

**xdme\_cmds**

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> xdme_cmds		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		November 24, 2024	

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>xdme_cmds</b>	<b>1</b>
1.1	XDME commands overview	1
1.2	AREXX	1
1.3	Block	2
1.4	Blockoperations	2
1.5	Buildin integer math	3
1.6	Commandshell	3
1.7	Control structures	3
1.8	GadtoolsBox	4
1.9	General	4
1.10	IO Operations	4
1.11	Input recorder	5
1.12	Keyboard	5
1.13	Menu Manipulation	5
1.14	Misc	6
1.15	Move in the text	6
1.16	Preferences and Settings	7
1.17	Requester	8
1.18	Search and Replace	8
1.19	Specal Features	9
1.20	Stack	9
1.21	Text Editing	9
1.22	Text formatting	10
1.23	Using and Defining Macros	10
1.24	Variable	10
1.25	Variables	11
1.26	Windows	11
1.27	MAP,UNMAP	11
1.28	MENUADD	13
1.29	FORCE	13

1.30 MENUON,MENUOFF,MENUADD . . . . .	14
1.31 CTAGS . . . . .	14
1.32 REF . . . . .	14
1.33 SCANF . . . . .	15
1.34 REPEAT . . . . .	16
1.35 IF,WHILE,IFELSE . . . . .	17
1.36 INDENT . . . . .	18
1.37 TLATE . . . . .	19
1.38 SETGEOMETRY . . . . .	19
1.39 GOTO . . . . .	19
1.40 REPLACE . . . . .	19
1.41 PROJECTINFO . . . . .	20
1.42 SELECT . . . . .	20
1.43 PICK,PUSH,POP . . . . .	20
1.44 OPENWINDOW . . . . .	21
1.45 JUSTIFY,UNJUSTIFY . . . . .	21
1.46 BREAKOUT . . . . .	22
1.47 SETDEFTITLE,SETDEFICONTITLE . . . . .	24
1.48 READTEMPLATE . . . . .	25
1.49 APPICON . . . . .	25
1.50 COMMANDSHELL . . . . .	26
1.51 DEFLIST,DROPLIST,ADDNODE,REMNODE . . . . .	27
1.52 GTB,LOADGTBPROJECT,DROPGTBPROJECT,OPENGT . . . . .	28
1.53 SPC . . . . .	29
1.54 Index . . . . .	30

# Chapter 1

## xdme\_cmds

### 1.1 XDME commands overview

This file contains a complete list of all commands of XDME sorted alphabetically and by topic.

- AREXX
- Block
- Blockoperations
- Buildin integer math
- Commandshell
- Control structures
- GadtoolsBox
- General
- IO Operations
- Input recorder
- Keyboard
- Menu Manipulation
- Misc
- Move in the text
- Preferences and Settings
- Requester
- Search and Replace
- Specal Features
- Stack
- Text Editing
- Text formatting
- Using and Defining Macros
- Variable
- Variables
- Windows

### 1.2 AREXX

PORT name cmd	Send cmd to ARexx-Port name
PROJECTINFO	Gives some information about the current project.
RX	ARexx macro, no args (RX macname)
RX1	ARexx macro, one arg (RX1 macname arg1)

---

RX2	ARexx macro, two args (RX2 macname arg1 arg2)
RXRESULT any	Copy any into RESULT in an AREXX-script.
SELECT what	make a window the current one.

### 1.3 Block

AUTOUNBLOCK what	clear old selection when a new block is defined (on) or give an error
BAPPENDSAVE file	Append currently marked text to file
BARPSAVE	save the block to a file with filerequester
BCOPY	insert currently marked text before cursor
BDELETE	delete currently marked text
BEND	Set end of block
BLOCK	Set start or end (if start is already set) of block
BLOCKTYPE type	type is LINE (old), CHARACTER (like everywhere else) or VERTICAL (rectangular block).
BMOVE	delete currently marked text and insert it before cursor line
BSAVE file	save the block to file
BSOURCE	execute currently marked text block as if it were a script file
BSTART	Set start of block
CLIPINS	Insert current contents of clipboard in the text
COPY	copy currently marked text into clipboard
LINEBLOCK	mark the current line
POPMARK	pop the block stack and highlight the popped block
PURGEMARK	clear the mark stack
PUSHMARK	push the currently marked block onto a stack and unhighlight the block
SWAPMARK	PUSHMARK, swap top two marks on stack, POPMARK
UNBLOCK	clear the block markers for the current window

### 1.4 Blockoperations

AUTOUNBLOCK what	clear old selection when a new block is defined (on) or give an error
BAPPENDSAVE file	Append currently marked text to file
BARPSAVE	save the block to a file with filerequester
BCOPY	insert currently marked text before cursor
BDELETE	delete currently marked text
BEND	Set end of block
BLOCK	Set start or end (if start is already set) of block
BLOCKTYPE type	type is LINE (old), CHARACTER (like everywhere else) or VERTICAL (rectangular block).
BMOVE	delete currently marked text and insert it before cursor line
BSAVE file	save the block to file
BSOURCE	execute currently marked text block as if it were a script file
BSTART	Set start of block
CLIPINS	Insert current contents of clipboard in the text
COPY	copy currently marked text into clipboard

---

LINEBLOCK	mark the current line
POPMARK	pop the block stack and highlight the popped block
PURGEMARK	clear the mark stack
PUSHMARK	push the currently marked block onto a stack and unhighlight the block
SWAPMARK	PUSHMARK, swap top two marks on stack, POPMARK
UNBLOCK	clear the block markers for the current window

## 1.5 Buildin integer math

ADD var val	add val to the value of var
DEC var	decrement the value of var
DIV var val	divide the value of var with val
INC var	increment the value of var
MATH1 arg1 arg2	long version for NOT INC NEG DEC; \$INFIXMODE decides if arg1 or arg2 is operator, the other arg is variablename
MATH2 arg1 arg2 arg3	long version for MUL MOD DIV SUB ADD; \$INFIXMODE decides if arg1 or arg2 is operator, the other arg is variablename
MOD var val	modulo divide the value of var with val
MUL var val	multiply the value of var with val
NEG var	negate the value of var
NOT var	logical not for the value of var
SUB var val	sub val from the value of var

## 1.6 Commandshell

CLOSECMDSHELL	close the command shell
CMDSHELLOUT txt	output a string to the command shell
OPENCMDSHELL	open the command shell with the filename in \$CMDSHELLNAME

## 1.7 Control structures

ABORT	abort the current command execution
BREAK	break out of the current loop (WHILE, REPEAT)
CONTINUE	skip to the end of the current loop (WHILE, REPEAT)
EVAL command	reinvoke the command interpreter; that command can be used to split long commandsequences to keep MAXIA small
EXECUTE comm	Execute a CLI command.
FORCE flags command	set special conditions for executing command;
IF cnd act	if (cnd) act
IFELSE cnd ifact elseact	if (cnd) ifact else elseact
PROJECTLOAD	Recall session
PROJECTSAVE	Save all window-dimensions, filenames and position of iconified windows.
REPEAT cnt comm	repeat comm cnt times.
SCANF ctlstr	scan the string at the current text position (C scanf) example: scanf %s
SOURCE file	source a script file. '#' in first column for comment
UNABORT	clear the ABORT flag (only in an ARexx script)

WHILE cnd act      while (cnd) act

## 1.8 GadtoolsBox

ACTIVATEGTBWINDOW project window open a window of a GTB project  
 CLOSEGTBWINDOW project window close a window of a GTB project  
 CONNECTGTBGADGET prj win gad list connect a gadtoolsgadget with a list  
 DROPGTBPROJECT project free the resources needed for a GTB Project  
 LOADGTBPROJECT project filename read a GTB .GUI File  
 OPENGTBWINDOW project window open a window of a GTB project  
 SETGTBGADGET project window gadget value set another value to a GTB gadget

## 1.9 General

(text)              enter text as if typed  
 key      execute a keymap as a macro (example -return)  
 header-item execute a menu item as a macro (example: Project-Save)  
           (case independent)

## 1.10 IO Operations

XDME will always save the text at the place where it came from, expect you have used the CD command to change the current directory.

APPENDSAVE file	Append current text to file
ARPINSFILE	INSFILE with filerequester
ARPLoad	NEWFILE with filerequester
ARPSAVE	SAVEAS with filerequester
BAPPENDSAVE file	Append currently marked text to file
BARPSAVE	save the block to a file with filerequester
BSAVE file	save the block to file
CD dir	set directory of current window to dir
CHFILENAME name	change the name of the working file
EXECUTE comm	Execute a CLI command.
INSFILE name	insert a file into the current text.
KEYLOAD filename	replace the current keymap with the contents of filename
KEYSAVE filename	save the current keymap into filename
MENULOAD filename	replace the current menustrip with the one from the file
MENUSAVE filename	write the current menustrip in a file
NEWFILE name	replace current text with new file
PATTERN pat	sets the pattern for the filerequesters.
PRINT text	Print text to the shell XDME was started in
PROJECTLOAD	Recall session
PROJECTSAVE	Save all window-dimensions, filenames and position of iconified windows.
READTEMPLATE filename	read in a file and replace all occurrences of \$(varname) with the contents of that varname
REQFONT	SETFONT with ReqTools fontrequester
REQINSFILE	INSFILE with ReqTools filerequester
REQLOAD	NEWFILE with ReqTools filerequester

---



SAVEAS file	save current text under a different name (title line name does change)
SAVECONFIG	save current editor configuration to s:XDME.prefs
SAVEOLD	save current text under current name
SOURCE file	source a script file. '#' in first column for comment
WRITETO file	write text to this file. The current name of the text is not changed.

## 1.11 Input recorder

RECEMD	end macro recording
RECPLOY	replay previously recorded macro
RECSAVE file	save previously recorded macro to a file. Execute with SOURCE
RECSTART	start macro recording
SIMPTR x y	simulate the mousemovement to windowpos x/y (pixels); that command is needed to replay saved macros, it is not helpful in any other situation

## 1.12 Keyboard

KEYLOAD filename	replace the current keymap with the contents of filename
KEYSAVE filename	save the current keymap into filename
MAP key map	map a key to a keymap
NEWKEYTABLE name	use a keytable or create a new one
REMKEYTABLE	delete the current keytable, if it is not the only one
UNMAP key	unmap a key
USEKEYTABLE name	search for a certain keytable and use it as the current one

## 1.13 Menu Manipulation

MENUADD hdr item cmd	add menu item
MENUCHKITEM menuname itemname variablename	write the current status (0 or 1) of an checkmarks in a variable
MENUCLEAR	delete entire menu
MENUDEL hdr item	delete menu item
MENUDELHDR hdr	delete menu header
MENULOAD filename	replace the current menustrip with the one from the file
MENUOFF	disable menus (multiple calls are stacked)
MENUON	This command will enable menus.
MENUSAVE filename	write the current menustrip in a file
MENUSETITEM menuname itemname status	set the status of a menu-item with checkmarks
NEWMENUSTRIP name	use a menustrip or create a new one
REMMENUSTRIP	delete the current menustrip, if it is not the only one
USEMENUSTRIP name	switch to menustrip name

## 1.14 Misc

ABOUT	display information about XDME
ESC	toggle manual command entry mode
ESCI MM arg	go into command entry mode prompting with arg
EXECUTE comm	Execute a CLI command.
NOP	no operation
NULL	no operation
PRINT text	Print text to the shell XDME was started in
QUITALL	leave XDME. If any text was modified, a safety check is performed for that text
RECALL	recall most recently entered command. Must be used from a keymap (c-esc).
REDISPLAY	force XDME to redraw the text
REM com	add comment
REQREPLACE	display replace requester ((c) 1994 by Karl Lukas)
UNDELIN E	insert most recently deleted line (only last line saved)
UNDO	undo current line (must be mapped to a key to work)

## 1.15 Move in the text

BACKTAB	backward tab
BOTTOM	Move to Bottom of File
COL n	Move cursor to column n or n characters left (-n) or right (+n)
DOWN	cursor down. If in commandline move to next line of commandline-history
DOWNADD	cursor down. If at bottom of text, add a line.
FIRST	move to column 1
FIRSTNB	Move to first non-blank in line.
GOTO dest	Goto to a position in the text.
LAST	move one beyond the last non-space in a line.
LEFT	cursor left
MAKECURSORVISIBLE	Scrolls an oversized screen so the cursor will become visible.
MATCH	find matching paren. Works with (), [], {}, '' and C-comments
PAGEDOWN	pagedown a partial page (see PAGESET)
PAGELEFT	page to the left as requested by StyleGuide.
PAGERIGHT	ditto to the right
PAGEUP	pageup a partial page (see PAGESET)
PING n	set a text marker (n = 0-9).
PONG n	move to a previously set text marker (n = 0-9)
RETURN	if AUTOINDENT is off: (FIRST DOWNADD) else insert line, split current line and indent like last line above.
RIGHT	cursor right
SCREENBOTTOM	Move cursor to the bottom of the screen.
SCREENTOP	Move cursor to the top of the screen
SCROLLDOWN	Scroll down without moving cursor
SCROLLLEFT	Scroll left without moving cursor
SCROLLRIGHT	Scroll right without moving cursor
SCROLLUP	Scroll up without moving cursor
TAB	forward tab
TOMOUSE	moves cursor to mouse position

TOP	Move to Top of File
UP	cursor up. If in commandline, move to previous line of commandline-history
WLEFT	move to beginning of previous word. If in the middle of a word, move to beginning of current word.
WRIGHT	move to beginning of next word

## 1.16 Preferences and Settings

what is one of on, off or toggle to switch the option on, off or to change its state.

ADDPATH path	Add the specified symbolic directory to XDME's special search path (see REF and CTAGS).
AUTOINDENT what	(De)Activate autoindent with RETURN
AUTOSPLIT what	(De)Activate autosplit. This is an alternative to WORDWRAP. AUTOSPLIT only breaks the line if it gets too long and doesn't touch the rest of the text.
AUTOUNBLOCK what	clear old selection when a new block is defined (on) or give an error
BBPEN pen	selects pen as the block-background-pen
BGPEN pen	set background pen for text
BLOCKTYPE type	type is LINE (old), CHARACTER (like everywhere else) or VERTICAL (rectangular block).
CHFILENAME name	change the name of the working file
DEBUG what	For programmers only Allows to set a flag for testing code
DOBACKUP what	specifies if XDME creates a .bak file before actually saving the text
FGPEN pen	Set pen for text
FOLLOWCURSOR what	XDME will make sure the cursor is visible if you switch it on with this command. Usefull on screens that extend over the visual area.
GLOBAL what	turn global search on/off. If XDME cannot find a string in one window, it will continue with the next one.
HGPEN pen	set highlight (block) pen
ICONACTIVE what	Should XDME activate the iconified window
IGNORECASE what	set case ignore for seaches.
INSERTMODE what	set INSERTMODE.
MAP key map	map a key to a keymap
MARGIN n	set WordWrap and paragraph formatting margin (related to WORDWRAP and REFORMAT)
MODIFIED what	set modified flag manually (what={on,off,toggle})
NICEPAGING what	Should PAGEUP and PAGEDOWN scroll the page immediately (on) or jump to the border first
PAGESET n	n PERCENT (0 to 100). page step size relative to the current number of rows in the window.
PATTERN pat	sets the pattern for the filerequesters.
PUBSCREEN name	open next window on screen name. Use an empty string to turn it off (ie. "pubscreen ``")
REMPATH path	Remove a directorys from XDME's special path.
RESIZE cols rows	Resize current window. E.G: (resize 70 23)
SAVECONFIG	save current editor configuration to s:XDME.prefs
SAVETABS what	Optimize file saves by crunching spaces to tabs. The default is OFF.

SETDEFICONTITLE string Sets the pattern for the window-title when iconified  
 SETDEFTITLE string Sets the pattern for the window-title.  
 SETFONT font sz Set the window's font. setfont topaz 11  
 SETPARCOL col Set the LEFT margin for word wrap mode paragraphing & reformat. MUST be less than MARGIN.  
 SIZEWINDOW geo change size and position of the current window to geo  
 SLINE what Should XDME not allow to go beyond the end of line and preserve the length of lines (default: no)  
 SPACING n Insert a gap of n pixels between lines  
 TABSTOP n Set tab stops every n. does not effect text load.  
 TASKPRI n Set the priority of XDME to n (-5..5)  
 TBPEN pen set pen for title bar background  
 TFPEN pen set pen for title bar text  
 TITLE title set window title manually  
 UNMAP key unmap a key  
 WORDWRAP what set word wrap mode (related to MARGIN)

## 1.17 Requester

ARPFONT SETFONT with fontrequester  
 ARPINSFILE INSFILE with filerequester  
 ARPLOAD NEWFILE with filerequester  
 ARPSAVE SAVEAS with filerequester  
 BARPSAVE save the block to a file with filerequester  
 PATTERN pat sets the pattern for the filerequesters.  
 REQFILE title flags defvalue display a synch ReqTools FileRequest; the result is put in \$REQRESULT.  
 REQFONT SETFONT with ReqTools fontrequester  
 REQINSFILE INSFILE with ReqTools filerequester  
 REQLOAD NEWFILE with ReqTools filerequester  
 REQNUMBER title format gadgets defvalue min max display a synch ReqTools NumberRequest; the result is put in \$REQRESULT.  
 REQPALETTE title defvalue display a synch ReqTools PaletteRequest; the result is put in \$REQRESULT.  
 REQSTRING title format gadgets defvalue display a synch ReqTools StringRequest; the result is put in \$REQRESULT.  
 REQUEST title body gadgets display a synch ReqTools EZRequest; the result is put in \$REQRESULT.

## 1.18 Search and Replace

FIND string Set the search pattern to string and do a NEXT  
 FINDR s1 s2 Set find and replace patterns and do one find&replace.  
 FINDSTR string Set the search string pattern  
 GLOBAL what turn global search on/off. If XDME cannot find a string in one window, it will continue with the next one.  
 NEXT find next occurance of search pattern  
 NEXTR find next occurance and replace  
 PREV find previous occurance of search pattern  
 PREVR find previous occurance and replace  
 REPLACE replaces the next strlen(findstr) chars with repstr  
 REPSTR string SET the replace string pattern

## 1.19 Specal Features

APPICON	The AppIcon Interface
BREAKOUT	Some words about Variable Expansion
COMMANDSHELL	The CommandShell Interface
GTB	The GadToolsBox Interface

## 1.20 Stack

DROPVAR var	remove the last pushed occurency of the variable var from the variable stack
PEEK item	like POP, but doesn't remove the topmost element from stack !
PICK item	like POP, but doesn't remove the topmost element from stack !
PICKVAR var	restore the last pushed contents of the variable var from the variable stack without modifying the variable stack
POP item	Pop something from the stack and store it in item. The special item AUTO stores the thing back where it was taken from.
POPMARK	pop the block stack and highlight the popped block
POPVAR var	restore the last pushed contents of the variable var from the variable stack and remove it
PURGEMARK	clear the mark stack
PURGEVAR var	remove all occurencies the variable var from the variable stack
PUSH item	Push an item on the stack.
PUSHMARK	push the currently marked block onto a stack and unhighlight the block
PUSHVAR var	push the contents of the variable var onto the variable stack
SWAP item	exchange the topmost item on stack with the actual item
SWAPMARK	PUSHMARK, swap top two marks on stack, POPMARK
SWAPVAR var	swap the contents of a variable with that of its last pushed entry in the variable stack

## 1.21 Text Editing

BACK	backspace, (delete char to left of cursor)
BS	backspace, (delete char to left of cursor)
DEL	delete, (deletes char under cursor)
DELIN	delete line
DELINES n	delete n lines
DOWNADD	cursor down. If at bottom of text, add a line.
INDENT what how	indent text. what specifies what to indent and how how to indent it.
INSERT text	insert some text at the current position ignoring \$INSERTMODE
INSFILE name	insert a file into the current text.
INLINE	insert line
INSLINES n	insert n lines at once

JOIN	join next line to line at cursor
JUSTIFY how	simple text justification.
OVERWRITE text	overwrite text at the current position ignoring \$INSERTMODE
PRINTF format	parameters create a string with printf-style format and its (up to 8) parameters and write it into the current text
READTEMPLATE filename	read in a file and replace all occurrences of \$(varname) with the contents of that varname
REFORMAT	reformat paragraph using the margin.
REMEOL	Remove text under and beyond the cursor.
RETURN	if AUTOINDENT is off: (FIRST DOWNADD) else insert line, split current line and indent like last line above.
SPLIT	Split line at cursor
TLATE how	Modify character under cursor.
UNJUSTIFY	removes extra spaces in a line

## 1.22 Text formatting

JUSTIFY how	simple text justification.
REFORMAT	reformat paragraph using the margin.

## 1.23 Using and Defining Macros

MACROLOAD name	load commandmacros from a file
MACROSAVE filename	save all commandmacros into a file with a special format
RET	terminate a macro (before reaching its end)
SETMACRO name nargs	body create/modify the commandmacro name with nargs arguments
SETMACROVAR name value	create/modify a macrolocal variable inside a macro
SMV name value	short for SETMACROVAR
UNSETMACRO name	delete the commandmacro name
UNSETMACROVAR name	deletion of a macro's local variable

## 1.24 Variable

DROPVAR var	remove the last pushed occurrence of the variable var from the variable stack
FLAG name what	change flag name by what
INSVAR var where value	Insert a string into the variable var at position where;
PICKVAR var	restore the last pushed contents of the variable var from the variable stack without modifying the variable stack
POPVAR var	restore the last pushed contents of the variable var from the variable stack and remove it
PURGEVAR var	remove all occurrences the variable var from the variable stack
PUSHVAR var	push the contents of the variable var onto the variable stack
REMVAR var where len	Delete len characters from the variable var at

---

	position where;
SET var str	create/modify an internal variable
SETENV var str	create/modify an enviroment variable (ENV:)
SETMACROVAR name value	create/modify a macrolocal variable inside a macro
SETTOGGLE flag	flip toggle entry flag = 0..255 t0..t31
SETTOGGLE flag	set toggle entry flag = 0..255 t0..t31
SETTOGGLE flag	clear toggle entry flag = 0..255 t0..t31
SETTVAR var str	create/modify a text-local variable
SMV name value	short for SETMACROVAR
SPC var value	Modify an internal XDME system variable
SWAPV var1 var2	try to swap the contents of 2 variables
SWAPVAR var	swap the contents of a variable with that of its last pushed entry in the variable stack
UNSET var	delete an internal variable
UNSETENV var	delete an enviroment variable (ENV:)
UNSETMACROVAR name	deletion of a macro's local variable
UNSETTVAR var	delete a text-local variable

## 1.25 Variables

## 1.26 Windows

ACTIVATEWINDOW	Make the active Textwindow active for Intuition
ICONIFY	iconify the window
NEWWINDOW	open new window using default window parameters
OPENWINDOW geo	open new window using specified geometry.
QUIT	close current window. If text was modified, a safety check is performed
REDISPLAY	force XDME to redraw the text
RESIZE cols rows	Resize current window. E.G: (resize 70 23)
SELECT what	make a window the current one.
SETDEFICONTITLE string	Sets the pattern for the window-title when iconified
SETDEFTITLE string	Sets the pattern for the window-title.
SETGEOMETRY x y width height	Set x/y position and width/height of XDME's window.
SHOWLOG	XDME collects all warnings internally. These can now be showed again with this command.
SIZEWINDOW geo	change size and position of the current window to geo
TITLE title	set window title manually
TOBACK	Move active window to back
TOFRONT	Move active window to front
UNICONIFY	uniconify the window

## 1.27 MAP,UNMAP

MAP key map  
UNMAP key

with these commands you can modify the keytable.

MAP adds or modifies an existing key-entry to the keytable

UNMAP deletes an existing key-entry

key consists of a set of qualifiers, a "-" and one code (both case sensitive)

The qualifiers are:

```
s == any shift (caps-lock with alpha-keys)
c == control
a == any alt
A == any amiga
L == left mouse button
M == middle mouse button
R == right mouse button
u == key-up ( release of a key )
x == extended qualifier x
y == extended qualifier y
z == extended qualifier z
r == repeat
```

the code-specification is one out of:

```
main keyboard:
[single lowercase character] == the key on you main keyboard - that shows it
[single uppercase alpha]    == the key on .... but that means a set s-qualifier

numeric keypad:
nk0 .. nk9      == numerik keypad 0..9
nk+, nk-, nk*, nk/, nk. == numerik keypad +,-,*,/,.
```

There are two more keys on your numeric keypad. With american or british mapping, they are used as nk( and nk). With most european mappings they are used as nk[ and nk]. With canadian mapping they are used as nk@ and nk\textdegree{} If there is a standart mapping, that does map these keys in another way, they are NOT supported yet. (please mail)

```
other special keys:
f1 .. f10    == Function keys
hel          == help-key
esc          == escape
del          == delete
bac, bs     == backspace ( <- )
tab          == tabulator
ent          == enter
spc, spa     == spacebar
up, dow, rig,
lef         == cursor keys

lmb, mmb, rmb == mouse buttons
mmo          == mouse movement with a mousebutton pressed
?m2, ?m3, ?m4 == (? in [lmr]) double/triple/quadruple clicks of a
               certain mouse button (times out!)
```

Note that as mousebuttons can act for qualifying as well as for specifying, the specifying mousebutton must also be qualifying e.g L-rmb doesn't work, but LR-rmb does.



Note that certain keys may be used as dead-keys; keys that are deadening when pressed without qualifiers are not accessible with this keyboard-interface. e.g. on german keyboards there is a key "'" next to "\" which cannot be accessed. On the other hand, if a key is deadening on a qualified level, but not when pressed without qualifiers, you can remap it e.g. on german keyboards alt-f is deadening, but -f not, so you can access a-f; and if you map a-f to any function, it loses its deadening function for [X]DME

EXAMPLE: <alt-f> <a> creates something like "á"  
 after a "MAP a-f (NULL)" the same key-sequence  
 creates a normal "a"

Keys with the x, y or z qualifiers set can't be accessed immediately. First you must set the extended qualifiers (see QUALIFIER command), then you may press the remaining qualifiers and keys. These qualifiers were introduced to allow something like the CTL-X prefix of (Micro)Emacs.

ATTENTION !

We check only the first three characters of the code-specification so e.g. -space is recognized as -spa. some people call this a feature, but in fact it is a BUG, as this method makes problems with rexx-commands which start with these 3 first characters. XDME will ignore such commands and do whatever this key is supposed to do.

## 1.28 MENUADD

MENUADD hdr item cmd

SUBnames are splitted at ^S,  
 AMIGAShortcuts are splitted at ^A,  
 CHECKItems are leaded by ^C (at subs the last/sub name),  
 BARItems are called ^B (at subs the last/sub name)

EXAMPLES:

```
menuadd demo demo^Stest^AW Add "demo/demo/test" with Hotkey
menuadd demo demo^S^B Add subitembar
menuadd demo demo^S^Ctest2 Add subitem "test2" with checkmark
menuadd demo ^B Add itembar
```

## 1.29 FORCE

FORCE flags command

To execute a command under special conditions. flag is

T	don't change title
S	quiet (no Screen updating)
F	unable to fail
R	no Requestors

```

D          enable debugging mode
`          simple eval

```

### 1.30 MENUON,MENUOFF,MENUADD

```

MENUADD hdr item cmd
MENUOFF
MENUON

```

This command will enable/disable menus. Users who have a whole bunch of MENUADD commands in their .EDRC should note that disabling menus at the beginning will speed up the MENUADD commands. Then reenable menus at the end. These calls are stackable in that if you call MENUOFF, say, twice, it will take two MENUON@ commands to restore menus. The reverse is not true.

### 1.31 CTAGS

CTAGS

(1.30B and beyond) Compatible with Aztec's CTAGS program. This command searches for the subroutine name under the cursor in the associated tags file ("tags" in the directory holding the file currently being edited). Also, the file "tags" in directories specified by the special XDME path (see ADDPATH and REMPATh) will be searched.

If the tag is found, it loads the file the subroutine resides in if necessary, then WindowToFront()'s the window, ActivateWindow()'s it, and GOTO's the line where the subroutine starts. The search within the source file is anchored to the left column. If the file is already loaded, it is not reloaded.

This enables a programmer to quickly trace subroutines over an arbitrary number of files.

The tags file contains one or more lines of the following format:

```

subroutine-name file-name /^search-pattern
      (that's a slash and a carrot, then the pattern)

```

```

-- EXAMPLE of 'TAGS' file:
setpen cmd1.c /^setpen(
do_up cmd1.c /^do_up(
--

```

SPECIAL NOTE: CTAGS will work even if you are not in the directory containing the file. You can thus place a tags file in the directory containing the files it references, and the filenames WITHIN the tags file need not be a full path.

### 1.32 REF

## REF

(1.28d and beyond). This is a very powerful new command that allows you to bring up a reference to a keyword with a single keystroke. This is useful for programmers who have on-line documentation or fully commented include files. XDME opens a window just big enough to fit the reference.

NOTE: The new CTAGS command may be more suited to your application.

The reference keyword is the alpha-numeric string currently under the cursor. REF will search the file DME.REFS in the directories listed by the special XDME path (see ADDPATH and REMPATh). The file must be built by the user and each line has the following format:

```
(keyword) (nolines/endstring) (file) (searchstr/@@seekpos)
```

Surrounding the keywords with `` or () is optional if the keyword does not contain spaces.

```
keyword      keyword under cursor
nolines/ends  either a number (the number of lines in the reference)
              or a string denoting the end of the reference when found.
file         the file containing the reference material
searchstr/@@  search string in file that indicates the beginning of
              the reference, or two at's (@@) and the seek position
              in decimal (like @@2343). The latter method is used
              mainly for reference- generator programs.
```

Upon finding a successful keyword match the specified file is opened and the search string searched for. If a seek position was specified no search is made and a seek is made to the beginning of the reference. If found, the indicated number of lines (if a number is specified for <nolines/ends>) or until a match with the endstring (if a string was specified) will be placed in a temporary file and a new XDME window brought up. The temporary file is then deleted.

When looking for matches, the compare is anchored at the beginning of each line in the file. Thus, any spaces in front of the string in the file must be duplicated.

T: must be assigned to a temporary directory, usually RAM: See the included example DME.REFS file. The most common things referenced are the autodocs and commented include files.

Some modification of the included DME.REFS files may be required due to differences in include file and autodoc format.

## 1.33 SCANF

### SCANF ctlstr

This is equivalent to the C scanf() function with the restriction that only one conversion is allowed, that conversion being a string. Thus:

```
scanf %s      will place the string under the cursor in the variable
               $scanf

scanf %4s      The first four chars of the string.

scanf %[0123456789]
               will scan the string while it contains
               specified chars (e.g. scan a number)

scanf %[~,]    will scan the string until it finds a ', '.
```

Moreover SCANF now allows to use some more usefull arguments:

```
w - one word
l - the WHOLE line
b - all that matches isalnum()
c - single character
t - text in one line
a - all that matches isalpha()
r - c-style comment
```

If you precede one of these with a '+' you will also get everything to the left else you get all that is to the right. Example:

```
dummy1[line] = ...
^
```

```
w: my1[line]
+w: dummy1[line]
a: my
+a: dummy
b: my1
+b: dummy1
c: m
+c: m
```

The variable \$scanf may be used as an argument in any command. Example: (infile \$scanf).

NOTE: If using \$scanf in a macro, you probably want to precede it with a \ to prevent it from being evaluated at macro-creation time.

## 1.34 REPEAT

REPEAT cnt comm

Repeat arg n times. Apart from being a number, n can also be one of:

```
line    Current line # (lines begin at 1)
lbot    #lines to the bottom, including current line
cleft   column # (cols begin at 0)
cright  #chars to eol, including current char under cursor
tr      #char positions to next tab
tl      #char positions to next back tab
```

---

Certain commands can abort a REPEAT loop. Specifically, any FIND[R], NEXT[R], or PREV[R] in which the search string is NOT found will abort a REPEAT. Most operations which can go out of bounds, such as UP, LEFT, RIGHT, DOWN, also abort a repeat.

Specifying -1 as n causes REPEAT to go on forever (well, actually, 0xFFFFFFFF times) or until an abort.

REPEAT may also be abbreviated. Simply type

```
15 left
```

This will move the cursor 15 times left. In this construction you must specify a number as repeat-count.

## 1.35 IF,WHILE,IFELSE

```
IF cnd act
IFELSE cnd ifact elseact
WHILE cnd act
```

If the specified condition is true, execute the argument. For WHILE, the argument is executed until the condition is false (be careful!), but you may abort while with CTRL-c (be fast !).

the optional '!' inverts the logic.

Conditions:

```
#      if toggle entry # is SET.  there are 256 toggles (0..255)
t      if On line 1
b      if On last line
l      if At column 0
r      if At end of line (spaces below and beyond)
m      if Text has been modified
i      if in insert mode
x[<=>]# if column position (starts at 1) is (any OR combo of
    <, =, or >) than some number.  Example:  x<=20
y[<=>]# if Line number (starts at 1) is (same as for x)
cl     character under cursor is lower case alpha
cu     character under cursor is upper case alpha
ca     character under cursor is alpha-numeric
cn     character under cursor is numeric
cb     cursor within a block
c[<=>]# character under cursor is ascii code # (# in decimal)
    optional conditionals as in 'x' and 'y'.
# can also be a "string":
```

```
if c="hwlllo" `right tlate "e"'
```

replaces "hwlllo" by "hello". The condition is true, if the cursor is on the first char of the string and the string follows behind:

```
This is a test.
      ^
```

```
ifelse c="is" `title c=is' `title c<>is':    c=is
ifelse c="test" `title TRUE' `title FALSE':  FALSE
```

```
# can be a group of chars:
```

```
while c=[a-zA-Z0-9] ...
```

```
is the same as
```

```
while ca ...
```

```
Beware of spaces in strings and groups: if there are any, you
MUST NOT forget to put the condition in XDME's parentheses.
```

## 1.36 INDENT

INDENT what how

The INDENT-command allows to indent and outdent text. It is intelligent in a way that you will NEVER lose any text with it. If the in/outdent would push characters over the limit (left or right), the line is left and INDENT continues with the next line.

```
what:  .      Current line
      n      line n
      $n     line which is marked by PING n
      t      whole text
      b      whole block
      bs     block start
      be     block end
      _      last line
```

You may create ranges with "what,what". (This is obviously useless with "t" and "b").

```
how:   -      If there is a minus, text is outdented instead of
          indented
      .      Align to multiple of indent-amount instead of just
          inserting some spaces at the beginning
      n      Set indent-amount to n
      t      Set indent-amount to tabsize
      c      Insert not at the beginning but at the current column.
```

Examples:

```
indent . .ct  Acts like tab. Text under cursor and beyond is
              aligned to next tabstop.
indent b .t   Indent the current block. NOTE: ALL lines are
              tabstop-aligned ! This may destroy some of
              your formating.
indent b -.t  dito but the block is outdented.
```

## 1.37 TLATE

TLATE how

translate character under cursor. how can be one of the following:

number     Replace character by the character with the code number.  
(i.e. 65=A). Number can be a decimal-, octal- or hexnumber.

[+-]number     As above, but the actual character is used as offset:  
'tlate +1' makes a 'B' from an 'A', 'tlate -1' does the  
reverse. The resulting char-code is truncated to 8bit.

"x"             Replace actual character by 'x'.

[Uu]     Make character uppercase.

[Ll]     Make character lowercase.

## 1.38 SETGEOMETRY

SETGEOMETRY x y width height

Set x/y position and width/height of XDME's window. The width and height are ignored in iconified state. If you use negative values, the positions/sizes are relative the current screen-size (0 0 -1 -1 will open a full-sized window !). If a size is null, it's left unchanged (move window only).

## 1.39 GOTO

GOTO dest

Goto to a position in the text. dest is one of

BLOCK/START     Beginning of block (if there's one)  
END             Last line of block (dito)  
n             to line n  
+n             n lines down  
-n             n lines up

## 1.40 REPLACE

REPLACE

replaces the next strlen(findstr) chars with repstr (ie. if deletes as many chars as are in findstr and replaces them by the replace-string). Useful in a mapping to find the text and another to actually replace the text:

```
map f1 `next`
map f2 `replace next`
```

f1 searches for the text, f2 replaces it and looks for the next occurrence.

## 1.41 PROJECTINFO

PROJECTINFO

Gives some information about the current project.

```
result_string="%s %d %d %d %d %d %d",
name, LeftEdge, TopEdge, Width,
Height, IWinX, IWinY
```

i.e. Name of current window, it's dimensions and position when iconified

## 1.42 SELECT

SELECT what

make a window the current one. what:

```
FIRST, LAST, NEXT, PREVIOUS    obvious, eh ?
WINDOW=name                    Make window name the current one
SAVE                           Remember current window
LOAD                           restore current window
```

You can SAVE the current window, select another (or more) and LOAD the current window again. This guarantees uninterrupted work for the user and is STRONGLY recommended.

## 1.43 PICK,PUSH,POP

```
PICK item
POP item
PUSH item
```

Push/pop an item on/from the stack. Items are

```
POS    - actual cursor position (like PING)
MODIFIED - modified flag
ICONMODE - iconify-state
TABSTOP - tab-size
SAVETABS - Should I convert spaces to tabs ?
MARGIN  - right margin
INSERTMODE - obvious :-)
IGNORECASE - for search
WORDWRAP - word-wrapping on/off
WWCOL   - col. of wordwrap
```

---



```

WINDOW    - position and size of window
ICON      - position of icon
PENS      - all pens
BLOCK     - like PUSH-/POPMARK
ED        - actual window. Like SELECT SAVE, but can be nested
AUTO      - POP only ! This automatically put the topmost thing from
stack to its original place.
DISCARD   - POP only ! Discards the topmost item from the stack.

```

## 1.44 OPENWINDOW

OPENWINDOW geo

open new window using specified geometry. Geometry is specified as: +/-leftedge+/-topedge+/-width+/-height, where negative numbers denote values relative to the width or height of the screen. For example, the following opens a nearly full-screen window leaving 10 pixels above, below, to the left, and to the right:

```
openwindow +10+10-10-10
```

The following opens a window in the upper right hand corner of width 320 and height 100.

```

openwindow -320+0+320+100
openwindow -320+0-0+100      (same thing)
openwindow -320-100-0-0      (lower right hand corner)

```

## 1.45 JUSTIFY,UNJUSTIFY

JUSTIFY how  
UNJUSTIFY

These commands format a single line of text. UNJUSTIFY removes all formatting (ie. all obsolete spaces). JUSTIFY inserts spaces. The following formats are available:

```

full      insert spaces between words until the last character
           in the line is at MARGIN.
left      removes all leading spaces
right     inserts spaces at the beginning of the line until the
           last character in the line is at MARGIN.
center    center line between left border and MARGIN.

```

If you want to reformat a whole paragraph, mark it as a block and use

```

GOTO BLOCK      goto beginning of paragraph
WHILE CB (      while in block ...
    JUSTIFY LEFT remove leading spaces
    UNJUSTIFY   remove all interword spaces
    DOWN )      next line

```

```

GOTO BLOCK      goto beginning of paragraph again
REFORMAT       collect words anew
WHILE CB (      while in block
    JUSTIFY how  justify the line
    DOWN )      next line

```

## 1.46 BREAKOUT

### BREAKOUT

It seems that some words must be said to variable expansion ...  
the bad thing is, I have not written the function, that's doing the  
expansion, so i cannot guarantee, that the following text is absolutely  
correct ...

The current Version of XDME's macro interpreter expands variables in  
preparation of command calls; furthermore, the macro language does not  
know anything else than commands ... (ok, there are other things  
than only commands: abbreviated commands (quoted text for write,  
and numbers for repeat ...), macros (which are treated like commands)  
and ARexx, but these themes are not discussed here ...)  
even "constructs" like IF, WHILE, REPEAT are commands, and so they  
have also arguments ... and there is no difference in the evaluation  
of their arguments compared to other commands ...

(in the following section we precede each example line with "%"  
and we use a non-existing command called "out", so the following  
2 macrodefinitions are useful when testing the examples ...  
the first one just ignores the leading "%" and the second displays  
its argument in the next line; the mapping one lets XDME  
send the current line to its macro interpreter)

```

% setmacro % 0 ()
% setmacro out 1 (firstnb down insline tab (-> \$arg1) title OK)
% map a-a (eval \$currentline)

```

let us suppose we had done the following variable assignments ...

```

% set alpha xx
% set cmp[1] Amiga
% set cmp[2] Atari
% set cmp[3] Clone
% set best 1
% set quality best

```

- \* A Variable name may contain only alphanumeric chars and/or "-", "\_" if You wanna use other charcters inside a variablename, it must be enclosed with parantheses or Single Quotation marks ( "(...)" or "'...'" both ways are called 'quotes' in the next paragraphs).
- \* The Interpreter currently has knowledge of something like 4 classes of characters, that are alphanumeric chars together with "-" and "\_" which make up continous blocks of text, whitespace (for XDME this is always SPACE (0x20), since TAB (0x07) ist translated to 0x20 when reading files ...) Special characters ( "\", "(", ")", "\"", "'", "\$" ) and all other chracters;  
as long as no special character is invloved, we can say Whitespace

is used as delimiter for arguments;

```
% out ah.that/is*very;interesting
-> ah.that/is*very;interesting
```

as soon as special charcters are involved the situation gets hairy ...

- "\" are ignored, instead the nex caracter looses all of its meanings and is just copied to the current argument ('escaping') so we can say

```
% out now\ we\ build\ a\ long\ string\ \w/\ whitespace\ and\ "$x"
-> now we build a long string w/ whitespace and "$x"
```

in order to get a "\", that caharcter must actually be doubled

```
% out \\
-> \
```

- "\$" introduces the next variable; as stated above, a variable name may contain only alphanumeric chars and/or "-", "\_", else it must be enclosed w/ quotes

```
% out $quality
-> best
% out $(cmp[1])
-> Amiga
% out $cmp[1]
-> $cmp[1]
(assuming 'cmp' in an unset variable))
```

lonely "\$" or sequences of "\$" like "\$\$" will probably disturb the variable expansion, (try to expand a variable of no name) so the following macro might break ("might" since this behaviour might be changed one day)...

```
% out $$
(probably no output ...)
```

- "(" and ")" as well as "\"" and "'" can disable the whitespace argument splitting ... ( "quoting" )

```
% out (hey, now w/out espases)
-> hey, now w/out espases
```

these quotes can also be stacked, but inside of "(/)" "\"/'" will loose their meaning and vice versa

```
% eval (out ( hello ( hahah ) ` )) out ( ' hohoho )
-> hello ( hahah ) `
-> ' hohoho
```

if a open-quote has no conterpart, it quotes the complete rest of the current string

```
% out ( sim sala
-> sim sala
```

if an close-quote has no counter part, the same as above;  
the tricky thing is: a leading close quote cannot have a  
matching open-quote

```
% out ) now we can type whatever we want ... ( ` ' ' )
-> now we can type whatever we want ... ( ` ' ' )
```

\* If a variable is unknown to the system, or it cannot be resolved due to other reasons, e.g. to low memory conditions, it is expanded to itself; additionally it may happen, that the variable Module also sets the abortflag (this behaviour is currently not defined, so it might depend on the variable name being used)

\* You can ask, if a variable exists by preceding its name with a questionmark ( "?" )  
so in the above example we could call  
% out \$(?alpha)  
-> 1  
(we need quotes, since a questionmark is else treated as a breaking (non-alnum) character ...)

\* if You want to expand nested variables, You must reinvoke the interpreter; for that purpose, You can use the command EVAL; please note, that - as stated above - also IF and WHILE are commands; for that reason, it might be necessary to put a lot of escapes in highly nested macros ...

```
% out ( the $(cmp[$$quality]) is the $quality )
-> the $(cmp[$best]) is the best
(probably no output, see above ...)
```

```
% eval (out ( the \$(cmp[\$$quality]) is the $quality ))
-> the $(cmp[1]) is the best
```

```
% eval (eval (out ( the \\$(cmp[\\$quality]) is the $quality )))
-> the Amiga is the best
```

Additional comment: when using AmigaGuide (v34) the above text may have many duplicated backslashes; this is caused by the fact that Multiview (v39f) does treat the backslash as a special character, so I had to choose between using single backslashes for AmigaGuide v34 which then are invisible for Multiview, or using them duplicated for v34 what is the right way for Multiview ...

## 1.47 SETDEFTITLE,SETDEFICONTITLE

```
SETDEFICONTITLE string
SETDEFTITLE string
```

This command allows to specify a pattern from which XDME will build a string and display it at the appropriate position. The string can contain any characters (like in printf()). The following characters are replaced by

a special string, however:

Sequence	Replacement
%%	A single % in the resulting string
%l	the current line
%L	the number of lines
%c	the current column
%C	the code of the character under the cursor in hex
%m	modified flag (either - or *)
%f	the current filename
%p	the last 20 characters of the current path
%b	the actual blocktype (L for line, N for character oriented and V for vertical

The defaults for XDME's title are: %l/%L %C %c %m %f %i

The defaults for XDME's icon are: %f

## 1.48 READTEMPLATE

READTEMPLATE filename

This command inserts file.

passing all lines starting with "\$\$" to EVAL (lines starting with "\$\$#" are ignored) and replacing all "\$(...)" and "\$'...'" by their values if matching variables do exist.

That function might be very useful for handling very formalistic structs, which do need only little work by user.

Since the parse is line-oriented, usage of linefeeds inside "\$(...)" and "\$'...'" is forbidden; however expanded values might contain linefeeds

## 1.49 APPICON

APPICON

That Package allows use of a Workbench AppIcon.

Currently the whole package is controlled via some special variables, not via commands;

the following five variables are used:

\$appicon - (BOOL) the status of the appicon:  
setting it to "1" makes the AppIcon appear,  
setting it to "0" makes the AppIcon disappear.

\$appiconname - (FILE) the icon to be used for the AppIcon Image;

plase note, that modification of that variable currently only has effect after the NEXT appearance of the AppIcon, the visible Image is not changed.  
Defaults to "XDME".

\$appicontitle - (STRING) the titlestring to be used in connection with the AppIcon (the same limit as for \$appiconname)  
Defaults to "XDME".

\$appicondropaction - (COMMAND) the command to be executed, whenever another icon is dropped onto the AppIcon; any "%s" in that variable are expanded to the full name of the dropped icon (w/ sprintf).  
Defaults to "newwindow newfile '%s'".

\$appiconclickaction - (COMMAND) the command to be executed, whenever user doubleclicks on the AppIcon.  
Defaults to "newwindow arpload".

## 1.50 COMMANDSHELL

### COMMANDSHELL

That Package allows use of an ansynchroneous Commandshell, a simple Console window to type commands into.

Most aspects of that package can be controlled via variables, but there are also 3 commands:

OPENCMDSHELL - makes the commandshell appear;

CLOSECMDSHELL - makes the commandshell dissappear;

CMDSHELLOUT text - write some to the commandshell

the following special variables complete the package:

\$cmdshell - (BOOL) the status of the commandshell;  
setting it to "1" makes the Commandshell open,  
setting it to "0" makes the Commandshell close.

\$cmdshellfile - (FILE) the file to be used for the commandshell;  
that file must be interactive.  
plase note, that modification of that variable currently only has effect after the NEXT open of the commandshell, the active commandshell is not changed.  
Defaults to "CON:0/11/640/60/XDME Command Shell/Close".

\$cmdshellprompt - (STRING) the string to displayed to signal the user, that he can enter a command;  
plase note, that modification of that variable currently only has effect after the NEXT update of the prompt, i.e. after the next time, something was written to the commandshell, or the user pressed return in the cmdshell;

the active prompt is not changed.  
 The prompt is expanded (via variable-expansion) each time,  
 it is displayed  
 Defaults to "`XDME> "

\$errorsoncmdshell - (BOOL) as long as that flag is set, and  
 the commandshell is open, all errors are displayed on the  
 commandshell.

\$warningsoncmdshell - (BOOL) as long as that flag is set, and  
 the commandshell is open, all warnings are displayed on the  
 commandshell.

## 1.51 DEFLIST,DROPLIST,ADDNODE,REMNODE

ADDNODE list where name value  
 DEFLIST name  
 DROPLIST name  
 REMNODE list where

preliminary interface to xdme lists ...

"/" and "#" are special characters in a meaning, that they must  
 not be used in the name of a list or a node (in fact "/" is  
 used as a separator between listname and nodename, and "#" is  
 a special name to indicate numbered access )

There are currently some ReadOnly system lists:

- "\*RefPaths\*" - list of the Paths to check with tags/refs  
 use ADDPATH/REMPATH to modify
- "\*MenuStrips\*" - list of available Menustrips  
 use NEWMENUSTRIP/DELMENUSTRIP to modify
- "\*KeyTables\*" - list of available Keytables  
 use NEWKEYTABLE/DELKEYTABLE to modify

a "\*AppIcons\*" list is to be introduced as soon, as multiple  
 Appicons are allowed ...

the Lists GTBProjects, MenuItems and all user defined Lists  
 are prepared for nested usage, but this feature is not yet  
 enabled

Currently the lists are pretty unusable (except for use in  
 connection with the GTB module, since it is possible to  
 connect any list with a number of GTB Listview gadgets,  
 which are automatically updated if the list is changed ...)

The User interface to Modifyable lists contains currently:

"DEFLIST name" and "DROPLIST name" for rootlevel list  
 management, and "ADDNODE list where name value" and  
 "REMNODE list where" for node manipulation; in that  
 case "where" can be "tail", "head", a number (perhaps  
 preceeded by "idx=") or "name=" followed by the  
 name of an entry in the list ...

Variable Interface is done via the following mechanism:

`$(<list>/<node>/*Value*)` -> a node's value (if exists)

`$(<list>/<node>/*Name*)` -> a node's name (always)

please note, that a name starting with a "#" is internally treated as a number, so "#z" is Node no.0 and "#10" is node number 10 - this indexname is also usable after the "name=" directive in ADD/REM-NODE or as a Name in the variable access ... (so You can e.g. say "title (First Text is `$( *Texts* \/#0 \/*Name* )`")

## 1.52 GTB,LOADGTBPROJECT,DROPGTBPROJECT,OPENGT

DROPGTBPROJECT project

GTB

LOADGTBPROJECT project filename

Interface To enable XDME to load and basically display (not everything is currently handled correctly) GadToolsBox Projects (namely ".GUI" Files)

the user has the possibility to draw his own asynch. requesters and load and display them from within XDME; each time, a menu is selected or a gadget is Selected (Select-UP), a XDME Command is called;

the command is currently build in the form

`projectname"-windowname"-gadgetname` "value

for gadgets and

`projectname"-windowname"-menuname` "value

for menus; value is the State of the toggle for a togglemenu, the active label for a listview, and so on; for button-gadgets and non-toggle menuitems, value is "".

(the format may be changed with the `$gtbformat` variable)

however I think about a more intuitive way of connecting variables, gadgets and menus, so that macro-solution might be removed in the near future ...

It is already possible to connect Listview Gadgets with Lists (see the Lists section ... ahem where? =8-}) so that every change of a List is immediately shown in the connected Listviews (sorry, but the Cycle Gadgets are not yet supported nor any other structure than Lists and LV-Gadgets, but this is a planned enhancement)

\*WARNING\* the GTB module is BETA and it is certainly not bulletproof - You should think twice about what You are doing ...

\*BUG\* (or not?) it is not defined whatever happens, if there is that Boopsi Pop-Image used inside a GTB Project, so better make sure it is not used ...

\*BUG\* (yea it is...) i have not (yet) added Fallback conditions,



to make sure a window does not exceed Screensize... in that concern ... we do currently use the Screenfont also, if a non-topaz font was defined in the GTB Project (Any help how I can check, if the user wanted the GTB Project's font?)

\*BUG\* (probably NOT subject of change) XDME ignores the settings for screens, it always uses the screen of the active TextWindow ...

\*BUG\* there is currently no support for Gadget Shortcuts in a GTB project (any suggestions?)

\*BUG\* there is currently no sufficient menu handling (e.g. setcheck is not possible)

## 1.53 SPC

SPC var value

This command allows access to almost every XDME internal variable, that is, it can replace almost every preferences command; additionally to the functionality of the current prefs commands, it can access some System Variables, which themselves have no preferences commands, like the AppIcon variables, the CmdShell Variables and some more this is a list of the variables that should be settable w/ SPC.

```
activetofront,      appicon,      appiconclickaction,
appicondropaction, appiconname, appicontitle,
autoindent,        autosplit,    autounblock,
bbpen,             bgpen,        block,
cmdshell,          cmdshellname, cmdshellprompt,
currentdir,        debug,        dobackup,
ed,               errorsoncmdshell, fgpen,
filename,          findstr,      followcursor,
globalsearch,      gtbformat,    hgpen,
icon,             iconactive,  iconmode,
icontitle,         ignorecase,  infixmode,
insertmode,        keytable,   margin,
menufontname,      menufontsize, menustrip,
modified,          nicepaging,  norequest,
parcol,            pens,       pos,
repstr,            reqpattern, reqresult,
rexport,          rxresult,   saveicons,
savetabs,          scanf,      shortlines,
showtitle,         simpletabs, sourcebreaks,
tabstop,           tbpen,      tfpen,
viewmode,          warningsoncmdshell, window,
windowcycling,     windowtitles, wordwrap
```

Please note, that each variable settable w/ SPC is also usable in the Varstack; however the ED, POS and BLOCK should be used w/ care.

for the completeness: the following vars cannot be changed w/ SPC

```

ascii,          colno,          comlinemode,
currentline,    currentword,    firstnb,
itemcheck,      lineno,        numlines,
prevnblne,      recentword,     reqresult,
restofline,     rexxport,       txtfontname,
txtfontsize,    version

```

## 1.54 Index

ABORT	abort the current command execution
ABOUT	display information about XDME
ACTIVATEGTBWINDOW	project window open a window of a GTB project
ACTIVATEWINDOW	Make the active Textwindow active for Intuition
ADD var val	add val to the value of var
ADDNODE list	where name value add a node to a list
ADDPATH path	Add the specified symbolic directory to XDME's special search path (see REF and CTAGS).
APPENDSAVE file	Append current text to file
APPICON	The AppIcon Interface
ARPFONT	SETFONT with fontrequester
ARPINSFILE	INSFILE with filerequester
ARPLoad	NEWFILE with filerequester
ARPSAVE	SAVEAS with filerequester
AUTOINDENT what	(De)Activate autoindent with RETURN
AUTOSPLIT what	(De)Activate autosplit. This is an alternative to WORDWRAP. AUTOSPLIT only breaks the line if it gets too long and doesn't touch the rest of the text.
AUTOUNBLOCK what	clear old selection when a new block is defined (on) or give an error
BACK	backspace, (delete char to left of cursor)
BACKTAB	backward tab
BAPPENDSAVE file	Append currently marked text to file
BARPSAVE	save the block to a file with filerequester
BBPEN pen	selects pen as the block-background-pen
BCOPY	insert currently marked text before cursor
BDELETE	delete currently marked text
BEND	Set end of block
BGPEN pen	set background pen for text
BLOCK	Set start or end (if start is already set) of block
BLOCKTYPE type	type is LINE (old), CHARACTER (like everywhere else) or VERTICAL (rectangular block).
BMOVE	delete currently marked text and insert it before cursor line
BOTTOM	Move to Bottom of File
BREAK	break out of the current loop (WHILE, REPEAT)
BREAKOUT	Some words about Variable Expansion
BS	backspace, (delete char to left of cursor)
BSAVE file	save the block to file
BSOURCE	execute currently marked text block as if it were a script file
BSTART	Set start of block
CD dir	set directory of current window to dir
CHFILENAME name	change the name of the working file
CLIPINS	Insert current contents of clipboard in the text

---

---

CLOSECMDSHELL	close the command shell
CLOSEGTBWINDOW	project window close a window of a GTB project
CMDSHELLOUT txt	output a string to the command shell
COL n	Move cursor to column n or n characters left (-n) or right (+n)
COMMANDSHELL	The CommandShell Interface
CONNECTGTBGADGET prj win gad list	connect a gadtoolsgadget with a list
CONTINUE	skip to the end of the current loop (WHILE, REPEAT)
COPY	copy currently marked text into clipboard
CTAGS	search for the tag under the cursor (see below)
DEBUG what	For programmers only Allows to set a flag for testing code
DEC var	decrement the value of var
DEFLIST name	create a list
DEL	delete, (deletes char under cursor)
DELIN	delete line
DELINES n	delete n lines
DIV var val	divide the value of var with val
DOBACKUP what	specifies if XDME creates a .bak file before actually saving the text
DOWN	cursor down. If in commandline move to next line of commandline-history
DOWNADD	cursor down. If at bottom of text, add a line.
DROPGTBPPROJECT	project free the resources needed for a GTB Project
DROPLIST name	delete a list
DROPVAR var	remove the last pushed occurrence of the variable var from the variable stack
ESC	toggle manual command entry mode
ESCI	go into command entry mode prompting with arg
EVAL command	reinvoke the command interpreter; that command can be used to split long commandsequences to keep MAXIA small
EXECUTE comm	Execute a CLI command.
FGPEN pen	Set pen for text
FIND string	Set the search pattern to string and do a NEXT
FINDR s1 s2	Set find and replace patterns and do one find&replace.
FINDSTR string	Set the search string pattern
FIRST	move to column 1
FIRSTNB	Move to first non-blank in line.
FLAG name what	change flag name by what
FOLLOWCURSOR what	XDME will make sure the cursor is visible if you switch it on with this command. Usefull on screens that extend over the visual area.
FORCE flags	command set special conditions for executing command;
GLOBAL what	turn global search on/off. If XDME cannot find a string in one window, it will continue with the next one.
GOTO dest	Goto to a position in the text.
GTB	The GadToolsBox Interface
HGPEN pen	set highlight (block) pen
ICONACTIVE what	Should XDME activate the iconified window
ICONIFY	iconify the window
IF cnd act	if (cnd) act
IFELSE cnd ifact elseact	if (cnd) ifact else elseact
IGNORECASE what	set case ignore for seaches.
INC var	increment the value of var
INDENT what how	indent text. what specifies what to indent and how how to indent it.
INSERT text	insert some text at the current position ignoring

---

```

                                $INSERTMODE
INSERTMODE what                set INSERTMODE.
INSFILE name                   insert a file into the current text.
INLINE                         insert line
INSLINES n                    insert n lines at once
INSVAR var where value Insert a string into the variable var at position
                                where;
JOIN                           join next line to line at cursor
JUSTIFY how                   simple text justification.
KEYLOAD filename              replace the current keymap with the contents of filename
KEYSAVE filename              save the current keymap into filename
LAST                           move one beyond the last non-space in a line.
LEFT                           cursor left
LINEBLOCK                     mark the current line
LOADGTBPROJECT project filename read a GTB .GUI File
MACROLOAD name                load commandmacros from a file
MACROSAVE filename            save all commandmacros into a file with a special format
MAKECURSORVISIBLE            Scrolls an oversized screen so the cursor will become
                                visible.
MAP key map                   map a key to a keymap
MARGIN n                      set WordWrap and paragraph formatting margin (related to
                                WORDWRAP and REFORMAT)
MATCH                         find matching paren. Works with (), [], {}, `` and
                                C-comments
MATH1 arg1 arg2               long version for NOT INC NEG DEC; $INFIXMODE decides if
                                arg1 or arg2 is operator, the other arg is variablename
MATH2 arg1 arg2 arg3          long version for MUL MOD DIV SUB ADD; $INFIXMODE decides
                                if arg1 or arg2 is operator, the other arg is variablename

MENUADD hdr item cmd add menu item
MENCHKITEM menuname itemname variablename write the current status (0 or 1)
                                of an checkmarks in a variable
MENUCLEAR                     delete entire menu
MENUDEL hdr item              delete menu item
MENUDELHDR hdr                delete menu header
MENULOAD filename             replace the current menustrip with the one from the file
MENUOFF                       disable menus (multiple calls are stacked)
MENUON                        This command will enable menus.
MENUSAVE filename             write the current menustrip in a file
MENUSETITEM menuname itemname status set the status of a menu-item with
                                checkmarks
MOD var val                   modulo divide the value of var with val
MODIFIED what                  set modified flag manually (what={on,off,toggle})
MUL var val                    multiply the value of var with val
NEG var                        negate the value of var
NEWFILE name                   replace current text with new file
NEWKEYTABLE name              use a keytable or create a new one
NEWMENUSTRIP name             use a menustrip or create a new one
NEWWINDOW                     open new window using default window parameters
NEXT                           find next occurance of search pattern
NEXTR                         find next occurance and replace
NICEPAGING what               Should PAGEUP and PAGEDOWN scroll the page immediately
                                (on) or jump to the border first
NOP                             no operation
NOT var                        logical not for the value of var
NULL                           no operation
OPENCMDSHELL                  open the command shell with the filename in $CMDSHELLNAME

```

---

---

OPENTGBWINDOW project window open a window of a GTB project  
 OPENWINDOW geo open new window using specified geometry.  
 OVERWRITE text overwrite text at the current position ignoring  
     \$INSERTMODE  
 PAGEDOWN pagedown a partial page (see PAGESET)  
 PAGELEFT page to the left as requested by StyleGuide.  
 PAGERIGHT dito to the right  
 PAGESET n n PERCENT (0 to 100). page step size relative to the  
     current number of rows in the window.  
 PAGEUP pageup a partial page (see PAGESET)  
 PATTERN pat sets the pattern for the filerequesters.  
 PEEK item like POP, but doesn't remove the topmost element from  
     stack !  
     PICK item like POP, but doesn't remove the topmost element from  
         stack !  
 PICKVAR var restore the last pushed contents of the variable var  
     from the variable stack without modifying the variable  
     stack  
 PING n set a text marker (n = 0-9).  
 PONG n move to a previously set text marker (n = 0-9)  
     POP item Pop something from the stack and store it in item. The  
         special item AUTO stores the thing back where it was taken  
         from.  
 POPMARK pop the block stack and highlight the popped block  
 POPVAR var restore the last pushed contents of the variable var  
     from the variable stack and remove it  
 PORT name cmd Send cmd to ARexx-Port name  
 PREV find previous occurrence of search pattern  
 PREVR find previous occurrence and replace  
 PRINT text Print text to the shell XDME was started in  
 PRINTF format parameters create a string with printf-style format and its  
     (up to 8) parameters and write it into the current text  
     PROJECTINFO Gives some information about the current project.  
 PROJECTLOAD Recall session  
 PROJECTSAVE Save all window-dimensions, filenames and position of  
     iconified windows.  
 PUBSCREEN name open next window on screen name. Use an empty string to  
     turn it off (ie. "pubscreen '')")  
 PURGEMARK clear the mark stack  
 PURGEVAR var remove all occurrences the variable var from the  
     variable stack  
     PUSH item Push an item on the stack.  
 PUSHMARK push the currently marked block onto a stack and  
     unhighlight the block  
 PUSHVAR var push the contents of the variable var onto the variable  
     stack  
 QUIT close current window. If text was modified, a safety check  
     is performed  
 QUITALL leave XDME. If any text was modified, a safety check is  
     performed for that text  
     READTEMPLATE filename read in a file and replace all occurrences of  
         \$(varname) with the contents of that varname  
 RECALL recall most recently entered command. Must be used from a  
     keymap (c-esc).  
 RECEND end macro recording  
 REPLAY replay previously recorded macro  
 RECSAVE file save previously recorded macro to a file. Execute with

---

---

	SOURCE
RECSTART	start macro recording
REDISPLAY	force XDME to redraw the text
REF	reference string under cursor (see below)
REFCTAGS	The ultimate command for hopping through source. First, we try CTAGS and if we don't find anything, we check REF.
REFORMAT	reformat paragraph using the margin.
REM com	add comment
REMEOL	Remove text under and beyond the cursor.
REMKEYTABLE	delete the current keytable, if it is not the only one
REMMENUSTRIP	delete the current menustrip, if it is not the only one
REMNODE list	where delete a node inside a list
REMPATH path	Remove a directorys from XDME's special path.
REMVAR var	where len Delete len characters from the variable var at position where;
REPEAT cnt comm	repeat comm cnt times.
REPLACE	replaces the next strlen(findstr) chars with repstr
REPSTR string	SET the replace string pattern
REQFILE title flags	defvalue display a synch ReqTools FileRequest; the result is put in \$REQRESULT.
REQFONT	SETFONT with ReqTools fontrequester
REQINSFILE	INSFILE with ReqTools filerequester
REQLOAD	NEWFILE with ReqTools filerequester
REQNUMBER title format	gadgets defvalue min max display a synch ReqTools NumberRequest; the result is put in \$REQRESULT.
REQPALETTE title	defvalue display a synch ReqTools PaletteRequest; the result is put in \$REQRESULT.
REQREPLACE	display replace requester ((c) 1994 by Karl Lukas)
REQSTRING title format	gadgets defvalue display a synch ReqTools StringRequest; the result is put in \$REQRESULT.
REQUEST title body	gadgets display a synch ReqTools EZRequest; the result is put in \$REQRESULT.
RESIZE cols rows	Resize current window. E.G: (resize 70 23)
RET	terminate a macro (before reaching its end)
RETURN	if AUTOINDENT is off: (FIRST DOWNADD) else insert line, split current line and indent like last line above.
RIGHT	cursor right
RX	ARexx macro, no args (RX macname)
RX1	ARexx macro, one arg (RX1 macname arg1)
RX2	ARexx macro, two args (RX2 macname arg1 arg2)
RXRESULT any	Copy any into RESULT in an AREXX-script.
SAVEAS file	save current text under a different name (title line name does change)
SAVECONFIG	save current editor configuration to s:XDME.prefs
SAVEOLD	save current text under current name
SAVETABS what	Optimize file saves by crunching spaces to tabs. The default is OFF.
SCANF ctlstr	scan the string at the current text position (C scanf) example: scanf %s
SCREENBOTTOM	Move cursor to the bottom of the screen.
SCREENTOP	Move cursor to the top of the screen
SCROLLDOWN	Scroll down without moving cursor
SCROLLLEFT	Scroll left without moving cursor
SCROLLRIGHT	Scroll right without moving cursor
SCROLLUP	Scroll up without moving cursor
SELECT what	make a window the current one.
SET var str	create/modify an internal variable

---

---

```

SETDEFICONTITLE  string Sets the pattern for the window-title when iconified
SETDEFTITLE     string Sets the pattern for the window-title.
SETENV var str   create/modify an enviroment variable (ENV:)
SETFONT font sz  Set the window's font. setfont topaz 11
SETGEOMETRY x y width height Set x/y position and width/height of XDME's
                    window.
SETGTBGADGET project window gadget value set another value to a GTB gadget
SETMACRO name nargs body create/modify the commandmacro name with nargs
                    arguments
SETMACROVAR name value create/modify a macrolocal variable inside a macro
SETPARCOL col    Set the LEFT margin for word wrap mode paragraphing &
                    reformat. MUST be less than MARGIN.
SETTOGGLE flag   flip toggle entry flag = 0..255|t0..t31
SETTOGGLE flag   set toggle entry flag = 0..255|t0..t31
SETTOGGLE flag   clear toggle entry flag = 0..255|t0..t31
SETTVAR var str  create/modify a text-local variable
SHOWLOG          XDME collects all warnings internally. These can now be
                    showed again with this command.
SIMPTR x y       simulate the mousemovement to windowpos x/y (pixels); that
                    command is needed to replay saved macros, it is not
                    helpful in any other situation
SIZEWINDOW geo   change size and position of the current window to geo
SLINE what       Should XDME not allow to go beyond the end of line and
                    preserve the length of lines (default: no)
SMV name value   short for SETMACROVAR
SOURCE file      source a script file. '#' in first column for comment
SPACING n        Insert a gap of n pixels between lines
  SPC var value  Modify an internal XDME system variable
SPLIT           Split line at cursor
SUB var val      sub val from the value of var
SWAP item        exchange the topmost item on stack with the actual item
SWAPMARK        PUSHMARK, swap top two marks on stack, POPMARK
SWAPV var1 var2  try to swap the contents of 2 variables
SWAPVAR var      swap the contents of a variable with that of its last
                    pushed entry in the variable stack
TAB             forward tab
TABSTOP n       Set tab stops every n. does not effect text load.
TASKPRI n       Set the priority of XDME to n (-5..5)
TBPEN pen       set pen for title bar background
TFPEN pen       set pen for title bar text
TITLE title     set window title manually
  TLATE how     Modify character under cursor.
TOBACK         Move active window to back
TOFRONT        Move active window to front
TOMOUSE        moves cursor to mouse position
TOP            Move to Top of File
UNABORT        clear the ABORT flag (only in an ARexx script)
UNBLOCK        clear the block markers for the current window
UNDELIN        insert most recently deleted line (only last line saved)
UNDO           undo current line (must be mapped to a key to work)
UNICONIFY      uniconify the window
  UNJUSTIFY     removes extra spaces in a line
  UNMAP key     unmap a key
UNSET var      delete an internal variable
UNSETENV var    delete an enviroment variable (ENV:)
UNSETMACRO name delete the commandmacro name
UNSETMACROVAR name deletion of a macro's local variable

```

---

---

UNSETTVAR var	delete a text-local variable
UP	cursor up. If in commandline, move to previous line of commandline-history
USEKEYTABLE name	search for a certain keytable and use it as the current one
USEMENUSTRIP name	switch to menustrip name
VCTAGS name	search for tag name
VREF name	reference name
VREFCTAGS name	like REFCTAGS, but looks for name
WHILE  cnd act	while (cnd) act
WLEFT	move to beginning of previous word. If in the middle of a word, move to beginning of current word.
WORDWRAP what	set word wrap mode (related to MARGIN)
WRIGHT	move to beginning of next word
WRITETO file	write text to this file. The current name of the text is not changed.

---