

Ram:AFileEnglish

COLLABORATORS

	<i>TITLE :</i> Ram:AFileEnglish		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		December 8, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	Ram:AFileEnglish	1
1.1	Ram:AFileEnglish.guide	1
1.2	AFileEnglish.guide/Introduction	1
1.3	AFileEnglish.guide/Installation	2
1.4	AFileEnglish.guide/Startup	2
1.5	AFileEnglish.guide/MenuPrinc	3
1.6	AFileEnglish.guide/DefStruct	4
1.7	AFileEnglish.guide/ModStruct	6
1.8	AFileEnglish.guide/DataAccess	6
1.9	AFileEnglish.guide/FileMenu	7
1.10	AFileEnglish.guide/EditMenu	8
1.11	AFileEnglish.guide/SpecialMenu	9
1.12	AFileEnglish.guide/ImportExport	10
1.13	AFileEnglish.guide/InputMask	10
1.14	AFileEnglish.guide/MsgSpec	11
1.15	AFileEnglish.guide/MaskSpec	12
1.16	AFileEnglish.guide/MenuSpec	13
1.17	AFileEnglish.guide/PrintSpec	14
1.18	AFileEnglish.guide/PrintFmtSpec	14
1.19	AFileEnglish.guide/Alias	15
1.20	AFileEnglish.guide/AREXX	15
1.21	AFileEnglish.guide/History	17

Chapter 1

Ram:AFileEnglish

1.1 Ram:AFileEnglish.guide

AFile 3.30

Copyright (C) 1993-1994 by Denis GOUNELLE.

Any commercial usage or selling without author's written authorization is strictly forbidden. You can copy and spread this program under the following conditions:

1. All the files must be provided
2. No file must be modified
3. You can't charge more than \$6 for the copy

In spite of several tests, no warranty is made that there are no errors in AFile. YOU USE THIS PROGRAM AT YOUR OWN RISK. In no event will I be liable for any damage, direct or indirect, resulting of the use of AFile.

Introduction	What is AFile
Installation	How to install AFile
Startup	How to start AFile
Main menu	Main menu description
Structure definition	How to create a new file
Structure modification	How to modify the structure of a file
Data access	How to access data
Importing/Exporting data	How to import/export data
Input masks	How to define input masks
Alias	How to define aliases
AREXX Interface	AREXX programming interface
History	Program's history

1.2 AFileEnglish.guide/Introduction

Introduction

AFile is a data file manager, that is a tool which lets you create and manager your files (addresses, video collections, clients, etc...) using an Intuition interface. There is not limitation to the number of fields or records. The program offerters the standard printing, sorting, and data importing/exporting functions.

AFile uses AREXX as it's programming language. You can create full input masks with background picture, field positionning, checking of entered data, menus and printing customization.

Criticisms and suggestions will always be welcomed. Write to:

M. GOUNELLE Denis

27, rue Jules GUESDE

45400 FLEURY-LES-AUBRAIS

FRANCE

You can also send a message to the following Internet address : "gounelle@alphanet.ch". Note that this mailbox is not mine, so please send only short messages. As I don't have direct access to the messages, don't expect an answer before a dozen of days.

Thanks to Yves PERRENOUD for it's numerous suggestions.

1.3 AFileEnglish.guide/Installation

Installation

AFile needs the mathieedoubbas.library which is provided by Commodore on system disks.

AFile is now localized, so it can adapt itself to your favorite language. All you have to do is to copy the good catalog file into the directory corresponding to your language. For exemple, if your default language is french, copy the francais.catalog file into the SYS:Locale/Catalogs/Francais directory, under the AFile.catalog name.

1.4 AFileEnglish.guide/Startup

Startup

The program can be started both from CLI and Workbench. In both cases, you can specify a file to use, with the usual method, so the program will jump directly to the visualisation/modification screen.

The following arguments may also be specified:

USEASL

asks to use the asl.library file requester, instead of the builtin requester. This option is ignored under Kickstart 1.3.

NOCASE

asks search and sort operations to be case independant by default (using interface only)

PRINTSCRIPT <script's name>

PRINTSCRIPT=<script's name>

specifies the name of the AREXX script to use to print records, instead of the builtin method.

The script will be called for each record to print, with record data directly accessible using the GETFIELD command. The standard output is automatically redirected to the printer, so you just have to use the "SAY" command to output data. See AREXX Interface.

You can also specify a printing script in input masks. See Input masks.

INPUTMASK <mask's name>

INPUTMASK=<mask's name>

specifies an input mask at startup. This is usefull mostly when a file name is also specified.

FONT <name>

FONT=<name>

specifies the font to use, rather than the default text font. The font name must be in the <Y size> format (e.g. "courier9"). AFile can't use a proportional font.

DEPTH <number of planes>

DEPTH=<number of planes>

specifies AFile's screen number of planes. This is usefull when an input mask loads a background picture, and you want to be sure to have the right number of colors.

AREXXWIN <>window specification>

AREXXWIN=<>window specification>

specifies the output window for AREXX scripts started from AFile. By default, AFile opens a window which covers all the screen, except the menu bar.

1.5 AFileEnglish.guide/MenuPrinc

Main menu

This menu appears when the program is started, if you didn't specified a file name as an argument. AFile uses the preferred text font and screen mode. This menu has the following items:

New file

Allows to create a new file. A file requester will allow you to specify the name of the file to create. See Structure definition.

Modify structure

Allows to modify the structure of an existing file. A file requester will allow you to select the file to modify. See Structure modification.

Open file

Allows to open an existing file. A file requester will allow you to select the file to open, then the visualisation/modification screen will appear. See Data access.

Executer script

Allows to run an AREXX script. A file requester will allow you to select the script to run. See AREXX Interface.

Quit

Terminates AFile execution.

1.6 AFileEnglish.guide/DefStruct

Structure definition

AFile handles files made of fixed length records. All the records have the same structure, made of an unlimited number of fields. Each field is defined by its name, size and type. The number of records in a file is limited only by the capacity of your hard disk.

Field's name can't be longer than 32 characters, and may contain any character. However, you shouldn't use spaces if you think you will use AREXX to access to the file. Of course, each field's name must be unique.

Field's size is limited to 65535 bytes, except for "DATE", "ENUM" and "BOOLEAN" fields, which are 8, 1 and 1 bytes long, respectively. For numeric fields, you can also specify the number of digits after the decimal point. Possible field's types are:

alpha

alphabetic value only (lowercase and uppercase letters, space, dash, single quote)

numeric

numeric value (real or integer)

alphanum

any value

date

date (day, month, year with century)

boolean

boolean value (TRUE or FALSE)

enum

value contained in a list specified when defining the file

The definition structure requester is made of a display area, where are displayed the field's definitions (name/type/length). You can use the scroller to scroll the display. The current field is displayed under a blue background. To select a field, you just have to click upon it's definition: it will automatically copied in the gadgets.

The gadget row at the bottom of the window allow the following operations:

Add

adds a new field as the last field

Insert

inserts a new field before the current field

Remove

removes the current field

Replace

replaces the current field. This allow (for example) to modify the name, size or type of a field.

To define a new field, you'll have to enter it's name and length in corresponding string gadgets, and to select it's type using the Type cycle gadget. If the field's type is "ENUM", you will also have to enter the list of possible values (separated by commas) in the Values gadget. Then you must click upon the Add gadget: the new field will be added to the list, and selected as the current field. If the field can't be added (no memory, name not unique, etc...) the screen will flash.

The definition requester also has a menu, with the following commands:

Copy structure

Copies the structure of an existing file. Note that the structure currently displayed in the requester will be lost ! A file requester will appear, so you can select the file which structure is to be copied.

Abort

Cancel structure definition, and go back to main screen without creating the file.

Save

Records structure definition, and go back to main screen after

creating the file.

Encrypt data

Data in the file will be encrypted, using a password that you will have to provide when you record structure definition. This password will be asked for each time the user will try to access to the data.

Caution ! If you ever forget this password, you will never be able to access to the data in the file !

1.7 AFileEnglish.guide/ModStruct

Structure modification

This allows you to modify the structure of an existing file: you can add some fields, remove some others, modify a field's length, etc... However, you can't modify a field's type, or delete all the fields.

The modification requester is quite the same as the definition requester (see Structure Definition). The only differences are that the Type gadget and Copy structure menu item are disabled.

Once the structure has been modified, use the Save menu item to record your changes: AFile will automatically convert the file into the new structure. Take care that deleted records will be lost during this operation, and that this conversion may fail if there is not enough free space on the volume where is the file.

If some data will be lost during the conversion (shorter or removed fields, deleted records, etc...), you will be asked to confirm the operation.

With the Options menu, you can remove or add data encryption. If the encryption option is enabled when you record the new structure, AFile will ask you to enter a password. This can also be used to change the password for a file: you just have to ask to modify the structure of this file, record the structure without any change, and enter the new password.

1.8 AFileEnglish.guide/DataAccess

Data access

The visualisation/modification screen is made of a display area where data is displayed, of a status line and of a row of gadgets.

The gadgets showing the data are the same width as the corresponding

field, except if the screen is not large enough. Each time you modify the value of a field, AFile will check that this new value is compatible with the field's type. Dates must be entered in the "DD/MM/YYYY" format (with century) or just "DD/MM/YY" (the current century will be added automatically). Entering "?" is an easy way to specify today's date (see Aliases).

The first status line shows the type and size of the current field. The second line shows the file's name, and current position in the file under the X/Y form, where X is the current record number and Y the last record number. When a sort has been activated, the (Sorted) word is displayed after the file's name. If the current record has been modified, the Modified word is displayed at the right of the status line.

The row of gadgets allows the following operations (from left to right):

- go to the first record (or shift-left key)
- go to the previous record (or left key)
- write the current record
- go to the next record (or right key)
- go to the last record (or shift-right key)
- go to a specified record (or g key)
- delete the current record (or DEL key)
- print n records starting from the current record (or p key)
- previous fields page (or up key)
- next fields page (or down key)

If the current record has been modified, the changes will automatically be written if you move to another record. When you are on the last record, asking to go to the next record will automatically create a new record.

All other operations may be started with menus.

File	The File menu
Edit	The Edit menu
Special	The Special menu

1.9 AFileEnglish.guide/FileMenu

The file menu
=====

Undelete

From AFile's point of view, deleting a record just mean setting a flag for this record, which tells this record is deleted and must not be accessed. The record is not actually removed: data is still here. The menu item allows you to clear all deletion flags, so the corresponding records may be accessed again.

Pack file

Actually removes deleted record (you will be asked for confirmation). After this, you won't be able to undelete these records, but the file will be smaller because the data for these records will be definitely removed. Take care that this operation may fail if there is not enough free space on the volume where is the file.

Informations

Displays some informations about the current file: name, number of fields, record size, number of available records, number of deleted records.

Print structure

Prints the file's structure, that is the definition of all the fields. You can print the structure to the printer, or the a file.

Close file

Close the current file, and go back to main screen.

Close and quit

Close the current file, and terminates the program.

1.10 AFileEnglish.guide/EditMenu

The Edit menu

=====

Cancel changes

Reads the current record from the file, so cancels all changes since last write.

Select mask

Selects an input mask. A file requester will appear, so you can specify the mask to load. See Input masks.

Forget mask

Forgets the current input mask.

Import

Imports data from an ASCII file. The new records will be appended to the end of the file. See Importing/Exporting data.

Export

Exports data to an ASCII file. All the records will be exported. See Importing/Exporting data.

Search

Searches a record with a given value in a specified field. A field request will appear, so you can specify which field the value is to be searched in. Then you will have to enter the value to search, and search options:

Exact

Searches for a field which is equal to the value (option set), or just containing the value (option cleared)

min = MAJ

Case independant (option set) or dependant (option cleared) search

Not equal

Reverse search (option set, will stop on the first record which field doesn't match) or normal search (option cleared)

The last two options are available only for "ALPHA" and "ALPHANUM" fields.

Seach next

Continues searching, starting from next record.

1.11 AFileEnglish.guide/SpecialMenu

The Special menu

=====

Add sort key

Add a sort key on a specified field. A first requester will appear so you can specify the field which is to be used for sorting. A second requester will allow you to specify sort parameters: increasing/decreasing order, case dependant/independant. Once the sorting done, AFile will display the first record in the sorting order. Then all your moves withing the records (first/previous/next/last) will be made in the sorting order.

Take care that AFile only does a "logical" sort: the order of the records within the file is not actually modified. Sorting a file may use a lot of memory if the file contains a lot of records. As long as the sort is activated, you won't be able to add or remove records, to modified a value in the field which is used for sorting, and to use the Undelete or Pack file menu items.

Remove sort key

Removes the last, or all sort keys. Once all the sort keys have been removed, you will be able to add or remove records, and the moves within the file will use the "physical" order of the records in the file.

Reorder records

This actually modifies the order of the records in the file (you will be asked for confirmation). Take care that this operation may

fail if there is not enough free space on the volume where is the file, and that all deleted records will be definitely lost.

Execute script

Allows to run an AREXX script. A file requester will allow you to select the script to run. See AREXX Interface.

1.12 AFileEnglish.guide/ImportExport

Importing/Exporting data

Selecting the Import (or Export) item brings up a requester, which allows you to enter all the required parameters to import (or export) data.

The first parameter is the name of the source (or target) ASCII file.

Then you will have to specify the field delimiter and record delimiter characters. These delimiters may be entered as a single character (i.e. "/") or as an ASCII decimal code (i.e. 10 for a new line character). If you want to use an ASCII decimal code, it must have at least two digits: "2" specifies character "2", "02" specified character which ASCII decimal code is 2.

Last, you will have to specify the order in which the fields are to be imported (exported): enter "1" for the first field, "2" for the second field, etc... If you don't specify a order number for a field, it won't be imported (exported). When importing data, the order number may be considered as a column number: it allows to ignore some data, or to swap two fields.

1.13 AFileEnglish.guide/InputMask

Input masks

Input masks allow to do a lot of things: customize data display, menus and printing, help the user when he (or she) is entering data, check the data he (or she) has entered.

From AFile's point of view, a mask is an AREXX script with comments in a special format. The general form of this script is like follows:

```
/*
 * Sample input mask for AFile
 *
 * $MSG
 *     <message specification>
 * $END
```

```

*
* $MASK <background picture>
*     <mask specification>
* $END
*
* $MENU
*     <menu specification>
* $END
*
* $PRINT <print script>
*
* $PRINTFMT <print format string>
*/

```

PARSE ARG field value

<AREXX intructions>

EXIT 0

You don't have to enter all the specifications: an input mask can only have a single \$MSG specification (for example). Within a specification, you may also enter comments:

```

/* $MSG
*     specification line
**     comment line (begins whith " **")
*     specification line
* $END
*/

```

To select an input mask, you have to use the "Select mask" menu item. To cancel the mask, use the "Forget mask" item. See The Edit menu.

You may also associate an input mask to a data file, so it will automatically be loaded each time the data file is opened. In order to use this possibility, the input mask file's name must be equal to the name of the data file, followed by the .mask suffix. For example, if you want to define an input mask for a data file named Addresses, it's name must be Addresses.mask. The two files must be in the same directory.

\$MSG	: Message specification
\$MASK	: Mask specification
\$MENU	: Menu specification
\$PRINT	: Print script specification
\$PRINTFMT	: Print format specification

1.14 AFileEnglish.guide/MsgSpec

\$MSG specification
=====

The specification lines must contains:

1. a field's name
2. a flag telling if the verification script is to call
3. a message

Here is an example:

```

/* $MSG
**      Field's name      Verify?      Message
*      NUMBER            -              Product number
*      QUANTITY          CHECK         Quantity supplied
*      SUPPLIER          -              Supplier's name
*      SUPPLY_DATE       CHECK         Date of supply
* $END
*/

```

The message is displayed above the status line when the corresponding field is selected. If the "CHECK" keyword is present in column two, the AREXX script which contains the input mask is called each time the value of the field is modified, with the field's name and the new value as arguments. The script must return a 0 return code if the value is accepted, or a non-0 return code if the value is refused. Date values will be supplied in the "AAAAMMDD" format, and boolean values will be either "T" or "F".

The script cannot use the SEEK, WRITE, APPEND, DELETE, SEARCH, SORT, RELEASE, SETFIELD, and CLOSE commands on the current field. See AREXX Interface.

1.15 AFileEnglish.guide/MaskSpec

\$MASK specification

=====

This specification is more powerfull than the \$MSG specification, because it allows to load a background picture, and to set the position of the fields on the screen. The specification lines must contains:

1. a field's name
2. the left offset (X position, in pixels)
3. the top offset (Y position, in pixels)
4. the field width (in pixels)
5. a flag telling if the verification script is to call
6. a message

Here is an example:

```

* $MASK          Work:Pictures/AFile/Address.pic
**      Field's name      XPos      YPos      Width  AREXX?  Message
*      NUMBER            400      4        -      -      Customer's number
*      FIRSTNAME         400      26       100    -      Customer's first name
*      NAME               400      48       100    -      Customer's name
* $END
*/

```

The (optional) file's name after the \$MASK word must correspond to a picture file, in the standard IFF format. This picture will be loaded as the background picture, while the input mask is loaded. The picture's palette will be loaded too.

AFile is able to load a picture which is not in the same resolution that the screen used. You may use the DEPTH argument to specify a number of colors (see Startup).

The X and Y positions, and the width, are used to display the fields on the screen. If you specify "-" for the width, the default field's width is used.

The message is displayed above the status line when the corresponding field is selected. If the "CHECK" keyword is present in column two, the AREXX script which contains the input mask is called each time the value of the field is modified, with the field's name and the new value as arguments. The script must return a 0 return code if the value is accepted, or a non-0 return code if the value is refused. Date values will be supplied in the "AAAAMMDD" format, and boolean values will be either "T" or "F".

The script cannot use the SEEK, WRITE, APPEND, DELETE, SEARCH, SORT, RELEASE, SETFIELD, and CLOSE commands on the current field. See AREXX Interface.

1.16 AFileEnglish.guide/MenuSpec

\$MENU specification

=====

The specification lines must contains:

1. the menu item's title
2. a flag telling if arguments are needed
3. the name of the AREXX script to run

Here is an example:

```

/* $MENU
**      Item's name      Arguments?      Script's name
*      Weekly stats     -              Work:Lib/Stats.rexx
*      Daily extract     ARGS,QUIET     Work:Lib/Extract.rexx

```

```
* $END
*/
```

The items will be added in a new menu: the Extras menu. The first ten items will have a keyboard shortcut. Each time an item will be selected, AFile will call the corresponding AREXX script, after having eventually asked some arguments (in the case the "ARGS" keyword is present in column two), and without output window if the "QUIET" keyword is present in column two.

1.17 AFileEnglish.guide/PrintSpec

\$PRINT specification
=====

The \$PRINT word must be followed by the name of the AREXX script to call to print records.

The script will be called each time a record is to print, with the fields of this record directly accessible using the GETFIELD AREXX command. The output will be redirected to the printer, so you will just have to use "SAY" command to send data. See AREXX Interface.

This specification overwrites the optional PRINTSCRIPT argument (see Startup), as long as the input mask is loaded. Any \$PRINTFMT specification before a \$PRINT specification will be ignored.

1.18 AFileEnglish.guide/PrintFmtSpec

\$PRINTFMT specification
=====

The \$PRINTFMT keyword must be followed by a format string, showing how to print records. For each record to print, data will be formatted using this string, before being sent to the printer.

The format string can contain the following specifications :

%num[:{l|c|r}[size]]

Replaced by the value of the "num"-th field, eventually left justified (l), centered (c) or right justified (r) on "size" positions. If "size" is omitted, the field's size is used.
Examples : %3 %2:r %11:c20

\$no[:{l|c|r}[taille]]

Replaced by the name of the "num"-th field, eventually left justified (l), centered (c) or right justified (r) on "size" positions. If "size" is omitted, the field's size is used.

\n

Replaced by a line-feed character

\f

Replaced by a form-feed character

\char

Replaced by the given character. This allows to include special characters like : %, \$, \

Any \$PRINT specification before a \$PRINTFMT specification will be ignored.

1.19 AFileEnglish.guide/Alias

Aliases

Aliases allow you to make data entering easier, by defining shortcuts. An example is the possibility to enter "?" in a DATE field, in order to have today's date. This possibility is built in AFile, but you can also define your own aliases, using a text file made of lines containing:

1. the field's name
2. the alias value
3. the corresponding value

Here is an example:

```
;Field Alias Value
COUNTRY FRG FEDERAL REPUBLIC OF GERMANY
COUNTRY USA UNITED STATES OF AMERICA
COUNTRY UK UNITED KINGDOM
```

With this example, each time the user will enter the "USA" value in the "COUNTRY" field, this value will automatically be replaced by the "UNITED STATES OF AMERICA" string. Take care that aliases substitution is performed before running an eventual verification AREXX script (see Input masks).

The aliases file's name must be equal to the name of the data file, followed by the .alias suffix. For example, if you want to define aliases for a data file named Addresses, the aliases file's name must be Addresses.alias. The two files must be in the same directory. The aliases file will automatically be opened when the data file is opened.

1.20 AFileEnglish.guide/AREXX

AREXX Interface

AFile has a full AREXX interface, so AREXX may be considered as AFile's programming language. The name of the AREXX port is AFile_rexx. You can run a script using the Execute AREXX menu item, either from the Project menu of the main screen, or from the Special menu of the data access screen. In this case, the current file is automatically selected as the working file, and the current record in this file is the record currently displayed.

The following commands are accepted:

OPEN name

Opens the given file, and go to the first record of this file

This file is NOT selected as the working file.

SELECT name

Selects the given file as the working file. This file must have been opened using the OPEN command.

CLOSE name

Closes the given file.

CLOSE ALL

Closes any opened file.

The following commands are applied to the working file:

APPEND

Appends a new record, and moves to this record.

SEEK FIRST

Moves the first record.

SEEK LAST

Moves the last record.

SEEK NEXT

Moves the next record.

SEEK PREV

Moves the previous record.

SEEK n

Moves the n-th record.

SEARCH name value [NOCASE] [EXACT] [NOTEQUAL]

Searches a record which field named "name" contains the "value" value. If the "NOCASE" option is specified, the search will be case independant. If the "EXACT" option is specified, AFile will search a field equal to the value. The "NOTEQUAL" option allow to find a field not equal to the value.

SEARCH NEXT

Searches next occurrence.

INFO

Returns information about the current file, in the RESULT variable:

"full pathname" <number of fields> <record size> <number of available records> <number of deleted records>.

SORT name [DECREASE] [NOCASE]

Adds the given field as a new sort key. If the "DECREASE" option is specified, the records will be sorted in reverse order. If the "NOCASE" option is specified, the sort will be case independent.

RELEASE {LAST|ALL}

Removes the last, or all sort keys.

The following commands are applied to the current record of the working file:

WRITE

Updates the current record.

DELETE

Deletes the current record.

GETFIELD name

Returns the value of the field named "name", in the RESULT variable.

SETFIELD name value

Modifies the value of the field named "name" to "value". Caution, the value must be specified within double quotes !.

FIELDTYPE name

Returns the type of the field named "name" ("A", "N", "S", "D" or "B")

FIELDLEN name

Returns the length of the field named "name", as declared in file's structure definition. For numeric fields, the decimal point is taken in account if the number of digits is not 0.

If failed, all these commands return 10 in the RC variable. They return 0 if all is ok.

1.21 AFileEnglish.guide/History

History

AFile is written in C language, and was developed on an Amiga 3000 UNIX-1 (10 Mb RAM, internal HD disk drive, two internal hard disks of 100 Mb and 160Mb, external SyQuest drive of 88 Mb, external floppy disk

drive, 1960 monitor and Star LC24-10 printer) connected by a null-modem cable to an A500 with 1 Mb RAM.

v1.00, 25-Jul-93, 60448 bytes
First released version.

v1.10, 07-Aug-93, 62632 bytes
Added input masks.

Added AREXX commands "FIELDTYPE" and "FIELDLEN".

When a sort is activated, displays "sorted on <field's name>" in the status line (instead of just "sorted").

v1.20, 06-Sep-93, 64336 bytes
The number of fields is no longer limited.

Sorting modified: field selection with a request box, added "lwr = UPR" option, AREXX interface enhanced.

You can select the file to import/export with a file requester.

v1.21, 12-Sep-93, 64500 bytes
Uses new versions of OuvreEcran(), RequeteFic() and GetDefaultTextFont() functions.

v1.30, 15-Sep-93, 65612 bytes
Localized.

v1.31, 07-Oct-93, 65704 bytes
When an AREXX script is finished, waits for the user to press RETURN before closing the output window.

Bug fixed: sorting numeric fields was made on the inverse order than asked.

v1.40, 14-Oct-93, 67036 bytes
You can modify the structure of an existing file.

Added aliases.

Reports errors occurring when loading input masks and aliases.

Bug fixed in string requests handling.

Some source cleanup.

v1.41, 17-Nov-93, 67292 bytes
Update the file header each time a record is added/deleted (instead of waiting for the file to be closed).

A few enhancements in source files.

When a request is displayed, locks main window with a Requester.

Bug fixed in refresh of gadget associated with boolean fields.

A few changes in order to support german catalogue.

v2.00, 28-Nov-93, 68960 bytes

You can now encrypt data in your files.

Added "PRINTSCRIPT" argument.

Bug fixed in fast copy of data fields: ln_Pred of first field, and ln_Succ of last field, were not updated.

v2.10, 11-Dec-93, 69508 bytes

Changed input mask files format: there is now a single file (the AREXX script) which both specifies messages and performs verifications. The old input mask files are still recognized.

You may have an input mask file be automatically loaded when a data file is open.

Status and gadgets lines are now just below the last field's gadget, instead of being at the bottom of the screen.

When creating/changing a file's structure, changing a field's type no longer modifies the field's length when the new type is "DATE" or "BOOLEAN".

v2.20, 25-Dec-93, 70876 bytes

Extended input masks: custom menu with \$MENU, print script selection with \$PRINT.

Compiled with SAS/C 6.50.

v2.21, 26-Jan-94, 70876 bytes

Bug fixed: didn't restore default public screen after running an AREXX script.

Shows my new address.

v2.30, 20-Feb-94, 71412 bytes

Handles keyboard : arrow keys (+ shift), "p", "g", "DEL".

Added USEASL argument.

In an input mask, a line beginning with " **" is a comment.

Default number of records to print is set to one.

The field requester has now a blue background.

Compiled with SAS/C 6.51.

v3.00, 26-Feb-94, 74696 bytes

Added \$MASK specification (full mask with background picture and field positioning).

Added DEPTH argument.

Enhanced internal file requester.

Enhanced error handling (now displays some messages...).

Documentation formatted by TexInfo/MakeGuide.

v3.01, 05-Mar-94, 74732 bytes

Bug fixed: displayed an error message when opening a file without corresponding .mask file.

Restores palette when a file is closed.

v3.02, 25-Mar-94, 75088 bytes

Bug fixed: used a graphics.library V39 function (so worked only under OS 3.1).

Bug fixed in AREXX message handling (some commands were refused)

Bug fixed: sending arguments to the verification AREXX script was not good.

Displays type and size of the current field, on the status line

v3.10, 06-Apr-94, 78960 bytes

Added new field type "ENUM"

Rewritten search fonction : now opens a window for search options, added "exact match" and "not equal" options

Added "FONT" argument

"ALPHA" type fields now only accept lowercase and upercase letters (with or without accents), space, dash and single quote

Displays current field's type and size in the status line

Adds the pathname of the current data file if a script file name has no path in an input mask

With full input masks (\$MASK specification), displays status lines and action gadgets in the bottom of the screen

Enhanced field request : if a data is missing when adding/inserting/replacing a field, the corresponding gadget is automatically activated

Several bug fixed (was eating memory, problems with loading of input masks, ...)

v3.11, 16-Apr-94, 79220 bytes

Added "NOCASE" argument

Some source cleanup/reorganization (data access interface)

Locks input for structure definition window when the file requester is displayed ("Copy structure" item)

Removed fast field definitions copy (can't be done in a "system

friendly" way, generated memory corruption)

Bug fixed in CLI argument parsing

v3.12, 21-May-94, 79220 bytes

The data access Intuition interface now handles century in dates

Bug fixed : "TOOL TYPES" were not scanned the good way

v3.13, 05-Jul-94, 79684 bytes

Added the AREXXWIN argument

The name of the public screen is "AFile"

A few bugs fixed in IFF pictures loading

v3.20, 27-Jul-94, 80368 bytes

Modified input masks format (added \$PRINTFMT specification,
CHECK/ARGS/QUIET keywords)

Added "Close and quit" item in "File" menu

Bug fixed: sometimes didn't redraw gadget bar

Bug fixed: after sorting or packing, if you select "forget mask"
the display was not good, and "select mask" made the system crash

Bug fixed: a lonely \$PRINT specification in an input mask was
ignored

v3.30, 25-Sep-94, 81060 bytes

The files may now be sorted on several keys
