XES

COLLABORATORS

	TITLE : XES	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

XES

1.1 XES 2.2

The XDME Excelsior Suite

by

Fergus Duniho

Please note that this guide is under construction.

Some links go nowhere.

1.2 Introduction

XES is an integrated suite of macros, menus, keymappings, programs, requesters, and ARexx scripts for XDME. XDME is a powerful and configurable programmer's editor based on Matt Dillon's DME. XES gives you the ability to use XDME for much more than just programming. It lets you easily use it for outlining and word processing, and it makes it easier to program in specific languages, such as C, by formatting your code properly. XES is the new name for XDMD, which stood for "XDME Macros by Duniho." I changed the name, because I want to encourage other people to contribute to it. XES is the successor to DMD, which was released for the last time a couple years ago. XES picks up the version numbering from XDME.

1.3 Editing

```
                Regions

                Rectangles

                Clipfiles
                        Clipboard
        Blocks
```

1.4 Requirements

All you need to begin using XES is XDME. This version is designed for XDME 1.84. If you have an earlier version, some of XES's features will not work. You can ftp the latest version of XDME from ftp.fh-konstanz.de in pub/amiga/. Since XDME requires WB 2.04, so does XES.

To make full use of XES's features, you should also have the following:

I. ARexx (for the ARexx scripts)

II. GadToolsBox.library and NoFrag.library (for the GTB requesters)

III. RexxArpLibrary.library and ARP.library (for some of the ARexx scripts)

IV. RexxReqTools.library and ReqTools.library (for Reqtools requesters and for some ARexx scripts)

V. XPKMaster.library, xPack, and various compressor libraries (for the ability to decrypt and load encrypted files, and for the ability to pack or encrypt files when saving)

VI. AlphaSpell -- This is a spelling checker that I have previously distributed with XES. Due to the size of its dictionary, I am distributing it in a separate archive called aspell4.lzx.

VII. Rxgen.library for the screen beeping during interactive spell checking.

I also recommend using xData with XDME. This will let XDME read XPK-packed files. I didn't incorporate this ability into XES, because xData does it more efficiently. You should already have ARexx, and you should be able to find the rest on the Aminet or on a local BBS.

1.5 Regions

A region is the text between the mark and the point. Each window has its own mark and point. The mark is a location you set with c-@ or with c-spc. The point is the location of the cursor in an active window. The region begins at whichever comes first, the point or the mark, and it ends just before whichever comes last. There are various commands for editing regions. These are mapped to the same key combinations as are similar commands in Emacs. I took the idea of editing regions straight from Emacs.

Here are the keymappings for editing regions:

c-w kills a region. It copies a region to the clipboard and deletes it.

a-w copies a region to the clipboard.

c-y yanks the contents of the clipboard. It inserts the clipboard's contents and marks it as a region.

Unlike Emacs, XES does not maintain a killring. It just puts the contents of a region into the clipboard. To understand regions better, I refer you to the documentation for GNU Emacs.

Please note that c-w and a-w set the blocktype to character and leave it that way.

1.6 Region Rectangles

A region rectangle is another concept from Emacs. It is like a region, but the mark and point mark the the corners of a box. There are various things that you can do with rectangles. You can delete them, clear them, or push them to the right. The commands which operate on vertical blocks treat them as rectangles. Since rectangles cannot be inserted like regular text, they are not stored in the clipboard. The cut and copy facilities, which are mapped to Right Amiga X and to Right Amiga C, copy vertical blocks to T:rectangle. You can insert rectangles with the "Yank Rectangle" subitem in the "Insertion" submenu of the "Edit" menu. There are various macros which operate on rectangles.

1.7 Clipfiles

Clipfiles were originally implemented in XDME as a substitute for real clipboard support. A clipfile is a file that you can save text to and insert text from as though you were cutting and pasting with the clipboard. Although XDME now has genuine clipboard support, I have reasons for using clipfiles as well. One advantage of clipfiles is that you can append text to them, whereas XDME has no command for appending text to the clipboard.

Furthermore, I have drawn on the Emacs concept of a register to make clipfiles even more useful. In Emacs you can copy text to registers instead of putting it on the killring. This allows for easy retrieval of the text in case you kill more text in the meantime. With XES, you can change the file used as a clipfile. You can also change it back to what it once was, so that you can copy text to a clipfile and keep it around until you need it.

Nevertheless, using clipfiles has at least a couple disadvantages over using the clipboard. One is that the commands for copying text to clipfiles currently work only with line blocks. The other is that other programs can't use them as easily as they use the clipboard.

The clipfile commands are mapped like the clipboard commands, except they use the left Amiga key instead of the right. They are A-c for copying, A-x for cutting, A-v for pasting, and A-a for appending.

1.8 Macros

BUILDING BLOCKS

Control

Math

Arrays

OTHER MACROS

Searching

Regions

Rectangles

Blocks

Deletion

1.9 Control Macros

do	Introduces a non-looping range
fail?	Forces a statement to carry through even if it fails, setting a flag that indicates whether it failed. The flag is called $fail.
loop	Creates an infinite loop
until	Executes a statement once and continues to execute it until the condition is satisfied.
when	Executes statement if condition is satisfied. Resumes execution at the end of a while, repeat, loop, until, or do range.

Together, do and when function much like SELECT and WHEN in ARexx. I wrote them for the sake of imitating this sort of control structure. Ex:

```
do (when cn (title (Number)) when ca (title (Letter)) title (Neither))
```

1.10 Math Macros

intcmp x y	Compares x to y. Sets flag 3 to 1 if they're equal. Sets flag 4 to 1 if x >= y.
get-sign x eval-sign	Sets $z to -1 if x<0, 0 if x=0, or 1 if x>0. Sets flag 3 to 1 if $z=0, and sets 4 to 1 if $z>=0.
minus x y z	Sets x to (y - z).
plus x y z	Sets x to (y + z).

1.11 Array Macros

get-array x y z	Stores $(yz) into x. y is a label and z an indice.
set-array x y z	Stores z in $(xy). x is a label and y an indice.
free-array x	Unsets all the members of array x.
unset-array x n	Unsets the nth member of an array.
unset-member x	Serves as a bridge between free-array and unset-array.

1.12 Searching and Replacing Macros

replace-block	Replaces every occurance of $findstr in a block, whatever type of block it is, with $repstr.

replace-all	Replaces every occurance of $findstr with $repstr.

1.13 Regions

block-region	Marks the region as a block.
mark-block	Marks a block as a region, point at top, mark at bottom.
mark-point x y	Stores x and y coordinates of point into x and y.
goto-mark	Moves the cursor to the mark.
mark-first?	Sets flag 7 to 1 if the mark precedes the point.
region-top	Puts the point at the top of the region, and the mark at the bottom.
region-end	Puts the point at the bottom of the region, and the mark at the top.
x-distance?	Sets $dx to the absolute difference between the point's column and the mark's column.
y-distance?	Sets $dy to the absolute difference between the point's row and the mark's row.
delete-region	Deletes the region between the mark and the point.

1.14 Rectangle Macros

topleft-rectangle	Goto top left of region-rectangle, set mark at bottom right
block-rectangle	Marks region-rectangle as a vertical block
dimrec	Calculates dimensions of Rectangle. Also creates a blank line equal to its width
delete-rectangle	Deletes rectangle
open-rectangle	Pushes the contents of a rectangle to its right
clear-rectangle	Fills rectangle with spaces
copy-rectangle	Copies rectangle to T:rectangle
kill-rectangle	Kills rectangle
yank-rectangle	Yanks killed rectangle and marks it as a region-rectangle

replace-rectangle	Replaces every occurrence of $findstr with $repstr within a region-rectangle

1.15 Block Macros

block-type?	Indicates the block type by setting flags 5 and 6. 5 off, 6 off = No block 5 on, 6 off = Line block 5 on, 6 on = Character block 5 off, 6 on = Vertical block
copy-block	Copies line and character blocks to clipboard. Copies vertical blocks to T:Rectangle.
cut-block	Copies block as per copy-block and then deletes the block.
delete-block	Deletes any sort of block.
read-block	Reads $block and returns information from it to the block-type? macro.
replace-block	Replaces every occurance of $findstr in a block, whatever type of block it is.
block-region	Marks the region as a block.
mark-block	Marks a block as a region, point at top, mark at bottom.

1.16 4.6.

delete-block	Deletes any sort of block.
delete-region	Deletes the region between the mark and the point.

1.17 5.

Major Modes

XPK support

#?.xsrc files

Diary

Pagination

Insert Literal

GTB Requesters

Spell Checking

1.18 Major Modes

XES provides XDME with many major modes. I took the concept of
major modes from GNU Emacs. A major mode is a configuration of keymappings that makes editing a particular sort of document easier. The major modes in XES are completely localized. This means that when you set a major mode, it is only for the window you set it in. So, for instance, you can have C mode in one window and Text mode in another, and you can switch back and forth between them without manually switching modes.

Usually, you will not have to set a major mode. If you load files with the AppIcon or with the requesters I mentioned in the previous section, XES will set the mode for you automatically. It will do this for you in one of two ways. If the file has an accompanying "#?.xsrc" file, XES will use the mode named in that file. Otherwise, it will choose a mode based on the name of the file. For example, files ending in "#?.c" will be loaded in C mode.

HTML Mode

1.19 HTML Mode

This is based on html-mode.el for Emacs. Html-mode.el has many key combinations for inserting HTML commands, and HTML mode for XDME inserts the same commands with the same key strokes. But there are some important differences. XDME's HTML mode has the added feature of recognizing whether the cursor is in a block. If it is within a block, XDME's HTML mode operates differently than Emacs'. XDME will usually perform an operation on an entire block. For example, c-c c-e, which will normally insert "<EM></EM>", will insert "<EM>" at the beginning of the block and "</EM>" at the end. Because XDME operates differently on blocks, I found no need to include html-mode.el's commands that operate on ranges. I also haven't included some of its fancier features, such as the ability to give each reference a unique name.

1.20 XPK Support

XES provides XDME with partial XPK support. It doesn't provide complete XPK support, because I use XDME with xData. This program lets you load XPK files into other programs, and I highly recommend getting it. XES provides XDME with what xData doesn't. With XES you can pack files every time you save them, and you can encrypt and decrypt files as you save and load them.

To pack or encrypt a file, you have to select an XPK method other than NONE, which is the default. You may select the method with the "Select Method" subitem in the XPK submenu in the Misc menu. This will pop up a file requester for the LIBS:Compressors directory, which is where all your XPK compressor libraries should be stored. Select the library you want, and that library will be used to pack or encrypt the file you selected it for everytime you save that file.

You should bear in mind that this requester will let you select an XPK method for one window only. So you can work on a file you want encrypted in one window, and a file you want left unpacked in another. If you want to encrypt a file, you should also select a password for the encryption. You may do this with the "Select Password" subitem in the XPK submenu. This password will apply only to the window you selected it for.

So that a file will be packed or encrypted whenever you save it, you must use the "Save As" requester mapped to c-x c-w or the "Save" subitem in the "Save" submenu. The keyboard shortcut for this is Right Amiga S, and it is also mapped to c-x c-s.

The ARexx script responsible for loading files checks whether a file was encrypted with FEAL or with IDEA before loading it. If the file was so encrypted, the script will pop up a password requester. If you get the password right, it will successfully decrypt the file before loading it into XDME. Otherwise, it will load the file in its encrypted form.

XES does all packing, encrypting, and decrypting of files with the "xpack" program, which comes with the XPK package. It writes temporary files to T:, uses xpack on these files, and then loads the file into XDME or moves it to the place you want to store it.

1.21 #?.xsrc files

An "#?.xsrc" file which XES sources whenever you load its accompanying file. It normally contains the major mode, the margins, and settings for printing it with paginate. You may also put anything else in an "#?.xsrc" file by editing it like an .edrc file.

To automatically create an "#?.xsrc" file for a file, you should select "Preferences" from the "Save" submenu in the "Project" menu. Previously, XES automatically created these files for each file it saved. I changed it because the automatic mode selection will suffice for setting the parameters for most files.

To have an "#?.xsrc" file make a difference when you load a file, you must use the AppIcon or one of the requesters XES supplies for loading files.

1.22 5.4.

XES makes it easy for you to maintain a secret diary of your private personal affairs. To make an entry in your diary, select the "Diary" subitem in the "Load" submenu of the "Project" menu. This will present you with a requester set to the directory that you keep your diary in. The default filename will be the entry for that day, even if it doesn't exist yet. Each diary filename will be a date in sorted date form. This will be an eight digit number with the year first, the month second, and the day last. It is in sorted date form so that the file requester will show the entries in order. To make an entry in your diary, just press RETURN.

Before it loads a file or creates a new entry, it will ask you for a password. If the entry you want to load already exists, it will use this password to decrypt the entry. It will also use this password to encrypt it whenever you save it. The diary is set to use IDEA encryption with the xpkIDEA.library.

Whenever you use the Diary requester to load a previous day's entry, it will load that entry in Read Only mode. This is to help keep your diary faithful to what you originally wrote.

To use the Diary facility, you need XPK and in particular the xpkIDEA.library. Before using it, you should edit Diary.xdme to print your name, instead of my own, in each diary entry, and to choose the directory you want to store your diary in.

1.23 5.5.

XES lets you easily paginate your files. It does this with the help of my "Paginate" program. You may set the parameters for paginate by selecting "Set Parameters" in the "Paginate" submenu of the "Format" menu. To know how to set Paginate's parameters, you should read "Paginate.man". If you're unhappy with the result, you can reload the original file as T:Text and try again.

1.24 Insert Literal

XES provides you with three different ways to enter characters that you can't or don't know how to type. The easiest way is to use the "Insert Literal" requester. This shows a requester of all printable characters. Just select the character you want with the mouse, and it will appear in the text. This requester is available in the "Insertion" submenu of the "Edit" menu, and its keyboard shortcut is Right Amiga L. To use this requester, you must have RexxArpLib.library installed. This requester will use whatever font you have selected for the window you are in.

The second way is with the "Insert ASCII" requester. This lets you insert any character at all by entering its ASCII number. This may be slightly quicker than the "Insert Literal" requester if you already know a character's ASCII value. But more importantly, it lets you insert any non-printable character, such as a formfeed or an escape. This requester is mapped to a-a. It is also available in the "Insertion" submenu

The third way is the way it's done in Emacs. Type c-q followed by the key combination for the character you want.

1.25 GTB Requesters

XES provides XDME with 8 GadToolsBox requesters:

Find & Replace

Pen Colors

Printing

Margins

Windows

Local Settings

AlphaSpell & Guesses

1.26 Find & Replace GTB Requester

The Find & Replace requester is mapped to right Amiga r and accessible as "Replace ..." in the Search-Replace submenu. It pops up an asynchronous requester with two string gadgets and five buttons. You can use the requester to search backwards and forwards for words, to do a selective search and replace, or to replace every occurrence of a string within either a block or the entire document. Replacing within a block works accurately for every block type.

1.27 Pen Colors GTB Requester

The pens requesters shows six palettes, one for each pen that XDME can set. Colors change as soon as you select a color on a palette. The requester doesn't go away until you click on the close gadget. So you can just keep clicking on colors until you find a color combination that you like. If you find a color combination you want to keep, you can save it by selecting "Save Config" in the Settings menu.

1.28 The Print GTB Requester

The Print requester has many string and integer gadgets for changing the parameters for Paginate, the program XES uses to print files in XDME. Paginate is a pagination program I wrote myself and have included with XES. You should read the manual for it to know more about it. I will not repeat details about it here.

The Print requesters let you print to PRT:, preview what the file will look like in its printed form, print it to a file, or append it to the end of another file in paginated form. Please note that the margins you can set with this requester are parameters only for Paginate. They are not XDME margins or printer margins.

1.29 The Margins GTB Requester

This requester lets you set the indent column, the left margin, and the right margin with slider gadgets. It does not let you make the left margin greater than the right or the right less than the left. Whenever you pop it up, it shows your current margins. It doesn't let you set any margin greater than 80. To do that, you should set it manually.

The indent column is useful mainly in text mode. It indicates how much the first line of a paragraph should be indented. Outline mode also uses it, but the outline commands calculate and set this value automatically.

1.30 The Window GTB Requester

The Window requester lets you select a window to go to. Whenever you use this requester to go to another window, it updates the menus, so that the checkmarks indicate the correct values for the window you're in. This requester replaces the Windows menus from previous versions of XES. This requester is not limited by the same restrictions as the menu was.

1.31 The Local Settings GTB Requester

Since local settings are different from window to window, the checkmarks for them in the menus can sometimes be inaccurate. So I took them out of the menus and put them into a requester. This requester keeps track of Wordwrap, Viewmode, Insertmode, Autoindent, Autosplit, Ignorecase, the XPK Password, and the XPK Method. It lets you select the XPK Method from a listview. XES calls the ARexx script xMethods.xdme to generate the list for this listview. It calls it only at startup.

1.32 AlphaSpell GTB Requesters

To spellcheck the document you're editing, select "Check Spelling" from the "Spelling" submenu of the "Misc" menu. AlphaSpell will then spellcheck the file, and XDME will load AlphaSpell's output into a listview gadget. This listview gadget belongs to a GadToolsBox GUI, which contains another listview and some buttons.

The first listview contains an alphabetized list of the words AlphaSpell didn't find. You can quickly review the list to find the words you know are misspelled. You can then use the "Prev" and "Next" gadgets to find each occurance of the words you've misspelled. Just click on the word, so that it appears in the bottom of the listview, then press on "Prev" to search backwards or on "Next" to search forwards. It is best to go to the top of the document before beginning any new search. You can do this as you normally would, since the requester is asynchronous. Unlike XDME's ordinary prev and next commands, the "Prev" and "Next" gadgets search for whole words. So if you're searching for a small word, it won't stop everytime it is part of a larger word.

If you know that a word in the first listview is spelled correctly, you can indicate that you want to put it in a user dictionary by clicking the "Learn" button. This will move the word from the first listview to the other one. If you make a mistake or change your mind, you can put a word back into the first listview by clicking on the "Remove" button. You can move the words from the second listview to your user dictionary by clicking on the "Save" button. Once you do this, these words will belong to your user dictionary, and they will disappear from the listview.

If you aren't sure how a word should be spelled, you can ask AlphaSpell to guess the correct spelling for the word you want by clicking on the "Guess" button. It will then pop up another GTB requester with its guesses in a listview. If you don't find what you're looking for, you can change the spelling in the string gadget and have it guess again by clicking on the "Guess" button in the new requester. You can keep doing this until you're happy with the results. Once you find the word you want, you can click on that word and then use the "Replace" button to delete the current word and put the new word in its place.

1.33 Spell Checking

XES uses AlphaSpell for spell checking. It writes the ↩
contents of
the window to a temporary file and uses AlphaSpell to generate a list of words that are not in the dictionary. For more information, read about the

AlphaSpell GUI's
or read the documentation for AlphaSpell.

1.34 History

XDMD v1.00

XDMD v1.10

XES v1.20

XES v1.25

XES v1.30

XES v1.31

XES v1.40

XES 2.0

XES 2.1

XES 2.2

1.35 XDMD v1.00

Released 29 October 1994

1.36 XDMD v1.10

Released 29 October 1994

I. By taking a short detour through ARexx, I implemented arrays in XDME. The macros "set-array," "get-array," "free-array," "unset-array," and "unset-member" all work with arrays. The ARexx script LoadArray loads a file into an array.

II. I used the array capability to redo how spell-checking works in XDME. Instead of using a requester, it uses LoadArray to put the list of unfound words into an array. It then redefines some keys on the numeric keypad.

A. nk1 goes to the last word in the array.

B. nk7 goes to the first word in the array.

C. nk+ moves forward through the array.

D. nk- moves backwards through the array.

E. nk2, nk4, nk6, and nk8 work as cursor keys.

F. nk3 and nk9 search forwards and backwards for the find string, which gets set to the current word in the array whenever you move through the array.

G. nk(puts the numeric keypad back in number mode.

III. Yesterday, before I figured out how to implement arrays, I did some new spell-checking routines with twin variable stacks. I left the code in .edrc for anyone interested in looking at it.

A. I added the "empty-stack" macro, because "purgevar" doesn't work.

IV. I fixed some bugs in s-bs and s-del. I added a-d, which works like M-d in Emacs.

A. I added the "delstr" macro.

V. I reset toggle 0 and set toggle 1, so that I can use variables as flags with "if," "ifelse," and "while."

VI. I added some numeric keypad modes.

A. "keypad-numbers" - your ordinary numeric keypad

B. "keypad-move" - the cursor movements written on the front. nk5 recenters the screen.

C. "keypad-spell" - the mode for spell checking, which I described earlier.

1.37 XES v1.20

Released 9 November 1994

I. Overhauled the menus.

A. Checkmarks indicate whether different settings are set. These get updated for whichever window you are in if you change windows with the menu.

B. Amiga key symbols have replaced "A-"s.

C. The menus have been rearranged somewhat.

II. New macros:

A. "Intcmp" macro compares two numbers. With its help, nk+ and nk- never take you out of bounds when spell-checking.

B. "Prev2" macro serches backwards for the closest of two different strings.

C. "Prevf" macro searches backwards for a string, and it sets a flag if the search fails.

D. "Pos" takes a column and line number as arguments and goes there.

E. "Pos-line" indents the current line n columns from the left.

F. "Text-format" formats a paragraph according to values stored in $indcol and $parcol. $Indcol indicates how much the first line of a paragraph should be indented. The outlining macros set these values and use text-format.

1. "Setindcol" sets the value of $indcol.

III. New major modes:

A. Fundamental - What used to be called "Programming."

B. Programming - The tab key indents code in a general way that is helpful for programming in any language. Use a more specific mode if one is available for the language you are programming in.

C. ARexx - The tab key properly indents ARexx code so long as the strings "do" and "select" are not parts of longer strings in the code.

D. Manual - For writing manuals.

IV. Miscellaneous changes here and there.

1.38 XES v1.25

Released 17 November 1994

I. An asynchronous gadtools find and replace requester.

II. More Emacs and Quasi-Emacs keymappings.

A. Registers sort of like in Emacs.

B. Many keymappings converted to equivalent in Emacs. For example, "Save As" is now mapped to c-x c-w instead of to f9.

C. C-q works like it does in Emacs. After pressing c-q, you can type any character. C-x k is now the key combination for closing a window.

D. The case conversion macros now work like the equivalent in Emacs, and they have the same keymappings.

E. Major modes can easily be chosen with a c-x m <key> combination.

F. Rectangle operations now have the same keymappings as they have in Emacs. Those which have no keymappings in Emacs are also mapped to a c-x r <key> combination.

III. #?.xsrc files for each file saved. These are files that XDME sources for an individual file after it loads it.

IV. The right and left Amiga keys are recognized as different.

V. Clipfiles.

VI. The AppIcon can be turned on and off from the Settings menu.

VII. Automatic recognition of files encrypted with xpkIDEA.library or with xpkFEAL.library.

1.39 XES v1.30

Released 6 December 1994

I. More GadToolsBox requesters: Pens, Print, Margins, and Windows. The Windows requester replaces the Windows menu.

II. A new "block-type?" macro. This one reads $block and even tells whether there is a block. Editing commands no longer require the cursor to be in the block to recognize that there is a block.

III. Paginate updated. The new version indents the text body at the same column as it does the footers and headers. I have not updated the documentation for it.

IV. Shortlines added to Settings menu. This is like my Smartkeys mode, but it is currently exhibiting some unwanted behavior. So I put Smartkeys back in.

1.40 XES 1.31

Released 23 December 1994

I. Various macros updated to work with shortlines mode, now that it is working right. I changed "right" to "col +1" in various places.

II. A new GTB requester for local settings. This replaces the items that were regularly updated in the menus. This saves time when switching between windows. This requester also keeps track of your XPK password and method. It lets you select a method from a listview gadget that contains all available XPK compressors.

III. Bug fix: Copy and Cut in the Edit menu now work correctly.

1.41 XES 1.40

Released 29 May 1995

I. New and Improved Modes.

A. Many modes now have many insertion commands mapped to c-c combinations. C mode lets you use c-c combinations to insert C code, HTML mode to insert HTML commands, etc. This feature is available in C, ARexx, HTML, Amigaguide, and Symbolic Logic modes.

B. HTML mode added. It is based on html-mode.el for Emacs.

C. ARexx mode much improved. It was buggy before, but now the tab key indents ARexx code almost as well as C mode indents C code. It is less perfect only because ARexx has a much more complicated syntax than C has.

D. LISP mode for parentheses intensive programming languages.

II. Useful new macros

A. Control macros

1. With the macros "do" and "when" I have created something like the SELECT-WHEN feature of ARexx or the switch-case feature of C.

2. The "until" macro works sort of like repeat-until in Pascal. It goes through once and repeats until the condition is satisfied.

3. The "loop" macro creates an infinite loop.

4. "Fail?" forces an operation to go through even if it fails, and it sets $fail to 1 if the operation does fail.

B. Search macros

1. I have implemented whole word searching with wsearch, wfind, and wprev. Prev2 now does whole word searching. Closer? tells which of two words is closer, going backwards.

III. Modes and GUIs loaded only if they're used. This saves on memory and, more importantly for me, the time it takes for XDME go through .edrc when it starts up.

IV. A GadToolsBox requester for AlphaSpell.

A. All unfound words appear in a listview.

B. It will learn words by saving them to a user dictionary.

C. It will guess at correct spellings for you.

V. Install script.

VI. XES has been modularized. It no longer puts everything into one big .edrc file. Instead, it consists of several XDME scripts, and it requires only one line in your .edrc file.

VII. ARexx scripts now go in XES:Rexx/ instead of REXX:.

VIII. Various bug fixes.

1.42 XES 2.0

I. Various bug fixes

A. SaveFile.xdme compensates for a new bug in XDME.

B. ARexx mode more accurately matches END's with the appropriate DO or SELECT.

C. Mode.xdme recognizes the new modes, and it is better at selecting the correct mode.

D. "REXX:" replaced with "XES:Rexx/" in various ARexx scripts.

E. Wholeword searching no longer overwrites words in caps.

II. New features

A. In Outline Mode, pressing the tab key will now properly format roman numeral headings of more than one letter in length, such as "II." or "VIII." But you will still need to use shift-tab for headings such as "I." and "V."

B. Diary.xdme now gets the Diary drawer and the author's name from environment variables instead of having these values hard coded in the script. The Install script sets these environment variables.

C. Interactive spell checking with AlphaSpell V.

D. GUI for AlphaSpell V. Better than the one for AlphaSpell IV.

E. New Rexx scripts

1. VER.xdme updates version strings in scripts.

2. Others. This is freeware. Expect better documentation from my shareware. Yes, I'm being lazy. Most of the REXX scripts simply aid in other tasks anyway, and you don't have to know about them to use them.

III. Some obsolete files deleted.

IV. Changing to version.revision version numbers. Jumped to version 2.0 to make this change clean. Each REXX script has its own version.revision number, where version 1.0 is how it was when I wrote VER.xdme.

1.43 XES 2.1

I. REXX files modified: VER.xdme, CopyDicts.xdme

II. XDME scripts modified: menus.xrc, xes.xrc, dict.xrc, arexx-mode.xrc, c-mode.xrc, keys.xrc

A. Version strings added to all XDME scripts. Version 1.0 is the default version for each file before I changed anything today, 14 August 1995.

B. "Select Dictionaries ..." menu item now works in menus.xrc

C. Default operations for control-c key combinations added.

1. "c-c v" updates revision in version string.

2. "c-c V" updates version in version string.

D. ARexx mode and C mode now have the following additions:

1. "c-c /" adds a border like this:

```
/*************************************************************************/
```

Its starts at the cursor and stops at the margin.

2. "c-c]" justifies a comment to the margin.

```
/* It looks like this.                                                   */
```

E. New requester for inserting ASCII values of characters. It is mapped to ca-a.

1.44 XES 2.2

Released 25 August 1995

I. Modified the following scripts: print.xrc, menus.xrc, c-mode.xrc, find.xrc, AlphaSpell.xrc, macros.xrc, and interspell.xrc.

II. Modified the following ARexx scripts: Paginate.xdme, Mode.xdme, and SaveWords.xdme.

Fixed bugs, adapted calls to AlphaSpell to use AlphaSpell's new

argument format, and switched some key mappings in C mode to reflect frequency of use.

1.45 Features

Here are some of XES's features:

I. Localized major modes

A. Text mode. This mode provides automatic indenting of paragraphs.

B. ARexx mode. This mode includes automatic line indentation and a set of macros for inserting various ARexx commands.

C. Two C modes, ANSI and K&R, complete with automatic line indentation.

D. HTML mode and Amigaguide mode for writing hypertext documents. Both include a set of macros for inserting various hypertext instructions.

E. LISP mode. This mode is also useful for Installer scripts, as well as for any other language that uses parentheses in the same way as LISP does. It provides automatic indenting for LISP-like code.

F. A general programming mode for other languages. This mode provides a modicum of automatic indenting to make programming a bit easier.

G. An outline mode for writing outlines, such as this one.

H. A proofs mode for writing logical proofs. (I'm a philosophy major.)

II. Automatic major mode selection upon loading.

III. The maintenance of a "Windows" menu that keeps track of all of XDME's windows.

IV. Different styles of editing integrated with each other.

A. Cutting and copying for all block types.

B. Regions and rectangles, a la Emacs.

V. XPK support, including automatic packing or encryption upon saving, and decryption with a password requester.

VI. The maintenance of a personal diary, complete with password encryption and decryption.

VII. Spellchecking via the AlphaSpell program and its accompanying dictionary.

VIII. Automatic pagination via the Paginate program.

IX. Various find and replace routines, including a Find requester and a Replace requester. The Replace requester lets you replace the word at the cursor, every occurance of the find string in the text, or every occurance in a block.

X. Various Emacs like keymappings.

XI. Various macros that make it easier to write other macros for XDME.

1.46 Installation

To install XES, click on the Install icon. To spellcheck with XDME, you will also need AlphaSpell V, which I have distributed as ASpell5.lha. I have distributed it separately, because it is shareware, and it may be used without XES. Install it by clicking on its Install icon.

1.47 Copyright

1.48 Distributability

1.49 Disclaimer

1.50 About the Author

I am Fergus Duniho. I am a Ph.D. candidate in the Philosophy department at the University of Rochester. I have finished everything but my dissertation, which will be on evil as a psychological attribute. More specifically, it's on whether we can rightly describe some people as evil, and on what it would mean to say that someone is evil.

I am also the author of the DDLI, a computerized personality indicator. Get ddli341.lha from the aminet and try it out.

My email address is: fdnh@troi.cc.rochester.edu