

BareED - A Brief Introduction

COLLABORATORS

	TITLE : BareED - A Brief Introduction		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY		January 20, 2025	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	BareED - A Brief Introduction	1
1.1	BareED 0.9 - a dirty introduction	1
1.2	This page is under construction....	2
1.3	How to deal with BareED so that no crash occurs...	3
1.4	Intro Install Requirements Language Use	5
1.5	BareED's button interface	11
1.6	The Amiga Rexx Interface of BareED	16
1.7	Amiga Rexx problems	22
1.8	Disadvantages and faults - oh no!	24
1.9	BareED is able to save icon imagery	26
1.10	Internals to BareED	27
1.11	Copyright and Distribution	29

Chapter 1

BareED - A Brief Introduction

1.1 BareED 0.9 - a dirty introduction

Please, do not expect too much from this file nor from BareED itself, BareED is currently only available as a pre-version (beta) that suits my need and perhaps yours. Although BareED runs very stable on my system, it is possible that BareED will fail on yours!

- What is BareED?

BareED is a simple text editor using the ASCII one character set for Amiga computers.

- Why another text editor when there are already hundreds of them out there?

I originally designed BareED as replacement for Notepad, the first word-processor for the Amiga, Amiga-ED, the batch and script editor, and Amiga-MEmacs, the text-editor.

What I liked most was that the colours could be chosen in Notepad and that Notepad gave the ability to use proportional fonts. Even Notepad is more powerful than BareED it has got so many limits that only a few people have used NotePad.

Using Amiga-MEmacs on an intuitive driven platform like the Amiga is a pain in the butt. Even the newer versions of Amiga-ED aren't comfortable to use nor they can be used to edit normal text files.

My favourite text-editor is the one shipped with the Devpac\$^1\$ package - clean and easy to use!

BareED is one of the few editors on the Amiga that make no use of the Amiga console device and therefore non-rigid colours can be used and of course non-fixed width fonts.

BareED also is one of the few editors that deal correctly with a sizeable editor window; even commercial editors have problems with that. So you can use BareED as notepad on your

Workbench that means that you size BareED's editor window to minimum and leave it anywhere on your Workbench desk open while you do other things. When you need it, for example to remember important stuff, you activate BareED only (without sizing the editor window to maximum) and enter the letters.

Although BareED is relatively quick on my system it is maybe dramatically slow on yours; that is due to the calculations that must be done before something is performed; nothing is rigid in BareED! Therefore it is quite modest of consuming memory but can slow down your machine to its limit while you scroll around in the text file or enter characters.

BareED has not been designed for native Amigas (like an A1200 or A4000) but for accelerated machines with a (for Amigas) quick CPU. As if that's not bad enough, BareED is only useable with a fast (and I mean fast!) interface to access system memory (forget any ZIII memory expansion card!) and with a 3rd party graphic device.

Additional informations can be found here, if you encounter any problems click the help-button in the status-bar of this window.

BareED's primitive button interface

ARexx supported macro commands, still under construction

Icon save and icon imagery

Disadvantages and faults of BareED

Internals to BareED

Copyright, distribution

\$^1\$) Devpac, © of HiSoft, UK

1.2 This page is under construction....

- System - the operating system, stuff to make the machine look like she is alive
- ASCII one - international standard format of characters that fit into a byte
- Character set - format of bit masks representing the keys on your keyboard
- Notepad - first Amiga word processor, 1985 (so far I know)
- MEMacs - text editor, 1986
- ED - text-editor for batch and script files, 1986, enhanced 1991
- Proportional fonts - visualised character set where each character can differ in width

- Fixed width fonts - visualised character set where each character has got the same width ←
- Font - character set that can be visualised, dropped as file
- Workbench - platform to perform things without touching the keyboard
- Console device - high level interface to simplify text handling and text output ←
- ARexx - Amiga specific implementation of Rexx (script interpreter)
- ZIII - expansion slot (Zorro) with 32 bit wide address range, update to ZII (24 bit) ←

1.3 How to deal with BareED so that no crash occurs...

To make BareED running you must first install (if not already done years ago) an Asl library module of at least version 38. If you did this, make sure you own OS 3 and a 68020 or better CPU. Use also the latest available SetPatch.

Click double on the icon labelled "BareED". A window appears. Here you can enter the characters; load in files, delete or modify characters in a specific file and of course save them back .

----- !!!! -----

NOTE: Currently there is no UNDO function in BareED implemented, so if you make a mistake it cannot be un-done. So be a bit carefully.

----- !!!! -----

Make sure when you try BareED for the first time on your system to save first any important things, close all 3rd party applications and be sure you have at least the minimum configuration that BareED presupposes. If BareED crashes your system it's very likely that some applications in your Workbench start-up drawer are incompatible to BareED. I have encounter also a crash that occurs when BareED is started after a commercial Basic compiler / interpreter quits. The crash occurs when BareED attempts to allocate memory. It's not a fault of BareED instead this compiler / interpreter corrupts the memory list of the system. So be (again) carefully.

If BareED appears but doesn't display the font set in the icon of the file you want to load in it is possible that the disk font library module or the font itself cannot be opened, either because of no more obtainable RAM or inaccessible files.

If the Asl screen mode requester module does not pop up the setting in BareED's icon set either to FOREIGN or NATIVE and you don't have a 3rd party graphic board support software running, or your machine does not support a DMA screen mode, respectively.

If you changed by mistake all line-feeds through Carriage-returns (form-feed) and BareED tells you that it cannot find any of those Carriage-returns you have to add one line-feed at the end of your very loooooonnnng line. Then, BareED will find them. This applies to all text files; make sure there is at least one line-feed.

By the way, you can enter a line-feed by pressing CONTROL + left Amiga + j; and for a Carriage- return you press CONTROL + left Amiga + m in the "Find & Replace" requester.

If you press the "Replace All" button do not activate the text editor window under any 3rd party graphic software emulation! If you do, you will encounter a software failure that ends in a GURU! This is caused by a corrupted memory list due to invalid calls to Exec PutMsg() and GadTools GT_PostFilerIMsg(). Since BareED doesn't use PutMsg() and GT_PostFilerIMsg() on its own it's very likely that this fault is caused by belated react and reply to Intuition messages since BareED is heavy busy doing a job (replacing strings). BareED 0.9428 tab handling has been re-written so that it is now drastically faster (where necessary - RpPrintLine(), SetCursorXY(), CursorLineEnd()). This should solve the above stated, but on a heavy loaded system and even under a slow CPU it might fail. I've also tried to lock the layer of BareED's editor window for the time being busy, but this has caused in critical circumstances deadlocks.

If BareED crashes your system upon starting from Workbench or Shell take a look of the given stack size. BareED requires on a native Amiga with OS 3 and no 3rd party graphic emulation 2200 bytes of stack. With OS 3 and a 3rd party graphic emulation system it requires at least 3000 bytes of free stack (tested with CyberGraphX and Picasso96). Newer versions of those Amiga graphic emulation systems may need more stacks. So if you encounter such a problem increase the stack size step by step by 1024 bytes until BareED starts up and runs correctly. This applies not only to BareED but also to all other applications that can handle both, the native Amiga graphic device and any 3rd party. In addition, BareED has got now a stack check implemented. If BareED does not pop up when started from Workbench, the amount of free stack memory has been chosen too low. Increase the stack size in this case as stated 6 lines before. If this happens when started from a CLI, a message will appear telling you that BareED cannot continue which such less stack (for your safety).

On my pure system (A4000 Desktop from '92) I have not installed any patches not written by me. The only exception is the Picasso96 software package (used to make that CyberVision 64/3D useable) and thus I do

not encounter any problems when using BareED, in mind the hints given here.

[Click here for "Amiga Rexx" related problems](#)

1.4 Intro Install Requirements Language Use

BareED is a new editor for your Amiga/Draco computer.

BareED is actually only available as beta application (not ready yet) - but ←
some simple files can be edited
anyhow.

It has a new concept (as opposite to other Freeware editors)

Designed for non-mono-space fonts (proportional fonts).

Does not use the system's console-device so that the colours can be chosen for ←
the background,
text, cursor, and text underneath the cursor and for the marked characters.

Currently it's a really "bare" but further releases may get stronger.

Because I have not the time to write a complete guide in how to use ←
BareED you should read this short
introduction carefully.

%%%

Installing BareED

(1)

To install BareED you have nothing else to do than to drag the program icon ←
of BareED into your favourite
drawer.

(2)

If you wish to store BareED into a currently non-existing drawer select "New ←
Drawer" from the Workbench
window menu. Make sure you have got selected a suitable medium and drawer where to ←
create the new one.

If the new directory has been created, execute step one as stated above.

To install the complete package of BareED including Rexx-script, button interface ←
and the German catalogue

simply drag "BareED's" drawer to your favourite position on your hard drive.

No "ASSIGN" to "BareED's" "home-directory" is needed to run BareED!

%%%

Requirements

BareED can be used up from OS 3.0 (perhaps also up from OS 2.0 when ←
the Asl library of OS 2.1 is
installed, but this has not been tested by me; furthermore, pen sharing and re- ←
loading will then not work!

There aren't any specific disk resident libraries and devices used by BareED ←
 other than those your machine
 was originally equipped with.

Requirements (minimum):

Exec v36
 Dos v36
 Graphics v33
 Layers v33
 Intuition v36
 GadTools v36
 Diskfont v33
 Icon v33
 Asl v38

CPU 68020, 512Kb RAM, OCS

Better:

Exec v36
 Dos v36
 Graphics v39
 Layers v39
 Intuition v39
 GadTools v39
 Diskfont v33
 Icon v33
 Asl v39
 Locale v38

CPU 68030, 2MB 32-Bit RAM, AA

Perfect:

Complete Kickstart v40 package (OS 3.1)
 68040 with onboard RAM
 3rd party graphic card plugged into Zorro 3 or Draco direct slot
 Installed Picasso96 software package
 Asl library v41/42 (Asl v40 has got several bugs...)
 Several proportional fonts either bitmap related or outlined
 Ttf (true type font engine) package by Richard Griffith

%%%

Installing a suitable catalogue (OS 2.1 (Kickstart v38))

To install the catalogue containing the native language strings you have first to ←
 create the catalogue:

Edit the ".ct" file
 Use "CatComp" utility (or an equivalent application) to produce the "catalog" ←
 file.
 Create a sub-directory on the medium where you have stored BareED called " ←
 catalogs".
 Create in this new created directory another one that is called exactly ←
 as your native
 language, e.g. "dansk".

Move the translated (via "CatComp") "catalog" to this location.

Fully:

"Work:Edit/Editors/BareED" - Home-directory of BareED

```
1> dir work:edit/editors/BareED
BareED      BareED.doc
1>
1> mkdir work:edit/editors/BareED/catalogs
1>
1> mkdir work:edit/editors/BareED/catalogs/dansk
1>
1> copy ram:BareED.catalog to work:edit/editors/BareED/catalogs/dansk
1>
1> dir work:edit/editors/BareED
BareED      BareED.doc
catalogs (dir)
1>
1> dir work:edit/editors/BareED/catalogs
dansk (dir)
1>
1> dir work:edit/editors/BareED/catalogs/dansk
BareED.catalog
1>
```

If you are an average user of the Amiga OS you might ask why to create a ↵
sub-directory in the drawer
where BareED stays instead of using "SYS:locale/catalogs/.../BareED.catalog".
The reason why is: If you delete from Workbench the BareED drawer BareED and ↵
used by it files will also be
deleted, i.e. you don't have to scan through your system-partition to look for ↵
BareED used files, simple - eeh?

%%%

Currently supported keys

```
-----
CURSOR-UP    - move cursor to previous line
CURSOR-LEFT  - move cursor one position to the left
CURSOR-DOWN  - move cursor to next line
CURSOR-RIGHT - move cursor one position to the right
```

```
SHIFT CURSOR-UP - move cursor to top of page, when cursor already set to this
                  position, cursor is moved to previous page
SHIFT CURSOR-LEFT - move cursor to start of line
SHIFT CURSOR-DOWN - move cursor to bottom of page, when the cursor already ↵
                  set to this position,
                  cursor is moved to next page
SHIFT CURSOR-RIGHT - move cursor to end of line
```

```
SHIFT DELETE    - delete from cursor position all characters to end of line
SHIFT BACKSPACE - delete from character left of cursor all characters to start of ↵
                  line
SHIFT RETURN     - insert Carriage-return code (character 13) into archive

CTRL CURSOR-UP  - move cursor to first character of archive
```

CTRL CURSOR-DOWN - move cursor to last character of archive

ALT CURSOR-LEFT - find previous word, number or single letter within current line
 ALT CURSOR-RIGHT - find next word, number or single letter within current line

ALT RETURN - insert FormFeed code (character 12) into archive

Amiga RETURN - terminate line with a linefeed (character 10) and auto-indent ↵
 the next line

CTRL A - arrange text to block format
 CTRL B - enter right margin for block format (requester pops up)
 CTRL C - change word's first letter into a capital
 CTRL F - find next occurrence (find string must have been already ↵
 entered in the
 Find/Replace requester)
 CTRL G - change letters of word underneath of cursor into capitals (↵
 association: GREAT)
 CTRL K - delete from cursor position all characters to end of line
 CTRL M - fence-match, point cursor to one of these characters: ([{ < > }] ↵
)
 CTRL N - find next occurrence (find string must have been already ↵
 entered in the
 Find/Replace requester)
 CTRL P - find previous occurrence (find string must have been already ↵
 entered in the
 Find/Replace requester)
 CTRL R - replace occurrence (find and replace strings must have been ↵
 already entered
 in the Find/Replace requester)
 CTRL S - change letters of word underneath of cursor into lower ↵
 case letters
 (association: SMALL)
 CTRL U - delete from character left of cursor all characters to start of line
 CTRL W - delete word or number underneath of cursor
 CTRL X - delete current line
 CTRL Y - delete current line
 CTRL Z - arrange text to block format but in an AmigaGuide compatible manner

+++ DON'T USE ONE OF THE FOLLOWING KEY COMBINATIONS

CTRL H	CTRL I	CTRL J	CTRL L
CTRL O	CTRL Q	CTRL T	CTRL [

+++++

CTRL DELETE - delete current line

BACKSPACE - delete character left of cursor
 DELETE - delete current character
 RETURN - terminate line with a linefeed (character 10) (paragraph)
 ENTER - same as RETURN but with auto-indent of characters

Left mouse button - move cursor to position of mouse pointer

Right Amiga B - start marking an area

Right Amiga X - cut away the marked area (goes into clipboard)
 Right Amiga C - copy the marked area (goes into clipboard)
 Right Amiga V - insert earlier in clipboard remembered marked area

Double mouse click - start marking an area

+++ A HINT +++

If you want to mark a really large number of characters use the mouse (it's faster ↵):

Click once on the character that represents the first to mark character

Click again on this character (so called double click)

Click mouse one character behind the last to mark character

- Area shown in selected mark-colour

- You can now cut, copy this area

+++ A REMARK +++

If you lay out your text using the AmigaGuide compatible mechanism (CTRL-Z ↵), ensure that a brace-left

character is preceded through the at-sign and that the brace-left character ↵ has got a following brace-right

character. Otherwise, in case not, the AmigaGuide compatible mechanism is ↵ broken up and the result (text

layout) is very difficult to restore to your original layout. Another ↵

disadvantage is that italic strings may not be

laid out correctly.

Find Requester

n & Shift N - find next occurrence

l & Shift L - find previous occurrence

p & Shift P - find previous occurrence

Number Requester

Return & Carriage-return - leave requester with valid result of number gadget

%%%

Settings and preferences

When "Create Icons?" is enabled BareED writes along with the archive the ↵ settings you have chosen for this

archive, for example the colours.

BareED has got no global preferences with two exceptions: You can enter in ↵

BareED's icon the monitor type

you wish the Asl requester module to display and whether you like to ↵ reserve the pens taken for the

knob-bank so that other application cannot use these pens and no false colours ↵ occur.

Both are entered normally through the use of the tool types.

MONITORTYPE=ALL|FOREIGN|LIKEWB|NATIVE

Where

ALL means to display any monitor available on your machine

FOREIGN means only to display non Amiga modes, i.e. you must have plugged in a 3rd party graphic device into your Z3 bus (for example)

LIKEWB allows only to display by Workbench supported modes, i.e. no

HAM, EHB, DPF, 15 bit and modes with an alpha channel

NATIVE means to display only those monitors that can be directly displayed by the Amiga hardware

Combinations of the above stated are allowed, such as

MONITORTYPE=FOREIGN|LIKEWB (e.g. for the Draco computer)

Or

MONITORTYPE=NATIVE|ALL

Please do not combine NATIVE and FOREIGN....

KNOBPENS=RESERVE

Tells BareED to reserve the pens taken for the knob-bank - in order to avoid each time re-mapping the colours to the new surround when other applications attempt to reserve pens for their own purpose, e.g. image viewers (like MultiView), Workbench games, icon patches (NewIcons) or OS 3.5 icon subsystem.

The option "IBM Compatible Save?" allows you to export your archive to another computer platform without doing the significant changes on your own. This means that the complete layout of your archive will be lost (on your Amiga) but that on an other machine like an IBM* compatible PC (e.g. under MS-DOS* software; especially under Microsoft Word 2000*) the archive layout will look close to that layout you have chosen on your Amiga. Those changes are not visible onscreen but when you load back this saved archive!

What BareED does, is to remove all line terminators (LineFeeds) for a paragraph. Because it is really hard to detect which lines belong to a paragraph on the Amiga platform you must prepare first your archive. Remove all non-standard text attributes (for example Escape codes and AmigaGuide macros / attributes). Then, enter the right margin (if you use the Helvetica font use a right margin of 153). Then change your layout of your archive to block format (tabulators are allowed). Now save your archive to a medium. BareED will now combine all lines that are not split into two and which are as long as your set right margin to

one paragraph.

```
*IBM      ® International Bureau Machines
*MS-DOS   ® Microsoft
*Word 2000 ® Microsoft
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Because this file is very short it's recommended that you use method "trial and error" when using BareED. BareED works just like other editors.

1.5 BareED's button interface

Before you start to use this feature of BareED make sure that you own a 68040 or higher CPU, OS 3.0 or better and at least 130 Kbytes (for the graphical details) of free memory. Ensure also that you are using a 256-colour screen with a resolution of at least 800 x 600 pixels (do not use the button interface if you only have one of the native Amiga graphic devices!). If this does not scare you, read ahead.

BareED offers a button interface that is disabled by default, due to the memory consumption and the time needed to set up the button interface. However, if you own a 68040+ CPU you may wish to use it.

No image for the button bank of BareED is limited to something. This means that the imagery may differ in width, height, depth and colour. By default no pens of the knobs are reserved in order to give any picture displayer the ability to display any imagery in the closest colour match. This can lead BareED to display those knobs in false colours when there are no more pens free, or when a picture displayer overrides BareED used pens - but that can be changed by BareED at any time, unfortunately, you have to tell BareED if the button bank appears ugly. If you don't agree with this, use the tool type setting that forces BareED to reserve these pens.

You should, even there is no such hard coded limit, limit your image to a height that can fit into the window when there is also enough room left to display one text line.

The imagery themselves have to be created through the use of an available paint program, e.g. Personal Paint. Your render image and the select image have to be of same size. You must define the border for the

knob. When you have made your imagery, save the render and the select image ↵
 for each knob to disk as a
 single file, where the render image is at the top and the select image at the ↵
 bottom - and no row splits them up!

```

|         |
|         |
|         | < Render Image
|         |
|_____|\_____ no room between
|-----| /
|         |
|         | < Select Image
|         |
|-----|

```

Suggested is an image with less than 48 pixels in width and height and that all ↵
 imagery have got the same palette.

You can easily use BitMapSaver (enclosed) to form the needed image format. You ↵
 should pay attention that
 BitMapSaver is only able to convert IFF-brushes (and pictures) and that it is a ↵
 palette-based program that is
 limited to 256 colours.
 You have to call BitMapSaver as follow:

```
1> bitmapsaver <name of iff image to convert> TO <knob file> OPT PEN RAW RAWHDR ↵
    CMAP RGB32
```

This image format can be directly read by BareED.

When you have created the knobs, you create an ASCII file which tells BareED to ↵
 use these imagery for the
 knob bank. Important is not the name you give an image (okay, it must be the ↵
 same as the filename), but the
 index.

```
2,knobs/Load.knob
3,knobs/Save.knob
4,knobs/New.knob
5,knobs/Close.knob
6,knobs/Print.knob
.....
```

In this example, those imagery will be taken from a sub-directory in the home ↵
 directory of BareED.

Currently, index numbers only from 2 to 22 are supported, where BareED ↵
 displays the imagery 5, 6, 11, 14
 and 22, 23 ghosted because they are internally unsupported.

The file describing the knobs must be labelled:

Button.cfg

And it must be in BareED's home directory.

You may change the order of the imagery to form your personal wish, e.g. you need the block layout button
most of all so you would enter as first entry this button:

```
20,knobs/Layout.knob
2,knobs/Load.knob
3,knobs/Save.knob
4,knobs/New.knob
..... and so on .
```

Following index' are assign to these functions:

- * 0 exit BareED
- * 1 move gadget within BareED's right window border
- 2 load a new file
- 3 save existing file out off editor
- 4 create a new, blank editor surround, ready to enter characters
- * 5 close the current editor surround
- * 6 print document
- 7 select screen mode and use it
- 8 select tab-step and use it
- 9 select a new font and use it
- 10 select pencils for document
- * 11 change palette of screen
- 12 snapshot marked block and save it as clip to the clipboard
- 13 insert clip in clipboard into current document
- * 14 un-do last "line remove"
- 15 write marked block to disk
- 16 insert a file into the current document
- 17 open the find-requester
- 18 open the replace-requester
- 19 open the go-to-requester
- 20 arrange current paragraph to block layout
- 21 arrange current paragraph to block layout but in an AmigaGuide compatible manner ↵
- * 22 open requester to execute an ARexx-macro
- * 23 set global preferences

(* = Unsupported)

BareED let you also choose the background pattern and colour beside the imagery and the vertical and horizontal distance between each image and the border of the knob-bank. There are synonyms to choose it. ↵

```
FILPAT    synonym for fill-pattern
PATPEN    synonym for pattern-pen
VSPACE    synonym for vertical spacing
HSPACE    synonym for horizontal spacing
ONEPAL    synonym for one palette
```


In order to make them available to BareED these synonyms have to be entered ↵
 in the same file where the
 knobs have been described, Button.cfg .

An example could look like this:

```
FILPAT=0
PATPEN=0
VSPACE=0
HSPACE=0
ONEPAL=0
```

The FILPAT is a long word representing a mask for set and unset pixels. ↵

Currently, it can be only entered as
 a decimal number.

If you want to disable anything use a semicolon in front of the to disable thing, ↵
 e.g.

```
FILPAT=1431677610
PATPEN=2
;VSPACE=1
;HSPACE=1
;ONEPAL=1
```

Comments may occur at any position within a line. Comments have to be ↵
 introduced through the semicolon.

Anything behind a semicolon to the line's end is ignored by BareED.

```
FILPAT=1431677610 ; hexadecimal ($) 5555AAAA, dual (%) ↵
01010101.01010101.10101010.10101010
PATPEN=2 ; use a white pattern (on my system)
VSPACE=1 ; use between each image (and between the pattern border and the image ↵
itself)
; one vertical line
HSPACE=1 ; ditto for horizontal
ONEPAL=1 ; all imagery will have the same palette thus tell BareED to compute ↵
the palette
; only once (30% faster)
```

Through the use of the button interface BareED gives you the ability to run DOS ↵
 scripts or commands. They
 are handled similar to normal knobs that call directly BareED intern functions.

First you enter the index, second the comma, third the filename, which ↵
 represents the image on disk, fourth
 one or several, blanks (or instead, tabs) and fifth enclosed within ↵
 parentheses the DOS command or script
 you want to use, e.g.

```
40,knobs/DosScript.knob (SYS:Utilities/MultiView)
```

You may of course add also an argument for the command you want to run, e.g.

```
40,knobs/DosScript.knob (SYS:Utilities/MultiView T:tempfile)
```

The index' that will allow to use DOS commands are index 40 to 63.

The main reason for me to implement this feature into BareED is to have easy access to ARexx scripts, so that following can be used:

```
40,knobs/ARexx_01.knob (SYS:RexxC/RX rexx/MyRexxScript.rx)
```

In this example the image for the knob is taken from a sub-directory in BareED's home-directory called "knobs" and the script for the Rexx server is called "MyRexxScript.rx" and it is taken from a sub-directory in BareED's home-directory called "rexx". The DOS-command is called "RX" and it resides in the "SYS:RexxC" drawer. By the way, the "RX" command will attempt to give away the work to the Rexx server, so that this server must be running (i.e. double click on the "RexxMast" icon in the SYS:System drawer before you can use your script). Starting with BareED 0.9518 you don't have to start the Rexx server manually. If the eleventh and the twelfth letter are capitals that point to the word "RX" BareED attempts to start the "RexxMast" program in the "SYS:System" drawer if the "RexxMaster process" is not already running. Note that no blanks may occur within the filename and pathname when the "RX" command is used. This is due to a bug in the "RX" command.

Example:

```
40,knobs/ARexx_01.knob (SYS:RexxC/RX BareED's Drawer for rexx scripts/My Rexx Script.rx)
```

The "RX" command will here try to get access to the file (!!!) BareED's.rexx. Even if you enclose such a string in double quotes it will not work since the "RX" command thinks in this case that this macro command is directly entered at the console. Example:

```
40,knobs/ARexx_01.knob (SYS:RexxC/RX "BareED's Drawer for rexx scripts/My Rexx Script.rx")
```

I strongly urge you to follow these guidelines:

Rexx scripts for BareED should be placed in one drawer and this drawer should reside in BareED's main (home) directory, e.g.

```
Work:Tools/BareED/      <- main (home) drawer of BareED
Work:Tools/BareED/BareED  <- objectfile of BareED
Work:Tools/BareED/Button.cfg  <- configuration file
Work:Tools/BareED/knobs/    <-drawer for knobs
Work:Tools/BareED/rexx/    <-drawer for rexx scripts
Work:Tools/BareED/rexx/Info.rx  <- a rexx script with a suffix
```

By the way, if your rexx script does not contain a suffix "ARexx" will add one on its own. This suffix is ever

".rexx". So "Info" becomes to "Info.rexx". You should remember this when you choose a name for your rexx script.

Now something smart:

BareED will deliver its Rexx port name in the clip variable BareED. The clip variable BareED will refer to the running copy of BareED which you have used to start the Rexx command / script through the use of the button interface (knob-bank). Example:

```
/* Rexx Demo - first line */
```

```
BareED_HOST = GetClip('BareED') /* Get the name, e.g. BareED.1, BareED.2 and so on
```

```
IF BareED_HOST = '' THEN DO /* Ensure we got it */
  CALL SetClip('BareED') /* Remove from ClipNode */
  EXIT 5 /* Error, no name! */
END
```

```
ADDRESS VALUE BareED_HOST /* Refer from now on to this running copy of BareED */
```

```
CALL SetClip('BareED') /* Remove from ClipNode */
```

```
.... here you can now let you script start
```

Of course it's not necessary to remove the clip via SetClip(BareED) but this saves memory thus I use it.

1.6 The Amiga Rexx Interface of BareED

First of all, this chapter and in general the ARexx-interface of BareED are under construction!

ARexx scripts can indirectly drive BareED, but at the moment BareED does not offer to execute ARexx scripts with one exception: the button interface of BareED gives you direct access to such scripts.

BareED is able to deal with strings containing up to three macro-commands where the first is called "Command-Index", the second "Object-Index" and the third "Parameter-Index".

So instead of writing:

```
GetAmountChars
  You write
Get Amount Chars
```

Which is first better to read and second for me easier to parse. The disadvantage is that it may collide with reserved ARexx symbol names or functions, e.g. Set Error Off, where "ERROR" is a reserved ARexx

command.

One goal of BareED's ARexx interface is the non-case-sensitive handling of over ↵
given strings. So you can

write:

```
get amount chars
```

Or

```
'get amount chars'      «« enclosed in single quotes
'get aMOunt chARs'
'GET AMOUNT CHARS'
```

It's non important to BareED! BareED splits a command line (string) into the ↵
above stated index' where hash
values are used instead of simple string comparison.

Currently BareED has got an ARexx interface with more than 80 macro-commands ↵
but only a few calls the
appropriate routines (under construction). Supported are:

```

additional string
-----|
|
Command-Index Object-Index  Parameter-Index «astr»  RESULT
-----
SET
ERROR    ON  |  OFF    -none-  ( useful when debugging )
ECHO     ON  |  OFF    -none-  ( useful when debugging )
FONT     "times.font,13"  -none-  should be clear....
FONTSTYLE 0,1,2,3 or combined -none-  try!
CLIPUNIT  0 - (almost) ~    -none-  0 suggested for textfiles
FIND     STRING    "string" -none-  the string to look for
REPLACE  STRING    "string" -none-  find string replaced with this one

FINDMODE -none-      -none-  Set default mode, case sensitive but no word ↵
        only search
        NORMAL      -none-  non case sensitive search
        WORDONLY    -none-  word only search

MARGIN   RIGHT    "number" -none-  set right margin in number of letters
TAB      SIZE     "number" -none-  "number" in amount of characters
BLOCKSTART -none-      -none-  set start point for marked area or erase it
BLOCKEND  -none-      -none-  set end point for marked area or erase it (not ↵
        necessary, anyhow...)
-----
GET
ARCHIVESTART -none-      address memory where characters start
ARCHIVEEND   -none-      address memory where characters end
REGION       -none-      address allocated memory for storage
REGION       SIZE       amount allocated memory for storage in bytes
ARCHIVENAME  -none-      filename without path
FILENAME     -none-      filename inclusive path

AMOUNT
CHARS        amount characters in archive
LINES        amount lines in archive

```

CHANGES amount of modifications
 CURRENT
 CHAR RC single character (RC unequal zero if end of archive)
 LINE RC complete string line (RC unequal zero if end of archive
 or only a paragraph)
 COLUMN offset in number of spaces in current line
 CURSOR
 X offset (in number of characters) in current line
 Y current line

 BVERSION -none- packed: version (high word), revision (low word)
 FONT -none- string: fontname - terminated by a comma - then font ↵
 height
 TAB SIZE amount of space characters a tab takes up
 CHARWIDTH "char" characters width in number of pixels
 BLOCK -none- add contents of clip to archive

PUT
 CHARS "string" -none- add string to archive
 CHAR "Q" -none- add single character to archive
 BLOCK -none- -none- copy marked block (written to clipboard)

LAYOUT
 NORMAL -none- -none- layout paragraph to normal block format
 GUIDE -none- -none- layout paragraph Amiga Guide compatible

MOVE

CURSOR
 LEFT -none-
 RIGHT -none-
 UP -none-
 DOWN -none-
 LINESTART -none-
 LINEEND -none-
 PAGESTART -none-
 PAGEEND -none-
 ARCHIVESTART -none-
 ARCHIVEEND -none-

 BLOCK -none- -none- cut and copy marked block (written to clipboard)

GOTO

 linenumber -none- -none-
 LINE linenumber -none-
 BOOKMARK 1 to 10 -none-

DELETE

CURRENT
 CHAR -none-
 LINE -none-
 WORD -none-
 TO
 LINEEND -none-
 LINESTART -none-
 CHAR
 LEFT -none-
 RIGHT -none-

```

      LINE      -none-      -none-
-----
LOCK
  ON      -none-      -none-  forbid modifications through user interface
  OFF     -none-      -none-  allow it
-----
ACTIVATE
  WINDOW  -none-      -none-  input stream set to editor window
-----
FIND
  NEXT    WORD      RC    unequal zero if none found
  PREVIOUS WORD      RC    unequal zero if none found
  NEXT    STRING    RC    unequal zero if none found
  PREVIOUS STRING    RC    unequal zero if none found
-----
REPLACE
  NEXT    -none-      RC    unequal zero if none found
  ALL     -none-      -none-  but requester pops up
-----
SAVE
  -none-   -none-      -none-  (but file-requester pops up!)
  "filename" -none-      -none-
-----
NEW      -none-   -none-      -none-
  "filename" -none-      RC    unequal zero if "filename" could not be loaded  ↵
  in
-----
QUIT     -none-   -none-      -none-
-----
TELL
  "string" -none-      -none-
-----
CASETELL
  "string" -none-      1 = yes, 0 = no
-----

```

Some notes at this point:

With "storage" I refer to the allocated memory region where characters can ↵
 be placed within. It is
 not identical to the physical address of the first character of the ↵
 file that has been previously
 loaded in, or to the first character you have entered in the text editor window.
 Instead, "archive" represent the group of single characters that are combined ↵
 together and which
 can be written to a medium as a single file.

The find/replace functions are still under construction (as the complete BareED ↵
 Rexx port).

The macro "FINDMODE" offers three choices:

FINDMODE - without any arguments, to set the initial state: case sensitive, no ↵
 search for "words only"
 FINDMODE - NORMAL, to search case insensitive
 FINDMODE - WORDONLY, to ignore combined words

I strongly urge you to call the FINDMODE command without an argument before you begin to search for a string, otherwise you run with unknown settings and unexpected results may happen.

```
move cursor archivestart
set find string "BareED"
set replace string "bare editor"
set findmode
set findmode wordonly
```

```
reps = 0
```

```
do while RC = 0
  find next string
  if RC ~= 0 then
    break
  replace next
  reps = reps + 1
end
```

```
tell "Found" reps "occurrences to replace!"
```

NOTE:

If you use "REPLACE NEXT" on its own - and currently the cursor does not point to the string you are looking for, "REPLACE NEXT" will only move the cursor to the next occurrence without replacing it. No error is returned, which means that you cannot distinguish between a simple cursor move and an exchange.

If you want to obtain information's from BareED then don't forget to use:

OPTIONS RESULTS

Otherwise BareED does not return a value or string. Following is an example that cares about all hints above stated:

```
/* Demo - First line */
address BareED.1 /* Refer from now on to first running copy of BareED */
options results /* Tell BareED to return values in case requested */

'set echo on' /* BareED should display any incomings */
'set error off' /* Pass through any encountered errors */

get amount chars /* Get amount of used chars, warning: zero possible - when archives empty! */

amount = result /* AMOUNT is a macro name of BareED, here used as variable, so the next time we refer to amount; we refer to the variable and not to the macro name!!! */
```

```
'get amount lines'      /* Using single quotes prevents ARexx to refer to the ←
    variable, the string is passed
    through to BareED as it! */
lines = result          /* LINES is also a BareED macro name, so see above */

get archivestart        /* Get memory address of first character or letter, thus ←
    archive pointer */
aptr = result

say "Archive at 0xD2X(aptr)", size in bytes:" amount "- where the archive ←
    contains" lines "lines."
```

[Click here for "Amiga Rexx" related problems](#)

By BareED reserved keywords in alphabetical order.

```
ACTIVATE  ALL      AMOUNT      ARCHIVE      ARCHIVEEND
ARCHIVENAME ARCHIVESTART  BLOCK      BLOCKEND  BLOCKSTART
BOOKMARK  BVERSION  CASETELL  CHANGES  CHAR
CHARS     CHARWIDTH CLIPUNIT    COLOR     COLORS
COLUMN    COMPUTE  CURRENT  CURSOR     DELETE
DEPTH     DOWN     DRAWERNAME ECHO      ERROR
FILE      FILENAME FIND       FINDMODE  FIRST
FONT      FONTSTYLE GET       GOTO      GUIDE
HEIGHT    INACTIVATE INFOWINDOW INITIALX   LAST
LAYOUT    LEFT     LEFTX     LENGTH    LINE
LINEEND   LINES    LINESTART LOCK      MARGIN
MARK      MOVE     NEW       NEXT      NORMAL
OFF  ON    PAGE     PAGEEND  PAGESTART
PENS     POSITION  PREVIOUS  PUT      QUIT
REGION   REPLACE  REQUEST   RESTORE  RIGHT
RPORT    SAVE     SCREEN    SET      SIZE
STRING   TAB      TELL      TO       UP
USED     WIDTH    WINDOW    WORD     WORDONLY
WORDS    X        Y
```

Reserved symbols by ARexx - not available to BareED

```
Abbrev()  Abs()  Addlib()  Address  Address()
AllocMem() Arg  Arg()  B2C()  BAddr()
BitAnd()  BitChg()  BitClr()  BitComp()  BitOr()
BitSet()  BitTst()  BitXor()  Break  Break_C
Break_D  Break_E  Break_F  C2B()  C2D()
C2X()  Call  Center()  Centre()  Close()
ClosePort() Compare() Compress() Copies()  D2C()
D2X()  Datatype() Date()  Delay()  Delete()
DelStr()  DelWord() Digits()  Do  Drop
Echo  Else  End  Eof()  Error
ErrorText() Exists() Exit  Export()  Find()
Forbid()  Form()  Forward()  FreeMem() FreeSpace()
Fuzz()  GetArg()  GetClip()  GetPkt()  GetSpace()
Halt  Hash()  HI  If  Import()
Index()  Insert()  Interpret  IoErr  Iterate
LastPos()  Leave  Left()  Length()  Lines()
```



```

MakeDir()   Max()   Min()   Next()   Nop
NoValue    Null()   Numeric  Offset()  Open()
OpenPort() Options  Otherwise Overlay() Parse
Permit()    Pos()   Pragma()   Procedure Pull
Push        Random() RandU()   RC      ReadCh()
ReadLn()    RemLib() Rename()   Reply()  Result
Return      Reverse() REXX...   Right()  RX
RXC  RXSET  Say   Seek()   Select
SetClip()   Shell  Show()   ShowDir() ShowList()
SigL  Sign()  Signal  SourceLine() Space()
StateF() StdErr StdIn  StdOut  Storage()
Strip()  SubStr() SubWord() Symbol()  Syntax
TCC  TCO  TE   Time()   Trace
Trace() Translate() Trim()   Trunc()  TS
TypePkt() Upper()  Value()  Verify()  WaitPkt()
When      Word()   WordIndex() WordLength() Words()
WriteCh() WriteLn() X2C()   X2D()   XRange()

```

1.7 Amiga Rexx problems

```

I   Unknown command
-----

```

When you encounter a problem that you cannot track down it's very likely that a function either of the rexx master or a rexx support library returned an error or even no error in the RC variable. In case that occurs ARExx tells you that the command (here BareED) returned the value 10.

For example I used this and similar fragments which caused a lot of error messages until it was solved:

```

Address BareED.1
Options Results

```

```

Delay( 2)
Put Char '0A'X

```

Seems to be okay to me. I opened the "rexxsupport.library" and therewith the Delay() function could be used. The mistake that I made was that I didn't cared about the RC variable where a function over gives the result of the function. In the ARExx manual is stated that a returned function code may not be explicit called, it's automatically done. I know that but never thought that the result of Delay() is passed immediately to BareED. In the example code above the ARExx-Server made following of the code:

```

Address BareED.1
Options Results

```

```

Delay( 2)

```

```
0      <<<< !!!!
Put Char '0A'X
```

When BareED encountered the character zero ("0") it didn't know how to handle it so it returned 10 (not known by me)! This mistake is not only visible when "Options Results" is used (application function, do and return a result).

This problem to solve is very easy; call a function so that ARexx knows that you don't care about a result. In the code fragment above it is done through:

```
Address BareED.1
Options Results
```

```
Call Delay( 2)      <<<< !!!!
Put Char '0A'X
```

The "CALL" command indicates that the RC variable has got no influence on the further "programme course". Thus RC is taken as unset by ARexx and therewith ignored, which leads ARexx to continue with "Put Char '0A'X" instead of "0". Nevertheless, the return code of the function Delay() can be check; it's not placed in the RC variable but in the RESULT variable!

```
I I BareED's macros become not recognize
-----
```

As already stated somewhere in this document, BareED has got a non case-sensitive interface to Arexx where a letter is ever treated like it would have been entered in upper case. The next is that BareED doesn't use single macro commands, e.g. "MoveCursorArchiveend", instead it will support "Move Cursor Archiveend". This has got the advantage to be more readable but can put anybody in trouble if he/she doesn't know which symbols ARexx and BareED reserve!

So a simple line like:

```
Set Error Off
```

Will cause trouble because "ERROR" is a reserved ARexx and BareED macro name! To avoid this enclose any probably from both used macro name in single quotes, e.g.:

```
'Set Error Off'
```

Now this string is passed through to BareED instead of being analysed by ARexx, because ARexx treats the

three words as one single line and therewith as a macro (which it does not understand).

Also, variable names can cause trouble, e.g.:

```
Get Amount Lines
Lines = RESULT      < OK
```

```
....bla bla bla
```

```
Get Amount Lines    < ERROR
Lines = RESULT
```

In the example I misused a reserved macro name as variable name, "Lines". Until the second "Get Amount Lines" is encountered all goes as it should, when now ARexx analysis's the second "Get Amount Lines" it encounters the "Amount" macro name followed by the "Lines" variable. If "Lines" has got the value 1300 ARexx would pass this string to BareED:

```
Get Amount 1300
```

Of course BareED cannot handle this. To avoid this there are several solutions, first: do not use variable names, which will also be used as macro commands by BareED, second: enclose the macro name, which collides with the variable name in single quotes, e.g.:

```
> Get Amount 'Lines' <, or which is in my opinion better, enclose all macro names in single quotes:
```

```
'Get Amount Lines'
```

As already told, it doesn't bother BareED if you write in lower or upper case or in mixed form. So you can even write:

```
'gEt aMoUnT lineS'
```

BareED knows how to handle this.

I I I Not know problems by author

Room for your extraordinary experiences with BareED and ARexx...

1.8 Disadvantages and faults - oh no!

Disadvantages and faults known by author

Does not work together with Nico François PowerSnap version xx. This means only that PowerSnap does not find the right characters for snapping since BareED reserves between each text-line a separate row.

Running on a screen with less than 4 colours and marking an area causes a little problem: cursor and marked area will be shown in the same colour. Thus it cannot be displayed where the marked area ends.

Paragraphs (blank lines) will not be shown in selected mark colour when within a marked area.

Tool types-values only allowed to set up as decimal counts: As I mentioned in the source of BareED "atoi" and alike functions of my compiler crashed my machine. Because of this I used the built-in OS function StrToLong() of the DOS library. This function accepts only decimal counts. With the introduction of OS 4.0 it might handle also dual and hexadecimal counts.

Slow deleting / inserting of characters: Currently BareED adds / removes each character instantly. This might be changed with a line-buffer. The problem with a line-buffer is that it has got a rigid size where the line doesn't, hmmm. The next problem is that the internal cursor of BareED which displays the visible cursor does not has "ground beneath its feet" in this special case. So, at the moment I don't like to touch the routines. Added a sub-function written in assembly that will reduce the time needed to move and copy characters.

Pen-selection not very kindly: A plan for a future version of BareED is to use a friendlier interface to choose the pens.

If you press a key and nothing happens BareED cannot obtain more RAM from the system. In this special case it's also not possible to save the file because due to the low available memory the "Asl" file-requester cannot be displayed. Perhaps a future version of BareED gives a warning if you will run out of memory.

The whole user interface of BareED is font-sensitive; it uses the screen's bar-title font which may be is proportional. When now the font is so tall that the window which is computed basing on this font's width and height cannot be opened, BareED does not fallback and uses the topaz-8 font. The next bug is that it does not display this error.

By the way, currently BareED does not display an occurred error caused by other libraries than Amiga-DOS.

System function ObtainBestPenA() inconsequent: First four and last four colours used ever by intuition

screens (multi-colour mode); if ObtainBestPenA() is used existing colours (pen index) not returned even if it is the same as the requested. Thus, I have to implement on my own a better support to share screen pens.

Set pen not cut-down to number of displayable colours. This comes up if you're currently using a screen-depth of e.g. 32 colours and then switch back to (say) 8 colours.

Currently BareED does not support more than 256 pens of a screen, this will change in the (far) future when BareED also offers to choose the colour values for pens.

Created catalogue files with 'CatComp' where the short cut for a menu-item is invalid (or double used) not recognized by BareED - and thus not corrected!

Underscore (short-cut) for gadgets not set and handled even if GadTools 37 is available.

Might trash window border when a font shall be displayed using font-style "italic". This is due to the fact that such a font is bend to the right side, in addition, several fonts draw even pixel out to the left side. Currently, BareED attempts to calculate the needed room left and right of the drawing area once, but it may fail.

Button interface render engine written in C (CHUNKY TO PLANAR) and thus it is slow. The next is that it will ever use eight bit deep biplanes - even the destination area isn't so deep.

BareED is very suspect to faults made by 3rd party applications. This can result in crashes. Be warned!

Non-carefully written system hacks will often lead BareED to fail! Since BareED doesn't trashes memory what has been not demanded by itself, no 3rd party application has the right to trash memory demanded by BareED.

BareED uses the allocated resources in their full size. Trashing even one of such a byte can result in a disaster.

1.9 BareED is able to save icon imagery

BareED has got as default a 4-colour built-in icon image. If you, for example, prefer icon of MagicWB, NewIcon or even GlowIcon type you would be disappointed when BareED would only save a 4-colour icon image to disk. To solve the problem, I implemented a routine which checks first if there is already an icon image on disk, i.e. you save a newer version of the text file. If it is, this one is taken instead of the built-in 4-colour icon image. If there is currently no icon image on disk, BareED checks if in its home directory is a

directory labelled "defs" that contains an icon image which suffix correspond ←
 with the one of the file you are
 going to write to disk. Example:

You want to save the file "BareED.guide" to disk.

In this example BareED will look for an icon in the drawer "defs" ←
 labelled "def_guide.info". The file
 "def_guide.info" is a normal Workbench icon. If BareED finds the file, this ←
 file (icon) is saved along with the
 text file instead of the 4 colour built-in icon of BareED.

You should note that a suffix must not be longer than 6 characters:

```
BareED.guide -> 5, ok
CpyLib.asm   -> 3, ok
Startup.c    -> 1, ok
Kernal.cpp   -> 3, ok
CreateKnobs.script -> 6, ok
Man.postscript -> 10, wrong - here the 4-colour icon would be used!
```

In the above example these icons must be present in the "defs" drawer:

```
def_guide.info
def_asm.info
def_c.info
def_cpp.info
def_script.info
```

From Workbench those files will be viewed as:

```
def_guide
def_asm
def_c
def_cpp
def_script
```

In order to save an icon image to disk the menu item "Create Icons?" must ←
 be turned on. You should pay
 attention that no tool type of an existing icon is stored in the new created one.

1.10 Internals to BareED

You may not imagine how much effort I spent to optimise BareED - not only to make ←
 it faster but also to make
 it as short as possible...

- BareED has been written with the complete renunciation of all 3rd party link ←
 libraries and object files
 - All standard functions like strlen, strcmp, strncmp, strncat and so on have ←
 been re-designed in plain C so
 that any compiler can use relative addressing mode to processor register a4 (←
 small data)
 - A complete new start-up-code, written entirely in plain C (almost)
-

- o gained about 11Kb of code
- o gained about 2700 long relocation entries (~16 Kbytes)
- o gained speed when accessing the operating system, especially Exec, GadTools and Intuition ←
- o auto-detach from CLI/Shell
- o no assign to PROGDIR needed (even when called from the console)
- o 3rd party graphic device compliant (no default PAL/NTSC screen setting)
- o coloured editor window
- o proportionnel font support
- o real tabulators
- o stack check at initialising time
- o Draco computer compliant
- o Enforcer/Mungwall/IO_Torture tested
- o extensive tests already done for OS 3.0, 3.1

Internal limits of BareED:

- Up to 2.1 milliards characters per archive
- Up to 2.1 milliards pixels per line
- Up to 2.1 milliards lines per archive
- Up to 65535 pages per archive
- No screen depth limit (although more than 8 get not really supported through the system (yet)) ←
- Any screen size greater than 640 pixels in width and 200 pixels in height
- Up to 75-pixel height for the font (due to the limit of the visible area; internally BareED can handle fonts ←
- up to 32767 pixel in height)
- Tabulator size up from 1 to endless (although any tabulator width greater than 2 milliards pixels will put ←
- BareED into trouble)
- Up to 16.7 million different colours per pen supported (24 bit, sorry no 48 bit support) ←
- Up to 2.1 milliards clipboard units but currently only one (lonely) block supported ←
- Only one font per text file supported
- Only one font style per text file supported

BareED consists at the moment of 3 files (system header files do not count here):

- startup.c 19799 bytes source 3468 bytes object -GNU-C compliant
- BareED.c 351747 bytes source 115388 bytes object -ditto-
- cpylib.asm 6282 bytes source 432 bytes object

BareED designed using:

- Maxon's MaxonC++ compiler V1 and V4 in C mode
- HiSoft's Devpac Amiga assembler V3 in 68000 mode
- HiSoft's Devpac Amiga debugger V3
- Cloanto's paint program Personal Paint V7
- Martin Apel's ADis disassembler V1
- Author's own graphical converter BitMapSaver V1
- Author's own hunk analyser DropHunk V1

Experimental compile runs under:

- AZTEC-C «« Not tested with BareED source codes higher than version 0.87 .

Version 0.87 okay (rely on warnings sprinkled out all over the source code!)
GNU-C «« large data mode ok., near will cause a lot of internal compiler errors
(spilled register -while setting up RawDoFmt() and several GadTools functions)
VBCC «« Not tested with BareED source codes higher than version 0.72 .
Version 0.72 okay (3rd pass of optimisation could not be used by me due to lack of RAM)

1.11 Copyright and Distribution

The copyright holder is:

Joerg van de Loo
Hoevel 15
47559 Kranenburg
Germany

It is allowed to re-distribute the load file of BareED and the enclosed documentation for free when no one takes explicit money for it. It is not allowed to re-distribute BareED on floppy disks (exception cover disks for Amiga magazines). It's okay to me to spread it through nets and CD-ROMs. The source-code of BareED is free available but only through me and no one has the right to make copies of it and spread it to 'friends'.

If you want to obtain a copy of BareED's source code then send a floppy disk already formatted (either 880 KB or 1.76 MB) and a self addressed envelope with 5 US\$ cash to my address. I will not ship BareED's source code through nets!

As stated in this document already BareED is a beta-release that means that it is not bug-free (and far away from perfect). So, if you encounter a bug which leads to a fatal crash or fault, which again has the result in lost of data or anything else that someone can imagine, I refuse to take any liability. Again, all use is at your own risk. I cannot be held liable for any probable made mistake or lost of something, including profit!