Tags by THOR. Whenever THOR encounters one of these in a message it will make the word enclosed by two such characters styled according to which characters found.

Style.fse will take the character it received as an argument and place one on either side of the word which is currently under the cursor. Style.fse will also enclose the word if the cursor is on a space after the word or if the cursor is on a special character like '!', '?', '.' etc. after the word.

1.43 Included scripts/AddAminetFileList.br

AddAminetFileList.br

This script is for adding an AmiNet RECENT/INDEX file list to the file database on the given system. Example command line:

```
rx AddAminetFilelist.br "News&Mail" "ram:INDEX"
```

1.44 Included scripts/AddFileList.br

AddFileList.br

This script is for adding a FLIM generated ABBS file list to the file database on the given system. Example command line:

```
rx AddFilelist.br "Ultima Thule BBS" "ram:utfiles.txt"
```

1.45 Included scripts/CheckDataBase.br

CheckDataBase.br

Checks the messages in one or ALL Systems. If the DELETE switch is used, all messages that are impossible to read will be marked as deleted.

1.46 Included scripts/CheckUserDataBase.br

CheckUserDataBase.br

Checks the the users in one or ALL Systems. If the DELETE switch is used, all users that are impossible to read will be marked as deleted.

1.47 Included scripts/ClearFileDataBase.br

ClearFileDataBase.br

Delete all fileareas in the file database on a System.

1.48 Included scripts/Filedescr2Comment.br

Filedescr2Comment.br

Search for the description of a file in THOR's filedatabase and add the description to the file as a filecomment. The script takes one argument on the commandline, and that is the path to a directory. All files in this directory, with a matching filecomment, will be searched for in the filedatabase of the BBS they were downloaded from, and their comment will be updated. The files must have been downloaded with NComm using the "Transfer/Options/File Comments" option.

Requirements:

- ~at least THOR 2.0 Pre1
- ~the existing filecomment should look like this:
" <bbsname>, <cpsvalue> cps" (NComm style)
- ~an existing and complete filedatabase
- ~the names of the BBSes must be the same in THOR and in NComm
- ~list, delete and filenote system commands in the searchpath

1.49 Included scripts/JoinConfs.br

JoinConfs.br

This arexx script is ment to be used when a conference changes name, and you have both the old and new conference in your database. It will add all messages in the new named conference to the old named

conf. Afterwards, the old named conference is renamed to the name of the new named conference.

1.50 Included scripts/PackBBS.br

PackBBS.br

This script will purge and pack (with Xpk settings you might have defined) all or a single System given on the command line.

1.51 Included scripts/PackData.br

PackData.br

This script will let you use the BBSREAD Arexx command PACKDATAFILE from cli. See the BBSREAD documentation for PACKDATAFILE documentation for further information.

1.52 Included scripts/Reply2Enter.br

Reply2Enter.br

This script will convert all active reply events on a system to enter msg events. This can be useful if the remote system has changed or packed conferences so that the events can't be sent as reply events anymore.

Replaces THOR:bin/Reply2Enter from previous 1.2x distributions.

1.53 Included scripts/SetLastRead.br

SetLastRead.br

This script will set the last read pointers for a ABBS system and is only useful on ABBS systems and conjunction with the supplied NComm script.

1.54 Command Reference/FSE

ARexx commands in the FSE

General Commands

HIDE	Hides the editor
NOP	Do nothing
QUIT	Quit the editor without any requester

REFRESH	Redraw the window
REVEAL	Bring back the editor after hiding it
Status Commands	
CURRENTCHAR	Returns the character under the cursor
FSESCREENNAME	Returns the name of the screen this FSE resides on
GETFSEREVISION	Returns the revision of the editor
GETFSEVERSION	Returns the version of the editor
GETLINE	Returns the contents of a line
ISFIRSTCALL	Returns first-call status
LINELENGTH	Returns maximum number of columns
MAXMSGLENGTH	Returns maximum number of lines
MSGFILENAME	Returns the filename of the file in the FSE
MSGLENGTH	Returns number of lines in this file
RECIPIENT	Returns the receiver of the message you're writing
RECIPIENTADDR	Returns the receivers address of the message
SENDER	Returns the sender of the message in the FSE
SENDERADDR	Returns the senders address of the message
SENDERDATE	Returns the date and/or time of the message
CONFERENCE	Returns the original conference of the message
XPOS	Returns cursor X-value
YPOS	Returns cursor Y-value
AUTOINDENT	Changes and returns the AutoIndent status
OVERWRITE	Changes and returns the Overwrite/Insert status
FSE Specific Commands	
BACKSPACE	The same as a backspace-press
BEGOFFILE	Moves cursor to beginning of file (1,1)
BEGOFLINE	Moves cursor to the first char on the current line
DELETELINES	Delete a number of lines
ENDOFFILE	Moves cursor to the end of the file
ENDOFLINE	Moves cursor to the end of the current line
INCLUDEFILE	Include specified or requested file
INSERTBUFFERLINES	Insert a number of lines from the buffer
INSERTINPUT	Insert string asynchronously
INSERTLINES	Insert one or more empty lines
INSERTSTRING	Insert string synchronously
INSERTTAG	Ask THOR for a tag to include
NEWLINE	The same as a <return> press
PASTECLIP	Paste clipboard contents at current cursor position
PLAYMACRO	Play macro number 1-20 asynchronously
SAVEFILE	Save file to specified or requested name
SETPOS	Move cursor to a specific position
Requester Commands	
REQUESTNOTIFY	Open a requester with text and gadgets
REQUESTSTRING	Request a string from the user

1.55 FSE/General/Hide

The HIDE command

Template:

,

Purpose:

Hide the editor.

Specifications:

The next ARexx-command sent will wake it up again.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

HIDE

1.56 FSE/General/Quit

The QUIT command

Template:

,

Purpose:

Quit the FSE.

Specifications:

Will quit without asking for confirmation to cancel the message.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

QUIT

1.57 FSE/General/Refresh

The REFRESH command

Template:

,

Purpose:

Redraw the FSE window.

Specifications:

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

REFRESH

1.58 FSE/General/Reveal

The REVEAL command

Template:

,

Purpose:

Bring back the editor after it has been hid with the HIDE command.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

HIDE

REVEAL

1.59 FSE/Information/CurrentChar

The CURRENTCHAR command

Template:

,

Purpose:

Return the character under the cursor.

Specifications:

Remember that the command may return "".

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
CURRENTCHAR
char = result
```

```
REQUESTNOTIFY TEXT '''char''' BT '_Ok''
```

1.60 FSE/Information/FSEScreenName

The FSESCREENNAME command

Template:

,

Purpose:

Return the name of the screen this FSE resides on.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
FSESCREENNAME
screenname = result
```

```
REQUESTNOTIFY TEXT '''screenname''' BT '_Ok''
```

1.61 FSE/Information/GetFSERevision

The GETFSEREVISION command

Template:

,

Purpose:

Return the revision of the editor.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
GETFSEVERSION
version = result
```

```
GETFSEREVISION
revision = result
```

```
REQUESTNOTIFY TEXT '""' || 'FSE version: ...
' || version || '.' || revision' "" BT '"_Ok"'
```

1.62 FSE/Information/GetFSEVersion

The GETFSEVERSION command

Template:

,

Purpose:

Return the version of the editor.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
GETFSEVERSION
version = result
```

```
GETFSEREVISION
revision = result
```

```
REQUESTNOTIFY TEXT '""' || 'FSE version: ...
```

```
'||version||.||revision'' BT '_Ok''
```

1.63 FSE/Information/GetLine

The GETLINE command

Template:

```
LINENUMBER
```

Purpose:

Return the contents of a line.

Specifications:

Returns the contents of the line number given on the commandline, or the current one if none specified.

Result:

Will return 0 on success and 10 on failure.

Warning:

If there are no lines in the message, or the line requested is out of range.

Example:

```
options results
```

```
address 'THOR_FSE.01'
```

```
/* Output contents of current line */
```

```
GETLINE
```

```
say result
```

```
/* Output contents of line number 10 */
```

```
GETLINE 10
```

```
say result
```

1.64 FSE/Information/IsFirstCall

The ISFIRSTCALL command

Template:

```
,
```

Purpose:

Determine whether this is the first time the editor is called on this message.

Specifications:

Returns:

"ON" : This is the first time the editor is called on this message (i.e. you may want to quote it.)

"OFF" : Edit message etc.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

options results

address 'THOR_FSE.01'

ISFIRSTCALL

if result = "OFF" say 'This is NOT the first time ...
this file is being edited.'

else say 'This IS the first time this file is being ...
edited.'

1.65 FSE/Information/LineLength

The LINELENGTH command

Template:

,

Purpose:

Return the number of characters in the current line.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

options results

address 'THOR_FSE.01'

LINELENGTH

say 'This line is '||result||' characters long.'

1.66 FSE/Information/MaxMSGLength

The MAXMSGLength command

Template:

,

Purpose:

Return the maximum number of lines in the FSE.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

MAXMSGLENGTH

say 'This editor has a maximum of '||result||' line(s).'

1.67 FSE/Information/MSGFileName

The MSGFILENAME command

Template:

,

Purpose:

Return the filename of the file currently being edited.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

MSGFILENAME

say 'The name of this file is '||result

1.68 FSE/Information/MSGLength

The MSGLENGTH command

Template:

,

Purpose:

Return number of lines in the file currently being edited.

Specifications:

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

MSGLENGTH

say 'This file has '||result||' line(s).'

1.69 FSE/Information/Recipient

The RECIPIENT command

Template:

,

Purpose:

Return the name of the receiver of the message in the editor.

Specifications:

Result:

Will return 0 on success and 10 on failure.

Warning:

In a normal reply-situation, SENDER=RECIPIENT if it isn't forwarded.

Example:

```
options results
address 'THOR_FSE.01'
```

RECIPIENT

INSERTSTRING 'This message is addressed to '||result

1.70 FSE/Information/RecipientAddr

The RECIPIENTADDR command

Template:

,

Purpose:

Returns the address of the receiver of the message in the editor.

Specifications:

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
RECIPIENTADDR
```

```
INSERTSTRING 'This message is addressed to the address '||result
```

1.71 FSE/Information/Sender

The SENDER command

Template:

,

Purpose:

Return the sender of the message in the editor.

Specifications:

The name returned by this command will be the name of the person who wrote the message that this message is a reply to. When replying a message in THOR this name will be the same as that returned by the RECIPIENT command. When the message is forwarded the names may differ.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
SENDER
```

```
if(result ~= "") then INSERTSTRING 'This message is a ...
reply to a message written by~'||result
else exit
```

1.72 FSE/Information/SenderAddr

The SENDERADDR command

Template:

,

Purpose:

Return the address to the sender of the message in the editor.

Specifications:

The address returned by this command will be the address of the person who wrote the message that this message is a reply to. When replying a message in THOR this address will be the same as that returned by the RECIPIENTADDR command. When the message is forwarded the addresses may differ.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
SENDERADDR
if(result ~= "") then INSERTSTRING 'This message is a ...
reply to a message written by~' || result
else exit
```

1.73 FSE/Information/SenderDate

The SENDERDATE command

Template:

,

Purpose:

Return the date and/or time at which the message in the editor was written.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
```

```
address 'THOR_FSE.01'  
  
SENDERDATE  
INSERTSTRING 'This message is dated: '||result
```

1.74 FSE/Information/Conference

The CONFERENCE command

Template:

,

Purpose:

Returns the original conference of the message in the FSE.

Specifications:

The conference name returned here is the conference that the message you are replying to was originally in.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results  
address 'THOR_FSE.01'  
  
CONFERENCE  
if(result ~= "") then do  
    INSERTSTRING 'Reply to a message in conference~'||result  
end  
else exit
```

1.75 FSE/Information/XPos

The XPOS command

Template:

,

Purpose:

Return cursor X-value.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
XPOS
INSERTSTRING 'The cursor is now on character ...
'||result||' on the line.'
```

1.76 FSE/Information/YPos

The YPOS command

Template:

,

Purpose:

Return cursor Y-value.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
YPOS
INSERTSTRING 'The cursor is now on line number ...
'||result||'.'
```

1.77 FSE/Information/AutoIndent

The AUTOINDENT command

Template:

STATUS

Purpose:

Change and/or return current AutoIndent status.

Specifications:

Valid specifiers are ON=Y=YES and OFF=N=NO. Returns ON or OFF depending on the current status.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
AUTOINDENT
INSERTSTRING 'The current AutoIndent status is: ...
' || result
```

```
AUTOINDENT OFF
INSERTSTRING 'AutoIndent is now OFF'
```

1.78 FSE/Information/OverWrite

The OVERWRITE command

Template:

STATUS

Purpose:

Change and/or return current Overwrite/Insert status.

Specifications:

Valid specifiers are ON=Y=YES and OFF=N=NO. Returns ON or OFF depending on the current status.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
options results
address 'THOR_FSE.01'
```

```
OVERWRITE
if(result = "OFF") then INSERTSTRING 'Insert is ON.'
else if(result = "ON") then INSERTSTRING 'Overwrite is ON.'
```

1.79 FSE/SpecificCommands/BackSpace

The BACKSPACE command

Template:

NUMBER

Purpose:

Delete the character to the left of the current cursor position.

Specifications:

NUMBER is the number of backspaces to insert.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

options results

address 'THOR_FSE.01'

/* Delete from cursor to left margin */

XPOS

BACKSPACE result

1.80 FSE/SpecificCommands/BegOfFile

The BEGOFFILE command

Template:

,

Purpose:

Move cursor to the beginning of the file.

Specifications:

Opposite of the ENDOFFILE command.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

BEGOFFILE

1.81 FSE/SpecificCommands/BegOfLine

The BEGOFLINE command

Template:

,

Purpose:

Move cursor to the first character on the current line.

Specifications:

Note that this does not include whitespace. Opposite of the ENDOFLINE command.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

BEGOFLINE

1.82 FSE/SpecificCommands/DeleteLines

The DELETEDLINES command

Template:

LINES

Purpose:

Delete a number of lines.

Specifications:

LINES is the number of lines to delete.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

/* Delete ten lines */

DELETEDLINES 10

1.83 FSE/SpecificCommands/EndOfFile

The ENDOFFILE command

Template:

,

Purpose:

Move cursor to the position behind the last character in the text.

Specifications:

Opposite of the BEGOFFILE command.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

ENDOFFILE

1.84 FSE/SpecificCommands/EndOfLine

The ENDOFLINE command

Template:

,

Purpose:

Move cursor to the position behind the last character on the current line.

Specifications:

Opposite of the BEGOFLINE command.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

ENDOFLINE

1.85 FSE/SpecificCommands/IncludeFile

The INCLUDEFILE command

Template:

NAME

Purpose:

Include file.

Specifications:

If no NAME is specified, a filerequester will open requesting a name from the user.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
address 'THOR_FSE.01'

INCLUDEFILE 's:startup-sequence'

/* Request file from user */

INCLUDEFILE
```

1.86 FSE/SpecificCommands/InsertBufferLines

The INSERTBUFFERLINES command

Template:

LINES

Purpose:

Insert lines from the buffer.

Specifications:

Inserts lines deleted with CTRL-Y or DELETELINES. Same as CTRL-B. LINES is the number of lines to insert.

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
address 'THOR_FSE.01'

/* Delete ten lines and reinsert them five lines below */

DELETELINES 10
SETPOS Y 5
INSERTBUFFERLINES 10
```

1.87 FSE/SpecificCommands/InsertInput

The INSERTINPUT command

Template:

STRING/A/F

Purpose:

Insert string asynchronously

Specifications:

The string will be inserted as if it were typed at the keyboard and may include control-codes like CTRL-B, CTRL-QY etc. Strings inserted with INSERTINPUT will be noticed by i.e. macro-learning (as opposed to INSERTSTRING).

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

INSERTINPUT 'String....'

1.88 FSE/SpecificCommands/InsertLines

The INSERTLINES command

Template:

LINES

Purpose:

Insert one or more empty lines at the current cursor position (same as CTRL-O).

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

INSERTLINES 7

1.89 FSE/SpecificCommands/InsertString

The INSERTSTRING command

Template:

STRING/A/F

Purpose:

Insert the string supplied synchronously.

Specifications:

Will not be noticed by i.e. macro-learning (as opposed to INSERTINPUT).

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

INSERTSTRING 'String!'

1.90 FSE/SpecificCommands/InsertTag

The INSERTTAG command

Template:

,

Purpose:

Ask THOR for a tag and include it at current cursor position.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

address 'THOR_FSE.01'

ENDOFFILE

INSERTTAG

1.91 FSE/SpecificCommands/NewLine

The NEWLINE command

Template:

,

Purpose:

The same as a <return> press.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
address 'THOR_FSE.01'
```

```
/* ten new lines */
```

```
do i = 1 to 10
```

```
    NEWLINE
```

```
end
```

1.92 FSE/SpecificCommands/PasteClip

The PASTECLIP command

Template:

,

Purpose:

Paste clipboard contents at current cursor position.

Specifications:

-

Result:

Will return 0 on success and 10 on failure.

Warning:

-

Example:

```
address 'THOR_FSE.01'
```

```
    PASTECLIP
```

1.93 FSE/SpecificCommands/PlayMacro

The PLAYMACRO command

Template:
 MACRO

Purpose:
 Play macro number 1-20 asynchronously.

Specifications:
 MACRO is the macro number to play.

Result:
 Will return 0 on success and 10 on failure.

Warning:
 -

Example:
 address 'THOR_FSE.01'

 PLAYMACRO 9

1.94 FSE/SpecificCommands/SaveFile

The SAVEFILE command

Template:
 FILENAME

Purpose:
 Save file. If no name specified, requests one from user.

Specifications:
 -

Result:
 Will return 0 on success and 10 on failure.

Warning:
 WARN is returned when an error has occurred.

Example:
 address 'THOR_FSE.01'

 /* save file as 't:temp' */

 SAVEFILE 't:temp'

 /* let the user choose filename */

 SAVEFILE

1.95 FSE/SpecificCommands/SetPos

The SETPOS command

Template:

X/K,Y/K

Purpose:

Move cursor to a specific position.

Specifications:

Note that 0 is a valid argument that does not affect the cursor. You may also leave out any arguments.

Result:

Will return 0 on success and 10 on failure.

Warning:

A WARN is returned if any of the coordinates are out of range.

Example:

```
address 'THOR_FSE.01'
```

```
/* place cursor on char number 10 on line 2 */  
SETPOS X 10 Y 2
```

```
/* moves cursor to (10,10) */  
SETPOS 10,10
```

```
/* moves cursor to position 10 on this line */  
SETPOS 10
```

```
/* moves cursor to line 10 with unchanged X-value */  
SETPOS 0 10  
SETPOS Y 10
```

1.96 FSE/Requesters/RequestNotify

The REQUESTNOTIFY command

Template:

TEXT/A,BUTTONTEXT=BT/A

Purpose:

Will open a requester with some information text in it, specified to the command.

Specifications:

TEXT can contain upto 300 characters and will be processed like the signature in THOR. This means that variables like \$ver and \$time can be used and will be translated. Max number of characters is 300, this is after any variables you use have been expanded, so with \$time and \$ver the limit for the

text in is about 280 characters.

BUTTONTEXT is the text to be in the gadgets and the shortcut can be set with '_' in front of the key that is to act as a shortcut. The text for each gadget is separated with a '|', and it's possible to define 1 or more gadgets.

Result:

If the command is not known to THOR or wrong parameters are specified, the command will return 10. The command will return 0 when the user chooses a gadget, and result will contain the number of the gadget selected. The value in result will be 1 (TRUE) for leftmost (positive) response, then each consecutive response will return 1 more, the rightmost (false) response will return 0 (FALSE) in result.

Warning:

-

Example:

```
options results
address 'THOR.01'
```

```
THORTOFRONT
```

```
REQUESTNOTIFY TEXT '"The time is now: $time\n\nOK will return ...
1, MAYBE will return 2\nand CANCEL will return 0"' BT ...
'"_OK|_MAYBE|_CANCEL"'
```

```
retval = result
```

```
if(retval = 0) then do
    say 'User pressed CANCEL'
end
```

```
if(retval = 1) then do
    say 'User pressed OK'
end
```

```
if(retval = 2) then do
    say 'User pressed MAYBE'
end
```

1.97 FSE/Requesters/RequestString

The REQUESTSTRING command

Template:

```
TITLETEXT=TITLE/A,BUTTONTEXT=BT/A,INITIALSTRING=ID,MAXCHARS/N/A
```

Purpose:

Will open a requester which accepts any string input you give it, upto MAXCHARS number of characters.

Specifications:

BUTTONTEXT is the text to be in the gadgets and the shortcut can be set with '_' in front if the key that is to act as a shortcut. The text for each gadget is separated with a '|', and it's possible to define 1 or more gadgets, although more than two gadgets won't do any good. If a single gadget is used, the command can only be canceled by entering no text in the gadget. 2 gadgets, no more and no less, is highly recommended for this requester.

Result:

If nothing is entered, the requester is canceled or fails for some other reason, the command will return 10. Will return 0 on success and result will contain the string entered.

Warning:

Note that the MAXCHARS can't be higher than 1000.

Example:

```
options results
address 'THOR.01'
```

```
THORTOFRONT
```

```
REQUESTSTRING TITLE 'Enter something:' BT '_Ok|_Cancel' ID 'Some text'
MAXCHARS 40
```

```
say result
```

1.98 FSE/General/NOP

The NOP command

Template:

```
,
```

Purpose:

Do nothing but return a value.

Specifications:

The NOP (NO-oPeration) instruction is provided to control the binding of ELSE clauses in compound IF statements.

Result:

Will return 0 on success and 10 on failure.

Warning:

```
-
```

Example:

```
address 'THOR_FSE.01' /* or 'THOR.01' */

if i = j then /* First (outer) IF */
  if j = k then a = 0 /* Inner IF */
  else NOP /* Binds to inner IF */
```

```
else a = a + 1    /* Binds to outer IF */
```