

GED.HYPER

COLLABORATORS

	<i>TITLE :</i> GED.HYPER		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		October 17, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	GED.HYPER	1
1.1	main	1
1.2	FEATURE LIST	2
1.3	LICENCE	4
1.4	INTRODUCTION	5
1.5	REQUIRED SYSTEM	5
1.6	GETTING STARTED	6
1.7	MOUSE HANDLING	7
1.8	DESCRIPTION OF MENUS	8
1.9	PROJECT MENU	8
1.10	project/about	9
1.11	project/user	10
1.12	project/clear text	10
1.13	project/more ed	10
1.14	project/open	11
1.15	project/open fast	12
1.16	project/open new	12
1.17	project/open original	12
1.18	project/insert	12
1.19	project/append	13
1.20	project/new name	13
1.21	project/current dir	13
1.22	project/save	14
1.23	project/save as	14
1.24	project/save as XPK	15
1.25	project/save & exit	15
1.26	project/bits	15
1.27	project/print	16
1.28	project/quit & unload	16
1.29	project/quit (window)	16

1.30	BLOCK MENU	17
1.31	block/mark	18
1.32	block/hide mark	18
1.33	block/cut	19
1.34	block/copy	19
1.35	block/paste	20
1.36	block/paste vertical	20
1.37	block/bcopy	21
1.38	block/bdelete	21
1.39	block/bmove	21
1.40	block/delete column	22
1.41	block/insert column	22
1.42	block/append text	22
1.43	block/column text	23
1.44	block/indent	23
1.45	block/sort	23
1.46	block/uppercase	24
1.47	block/lowercase	24
1.48	block/save as	24
1.49	block/print	24
1.50	LAYOUT MENU	25
1.51	layout/set right margin	26
1.52	layout/use current margin	26
1.53	layout/word wrap on/off	26
1.54	layout/templates on/off	27
1.55	layout/block left/right	27
1.56	layout/block left	27
1.57	layout/block right	28
1.58	layout/block center	28
1.59	layout/section block	28
1.60	layout/section left	29
1.61	layout/section right	29
1.62	layout/section center	29
1.63	layout/AutoCase	30
1.64	layout/right-to-left	30
1.65	FIND MENU	30
1.66	find/find	31
1.67	find/find next	31
1.68	find/find previous	32

1.69 find/replace	32
1.70 find/replace next	32
1.71 find/count	33
1.72 find/check	33
1.73 find/matching bracket	33
1.74 find/reference	33
1.75 find/reference...	34
1.76 find/complete	34
1.77 find/ASCII table	35
1.78 find/ASCII insert	35
1.79 find/show code	35
1.80 find/character set	36
1.81 find/insert code	36
1.82 find/toggle case	37
1.83 find/functions	37
1.84 CONTROL MENU	38
1.85 control/go to line	39
1.86 control/top-bottom	39
1.87 control/to last change	40
1.88 control/store position	40
1.89 control/recall position	40
1.90 control/fold all	40
1.91 control/unfold all	42
1.92 control/insert	42
1.93 control/toggle TAB mode	42
1.94 control/NumPad = movement	42
1.95 control/window arrange	43
1.96 control/window center	43
1.97 control/window zip	43
1.98 control/window enlarge	44
1.99 control/preview	44
1.100control/freeze window	44
1.101control/next window	45
1.102control/previous window	45
1.103control/iconify	45
1.104MISC MENU	45
1.105misc/help	46
1.106misc/source files	47
1.107misc/filter	47

1.108misc/search file	48
1.109misc/line swap	48
1.110misc/line double	48
1.111misc/line pick	48
1.112misc/line push	49
1.113misc/undo line	49
1.114misc/statistics	49
1.115misc/shell	50
1.116misc/calculator	50
1.117misc/HiSpeed	50
1.118misc/files	51
1.119misc/insert date	51
1.120misc/insert time	52
1.121misc/insert path	52
1.122misc/command	52
1.123misc/last error	52
1.124MACRO MENU	53
1.125macros/edit macro	53
1.126macros/run text as macro	53
1.127macros/sequence record	54
1.128macros/sequence load	54
1.129macros/sequence save	55
1.130macros/sequence play	55
1.131macros/play many	55
1.132macros/macros c	56
1.133macros/macros others	56
1.134macros/GUIMake	58
1.135CONFIG MENU	60
1.136config/references	61
1.137config/file hunter	61
1.138config/API	62
1.139config/menus	64
1.140config/mouse	65
1.141config/keyboard	65
1.142Event definition	66
1.143Magic codes	67
1.144config/dictionary	67
1.145config/templates	68
1.146config/indention	69

1.147config/TABs	69
1.148config/display	69
1.149config/GUI	71
1.150config/layout	72
1.151config/printer	73
1.152config/misc	73
1.153config/save	75
1.154config/load	75
1.155User defined gadgets	75
1.156Keyboard	76
1.157Cursor keys	76
1.158HELP key	77
1.159TAB key	78
1.160RETURN key	78
1.161DEL key	79
1.162ESC key	79
1.163F-Keys	80
1.164ARexx port	81
1.165Select a host	82
1.166Lock a window	83
1.167Do your Job	83
1.168Unlock GUI	84
1.169Internal commands	84
1.170Command list	85
1.171API	89
1.172BACK	89
1.173BEEP	89
1.174BIND	90
1.175BITS	90
1.176BLOCK	90
1.177BRACKET	91
1.178CLIP	91
1.179CMD	91
1.180CODE	92
1.181COLON	92
1.182CR	92
1.183DEL	93
1.184DELETE	93
1.185DIR	93

1.186DJUMP	94
1.187DOWN	94
1.188DPAGE	94
1.189ENDWORD	94
1.190EXALL	95
1.191EXTRACT	95
1.192FDOWN	95
1.193FILE	96
1.194FIND	96
1.195FIRST	96
1.196FIX	97
1.197FOLD	97
1.198FORMAT	97
1.199FREEZE	98
1.200FUNC	98
1.201FUP	98
1.202GOTO	99
1.203GREP	99
1.204GUI	99
1.205HELP	100
1.206HUNTER	101
1.207INDENT	101
1.208INFO	101
1.209INSERT	102
1.210KEY	102
1.211LAYOUT	102
1.212LEFT	103
1.213LINES	103
1.214LOCK	103
1.215MACRO	104
1.216MARK	104
1.217MAXDOWN	104
1.218MAXUP	105
1.219MENUS	105
1.220MISC	105
1.221MODE	106
1.222MORE	106
1.223MOUSE	106
1.224NAME	107

1.225NEW	107
1.226NEXT	107
1.227NOTIFY	107
1.228OPEN	108
1.229PATH	108
1.230PHRASE	108
1.231PING	109
1.232PONG	109
1.233POP	110
1.234PREFS	110
1.235PREV	110
1.236PREVEND	110
1.237PRINT	111
1.238PROJECT	111
1.239PUSH	111
1.240QUERY	112
1.241QUIT	114
1.242REFRESH	115
1.243REMAP	115
1.244REPLACE	115
1.245REQUEST	116
1.246RIGHT	116
1.247RUN	116
1.248RX	117
1.249SAVE	117
1.250SCREEN	118
1.251SET	118
1.252SHIFT	118
1.253SMARTCR	119
1.254SUFFIX	119
1.255TAB	119
1.256TABS	120
1.257TASK	120
1.258TEXT	120
1.259TMPLATE	121
1.260UJUMP	121
1.261UNDO	121
1.262UNLOCK	122
1.263UP	122

1.264UPAGE	122
1.265USE	122
1.266VIEW	123
1.267VLEFT	123
1.268VRIGHT	124
1.269WINDOW	124
1.270XREF	124
1.271Input events	125
1.272GENERAL HINTS	126
1.273CREDITS	127
1.274HOW TO REGISTER	127
1.275Registration site Germany	128
1.276Registration site Belgium	129
1.277Registration site France	130
1.278HOW TO GET UPDATES	130
1.279HOW TO CONTACT AUTHOR	131
1.280 GoldED	131

Chapter 1

GED.HYPER

1.1 main

Feature list

ARexx port

Licence

Command list

Introduction

General hints

Required system

Credits

Getting started

How to register

Mouse handling

How to get updates

Description of menus

How to contact author

Keyboard

Index

 Menus

Project menu

Control menu

Block menu

Misc menu

Layout menu

Macro menu

Find menu

Config menu

Suggested reading pattern: Top to Bottom. Novice users should skip the sections "ARexx port" and "Command list".

1.2 FEATURE LIST

FEATURE LIST

- o OS3.x look & functions (
 - AppWindows
 - & more)
- o
 - Folding
 - of paragraphes (unlimited nesting available)
- o Menus fully customizable (easy-to-use requesters)
- o
 - ARexx port
 - (approx. 420 commands/options)
- o
 - QuickReference
 - capability (includes, sources, ...)
- o
 - APC
 - (TM): Automatic phrase completion, based on dictionary
- o
 - shifting
 - (two speeds)
- o Smooth display, fast scrolling
- o
 - HotKey
 - support
- o unlimited number of windows
- o
 - Formatter
 - (aligned, block, centered),
 - WordWrap
 - o automatic indention (left margin)
- o
 - SmartIndention
 - (code dependend, e.g. after 'if')
- o

- o AutoBackup
(any interval, any backup path)
 - o MenuHelp
(AmigaGuide)
 - o localized (English/German so far)
 - o right-to-left
input mode available
 - o printer control (linefeed/spacing/style)
 - o Multiselect
of files (e.g. join files together)
 - o direkt
XPK support
: (de)crunching
 - o ASCII character selection window
 - o flexible GUI handling: any
display mode
/
font
 - o protection bits support, file comment support
 - o clipboard
support (snapping): Cut & paste
 - o AutoLoad
of project file
 - o FastLoad
mode
 - o AutoCase
(TM) correction (based on user's dictionary)
 - o automatic
parenthesis check
available
 - o QuickFunc
jump table display for many prog. languages
 - o insertion of columns
,
removal of columns
 - o user friendly (about 25 requesters)
 - o fixed/regular/
dynamic TABs
; solid/tranparent TABs
 - o character set remap
(e.g. Amiga to MS-DOS)
 - o four
speeds of scrolling
 - o quick starter ED
 - o startup macro
 - o powerful
-

- macro recording
 - o
- templates
 - o
- global search
- across file boundaries
- o
 - file hunter
 - (extract file name under cursor, search file)
- o online
 - spellchecker
 - o
 - user defined gadgets
 - o
 - search/replace history
 - o asynchronous printing
- o icon
 - dock
 - o
 - preview
 - display mode
- o and many more ...

1.3 LICENCE

LICENCE

This licence agreement affects all programs, manuals and disks of the editor package GoldED, release 0.99 or later, except files within the support directory. Violations of any of the topics below will cause this licence to expire, i.e. terminate your rights to use or otherwise handle (e.g. distribute) the package.

You may duplicate/distribute the GoldED package EXCEPT THE KEYFILE DRAWER as long as you don't charge money for it apart from compensation for media expenditures. Keyfiles are ©1994 Dietmar Eilert. You may NOT distribute keyfiles or keyfile-related data (i.e. the keyfile drawer). Distribution of keyfiles is a violation of copyright laws and will be prosecuted by the copyright owner(s).

You are not allowed to include the demo version into any kind of software collection apart from the Amiga Library (compiled by Fred Fish) and BBS file areas without written permission of the author, including, but not limited to, CD ROMs and disk magazines. The demo version may not be distributed as part of a bundle (e.g. bookware, disk magazine).

The copyright owner(s) reserve the right to interdict distribution at any time if the distributor fails to obey to this licence from the copyright owner(s) point of view.

You may not modify the package: you may not change the compilation, you may not modify or translate files and you may not add or remove files. All

translation rights reserved.

The copyright owner(s) provide this program "as is". The entire risk of using this program is with you. The copyright owner(s) won't ever be liable for damages to you - whether they arise from the use of this package or the inability to use it, whether they are direct or consequential, including, but not limited to, the loss of data or the generation of inaccurate data.

1.4 INTRODUCTION

INTRODUCTION

The basic design goals of this editor were to make it as user friendly as possible. And make it as fast as possible. A lot of attention has been put to the general look & feel, performance considerations and full support of the Amiga OS. The whole design is based on OS2.x/3.x, a fast CPU, sufficient RAM and a hard disk. The makers of GoldED neither cared much about memory consumption nor did they pay attention to OS1.3 compatibility since this would have meant to make concessions to speed or general handling.

GoldED's way of working is basically event based: the editor waits for events like a pressed key or a menu selection and then calls a dispatcher to perform appropriate action. Action is not hard coded: you may assign any of the editor's internal functions to an event (though most people won't do any assignments at all but use the default configuration). For example you may assign the 'open file' function to the 'A' key by using GoldED's keyboard requester. Or assign the text "don't panic" to the A key. Or assign an ARexx macro script to it. Or a DOS command. Or just leave it as it is. No matter what kind of interface to GoldED you use (the menus, the keyboard or its ARexx port): all of them do support the same command set. It's easy to use, straight-forward and flexible. GoldED doesn't use an internal macro language like DME: Its internal functions are basically function calls, performing some kind of action. It fully relies on ARexx as far as conditional statements like IF ... THEN are concerned.

1.5 REQUIRED SYSTEM

REQUIRED SYSTEM

Minimum requirements are OS2.04, 68000 and 1 MB RAM. GoldED has NOT been tested with a 68000. Suggested minimum system for average performance (I'm serious about that) is OS3.0, 68020 & 2 MB RAM. Some advanced features require OS2.1, many other features even require OS3.x. We strongly recommend to install/rekick OS3.x if your Amiga is capable of it. GoldED is *much* faster under OS3.x.

Software requirements (all these files have been released to the public already - check Fish disks):

- o reqtools library

- o xpk library
- o amigaguide library

1.6 GETTING STARTED

GETTING STARTED

Start the editor either by doubleclicking its icon or by typing its name at shell level. Or use the HotKey combination (i.e. right ALT + right SHIFT & RETURN) if the editor is present in the background already. Depending on whether or not AutoLoad (see config/misc) is enabled, it might happen that a file is loaded automatically even though you have provided no file name. Don't get confused - it's a feature, not a bug :-)

QuickStarter

Besides the main editor GoldED, a quick starter ED is provided. This program is a small (4 KB) GoldED frontend. You can use it AS IF it were a real editor. For example you might type 'ed letter' to edit the file letter if the quick starter were named 'ed'. The big advantage of a quick starter is its ability to pass a new job to an already running instance of GoldED (if there is any). Needless to say that this happens very fast. The quick starter may be made resident (the main editor may not). The source code of ED is available in the "GoldED:Tools/EDSource" drawer.

Arguments

GoldED accepts four kinds of arguments: a list of file names to load, the name of a public screen to open on (after the SCREEN keyword), a configuration file to use (CONFIG keyword) and finally the HIDE option. The SCREEN/CONFIG/HIDE arguments may be passed as command line options as well as icon tool types (don't use quotes around file names within tool type entries). Example:

```
GoldED s:startup-sequence CONFIG s:MyPrefs
```

If you want the editor to stay in the background initially (waiting for

```
HotKey
activation), use the HIDE option but don't supply file names:
```

```
GoldED HIDE
```

The quickstarter ED additionally supports the STICKY option (see below). It will ignore the CONFIG/SCREEN options if it is able to pass the list of files to a running instance of GoldED. Example:

```
ED mail:answer CONFIG s:BBS.prefs STICKY
```

The quickstarter has been designed for synchronous operation (unless the HIDE

option is used to run the editor in the background). It won't detach itself from a shell window unless you RUN it. If the STICKY option is not specified, a call to ED returns after the last GoldED window has been closed. If STICKY is specified, a call to ED returns after the window opened by ED is closed (GoldED itself may keep on running if there are further windows to handle). This is designed to be used in command files where you want the execution of the batch file or command script to wait until the user has finished editing a particular file. ED will return (using the sticky option) when the user quits out of the file.

MenuHelp

The editor's help facility is based on Commodore's AmigaGuide library. Simply doubleclick the 'manual' icon - its default tool is AmigaGuide. You might use 'MultiView' of OS3.0 to read the 'GoldED.guide' file, too. GoldED itself supports menu help: press the <HELP> key during selection of a menu item (within GoldED) to get explanations related to the item you selected.

Language

Locale library of OS2.1 or better is supported as far as GoldED's requesters are concerned: if you have selected German to be your default language (by using the 'locale' preferences of your Workbench), requesters as well as time/date strings will be German. So far only English and German are available; other languages might be available in the future. Locale settings do not affect menus because menus are not part of the editor itself but external text files. Use

config/menus

to edit/load menus. The default menu

definition file is available in German & English. It is installed by the Install utility when installing the GoldED package. You may load a new menu definition file at any time (see

config/menus

).

GUI (Graphical User Interface)

The editor's user interface supports the look & feel of OS2.x. Most gadgets offer keyboard activation: if a character of the gadget inscription appears underscored, it may be used as shortcut. Thus you can move a slider gadget or activate a button gadget by a single keystroke. Use the shift key simultaneously to toggle 'direction' of changes - e.g. to move a slider gadget one step to the left instead of to the right. Use the 'Amiga' key simultaneously to activate another gadget from within a string gadget (without R-Amiga your input would go to the string gadget). Amiga-X will clear string gadgets. CTRL+L inserts a form feed code.

1.7 MOUSE HANDLING

MOUSE HANDLING

A simple mouse click into a text window will position the cursor. Doubleclick into the window to mark the word under the cursor. Hold down the mouse button

and drag the mouse pointer over the text to mark lines or single words. This editor offers two marker resolutions: character resolution (standard) and line resolution. Hold down the SHIFT key while marking to change to line mode.

1.8 DESCRIPTION OF MENUS

DESCRIPTION OF MENUS

Since GED offers almost unlimited user configuration, there is no 'fixed' appearance, neither as far as colors, resolution or fonts nor as far as menus & functions are concerned. This manual describes the default configuration. Use

```
config/load
to load a settings file.
```

The standard menus are:

```
Project menu
Control menu
Block menu
Misc menu
Layout menu
Macro menu
Find menu
Config menu
```

1.9 PROJECT MENU

PROJECT MENU

menu tree of project menu

```
project/about
project/new name
project/user
project/current dir
```

```

project/clear text
project/save
project/more ed
project/save as
project/open
project/save as XPK
project/open fast
project/save & exit
project/open new
project/bits
project/open original
project/print
project/insert
project/quit & unload
project/append
project/quit (window)

```

The project menu offers a variety of commands related to general handling of a document, especially as far as basic input/output functions are concerned (load/save or printing). ↔

1.10 project/about

```

project/about of
PROJECT MENU
Displays version ID. Furthermore the name of this task's ARexx port and this

```

task's screen name are displayed. If you told the editor to open a custom screen, this screen is always public, i.e. you are invited to run other tasks on the same screen (e.g. type SHELL CON:0/11/640/100/Shell/screenGOLDED.1 to open a shell window on GoldED's screen). See

```

config/display
for information

```

on how to make other programs appear on GoldED's screen.

1.11 project/user

project/user of:

PROJECT MENU

If you are a registered user of GoldED, your name appears if \leftrightarrow
select this

menu item. The unregistered release shows a general copyright information.

1.12 project/clear text

project/clear text of:

PROJECT MENU

Clears contents of current window. Protection bits are set to \leftrightarrow
the defaults

(see

config/misc

) before further actions take place. The path is reset to

the current path, the file name is reset to "unnamed". You are asked for confirmation if the text has been changed since loading.

1.13 project/more ed

project/more ed of:

PROJECT MENU

Opens a further window for input. The window size is read \leftrightarrow
from the

configuration file (see

config/load

). If you have enabled

CenterWin

(see

config/misc

, default is ON), the new window is centered on screen, otherwise the window position is read from the configuration file. Use

config/save

to save window dimensions & position of your current window as default settings. This editor offers many functions related to window management, e.g. it is able to 'arrange' your windows on screens. Autoscroll screens are fully supported: window functions consider the visible section only. See

control/window arrange

for more details.

1.14 project/open

project/open of:

PROJECT MENU

Discards current text if any was loaded (same as

project/clear text

) and

ask for new file(s) to load.

Multiselect

Like most file requesters of GoldED, the requester used by this function supports multi selection: you may select more than one file. Treatment of multi selected files depends on the context: while this function will open a new window for each file,

project/append

loads all selected files to one

window.

AppWindows

GoldED's text windows are so called AppWindows: it is possible to drag icons (of text files) over a text window. These files are appended to the text of that window. Multi selection is supported: You may drag more than one file over a window using extended selection (hold down the shift key while you select icons).

Crunched XPK files (see

project/save as XPK

) are recognized and

decompressed while reading if the

XPK

libraries is available. This function

checks for TABs (dez. 8); TABs are replaced by SPC (dez. 32) while loading.

See

config/tabs

if you want to influence substitution. LOAD uses an input

buffer of about 16KB to speed up operation. However, if you want even better performance, use

project/open fast

instead.

If AutoFold (

config/misc

) is ON, the file is scanned for folded sections

after loading. You should disable AutoFold if you don't want to use the folding capability to prevent scanning for fold markers and thus save time.

Warning: Do never attempt to edit binaries (programs). GoldED is a text editor, not a file monitor. It will change the data in a way suitable for text files but definitely unsuitable for binaries (e.g. remove CR return codes, substitute spaces, clear the executable-bit).

1.15 project/open fast

```
project/open fast of:
PROJECT MENU
FastLoad
```

A fast replacement for

```
project/open
```

. This option requires an IO buffer of

the original file's size and it doesn't check for TABs. However, since GoldED never writes TABs (a concession to speed), this function might suit you very well if your Amiga has a decent amount of RAM. This function is about 50% faster than 'slow' load. Use

```
project/open original
to reload a file in slow
```

mode (with TAB substitution enabled) if you discover after loading that it contains TAB's ('strange' characters at beginning of lines). GoldED automatically falls back to slow load if it detects TAB codes within the first 1500 bytes of a file.

1.16 project/open new

```
project/open new of:
PROJECT MENU
Load a document but don't load it to the current window as
project/open
would do. Instead, a new window is prepared.
```

1.17 project/open original

```
project/open original of:
PROJECT MENU
Reload current file from disk. Useful after you have made some ↔
changes but
want to switch back to the original. This function uses 'slow loading' (see
project/open fast
), i.e. TAB's are substituted by spaces.
```

1.18 project/insert

```
project/insert of:
PROJECT MENU
```

Insert a file at current cursor position (before current line). A file requester will pop up, asking you for one or more files to be inserted. Hold down the SHIFT key to select more than one file (this feature is called 'multiselection').

1.19 project/append

project/append of:

PROJECT MENU

Append one or more files to your current text. A file requester will pop up, asking you for one or more files to append to your text. Quite useful to join a couple of files together.

1.20 project/new name

project/new name

Change the name of current text (you are prompted for a new one). Only the document in memory is affected - no disk file is renamed. Since one usually would use

PROJECT/SAVE AS

to save a file to a new location, this function is rarely used.

1.21 project/current dir

project/current dir of:

PROJECT MENU

Set the 'current path' to wherever you want. The current path is used by many

functions (e.g.

project/open new

) as default path. It is used by all menu

items of type DOS, too (e.g. 'new shell'). If you have set the current path to 's:', a function like

project/open new

would list the s-directory when

asking for a file.

Setting the current path doesn't change the name of an existing text - use the menu command

project/new name

to change the name. Some functions (e.g.

project/open

) don't care about the current path - they extract path information from the current document's name. Use

project/clear text

to

reset the path of an empty text window to the current directory.

1.22 project/save

project/save of:

PROJECT MENU

Save a text, using the current name displayed in the window's ↔

title bar. Old

copies (no matter how protection bits have been set) are overwritten without asking for confirmation if 'overwrite' (

config/GUI

) mode has been set. Turn

on backup creation if you want the editor to backup the old version if one is available; see

config/misc

for more information on backups. You might even

ask the editor to backup your files regularly (e.g. every 10 minutes) to any directory (see

config/misc

as well). This function is disabled for windows

of type read-only to prevent the user from accidentally overwriting important files. QuickRef windows are read-only (see

QuickReference

).

1.23 project/save as

project/save as of:

PROJECT MENU

Same as

project/save

but gives you the opportunity to enter a new file name

before the text is written to disk. This function is disabled for windows of type read-only to prevent the user from accidentally overwriting important files. QuickRef windows are read-only (see

QuickReference

).

1.24 project/save as XPK

project/save as XPK of:
 PROJECT MENU
 XPK support

Save current file in compressed and/or encrypted mode (about 50% less disk space required depending on chosen compressor). See

config/misc
 on how to

select compression mode. This function requires the complete set of XPK libraries which has been released as FD. Note: other editors/programs might not be able to handle crunched files, so be careful when using this function. Don't crunch your sources - the compiler won't like it.

1.25 project/save & exit

project/save & exit of:
 PROJECT MENU
 Same as
 project/save
 followed by
 project/quit (window)
 : Save current text

and close window. Exit GoldED if the last window has been closed. GoldED won't close a window or exit if the SAVE operation fails (e.g. disk full error). This function is disabled for windows of type read-only to prevent the user from accidentally overwriting important files. QuickRef windows are read-only (see

QuickReference
).

1.26 project/bits

project/bits of:
 PROJECT MENU
 Edit protection bits of the current text. Have a look at ↔
 your Amiga DOS

manual if you are unfamiliar with these bits. In generally you should set the S (script) bit for batch files but let the other bits untouched. Changes won't have any effect until you save the text. These bits are set to a default state after performing

project/clear text
). Use

config/misc
 to

define the default state.

1.27 project/print

project/print of:
 PROJECT MENU
 Send current text to printer 'PRT:'. See
 config/printer
 for information on

how to affect output style. DeskJet/LaserJet owners should use the HiSpeed printing facility of the

misc
 menu instead (
 misc/hispeed
). HiSpeed is

shareware; a registered HiSpeed release is part of the GoldED Pro and GoldED Pro/NET registration.

1.28 project/quit & unload

project/quit & unload of:
 PROJECT MENU

Close current window. Exit from GoldED if the last window has been closed. ↔

You are asked for confirmation if you attempt to exit without having saved your text so far (unless you didn't change the text at all). You are NOT asked for confirmation if only preferences have been changed (see

config/save
). GoldED is unloaded from memory by this menu after the last window has been closed (the editor is unloaded even if the 'resident' option is enabled; see

config/misc
). Use
 project/quit (window)
 if you want to

have the resident setting considered.

1.29 project/quit (window)

project/quit (window) of:
 PROJECT MENU

Close current window. Exit from GoldED if the last window has been closed. ↔

You are asked for confirmation if you attempt to exit without having saved your text so far (unless you didn't change the text at all). You are NOT

asked for confirmation if only preferences have been changed (see
config/save
).

GoldED's memory management is asynchronous - after having closed a window you don't have to wait for the memory to be freed (this is done by a background task). Your Amiga might appear to be slightly slower than usual while the background task is busy - especially if the text buffer has been large (200 KB or more).

HotKey

If hotkey support (see
config/misc
) is enabled, GoldED will not be removed from RAM even after closing the last window. Instead it will wait for a hotkey combination (right ALT & right SHIFT & RETURN). Press these keys to make the editor reappear. Or use the commodities exchange program of your workbench. HotKey activation will give you a very fast response time since the editor won't have to be reloaded from disk.

1.30 BLOCK MENU

BLOCK MENU

menu tree of block menu

- block/mark
- block/insert column
- block/hide mark
- block/append text
- block/cut
- block/column text
- block/copy
- block/indent
- block/paste
- block/sort
- block/paste vertical
- block/uppercase

block/bcopy
 block/lowercase
 block/bdelete
 block/save as
 block/bmove
 block/print

block/delete column

All functions of the block menu are related to the management of ↔
 'blocks',

which are marked sections of lines. Usually one would use menu functions or corresponding keyboard shortcuts to mark lines, however the mouse may be used as well: simply hold down the left mouse button while you drag the pointer over a desired section of lines. You can have only one block per document.

1.31 block/mark

block/mark of:
 BLOCK MENU

Mark beginning or end of a block. This command will mark whole ↔
 lines only;

use the

mouse

if you want to mark single words or characters. If you call this function for the first time (no marked lines so far), the editor will remember the current cursor position as START of a new block. If you call this function a second time, the editor remembers the current cursor position as block END - all lines between START and END are highlighted. START and END are symbolic names. START doesn't necessarily have to be smaller than END. Things are handled differently if you use this function after a block has already been selected: If the cursor position is closer to the end of the current block than to its start, the end position is updated. Otherwise the start position is set to the cursor's line. You might use

block/hide mark

to get rid of a block, i.e. to unmark lines.

Some functions of this editor require line resolution as far as blocks are concerned - for example you can't mark a single word and block-format it. GoldED will promote blocks to paragraphs if required.

1.32 block/hide mark

block/hide mark of:

BLOCK MENU

Turn block off, ie. don't have any text section highlighted. ↔

Useful after

block/mark

to get rid of a block selection.

1.33 block/cut

block/cut of:

BLOCK MENU

Cut selected parts (see

block/mark

or

mouse

) from text. These lines are

written to the clipboard, ready to be inserted into any application supporting the clipboard device (e.g. into a shell window by pressing AMIGA & V or into any GoldED document using

block/paste

). The clipboard offers

several storages called "units" and thus is able to keep many blocks of data; standard unit (used to exchange data between applications) is unit 0. You shouldn't use these clipboard-based functions to move/copy data within a single GoldED document;

block/bcopy

or

block/bmove

would perform the same

task much faster.

1.34 block/copy

block/copy of:

BLOCK MENU

Copy marked parts (see

block/mark

or

mouse

) to the clipboard device,

ready to be inserted into any application supporting this device. The clipboard offers several storages called "units" and thus is able to keep many blocks of data; standard unit (used to exchange data between applications) is unit 0. This function doesn't affect your current text (as

block/cut

would). See
 block/paste
 for more details.

1.35 block/paste

block/paste of:

BLOCK MENU

Insert clipboard contents (if any are available) into current text ←

. An empty

clipboard is reported as "clipboard error". Single words found in the clipboard are inserted at the current cursor position. Paragraphes (i.e. multiple lines) found in the clipboard are treated differently: they are inserted before the current line.

Clipboard

The 'clipboard' (actually the 'CLIPS:'-directory) is used by many applications to exchange data. It offers several storages called "units" and thus is able to keep several blocks of data simultaneously. GoldED can access any unit (see

CLIP

command), though usually only unit 0 is used. Postings

to the clipboard are IFF files; GoldED supports IFF/FTXT clipboard access. For example, you could mark a text within a shell window, press AMIGA + C to copy these lines to the clipboard unit 0 and then reinsert them into a GoldED window using AMIGA + V. A paste operation won't remove the data from the clipboard, i.e. you can call this function several times.

1.36 block/paste vertical

block/paste vertical of:

BLOCK MENU

Insert

clipboard

contents "vertically" at current cursor position: the

clipboard contents are mixed to the existing lines. Depending on the writing mode (

control/insert

) text is either inserted or the current text

overwritten by this operation. While insert mode is recommended for pasting single words, overwrite mode should be used to create multi-column documents.

You will get a 'clipboard error' if the clipboard is empty. Use

block/copy

to move text sections to the clipboard.

1.37 block/bcopy

block/bcopy of:
 BLOCK MENU
 Copy marked section (see
 block/mark
 or
 mouse
) to current cursor position.

Useful to duplicate sections within a single document (while you would have to use a

block/cut
 /
 block/paste
 pair to exchange data between different

windows or different applications).

1.38 block/bdelete

block/bdelete of:
 BLOCK MENU
 Delete highlighted section (see
 block/mark
 or
 mouse
). It is not possible

to undo this operations, so better be careful. If you don't want to miss an undo option, use

block/cut
 instead (
 block/paste
 to undo) - however those

functions are no as fast as this one.

1.39 block/bmove

block/bmove of:
 BLOCK MENU
 Move highlighted area (see
 block/mark
 or
 mouse
) to current cursor

position. This function is useful to move sections of lines within a single document (while you would have to use a

block/cut
 /
 block/paste

pair to
 move text from one window to another window/application).

1.40 block/delete column

block/delete column of:
 BLOCK MENU
 Removal of columns

Delete a column from highlighted lines (see
 block/mark
). Move cursor to

desired column before you call this function (e.g. move cursor to column 10
 if you want to delete this column from all block line). You shouldn't use
 this function if you simply want to change indentation of a paragraphe because;
 use

block/indent
 instead.

1.41 block/insert column

block/insert column of:
 BLOCK MENU
 Insertion of columns

Insert a column into highlighted lines (see
 block/mark
 on how to mark

lines). Move cursor to desired column before you call this function. Example:
 Move cursor to column 40 if you want to insert one space character before
 this column into all marked lines.

1.42 block/append text

block/append text of:
 BLOCK MENU

Append text to marked lines. Example usage: Mark some ↵
 lines (see

block/mark

) and call this function. A requester will pop up, asking you for
 a text to be appended. If you enter ';', a semicolon would be appended to
 each line you have marked.

1.43 block/column text

block/column text of:

BLOCK MENU

Insert a text into marked lines at current cursor position. ←

Example usage:

Mark some lines (see

block/mark

), move cursor to desired column (e.g.

column 1) and call this function. A requester will pop up, asking you for a text to be inserted. If you enter 'Prototype ', this word would be inserted at the beginning (column 1) of each block line. This function is quite useful if you want to create tables (for example you could insert a '|' to get a vertical line).

1.44 block/indent

block/indent of:

BLOCK MENU

Shifting

Change indentation of marked lines (

block/mark

). Use arrow gadgets to shift

text left or right. Currently selected keyboard-TAB distance (see

config/tabs

) is used as default indentation step, however you may change this value using the step gadget. An indent function (two speeds) is assigned to keyboard's cursor keys, too (see

cursor keys

).

1.45 block/sort

block/sort of:

BLOCK MENU

Sort selected lines (see

block/mark

) alphabetically. This function is not

case sensitive (i.e. 'A' and 'a' would be considered equal).

1.46 block/uppercase

block/uppercase of:

BLOCK MENU

Make all characters of highlighted lines (see
block/mark

) uppercase. This

function uses the locale library if available to treat non-ASCII characters (e.g. 'ß') the right way. Locale library is part of OS2.1 and OS3.0 or better. It is not part of OS2.04.

1.47 block/lowercase

block/lowercase of:

BLOCK MENU

Make all characters of highlighted lines (see
block/mark

) lowercase. This

function uses the locale library if available to treat non-ASCII characters (e.g. 'ß') the right way. Locale library is part of OS2.1 and OS3.0 or better. It is not part of OS2.04.

1.48 block/save as

block/save as of:

BLOCK MENU

Save marked lines (see
block/mark

) to disk. You will be asked for a file

name. Do not use this function to move text from one text to another - use the clipboard instead (see

block/copy

).

1.49 block/print

block/print of:

BLOCK MENU

Print marked lines (see
block/mark

). The standard preferences printer is

used. See

config/printer

if you want to affect output settings like quality
or linefeed.

1.50 LAYOUT MENU

LAYOUT MENU

menu tree of layout menu

```

layout/set right margin
layout/block center
layout/use current margin
layout/section block
layout/word wrap on/off
layout/section left
layout/templates on/off
layout/section right
layout/block left/right
layout/section center
layout/block left
layout/AutoCase
layout/block right
layout/right-to-left
Formatter

```

All functions of the layout menu are related to formatting a text. They are of no use when writing source code - which is the main purpose of GoldED. But they should turn out useful if you edit normal text files (e.g. your e-mail).

Paragraphe vs. block

Some of the formatting functions actually are block functions: they do affect marked lines only. See

```
block/mark
```

if you don't know how to mark lines.

Others do affect the 'current paragraphe' of a text. For example you might move the cursor to this star '*' and then choose

```
layout/section left
```

: the
 lines from 'Some ...' (beginning of paragraphe) to the end of this paragraphe would be made left aligned. The editor determines the end of a paragraphe by looking for an empty line (however some lines appear to be empty but actually contain multiple spaces - these lines are not considered empty).

1.51 layout/set right margin

layout/set right margin of:
 LAYOUT MENU
 Set righth margin to current cursor position (margin values are ←
 considered by
 formatting operations like
 WordWrap
). Example usage: move cursor to column
 80, then call this menu. This will set the right margin to column 80.

1.52 layout/use current margin

layout/use current margin of:
 LAYOUT MENU
 Formatting functions ignore left border settings (
 config/layout
) if this
 menu appears checked; the left margin of the current line is used instead.

1.53 layout/word wrap on/off

layout/word wrap on/off of:
 LAYOUT MENU
 WordWrap

Toggles word wrap mode on/off. Current status is displayed in screen's title bar (WRAP). If word wrap is on, the editor will reformat the current paragraphe (left aligned) if cursor moves behind right margin (see

config/layout
 on how to set right margin). Word wrap is very useful if you work on a plain ASCII text: You won't have to bother about pressing the enter key - the editor will switch to a new line automatically if the current lines is full.

Do never use word wrap if you work on a source file - the source would get mixed up as soon as the editor attempts to reformat a section of lines. Do

not use word wrap when creating tables or any other kind of formatted output for the same reason, too.

1.54 layout/templates on/off

```
layout/templates on/off of:
  LAYOUT MENU
  Templates
```

Toggles template mode on/off. Current status is displayed in screen's title bar (TMPL). If template mode is on, the editor will look for search patterns (templates) during user input. If a template is found, template-specific action as set up by the user is performed. Using templates you could make the editor replace "SNC" by "sincerely" immediately while you are typing. Several action types are available, including playback of recorded sequences

```
,
```

```
internal commands
or ececution of ARexx macros. Template setup is described
```

in the

```
Config/Templates
section.
```

1.55 layout/block left/right

```
layout/block left/right of:
  LAYOUT MENU
  Reformats marked lines (see
  block/mark
  ) to make them appear left & right
```

aligned. The block's last line is made left aligned. Empty lines are not removed during formatting, i.e. your document's basic structure remains unchanged. Multiple spaces are removed, so better do not attempt to format tables. Use

```
config/layout
to set block width and the left border.
```

1.56 layout/block left

```
layout/block left of:
  LAYOUT MENU
  Reformats marked lines (see
  block/mark
```

) to make them appear left aligned.

Empty lines are not removed during formatting, i.e. your document's basic structure remains unchanged. Multiple SPC's are removed, so better do not attempt to format tables. Use

```
config/layout
to set maximum line width
```

and the left border.

1.57 layout/block right

layout/block right of:

```
LAYOUT MENU
```

Reformats marked lines to make them appear left aligned. Empty ↔
lines are not

removed during formatting, i.e. your document's basic structure of paragraphes remains unchanged. Multiple SPC's are removed, so better do not attempt to format tables. Use

```
config/layout
to set maximum line width
```

and the left border.

1.58 layout/block center

layout/block center of:

```
LAYOUT MENU
```

Reformats marked lines to make them appear centered within ↔
currently selected

layout area. Use

```
config/layout
to set the layout area width and the left
```

margin. Empty lines are not removed during formatting, i.e. your document's basic structure of paragraphes remains unchanged.

1.59 layout/section block

layout/section block of:

```
LAYOUT MENU
```

Reformats current paragraphe (see
Paragraphe vs. block
) to make the lines

appear left & right aligned. The last line of a paragraphe is not affected -it is made left aligned. A line is a 'last' line if its successor is an empty line (however some lines appear to be empty but actually contain spaces -these lines are NOT considered empty). Empty lines are not removed during formatting, i.e. your document's basic structure of paragraphes remains

unchanged. Multiple SPC's are removed, so better do not attempt to format tables. Use

```
config/layout
to set block width or the left border.
```

1.60 layout/section left

```
layout/section left of:
LAYOUT MENU
Reformats current paragraphe (see
Paragraphe vs. block
) to make the lines
```

appear left aligned. Empty lines are not removed during formatting, i.e. your document's basic structure remains unchanged. Multiple SPC's are removed. Use

```
config/layout
to set layout width and left border.
```

1.61 layout/section right

```
layout/section right of:
LAYOUT MENU
Reformats current paragraphe (see
Paragraphe vs. block
) to make the lines
```

appear right aligned. Empty lines are not removed during formatting, i.e. your document's basic structure remains unchanged. Multiple SPC's are removed. Use

```
config/layout
to set layout width respectively left border.
```

1.62 layout/section center

```
layout/section center of:
LAYOUT MENU
Reformats current paragraphe (see
Paragraphe vs. block
) to make the lines
```

appear centered within currently selected layout area. Layout width and left margin are set by

```
config/layout
. Empty lines are not removed during
```

formatting, your document's basic structure remains unchanged. Multiple SPC's are removed.

1.63 layout/AutoCase

layout/AutoCase of:
 LAYOUT MENU
 Toggle
 AutoCase
 mode. AutoCase correction is based on the user dictionary;

see

config/dictionary

.

1.64 layout/right-to-left

layout/right-to-left of:
 LAYOUT MENU
 Toggles right-to-left input mode: user input will appear from ↔
 right to left in

reversed mode. Reversed mode will help you to process files written in languages like Hebrew. Backspace, Delete and Return change their behaviour in reversed mode as well - for example the Return key would move the cursor to the "last" column instead of the first column. Use

config/layout
 (right border) to set

the last column.

1.65 FIND MENU

FIND MENU

menu tree of find menu

find/find

find/reference...

find/find next

find/complete

find/find previous

find/ASCII table

find/replace
find/ASCII insert
find/replace next
find/show code
find/count
find/character set
find/check
find/insert code
find/matching bracket
find/toggle case
find/reference
find/functions

1.66 find/find

find/find of:
FIND MENU

Shows a requester to enter the text to search for. You may choose \leftrightarrow the search

to be case sensitive or not by using the appropriate checkmark gadget. Use the OK gadget to go to the next (i.e. after current cursor position) occurrence of the text to search for. Use the FIRST gadget to look for the first occurrence of the search pattern. Note: case-sensitive search is much faster than case-insensitive search.

Search/replace history

Use the arrow gadget to open a search/replace history listview. The listview will offer some of the previously used search/replace strings as well as the word under the cursor for fast selection.

1.67 find/find next

find/find next of:
FIND MENU

Go to the next occurrence (i.e. after current cursor position) of \leftrightarrow the pattern

to search for. Use
 find/find
 to enter search text.

1.68 find/find previous

find/find previous of:
 FIND MENU
 Go to the previous occurrence (i.e. before current cursor ←
 position) of the
 pattern to search for. Use
 find/find
 to enter the search text.

1.69 find/replace

find/replace of:
 FIND MENU
 Shows a requester to enter both, the text to search for ←
 as well as a
 replacement. You may choose the operation to be case sensitive or not by
 using the appropriate checkmark gadget. Select the NEXT gadget to replace the
 next occurrence of the search text by the replace text. Use the ALL gadget to
 replace all occurrences of the search text. Decide for the BLOCK gadget if
 you want the replacement operation to be restricted to marked lines (see

 block/mark
).

Use the arrow gadget to open a search/replace history listview. The listview
 will offer some of the previously used search/replace strings (if any) as
 well as the word under the cursor for fast selection.

1.70 find/replace next

find/replace next of:
 FIND MENU
 Replaces next occurrence of search pattern by replace text. ←
 Replacement as
 well as the search pattern are set using the
 find/replace
 requester.

1.71 find/count

find/count of:

FIND MENU

Shows a requester to enter a search text. You may choose the search operation ↔

to be case sensitive or not by using the appropriate checkmark gadget. Finally select the COUNT gadget to count all occurrences of the search pattern within your text.

1.72 find/check

find/check of:

FIND MENU

Checks for correct use of braces '(' within current line. You will be warned ↔

if there are more opening braces than closing ones or vice versa. Nesting is checked, too. You may turn on automatic checking after each line using

config/dictionary

.

1.73 find/matching bracket

find/matching bracket of:

FIND MENU

Move cursor to matching bracket. Handles (<>)... depending on character under ↔

cursor. Useful to check levels of execution within a program (move cursor over first opening parenthesis within a C-function, then use this function).

1.74 find/reference

find/reference of:

FIND MENU

QuickReference

Does try to open a help text related to the word your cursor is placed over. Example usage: Type 'struct RastPort', move cursor over 'RastPort', then call this function. If the reference system is set up (see below), a new text window should pop up, showing you a file 'graphics.h' from your compiler's include directory. The cursor will be placed in the first line of the structure definition 'RastPort'. The reference file is read-only, i.e. you

may change its contents but you may not save it (this is to prevent you from accidentally overwriting important files).

Setting up the reference system

In order to have this command work properly you'll have to set up GoldED's reference system first (i.e. tell the editor where to look for files like 'graphics.h'): Use

config/references

to do this. Setting up the reference

system basically consists of selecting the files or directories to be referenced (e.g. your includes directory); it's a matter of a few seconds. GoldED is shipped without the reference system set up.

What files can be referenced ?

It is possible to reference a lot of different file types like source codes (C, BASIC, Pascal, Assembler), autodocs or C-header files (*.h). See

config/references

for details. Example: you might set up the reference

system to know the functions of your current programming project (probably consisting of many files). After having done this you could move the cursor over a function call of one of your own functions and then use find/reference. A new window would pop up, presenting you the lines of a file where this function is defined.

1.75 find/reference...

find/reference... of:

FIND MENU

Prompts for a string to be referenced. Example usage (if the ↔
reference system

is set up): Enter 'Window', then use OK. A new window would pop up, presenting you the header file from your compilers includes directory where a structure 'Window' is defined. See

find/reference

for further explanations.

1.76 find/complete

find/complete of:

FIND MENU

APC (Automatic Phrase Completion)

Trys to 'complete' the word your cursor is placed over. Example usage: type 'swin', then call this function. 'swin' would be replaced by 'struct Window' if the C-dictionary is present (see

config/dictionary
 on how to
 load/edit/create a dictionary). You might type 'swindow' as well - it would be replaced by 'struct Window', too. You might even type 'swdow'. Or 'struwi'. However, something like 'wind' wouldn't be recognized since the abbreviation's first letter must always be the same as the first letter of the full form. This function uses the dictionary facility; the larger your dictionary grows the more detailed your abbreviations will have to be to ensure unique identification. A keyboard shortcut of this function is assigned to the
 ESC key
 .

1.77 find/ASCII table

find/ASCII table of:
 FIND MENU
 Opens a character selection requester: All characters of your ←
 current font
 are displayed in a table. Pick the character you are looking for; it will be inserted at current cursor position.

1.78 find/ASCII insert

find/ASCII insert of:
 FIND MENU
 Prompts you for an ASCII code. Enter desired number (e.g. 65), ←
 then press
 return. The character will be inserted at current cursor position. The ASCII insert function is useful to embed control codes into your text. Example usage: Insert the codes 27 91 49 109 (calling this function 4 times) at the top of a document. This sequence is recognized by the printer device as 'turn bold on' command. The printer would switch to bold mode if the text were sent to the printer device (using
 project/print
).

Often-used sequences should be assigned to menu entries instead of using this requester; see

config/menus
 on how to create menu entries of type 'text'.

1.79 find/show code

```

find/show code of:
  FIND MENU
  Show ASCII code of character under cursor. This might be useful ↔
  to identify
'garbage' characters. Example usage: After loading of a large text file in
fast mode (see
  project/open fast
  ; TAB substitution turned off) you discover
strange characters at the beginning of some lines. Using this function you
might find out that those characters actually are TAB codes (ASCII code 9).
You then would have to reload (see
  project/open original
  ) the text to have
TABS replaced by spaces (use
  find/character set
  to get rid of other
'strange' codes).

```

1.80 find/character set

```

find/character set of:
  FIND MENU
  Character set remap

```

Asks you for a character set translation file before GoldED attempts to remap the current text. Remapping means that each character is replaced by another character defined in the translation file. The result solely depends on the translation file. If you load the 'AmigaToMSDOS' file using the REQ(uester) gadget, the text would be remaped in a way to make it readable by MSDOS machines. You could use the 'MSDOSToAmiga' to do it vice versa (make MSDOS files readable for the Amiga). Or load 'StripControl' to have non-printable 'garbage' characters removed. Finally StripNonASCII is useful to remove non-ASCII characters (many e-mail networks do not allow non-ASCII codes).

Most translation files are 'lossy': some characters won't get translated. For example MSDOS doesn't know the '@' character, thus a '@' couldn't be remapped properly by an AmigaToMSDOS translation file (a 'c' would be used instead).

1.81 find/insert code

```

find/insert code of:
  FIND MENU
  FF (same as pressing CTRL + L):

```

Inserts a form feed code (ASCII 12) at current cursor position. This code is recognized by printers. If the text is sent to the printer device (using

```

    project/print
), the printer would eject a page when it encounters this
code.

```

ESC (same as pressing CTRL + ESC)

Inserts an ESC code (ASCII 27) at current cursor position. This code introduces many command sequences understood by the printer device as well as by the console device (see

```

    FIND/ASCII insert
).

```

1.82 find/toggle case

find/toggle case of:

```
FIND MENU
```

```
Toggle case of character under cursor (make it uppercase if it ←
is lowercase
```

so far or vice versa).

1.83 find/functions

find/functions of:

```
FIND MENU
```

```
QuickFunc
```

All functions of this submenu will scan the current text for functions, structure definitions or AutoDoc entries. A listview is made up for fast selection: click at a function name to jump to where this function is defined. Several scan modes are available: Select C if you are looking for C-functions, PASCAL if you are looking for Pascal procedures/ functions, BASIC if you are looking for Basic PROCEDURES or SUBROUTINES, ASSEMBLER if you are looking for labels beginning with an underscore (e.g. `_main`), HEADER if you are looking for structure definitions and finally AUTODOC to scan AutoDoc-like files (programmer manuals). QuickFunc heavily depends on the way of formatting: Don't use a left margin in source codes. Function definitions must start at column one. Some indentation schemes are not handled properly. For example C-function headers (function name, parenthesis, arguments, parenthesis) must fit within one line to be recognized.

You may preselect a default scan mode for special file names using the pattern gadget below the listview. For example you could set the default file extension of the C-mode to `*.c`. Thus the C-mode would be used when referencing a file like `main.c`. "Referencing" means scanning the file for keywords (see

```
config/references
```

```
) or displaying the QuickFunc list ("show
```

all" menu).

Besides making GoldED create a jump table you may use the 'current phrase' option to move to a definition related to the word under the cursor. Example usage: Switch to C scanning mode. Then place the cursor over a 'C' function call like "CleanUp()" and use 'current phrase': GoldED will look for a function definition of "CleanUp()" within the current file and move to that function (if available).

Custom scan functions (advanced programmers only)

GoldED supports custom scanning functions: select the <mode> gadget to open a setup requester, then load a handler using <add>. The handler is LoadSeg()'ed by GoldED, i.e. it is expected to be an executable. The external handler will be called for each line of a text. It will receive the address of a line's string pointer (char **) in A0. The line's length is available in D0. The handler will have to examine this line. It is expected to return NULL if the line is of no interest from the handlers point of view (e.g. if you write a handler to look for #defines, return NULL if the line doesn't contain a #define). Or it may return the length of a result string to be displayed within the jump table. Set the string pointer (address passed within a0) to the result string's address in this case. Example code is shipped with GoldED (GoldED:Tools/GEDScan).

1.84 CONTROL MENU

CONTROL MENU

menu tree of control menu

- control/go to line
- control/window arrange
- control/top-bottom
- control/window center
- control/to last change
- control/window zip
- control/store position
- control/window enlarge
- control/recall position
- control/preview
- control/fold all
- control/freeze window


```

control/unfold all

control/next window

control/insert

control/previous window

control/toggle TAB mode

control/iconify

control/NumPad = movement

```

1.85 control/go to line

```

control/go to line of:
CONTROL MENU

```

Asks you for a line to go to. First line of a document is ←
 considered to be

line 1. Use the 'unfold' (checkmark-)gadget to decide whether you want to have folded sections unfolded if necessary (see Folding

). If unfolding is

enabled, line numbers are absolute, i.e. if you enter 255, the editor would jump to line 255, no matter whether this line is folded (unvisible) so far or not. If you don't enable unfolding, line numbers are based on the number of visible lines. The editor would jump to the 255th visible line. The actual line number of that line - if all folds were unfolded - is either 255 (no folds before this line) or greater (some folds before this line). The line numbers displayed below the window titles are based on the number of visible lines, too. They are not absolute. If you would perform an absolute jump to line 300, a number less than 300 might be displayed if your text contains folds (i.e. if not all lines of the text are visible).

1.86 control/top-bottom

```

control/top-bottom of:
CONTROL MENU

```

Moves the cursor to line one if it has been positioned near the ←
 end of your

text so far. Moves it to the last line of your text if it has been close to the top so far.

1.87 control/to last change

control/to last change of:

CONTROL MENU

Moves the cursor to the line of last change (or at least ←
close to that

position it if the last operation was a 'delete line').

1.88 control/store position

control/store position of:

CONTROL MENU

Records the position of the cursor & the view area of the text in ←
the window.

Use

control/recall position

to recall the cursor and view area. You may

store positions for each text window seperately. GoldED supports 10 bookmarks
for each text (5 found within this submenu - see

config/menus

for

information on modifying menus).

1.89 control/recall position

control/recall position of:

CONTROL MENU

Recall cursor position/view stored by

control/store position

from one of

five bookmarks. If you simply want to jump back to the line of your last
operation at some time or other you do not have to use bookmarks. Use

control/to last change

instead.

1.90 control/fold all

control/fold all of:

CONTROL MENU

Scans text for fold markers. Folds all text sections ←
surround by fold

markers (see below for general information on GoldED's folding facilities).

There is a keyboard shortcut quite similar to this function: CTRL + HELP; it toggles all folds: if the cursor is placed over a fold header, all folds are unfolded. If the cursor is placed over a normal text line, all sections surrounded by fold markers are folded.

Folding

One of this editor's most useful features is its folding capability: Folding means hiding some lines of a file temporarily. This is quite useful if you are working on a large sourcefile but don't want to get lost in thousands of lines. Simply fold away all functions you are not interested in. Unfold the ones you are working on.

How to fold lines

If you want to hide a section of lines, enclose it into 'fold markers'. Fold markers are plain character sequences. Default sequence is `///` (which is regarded as a comment by most K&R and C++ compilers). Example - type:

```
/// "important function"

void
main()
{
    puts("fold me !");
}

///
```

Now place the cursor over any line of the example above (except the last line) and press the HELP key. The lines above vanish, a single new line - the 'fold header' - appears:

```
> important function
```

To unfold that header, press the HELP key once more. You may fold as many sections of a document as you like. Use CTRL + HELP to unfold all folds upon a single keystroke. You may even have folds within folds up to any level (nested folding), however this requires the use of different markers for beginning/end of a fold section (see

`config/misc`

). Fold markers may be set

to any string of up to 10 letters. The shorter the faster. They must start in column 1 to 5 (to speed up scanning). In generally you should choose a sequence regarded as comment by your compiler. Or embed the fold markers into comments - for example after a REM if you are a BASIC programmer.

Fold headers are write protected since they aren't normal lines: keyboard input is suppressed if the cursor is placed over a fold header. Folded sections are not recognized by find & replace operations. Unfold a fold if you want to change it. However, block operations (see

`block menu`

) do work.

It is possible to copy, duplicate, cut or remove a fold if it has been marked as a block. Saving or printing a file is not influenced by folding - the text is treated as if all folds were unfolded.

1.91 control/unfold all

control/unfold all of:
CONTROL MENU

Scans text for fold headers. If some are found, they are ←
unfolded. Have a

look at the

Folding

chapter for information on GoldED's folding facilities.

1.92 control/insert

control/insert of:
CONTROL MENU

Toggles writing mode from insert to overwrite and vice versa. In ←
insert mode

keyboard input is inserted into the text without overwriting the existing text. In overwrite mode the existing text is replaced by your input. The current mode is displayed in the window's title bar. It is either INSR (insert) or OVER (overwrite).

1.93 control/toggle TAB mode

control/toggle TAB mode of:
CONTROL MENU

Toggles TAB mode from solid to light and vice versa. Solid TAB' ←
s actually

insert blank characters into your text. Depending on whether insert mode is on or off (see

control/insert

), the existing text is either indented or

overwritten. Light TABs do not overwrite anything - the cursor is simply moved to the next TAB position. See

config/TABs

on how to set TAB

positions.

1.94 control/NumPad = movement

control/NumPad = movement of:
 CONTROL MENU
 Toggle NumPad mode. Standard assignments (i.e. numbers) are ←
 used if this
 option is disabled. PC-bindings (e.g. PgUp = previous page) are used in
 extended mode (option enabled).

1.95 control/window arrange

control/window arrange of:
 CONTROL MENU
 Arranges windows on screen. If you have only one open window, the ←
 window is
 resized to cover the complete screen. If you have two open windows, the
 windows are made to share the display without overlapping. The screen's title
 bar is left free (unless you drag the screen down). You may assign extra
 space to the current window using the 'weight'-gadget of
 config/Gui
 . This
 function handles autoscroll screens (screens larger than the display)
 properly - the window(s) are arranged within the visible section.

1.96 control/window center

control/window center of:
 CONTROL MENU
 Centers current window on screen. If the screen is an autoscroll ←
 screen (i.e.
 larger than the display), the window is centered within the screen's visible
 section.

1.97 control/window zip

control/window zip of:
 CONTROL MENU
 Zips current window as if the window's zip (zoom) gadget were ←
 used: The OS
 keeps track of two alternative window sizes for each window. This functions
 toggles between them. The minimum window size is limited by GoldED (the
 editor ensures that at least one line of text can be displayed; besides it
 keeps the column/line display readable).

1.98 control/window enlarge

control/window enlarge of:

CONTROL MENU

Enlarge current window to make it as big as the screen. The screen's title ←

bar won't get covered. This functions handles autoscroll screen (screens larger than the display), too: only the visible section is considered.

1.99 control/preview

control/preview of:

CONTROL MENU

Toggles display mode of current window from 'normal' (standard ← text font

used) to 'preview' (preview font used; ususally a small font to increase the overall view). See

Preview

for more details.

1.100 control/freeze window

control/freeze window of:

CONTROL MENU

FREEZE WINDOW

Freeze text buffer, i.e. close the window but don't free the text itself. Useful to have several text files present without cluttering the display. Use UNFREEZE to reopen the window. Freezing the last window will leave you without menus - use the

hotkey

to reopen display: right ALT plus right

SHIFT plus RETURN.

UNFREEZE

Asks you for a frozen window to reopen. This functions tries to restore the old window position/size. See

control/freeze window

for information on how

to freeze a window.

SWAP

Asks you for a frozen text buffer to reopen; the current window is frozen instead. A new window will pop up if there is no frozen window.

1.101 control/next window

control/next window of:

CONTROL MENU

Activate the 'next' window (based on order of window creation). A ↔
very handy

function since you won't need the mouse for window activation. Simply flip from one window to another using this command.

1.102 control/previous window

control/previous window of:

CONTROL MENU

Activate the 'previous' window (based on order of window creation) ↔
. This menu

is a reversal of

control/next window

.

1.103 control/iconify

control/iconify of:

CONTROL MENU

Iconify GoldED. All windows as well as the GoldED screen (if the ↔
editor did

use an own screen) are closed. This function depends on the availability of the workbench screen where the editor attempts to place an AppIcon. Iconify won't work if the workbench is closed.

AppIcon

The AppIcon may be used to pass new files to GoldED: Simply drag icons of text files over it. Multi select is supported (i.e. you may drag more than one file at once over the icon using extended selection: hold the SHIFT key down while you select icons). The editor will open a new window for each file. Doubleclick at the icon to wake GoldED up again.

1.104 MISC MENU

MISC MENU

menu tree of misc menu

misc/help
misc/shell
misc/source files
misc/calculator
misc/filter
misc/HiSpeed
misc/search file
misc/files
misc/line swap
misc/insert date
misc/line double
misc/insert time
misc/line pick
misc/insert path
misc/line push
misc/command
misc/undo line
misc/last error
misc/statistics

1.105 misc/help

misc/help of:
MISC MENU

AmigaGuide is made to display the main page of GoldED's manual (' ← database' in terms of AmigaGuide). You may use config/menus to select a new database.

This feature depends on the amigaguide library - if it is not available, you won't get any help.

1.106 misc/source files

misc/source files of:

MISC MENU

Open project definition requester. Used to specify all source files that

belong to a project. Doubleclick on a filename of this list to load the corresponding file. Multiselect is supported (requires OS3.0 or better): Hold the SHIFT button down while selecting files and use the OPEN gadget to make the editor load all selected files. GoldED itself offers only limited support for this list: The editor provides a few basic functions (e.g. adding/removing files or

misc/filter

. The project list has been implemented

to support the development of third party utilities (e.g. make tools or printing utilities). Further processing has to be done by external programs:

List access (programmers only)

Programs may send a

QUERY

command to GoldED's

ARexx port

in order to

obtain a pointer to a linked list of nodes: QUERY PRJLIST. Example source code is shipped with GoldED; check the GoldED:Tools/PRJSource directory. The node->ln_Name fields of the list's nodes will point to source file names (NULL terminated). You may pass this list to a listview. Use of this list requires a previous LOCK to ensure a valid list. The list is read-only. Use the

PROJECT

command to add or remove files. Do not modify the list on your own.

1.107 misc/filter

misc/filter of:

MISC MENU

Global search

Shows a requester to enter the text to search for. Examines all files found within the project list (see

misc/source files

). If the search pattern is

not found within a file, the filename is removed from the list (OS2.0). The name is highlighted if the pattern is found (OS3.0). You may choose the search to be case sensitive or not by using the appropriate checkmark gadget. Doubleclick at a filename to load the corresponding file. Hold the SHIFT button down while doing so if you don't want to lose the other marks.

Note: case-sensitive search is much faster than case-insensitive search.

1.108 misc/search file

```
misc/search file of:
  MISC MENU
  File hunter (suggested by David Göhler)
```

Extract file name from text under cursor. The editor knows about several file name delimiters used by different programming environments (e.g. <...> or quotations marks). It will decide for the word under cursor (surrounded by spaces) if no delimiters are found. Tries to locate and open that file. Searches the directory of the current text as well as default directories set up by

```
  config/file hunter
  (unless the file name is absolute). Tries to
append a default suffix (see
  config/file hunter
  ) if the file has not been
found. Useful to follow file links found in many programming languages (e.g.
#include of 'C' source codes).
```

1.109 misc/line swap

```
misc/line swap of:
  MISC MENU
  Swaps current line with next line. The function is ususally ↔
  used via the
ARexx interface to sort a text.
```

1.110 misc/line double

```
misc/line double of:
  MISC MENU
  Doubles current line. Faster than using the clipboard or
  block/copy
  .
```

1.111 misc/line pick

misc/line pick of:

MISC MENU

Delete current line from text. The line is not lost - it is ←
put to a

pick-push ring buffer (last in, first out). You may reinsert it anywhere else by using

misc/line push

. This function is assigned to the keyboard, too (CTRL-DEL). Note: keyboard access (CTRL-DEL) will give you a MUCH BETTER response time than menus due to the Amiga's OS (keyboard queue settings unfortunately don't affect menu shortcuts). Example usage: Delete 3 lines, move to another line and call

misc/line push

three times to reinsert the lines. The pick/push buffer holds up to 50 lines.

1.112 misc/line push

misc/line push of:

MISC MENU

Inserts the last line of the pick/push buffer before ←
current line. See

misc/line pick

for further explanations.

1.113 misc/undo line

misc/undo line of:

MISC MENU

Restores the previous contents of a line after you have changed ←
it. You can

undo a line as long as the cursor has not been moved to another line - leaving a line means accepting the change. This function can not be used to undo a delete line operation. Use

misc/line push

to reinsert a deleted

line.

1.114 misc/statistics

misc/statistics of:

MISC MENU

Presents a statistic overview of a text. Bytes, lines, folded blocks and non-ASCII-characters (codes above 127) are counted. Additionally the width of the longest line is determined. This function treats the text as if it were saved (i.e. all folds unfolded, CR codes appended to the lines).

1.115 misc/shell

misc/shell of:

MISC MENU

Opens a shell window on the screen used by the editor. The window is arranged

on screen, i.e. it will open within the visible section of overscan screens. GoldED takes care of providing valid path information: if you have a PATH SYS:C2 ADD command in your startup-sequence, the shell window will know about it.

1.116 misc/calculator

misc/calculator of:

MISC MENU

Tries to run the calculator of your workbench (must be placed in sys:tools).

The calculator is made to appear on the screen used by GoldED even if a custom screen is used.

1.117 misc/HiSpeed

misc/HiSpeed of:

MISC MENU

Tries to run the HiSpeed printer tool - the program is made to open its

window on the screen used by GoldED.

HiSpeed

HiSpeed has been designed for PCL printers like Hewlett Packard's DeskJet family. It is used to reduce the amount of paper when printing large files by using a small typeface. Up to 8 A4 pages are redirected to a single A4 sheet. It is fast. Speed depends on your DeskJet model; DeskJet+: about 4 pages/minute if you switch to HiSpeed mode. It should be a nice utility for people who have to do a lot of printing. Some of HiSpeed's features are:

- o workbench interface
- o spooler (job list)

- o AppWindow/AppIcon support
- o shell interface
- o ARexx port
- o single or double sided printing
- o descending printing available
- o free layout - e.g. two columns
- o ANSI ESC sequences supported
- o Linefeed adjustable
- o HiSpeed mode: fast printing
- o preview (WYSIWYG)
- o many fonts supported
- o page headers
- o numbering of lines
- o index/appendix creation
- o protrait/landscape
- o book mode to print A5 books

Please read the HiSpeed manual (see GoldED:Tools/HiSpeed) for more information on this tool. HiSpeed is shareware. The registered version is not part of the GoldED light distribution but it is part of the GoldED pro package; see

how to register

.

1.118 misc/files

misc/files of:
MISC MENU
DELETE FILE

Asks you for files to delete (hold the SHIFT button down to select multiple files). You are asked for confirmation. It is not possible to delete delete-protected files.

RENAME FILE

Ask you for a file to rename; you are asked for new names if you select one or more files.

CREATE DIRECTORY

Asks you for a directory to create. Attempting to create an already existing directory will result in an 'object in use' error.

1.119 misc/insert date

misc/insert date of:
MISC MENU
Inserts current date (e.g. "Boston, 3/5/93") at cursor position. ←
You should

set the environment variable USERTOWN to the name of your town using the DOS command setenv (e.g. setenv USERTOWN "Boston"). DOS/setenv uses the 'env:' directory to store environment variables. Unfortunately this directory is placed in RAM, so your settings are lost after a reset. Copy env:usertown to envarc:usertown to prevent this.

1.120 misc/insert time

misc/insert time of:
MISC MENU
Inserts time string at cursor position. Formatting depends on ↔
the locale
library (see
language
).

1.121 misc/insert path

misc/insert path of:
MISC MENU
Inserts a file name at cursor position. A file requester is ↔
offered for easy
selection.

1.122 misc/command

misc/command of:
MISC MENU
Asks you for one of GoldED' internal commands to be executed (↔
see list of
internal commands
). This function is assigned to SHIFT ESC, too. Parsing is
done by the OS function ReadArgs, which is used by most CLI commands, too, so
same rules as usual apply (arguments containing spaces have to be quoted).

1.123 misc/last error

misc/last error of:
MISC MENU
Shows the last error message once more (if there has been any so ↔
far).

1.124 MACRO MENU

MACRO MENU

menu tree of macro menu

```

macros/edit macro
macros/sequence play
macros/run text as macro
macros/play many
macros/sequence record
macros/macros C
macros/sequence load
macros/macros others
macros/sequence save
macros/GUIMake

```

1.125 macros/edit macro

macros/edit macro of:

MACRO MENU

Changes to GoldED's macro directory and asks you for an ARexx ↔
macro to load.

You should save your current text before or open a new window. GoldED's ARexx macros use 'GED' as suffix (e.g. number.ged). All macros addressing GoldED must use a special protocol to register with GoldED before performing operations to prevent race conditions if user & macro are trying to control the editor simultaneously (see

ARexx port

). We have provided an empty macro

(empty.ged) to be used as basis for own development efforts. Simply load this macro, insert your code and save it under a new name.

1.126 macros/run text as macro

macros/run text as macro of:

MACRO MENU

Tries to execute the current text as macro. You should save the ↔
text before

you call this function since the copy on disk is executed, not the text in memory. All ARexx macros have to start with a comment (`/* ... */`) to get recognized by the ARexx server. If the first line of the text isn't a comment, nothing happens. A script called by this functions has its host set up properly already (i.e. you don't need an ADDRESS command). Example: type these lines (without left margin) and then select 'run as macro' to have them executed:

```
/* this is a test */

'LOCK CURRENT'
'REQUEST BODY="Hi!"'
'UNLOCK'
```

1.127 macros/sequence record

```
macros/sequence record of:
  MACRO MENU
  Macro recording - Sequences
```

Start (1st call) respectively stop (2nd call) recording of a keyboard/command sequence. During recording all key presses and menu selections are logged. Mouse movements/clicks are not recorded. Use

```
macros/sequence play
to replay
```

a recorded sequence. Use

```
macros/sequence save
to write the sequence to
```

disk, ready to be used at some other time. If you open a requester while recording you will be asked whether this requester should pop up in playback mode, too (unless it belongs to the config menu). If you disable the requester, GoldED will use the selections made at recording time (e.g. if you moved the cursor to line 100 during recording, it will be moved to line 100 in playback mode, too. No GOTO requester will appear). Macro recording is based on low-level events like keystrokes or menu selections. For example pressing of the F10 key is recorded as "F10 has been pressed", no matter what action has been assigned to that key. If you change key bindings or menus after recording a macro, the macro will behave differently. This command is assigned to the SHIFT-F10 key, too.

1.128 macros/sequence load

```
macros/sequence load of:
  MACRO MENU
  Load a recorded sequence to be replayed by
  macros/sequence play
  (or by
```

pressing the F10 key). In generally it is more convenient to assign often

used sequences to keys or menu items (see
 MACRO
 command] instead of using
 this function.

1.129 macros/sequence save

```
macros/sequence save of:
  MACRO MENU
  Save a recorded command sequence, created by
  macros/sequence record
```

.
 Sequences related to GoldED should be written to the GoldED:Macros drawer.
 File extension should be "*.seq".

1.130 macros/sequence play

```
macros/sequence play of:
  MACRO MENU
  Replay a previously recorded keyboard/command sequence (use ←
  SHIFT+F10 or
```

```
  macros/sequence record
  to enter/leave recording mode). This command is
  assigned to the F10 key, too. If you want to replay the macro several times,
  use
```

```
  macros/play many
```

1.131 macros/play many

```
macros/play many of:
  MACRO MENU
  Replay a previously recorded keyboard/command sequence several ←
  times (use
```

```
SHIFT+F10 or
```

```
  macros/sequence record
  to enter/leave recording mode). Replay
```

of the sequence will stop if an error occurs during execution (e.g. if the
 find function reaches the end of a file).

1.132 macros/macros c

macros/macros c of:

MACRO MENU

This menu offers several C-related ARexx macros (GoldED's ARexx port is ←

described in the

ARexx port
section of this manual):

MARK

Mark all lines between nearest pair of curly brackets.

FUNCTION BODY

Insert empty function body ('C' style function). You will be asked for the function's name (e.g. "main") and the return type (e.g. UWORD). An empty function body is inserted at current cursor position, an empty comment placed above the function.

ADD SWITCH

Insert switch body. You could use ADD CASE to add further CASE branches.

ADD CASE

Add a further CASE branch to the last SWITCH statement. The new branch is inserted as first CASE line.

DMAKE

Does look for a makefile called 'dmakefile' within the directory of the current text. If one is found, dmake of the DICE C compiler is evoked (won't work if you don't own DICE, of course). This menu command actually is a macro. It fails if the ARexx server REXXMast is not running in the background. Usually the ARexx server is installed during startup (s:startup-sequence): run >NIL: sys:system/REXXMast.

COMPILE & LINK

Compiles and links the text using DCC of the DICE C distribution (won't work if you don't own DICE). To be more precisely: a temporary copy of your text (T:TEST.c) is created, this is passed to DCC to create an executable called T:TEST. If DCC has compiled/linked your file (without error/warnings), you are asked whether you want to run it.

1.133 macros/macros others

macros/macros others of:

MACRO MENU

This menu offers all-purpose ARexx macros (GoldED's ARexx port is ← described in the ARexx port section of this manual):

FOLD BLOCK

Fold marked lines (see block/mark on how to mark lines). You are asked for a comment to be used as fold header. Please read the Folding section of this manual if you are unfamiliar with GoldED's folding feature.

REVISION (by Marius Gröger)

Update version string, revise history: This function will scan your text for a version string according to the programmers' style guide (published by Commodore). If one is found, the revision number is increased by one. Versions strings (e.g. \$VER GED 1.0 (1.10.93)') are emedded into programs to provide required information for the AmigaDos command VERSION (VERSION scans files for the '\$VER:' keyword). Additionally, you are asked for a short comment about the latest version if a '\$HISTORY:' section has been detected within current text. The comment is added at the top of the history list. Example text header understood by the revision command:

```
char *Version = "$VER: revision 0.8 (21 Jul 1993)";

/*
  $HISTORY:

  18 Jul 1993 : 000.003 : added commandline args
  18 Jul 1993 : 000.001 : initial release
*/
```

This command actually is a macro. It won't work if the ARexx server REXXMAST is not running in the background. Usually the ARexx server is installed during startup (s:startup-sequence or s:user-startup): run >NIL: sys:system/REXXMAST

NUMBER LINES

Numbers a text. You are asked for the start value and the step value to be used. Choose 1000 followed by 10 if you want to get the sequence 1000, 1010, 1020, ... This command actually is a macro. It won't work if the ARexx server REXXMAST is not running in the background. Usually the ARexx server is installed during startup (s:startup-sequence or s:user-startup): run >NIL: sys:system/REXXMAST

EXECUTE LINE

Execute the current line as DOS command. Useful to execute compiler calls being part of the source code; example file header (place the cursor over the "dcc ..." line before calling this macro):

```

/* -----
ED v0.91 - GoldED quick starter, ©1993 Dietmar Eilert. DICE:

dcc main.c -// -proto -mRR -mi -r -2.0 -o ram:ED

-----
*/

```

ASSEMBLE

Assemble the current file which is expected to be assembler source code: A temporary copy of the current file named "t:test.asm" is created and passed to the assembler in order to create an object file "t:test.o" (pass 1). The object file (if the assembler has completed successfully) is passed to the linker in order to create an executable "t:test". This macro depends on the A68K assembler (copyright 1985 by Brian R. Anderson, AmigaDOS conversion copyright 1991 by Charlie Gibbs) and the linker "Blink"; these programs are not shipped with GoldED.

USE PATH

Set default path to path of current text (the default path is considered by project/open new and several other functions).

1.134 macros/GUIMake

```

macros/GUIMake of:
  MACRO MENU
  GUIMake

```

Display GUIMake copyright information. GUIMake is a project manager for DICE-C. The GUIMake package has been developed by Rico Krasowski - included with kind permission of the author. The purpose of GUIMake is to replace makefiles by a GUI-driven program. Besides comfortable file management (compiling/linking) GUIMake offers very handy error handling facilities. For example GUIMake might make the editor jump to the first erroneous line after compilation. GUIMake is a stand alone program -communication between GUIMake and GoldED is based on ARexx. Have a look at the GUI guide file (tools directory) for more detailed information on GUIMake and its ARexx port. This function as well as all other menu entries related to GUIMake are not available if you have disabled DICE support during installation of GoldED. GUIMake access depends on the ARexx server REXXMast which must be running in the background. Usually the ARexx server is installed during startup (s:startup-sequence): run >NIL: sys:system/REXXMast.

OPTIONS

Open options window of GUIMake (used to set compiler options). Have a look at the GUIMake manual for a more detailed description. This function isn't

available if you have disabled DICE support during installation of GoldED.

MODULES

Open modules window of GUIMake. Used to select files related to your project: source files, header files, ... Have a look at the GUIMake manual for a more detailed description of Rico's make utility.

CONFIG

Open config window of GUIMake (used to set up a project). Have a look at the GUIMake manual for a more detailed description. This function isn't available if you have disabled DICE support during installation of GoldED.

ERROR: FIRST

Make GoldED jump to the first erroneous line (either error or warning) after compilation (see COMPILE & LINK) has completed. The original DICE error/warning text is displayed below the window's title bar. This function isn't available if you have disabled DICE support during installation of GoldED.

ERROR: NEXT

Make GoldED jump to the 'next' error/warning. The line number is extracted from the file T:ERRORS produced by the compiler/GUIMake during compilation (see COMPILE & LINK). The original DICE error/warning text is displayed below the window's title bar. This function isn't available if you have disabled DICE support during installation of GoldED.

ERROR: PREVIOUS

Make GoldED jump to the 'previous' error/warning line (after having used the NEXT option). The original DICE error/warning text is displayed below the window's title bar. This function isn't available if you have disabled DICE support during installation of GoldED.

SHOW ALL

Load error file created during compilation (COMPILE & LINK). This function isn't available if you have disabled DICE support during installation of GoldED.

COMPILE & LINK

Compile & link files of your current project (projects are set up using the MODULES requester). Won't recompile a file unless it has been changed since last compilation (see COMPILE & LINK NEW). Have a look at the GUIMake manual (BUILD command) for a more detailed description. This function isn't available if you have disabled DICE support during installation of GoldED.

COMPILE & LINK NEW

Compile and link all files of your current project, no matter whether files have been changed or not since last compilation. Use MODULES to specify source files, header-files and object files related to your project or OPTIONS to set compiler options. Have a look at the GUIMake manual for a

more detailed description. This function isn't available if you have disabled DICE support during installation of GoldED.

RUN EXECUTABLE

Run the executable created by DICE/GUIMake (COMPILE & LINK). This function isn't available if you have disabled DICE support during installation of GoldED.

1.135 CONFIG MENU

CONFIG MENU

menu tree of config menu

config/references

config/TABs

config/file hunter

config/display

config/api

config/gui

config/menus

config/layout

config/mouse

config/printer

config/keyboard

config/misc

config/dictionary

config/save

config/templates

config/load

config/indentation

Use these functions to adjust GoldED to your likings; don't forget to use ↔

config/save

before you leave the editor; your definitions would be lost otherwise.

1.136 config/references

config/references of:

CONFIG MENU

Open a requester to set up the reference system. Whenever the user ← asks for a

reference (see

find/reference

), GoldED searches the reference database for

a match. The database is an index file, consisting of keyword-filename pairs: one reference file (eg. graphics/rastport.h) for each keyword (e.g. struct RastPort) GoldED knows about. If an entry matching the request is found, the corresponding reference file is loaded.

Database creation

Use the string gadget to select a database. The editor is shipped without a database, you have to create the first one on your own. Suggested file name is 's:GoldED.refs'. A new data base initially is empty. All you have to do in order to fill the database is selecting some files or directories using the file/directory gadgets below the list. Then choose 'create' to make GoldED scan all selected files and directories (including subdirectories) for keywords and create an index file. Scanning mode depends on the file name: if the file name suffix is '.c', function names are extracted. If the suffix is '.h', structure definitions are extracted. Several other types are recognized, too (see

find/functions

). You may even change the default file

extensions recognized by the scanner (see

find/functions

) or add your own

scanners. The index file is examined every time you are going to look for a definition (

find/reference

); you may keep the index file resident (RAM

gadget) for the sake of speed, though this might consume a lot of memory.

1.137 config/file hunter

config/file hunter of:

CONFIG MENU

Open file hunter window. Used to set up default drawers where the ← editor is

going to look for a file if requested by

misc/search file

. Subdirectories

are examined, too, if the "RECURSIVE" gadget is selected. You may specify a default suffix for each directory. The editor will append this suffix to the file name if it isn't able to locate the file without the suffix (the suffix must be given in the form *.suffix). Example usage: add the include directory of your C compiler to the directory list; set the suffix to "*.h". Now place the cursor over the following file name (between the brackets - this is 'C' code) and use

```
misc/search file
:

#include <amiga20/exec/exec.h>
```

The editor will then search the include directory for a file called "amiga20/exec/exec.h". Note that "amiga20/exex/exec.h" is a relative path without drive specification, so GoldED wouldn't be able to resolve it without the file hunter. You may add a suffix (e.g. *.tex) without reference to a known directory, too, using the "FILE" gadget: A "*.*" symbol instead of a directory name is added to the list in this case. Example: If the hunter is asked to look for a file called "tex:text", it would try "tex:text.tex", too.

1.138 config/API

```
config/API of:
CONFIG MENU
Application Interface
```

Use this requester to select clients to be launched during startup of GoldED. Clients are external programmes, sharing information with the editor (using a special, message-based protocol). Launching clients is a flexible approach to increase the abilities of this editor by third party programmers. Four clients (some of the include C source code) are shipped with GoldED:

Dock : user defined icon bar

Have a try and add GoldED:tools/GEDDock/dock as client: You'll get a ToolManager dock (icon bar) next to your text windows, providing often used functions at a simple mouse click (requires ToolManger library 2.0; ToolManger is ©1990-1994 Stefan Becker). Experiences users may change position, looks and functions of this dock by editing the dock configuration file "GoldED:API/dock/dock.prefs". Example settings file:

```
; dock settings

ADD AREXX COMMAND="'ADDRESS %s; FREEZE CURRENT' " ICON="icons:FREEZE.iff"
ADD AREXX COMMAND="'ADDRESS %s; FREEZE SWAP' " ICON="icons:EXCHG.iff"

DOCK X=0 Y=0 HORIZONTAL COLUMNS=1
```

A dock configuration file may consist of empty lines, comments (introduced by a semicolon) and command lines. The commands ADD and DOCK are available: DOCK is used to specify the basic look of the dock (including position and orientation), ADD is used to add dock entries. Dock entries are a combination of icon and action; two action types are available: AREXX (a

command is sent to the ARExx servers if the user clicks at an icon) and EXEC (a shell command is executed). Syntax (compare internal commands):

command	option	description
ADD	AREXX/S	set action type to 'ARExx'
	EXEC/S	set action type to 'program'
	COMMAND/K/A	command (%s is replaced by host port name)
	DIR/K	current directory
	OUTPUT/K	output file
	ICON/K/A	iff icon file

Comment: Use single quotations marks to have the command string evaluated by the Arexx server. Use double quotation marks to make the server execute a script.

command	option	description
DOCK	X/N	x position of dock
	Y/N	y position of dock
	HORIZONTAL/S	orientation (default is vertical)
	COLUMNS/N	number of columns

Comment: Do not specify X/N to make the client position the dock at the rightmost screen position. Do not specify Y/N to make the client position the dock below the title bar.

Save the dock configuration file after having made your changes. Open GoldED's API requester and use OK to restart all clients (thus making the dock client reload its configuration).

spellchecker (Spell & SpellIT)

Include the GoldED:API/spell/spell client to add online spell checking capabilities to GoldED: If this client is active, your input is spellchecked while you are typing (the last word is checked every time you type a whitespace character). Detection of errors causes audible beeps. Add "GoldED:API/spell/spell ASK" to your list of clients if you want a list of suggestions upon error detection. This client is based on the ISpell freeware package (available on Fish disks). ISpell has to be installed before you can use this feature.

Besides Spell you may use the (more powerful) SpellIT client to add online spell checking to GoldED. Online documentation (AmigaGuide) of this program is available in the GoldED:API/SpellIT drawer.

command set extensions

The API interface provides ways and means to extend the command set of GED: Have a try and include the GoldED:API/rexx/rexx client example to add two new commands, DISPLAYBEEP and ABOUT. Once installed, these commands may be used as any other built-in command (see

internal commands), i.e. you may use them within menus, key bindings, ARExx macros or in the command

requester: Choose
 misc/command
 to open the command requester and enter
 ABOUT.

1.139 config/menus

config/menus of:

CONFIG MENU

You may have as many menu titles (left listview) as you like ←
 - up to

intuition's limit. Each menu may have as many menu entries (middle listview)
 or submenus (right listview) as you like (up to intuition's limit once more).
 Doubleclick at a menu item to set its action (see
 event definition

). Use

the arrow gadgets to move a listview entry one position up or down. Use the
 'key' gadget to set a menu shortcut (a key to be used in conjunction with the
 right Amiga key). Select the arrow gadget to get a list of "unused"
 shortcuts. Only ASCII characters (ie. codes <= 128) are listed; the use of
 non-ascii ("national") characters is discouraged if you want to share your
 menus with other users from other countries. Menu shortcuts are expected to
 be single characters; they are not case sensitive. However, you may specify
 longer strings up to ten characters, too (requires OS3.0+). These strings
 (e.g. "CTRL-V") will appear in your menus though they will not act as
 shortcuts. Recommended usage is a reference to corresponding key bindings
 (see

config/keyboard

).

You may attach internal boolean variables to menus using the 'checkmark'
 gadget. The current state of these variables (on/off) will be represented by
 a checkmark in the final menu. Keep in mind that attaching variables is just
 a rendering operation. You'll have to assign appropriate commands (i.e.
 commands affecting the attached variable) to a menu if you want to change the
 state of a checkmark by selecting the menu (see

event definition

).

USER variables

Besides preset variables (e.g. INSERT reflecting the current writing mode)
 there are 20 boolean user variables for your private use. Use the

QUERY

command to get their current value (e.g. QUERY USER20) and

SET

to set them

(e.g. SET USER=20 VALUE=TRUE). Typical usage would be the management of
 compiler flags. Example: Create a menu item called "030-code". Attach the
 USER1 variable to it and set the action to "SET USER=1 VALUE=TOGGLE". Now you
 are able to toggle the state of USER1 from TRUE to FALSE by selecting the
 menu, the current state being represented by a checkmark. Finally, you would
 have to write a 'compile' macro, reading the state of USER1 (QUERY USER1) and

taking appropriate action.

The 'hyper' string gadget selects a guide file ('database'), containing menu descriptions. This database is used as default database by all commands related to help handling (e.g.

```
misc/help
).
```

Activate the 'leave out' gadget if you want to leave out a menu item as window border gadget (see

```
user defined gadgets
).
```

You should assign short names to these menu items since the number of user defined gadgets is limited by the available space within the window titles.

1.140 config/mouse

config/mouse of:

```
CONFIG MENU
```

Open mouse configuration requester. This requester is used to map \leftrightarrow commands to

the mouse buttons. You can edit the left button as well as the middle button if your mouse features one. You can not remap the right mouse button (i.e. the menu button). Single clicks, double clicks and qualifier combinations (SHIFT, ALT, CTRL) are available. Event definitions are described in the

```
event definition
section of this manual.
```

1.141 config/keyboard

config/keyboard of:

```
CONFIG MENU
```

Open keyboard binding requester used to map desired action \leftrightarrow to keys or

key-qualifier combinations (GoldED recognizes the qualifiers ALT, CTRL and SHIFT). As long as you don't bind any action to a key, the default keymap is used whenever it is possible: Pressing the 'A' key would insert an 'A' at current cursor position. Non-character keys (like the cursor keys) are initially unset, i.e. they wouldn't have the expected effect like moving the cursor (however GoldED is shipped with this stuff already set up). To make the cursor keys move the cursor you would have to bind 'move cursor' commands to these keys. Setting up a key(-combination) is easy: Simply use the record gadget and press the desired keys. A new requester will pop up, asking you for what action is to be assigned to this event (see

```
Event definition
below). However, some key combinations are consumed by the OS  $\leftrightarrow$ 
and thus not
```

available for remapping ('dead keys'). For example pressing ALT-G will not result in a character but influence the next event ("a" after ALT-G turns into "à"). The following keys are dead keys:

```

`      ALT-H
ALT-F  ALT-J
ALT-G  ALT-K

```

1.142 Event definition

Event definition

You may assign any of the editor's
internal commands
to a given event

(event = keystroke or menu selection). Or a DOS command. Or a macro. Or just a string. Write the command, script or string (use quotation marks !) to the CMD gadget and set the event type; supported event types are internal, arexx, shell, text or dummy. If you set the event type to 'dummy', you would disable the event (i.e. nothing would happen if the event is encountered). The DIR gadget may be used to set the current directory while the event is processed. This is supported for events of type shell only. You may set the output handle (e.g. 'con:0/0/640/400') using the OUTPUT gadget; this is supported for events of type 'arexx' and 'shell' only. GoldED defaults to opening a console window on its screen if the output handle is omitted. The string contents of the CMD/OUTPUT/DIR gadgets are 'interpreted' before use (except if the event is of type 'internal'); see

Magic codes

.

Select 'shanghai' if you want to make all windows use GoldED's screen during event processing. You have to specify how long shanghai mode is to be turned on; units are seconds. Turn Async ON to make events of type 'shell' asynchronous. Finally you may assign a help text to an event using the 'hyper' gadget: enter a node name of the 'current' AmigaGuide database (see

config/menus

). The help text assigned to a menu event is displayed during

MenuHelp

processing. If you don't provide a node name, GoldED defaults ←
to

use M<menu number>.<item number> (e.g. M1.2). You may use the '@' character to select a database (i.e. to override the default database) . Example usage: GOLDED:GOLDED.GUIDE@MAIN would make the editor look for a node 'MAIN' within the amigaguide file 'GOLDED:GOLDED.GUIDE'.

Multiple commands

You may assign any number of commands to a single event (menu item or key). You may mix command types (e.g. a shell command followed by one of GoldED's

internal commands

), too, but all commands will share the general settings
of the item requester (e.g. all shell commands will be asynchronous if ASYNC

is checked). Please keep in mind that ARexx processing is always asynchronous: after the command has been sent to the ARexx server GoldED will proceed immediately. Don't make assumptions about whether the ARexx command has already completed then (in most cases it won't - ARexx isn't that fast ;-). In generally you should avoid mixing ARexx commands with other commands.

1.143 Magic codes

Magic codes

Interpretation of strings (see event definition) means that some predefined symbols like \DATE are replaced by their actual value if the string is finally referenced by GoldED; quote a string if you don't want it to be interpreted. The following keywords are supported:

```
"<characters>" ..... character constant
%<number> ..... inserts ASCII code <number>
\n ..... return
\t ..... tab
\b ..... move cursor left
\" ..... quote
\NAME ..... current file name
\CON ..... window dimension string
\DATE ..... current date
\TIME ..... current time
\SCREEN ..... screen name
\HOST ..... name of GoldED's ARexx port
$<name> ..... environment variable <name>
```

1.144 config/dictionary

config/dictionary of:

CONFIG MENU

Opens requester to edit the dictionary. The dictionary is used by the editors ↔

APC

facility (to complete expressions) as well as by its AutoCase support ↔

```
E dictionary ..... ©1994 Andreas Weiss
ARexx dictionary ..... ©1994 Andreas Weiss
C dictionary ..... ©1994 Dietmar Eilert
KickPascal dictionary ... ©1994 Stefan Kraus
```

AutoCase

If you enable AutoCase checking, the editor will search the current line for words present within the dictionary, too. This check is performed when the cursor leaves the current line. It is case-insensitive (e.g. Rastport and RastPort would be recognized as the same expression). If a match is found, it is replaced by the dictionary entry, thus possibly correcting case.

General hints

Don't make the dictionary too big - the smaller it is, the more efficient it will be. Only add unique names to it - otherwise the editor might try to correct the spelling of a word even if you don't want it. For example it wouldn't be a good idea to add 'RastPort' for this would prevent you from using a variable 'rastport' within your program. However, 'struct RastPort' is fine, since this is the only way to write this kind of structure definition (at least as far as Amiga C programmers are concerned). Make trailing spaces part of the dictionary entries: For example there is always a space after the "int" keyword as far as C sources are concerned, so don't make "int" part of the dictionary but use "int " (this prevents the editor from replacing INTERNATIONAL by international).

Parenthesis check

Toggle the ()-check gadget to ON if you want the current line to be checked for correct use of braces as soon as the cursor leaves it (see find/check).

Unfortunately this check is performed if the display is shifted, too, due to the internal design of GoldED. Don't use this option for 'free-style' programming languages like C.

1.145 config/templates

config/templates of:

CONFIG MENU

Open requester to set the templates

recognized by the editor. Templates are

patterns the editor is looking for while the user is typing (if templates are turned on:

layout/templates on/off

). Only single words may be added as

search patterns since the template scanner examines the current word only during user input. If a template is found, the pattern is removed and template-specific operations are performed. You may assign either recorded

sequences

or events (arrow gadget; see event definition

) to templates.

1.146 config/indentation

```
config/indentation of:
  CONFIG MENU
  AutoIndentation , SmartIndentation
```

Open requester to set the indentation scheme. Turn AutoIndentation ON, if you want the cursor to be indented the same amount as the previous line after a CR (see

```
return key
```

). Turn SmartIndentation ON if you want automatic indentation after user defined keywords (e.g. after IF); use the listview gadget to set desired keywords. Supported smart indentation types are:

```
-->    next line: cursor indentation
<--    next line: cursor outdentation
>>>   shift current line right
<<<   shift current line left
<<< -> shift current line left; next line: indentation
```

1.147 config/TABs

```
config/TABs of:
  CONFIG MENU
```

```
Open TAB configuration requester. GoldED supports several ↔
modes as far as
```

TABs are concerned: solid TABs as well as light TABs are available (see:

```
control/toggle TAB mode
```

. Additionally you may decide for distinctive TABs, regular ones or dynamic TABs: while regular TABs are set using the 'tab key' slider (e.g. to every 4th column), distinctive ("fixed") TABs are set using a listview; simply enter desired TAB positions. Last but not least dynamic TABs are available: In dynamic mode GoldED will examine the last line(s) to determine appropriate TAB positions. If none are found, regular TAB's are used.

GoldED never uses TABs when writing a file to disk, however some other editors do, usually replacing eight spaces by a single TAB. GoldED resubstitutes these TAB codes by spaces while loading (see

```
project/open
```

```
).
```

Usually one TAB code is replaced by eight spaces. Use the 'tab file' gadget to change this. Setting it to four would make GoldED use four spaces for each TAB.

1.148 config/display

```
config/display of:
  CONFIG MENU
  Display mode   Font
```

Open requester to set display properties (i.e. resolution, fonts, window look and more). Some gadgets of this requester are 'dangerous' since they require closing down the current windows/display temporarily: the editor might not be able to reopen its display if you are short of memory.

You may select four different fonts to be used for the text, within requesters or for menus (if the editor runs on a custom screen). These fonts (except the screen font) have to be fixed width ones.

Preview

Besides the standard text font you may specify a preview font. This font is used by GoldED if you switch a window to preview mode (`control/preview`).

The preview font should be considerably smaller than the standard text font. Intended usage is increasing the overall view on the fly, thus avoiding to loose track while working on complex sections of a source code.

Icons

Enable/disable use of icons within requesters. Since the editor's icons have been designed with the OS2/OS3 color scheme they might not look that good if you use your own color scheme. Use this gadget to turn icons on/off.

Pens

You may set the pens used by the OS (and GoldED) to render the user interface. Some pens are not adjustable under OS2.1 or earlier. For example setting the menu background pen used to render menus requires at least OS3.0.

Shanghai

Turn shanghai mode ON if you want to force all windows ususally opening on the default public screen (i.e. the workbench screen) to open on GoldED's custom screen. Use not recommended if GoldED's screen is a one-plane screen (many programs won't look that pleasing then).

Chunky pixel

Many external graphics boards don't use a bitplane representation of graphics (as the current Amiga chipsets do) but a chunky pixel organization. Use the 'chunky mode' gadget to configure GoldED according to your hardware. Usually (chunky pixel gadget not checked) GoldED will try to speed up the display by restricting output to single planes - this will give you a considerable speed increase as long as a native plane-based Amiga chipset (e.g. the ECS chipset) is used. But it won't give you a speed increase at all if your external graphics board is based on a chunky pixel organization. In fact write-protecting planes might even slow down output, so switching GoldED to chunky mode might be a good idea if you own such a board. Be careful to have this gadget set properly to avoid loss of performance. Better leave it untouched (unchecked) if you feel unsure about this option.

1.149 config/GUI

config/GUI of:
 CONFIG MENU
 Open requester related to several features of GoldED's user interface: ←

CenterWin

If "center windows" is enabled, window positions are not read from the configuration file. Instead, GoldED attempts to center windows on screen (only visible section considered).

autoArrange

Windows are rearranged after a window has been closed or a new one has been opened if "AutoArrange" is on (compare control/window arrange). If you want to have the windows rearranged after control/next window , too, add a WINDOW ARRANGE=0 command to this menu (menu definition: config/menus).

weight

Used to assign extra space to the current window during window arrangement (e.g. caused by control/window arrange). Choose a weight of two in order to make the current window appear twice as big as the other windows.

margins

Sets a top/right margin to be left free during window arrangement (e.g. caused by control/window arrange). Useful to prevent a (ToolManger-)dock from being covered by text windows (ToolManager is ©1990-1994 Stefan Becker).

fast scrolling

GoldED will speed up scrolling after the cursor has reached the borders of a text window if the FastScroll gadget is checked. You might want to turn this feature off in monochrome mode (provided you own a fast A4000) to slow down the display.

status bar

1.151 config/printer

config/printer of:

CONFIG MENU

Printer definition requester. Used to define printing mode for ←

project/print

as well as

block/print

. Since all output of GoldED goes to

the standard printer driver, this works with any printer (while

misc/HiSpeed

supports PCL printers only). Contents of the 'init' string gadget are sent to the printer after all other initialization (e.g. quality selection) has been done. This gadget may be used to pass printer specific data, too: The gadget's contents are interpreted (see

magic codes

), you may

easily pass a so called aRaw to the printer device: '27 [<bytes> 34 r'. To send a 7-bytes command (e.g. "0123456") to the printer, enter:

```
%27 "[7" %34 "r" "0123456"
```

1.152 config/misc

config/misc of:

CONFIG MENU

Open main preferences requester. This requester is used to set several 'Auto' ←

features as well as the backup handling, default protection bits, fold markers and XPK compression mode (see

project/save as XPK

):

AutoFold AutoLoad

If AutoFold is enabled, the editor looks for fold markers after loading; if some are found, the corresponding sections are folded (see

Folding

);

happens before the text is displayed. Use the fold marker gadgets (start/end) to set the marker sequences for start respectively end of folds. The shorter this sequences are, the faster folding will be. Using the same sequence for marking start/end of sections to fold speeds up folding even more. However, this would prevent you from using nested folding: nested folding requires different markers.

If AutoLoad is enabled, GoldED attempts to load your last project during startup - unless file names are specified.

XPK compression mode

The XPK listview presents a list of available XPK compressors, found within the `libs:compressors` subdirectory. Additionally you may enter a password (used by some XPK encryption libraries) and set efficiency for compression (0% to 100%). Please read the original XPK documentation.

Backup creation: AutoBackup

The editor attempts to backup old copies of a file before saving a new version if 'create *.bak' is enabled (otherwise the old version is overwritten). Backups are written to any path you like (see backup path gadget). Enable AutoSave if you want backups of your windows every x minutes (use gadget below AutoSave to enter period). Toggle 'ask' gadget to ON, if you want to be asked for confirmation of AutoBackup events.

Create *.info

If 'create *.info' is ON, the editor generates an icon for each file saved to disk (unless an old one exists). The default tool of icons created by the editor is GoldED.

File protection bits

These gadgets (read/del/write/script) are used to set the default bits for a new text (e.g. created by `project/more ed`). Use `project/bits` to set the actual bits of a document.

Startup macro

The startup ARexx macro is executed once (asynchronously) after the editor has been launched, windows already open. Have a look at the section about the editor's

ARexx port

as far as ARexx programming is concerned. You could use the macro to customize the GoldED environment according to the type of files loaded during startup (e.g. switch to a C programming environment). Leave the startup gadget empty if you don't need automatic execution of a startup macro.

HotKey support (stay-in-ram gadget)

GoldED supports

HotKey

activation: if hotkey activation is enabled (stay-in-ram gadget checked) the editor will not be removed from memory even after the last window has been closed. Instead it will wait for a hotkey combination (right SHIFT & right ALT & RETURN) before it attempts to reopen its screen again. HotKey activation will give you a quick response time though it will consume some memory, too. You might want to disable this feature if you are short of RAM. Use the `QUIT UNLOAD` command (see

`misc/command`

) to remove the editor completely from RAM (you could use the commodities exchange program of your workbench, too).

overwrite

Existing files are overwritten (e.g. by
Project/Save
) without warnings if
this gadget is checked.

load twice

Disable <load twice> to make GoldED look for existing, ram-resident copies of text files before loading the file from disk (considering frozen buffers without windows, too); you are asked whether you want to use the RAM copy if one is found.

1.153 config/save

config/save of:

CONFIG MENU

Save configuration to a preferences file. Has to be used ↔
after GoldED's

settings have been changed if you don't want to lose your definitions. Default settings file is envarc:GoldEd/GoldED (this file is used during the editor's startup if no other configuration is specified).

1.154 config/load

config/load of:

CONFIG MENU

Load a configuration file & adjust to the new settings. ↔
This operation

includes closing down and reopening all windows; might be a dangerous call if not enough memory is available (i.e. if the editor isn't able to reopen the windows).

1.155 User defined gadgets

User defined gadgets

GoldED offers user-definable gadgets within window titles: useful to gain quick access to often used functions. You may 'leave out' any menu item as gadget (see

config/menus

).

1.156 Keyboard

Keyboard

Please read this chapter carefully if you want to take full advantage of this editor's features. Keys usually perform different tasks depending on what qualifier key(s) is/are pressed simultaneously. Qualifier keys are SHIFT, ALT or CTRL. For example the cursor keys map to seven different functions. Key bindings are not fixed; use

```
config/keyboard
to adjust them to your likings.
```

This manual describes the default setup. The following descriptions are available:

Cursor keys

TAB key

HELP key

ESC key

RETURN key

F-keys

DEL key

1.157 Cursor keys

Cursor keys

Speeds of scrolling

UP/DOWN + ALT

This sequence provides fast scrolling (up or down) - it is one of GoldED's most useful key combinations. The cursor won't move during fast scrolling (i.e. it will stay in the middle of the screen if it was there before you switched to fast scrolling).

UP/DOWN + SHIFT

Go to next (DOWN) or previous (UP) page. Pages do overlap to make navigation more comfortable.

UP/DOWN + CTRL

Fast jump: the cursor moves to the next quarter of your text. Useful to roughly set a new position before using fast/normal scrolling for fine tuning.

LEFT/RIGHT + ALT

Shifts the display area to the left or right. Usually display is shifted automatically if the cursor reaches the right/left window borders. This function is useful if you want to shift the display without moving the cursor at all.

LEFT/RIGHT + CTRL

This is a shifting function: mark same lines using
 block/mark
 , then use

this key combination to shift (indent) the block. Mainly used by programmers to ensure a proper indentation scheme. Usually the marked lines are shifted by one column; use the SHIFT key simultaneously to set shifting distance to TAB distance.

LEFT/RIGHT + SHIFT

Moves cursor to the beginning of the next (RIGHT) or the previous word (LEFT).

LEFT/RIGHT + SHIFT + ALT

Moves cursor to the end of the current/next (RIGHT) respectively previous (LEFT) word.

1.158 HELP key

HELP key

HELP

Fold/unfold current section: Unfold if cursor is placed over a fold header (see

 Folding
), otherwise look for fold markers & fold lines between markers (cursor must be placed between a fold start marker and a fold end marker).

HELP + CTRL

Fold/unfold the whole text: Unfold all folded sections if cursor is placed on a fold header (see

 Folding
), otherwise fold all sections surrounded by fold markers.

1.159 TAB key

TAB key

TAB (+ SHIFT)

Move cursor to next TAB position. This editor supports simple/regular TABs (e.g. every 4th column) as well as fixed TABs (any column you want) and dynamic TAB's. Use

config/tabs

to set mode & TAB positions. Press the SHIFT

key simultaneously if you want to jump to the previous instead of the next tab position (backtab). TABs are either solid (i.e. they behave as if they were a sequence of spaces) or light (i.e. they simply move the cursor without inserting any character); use either

control/toggle tab mode

or

config/tabs

to switch from light to solid and vice versa.

Dynamic TABs

GoldED supports dynamic TAB's (suggested by David Göhler): in dynamic TAB mode (set by

config/TABS

the editor will examine the previous line(s) to

determine appropriate TAB positions. Quite useful if you are about to edit assembler sources. Dynamic TABs default to regular TABs, if the lines above the current line are empty.

TAB + ALT (+ SHIFT)

Usually either distinctive or normal TABs are active. However, you can switch to distinctive tabs on the fly by holding down the ALT key while using TAB or TAB SHIFT.

1.160 RETURN key

RETURN key

RETURN

Split current line at cursor position & move cursor to next line. This editor supports

AutoIndention

: if you press the RETURN key, the current line's

indention is used as default indention for the next line (i.e. if the current

line is indented by four columns, pressing <CR> will move the cursor to the fourth column of the next line).

RETURN + SHIFT

Same as RETURN but the current line is not splitted no matter where the cursor is positioned so far.

RETURN + CTRL

Inserts an empty line: the cursor is not moved at all but a new line is inserted before the current line.

1.161 DEL key

DEL key

DELETE (+ SHIFT)

Delete character at current cursor position, shift rest of line one position to the left. Press the shift key simultaneously if you want to delete until the end of line. You can undo this operation using

misc/undo line

.

DELETE + CTRL

Delete the current line. You can recall up to 50 deleted lines using

misc/line push

since lines are not actually lost but put to the pick/push

buffer.

DELETE + ALT

Delete the current word. You can undo this operation as long as the cursor doesn't leave the current line using

misc/undo line

. Additionally, up to

100 deleted words are put to a pick-push puffer; use DEL-ALT-SHIFT to recall them.

DELETE + ALT + SHIFT

Reinsert previsouly deleted word (see above).

1.162 ESC key

ESC key

Trys to 'complete' the word your cursor is placed over. Example usage: type 'TIG', then press the ESC key. 'TIG' would be replaced by 'TAG_IGNORE' if the C-dictionary is present (see

config/dictionary
on how to load/edit/create a

dictionary). GoldED uses a (simple) pattern matching algorithm to find appropriate dictionary entries, so you might use other abbreviations than 'TIG', too (e.g. 'TAGI'). This will work as long as the first letter of the short form is the first letter of the full form. However, the larger your dictionary grows the more detailed your abbreviations have to be to ensure unique identification.

1.163 F-Keys

F-Keys

Use the

config/keyboard
menu to assign strings, shell commands, ARexx
macros or one of the editor's
internal commands
to any key including the

function keys. However, some commodities do their own function key mapping. If such a commodity is installed, the editor won't notice function key events. Default mappings of the function keys are:

key	decription	see
F1	open file project/open	
	F2	merge file
	project/insert	
	F3	print file
	project/print	
	F4	hide block
	block/hide mark	
	F5	mark line
	block/mark	
	F6	find next
	find/find next	
	F7	next page
	cursor keys	
	F8	set insert mode
	control/insert	
	F9	toggle TAB mode
	control/toggle TAB mode	
	F10	play macro
	macros/sequence play	

	SHIFT	
& key	decription	see

F1	save as	
	project/save as	
	F2	save as XPK
	project/save as XPK	
	F3	quit
	project/quit (window)	
	F4	clear text
	project/clear text	
	F5	mark line
	block/mark	
	F6	replace
	find/replace	
	F7	page up
	cursor keys	
	F8	overwrite mode
	control/insert	
	F9	project setup
	misc/source files	
	F10	record macro
	macros/sequence record	

1.164 ARexx port

ARexx port

ARexx macros vs. recorded sequences

GoldED offers two kinds of macros for automated control: ARexx scripts and recorded sequences. ARexx scripts are programs quite similar to programming languages like BASIC. They are evoked by GoldED (e.g. by setting up a menu item of type 'ARexx'; see

config/menus

), but actually executed by the ARexx

master server (part of the Amiga operating system). The ARexx server will examine the script and send messages to GoldED as well as receive messages from GoldED during execution. For example the ARexx server could ask GoldED to jump to a special line if it detects a GOTO command within a script. GoldED would tell the ARexx server whether the operation has been successful. Due to the flexibility of ARexx this is a very powerful method to automate control of GoldED. However this approach requires at least some basic knowledge of ARexx. If you need automated control but are not interested in ARexx you might want to use GoldED's ability to record command sequences instead: Enable recording using

macros/sequence record

and perform a

sequence of commands. In other words: make the editor 'learn' how to do it. Once you have recorded a sequence you may replay it as many times as you like (see

macros/sequence play

). You may save sequences

macros/sequence save

)
 or assign them to events like menus or keystrokes using GoldED's
 MACRO
 command. Sequences are far less powerful than ARexx script. But ↔
 they are far
 more handy, too.

This section describes the editor's ARexx interface. You are expected to be familiar with ARexx basics, i.e. you should know about the purpose of ARexx, how to write scripts, how to talk to applications, ...

ARexx basics

ARexx control of this editor is somewhat complicated since you never know how many editor tasks are running, how many windows are open or what the user is doing when ARexx wants to take over control. It is therefore quite important to obey to certain rules which are to be discussed now:

1.
Select a host
2.
Lock a window
3.
Do your job
4.
unlock GUI

1.165 Select a host

Select a host

If you run ARexx scripts from within the editor (i.e. if you set up menu items of type 'ARexx' or if you execute the current text as macro using

macros/run text as macro

), any script commands which are not part of ARexx itself are sent to the editor (the 'host') automatically. However, if your script is evoked from a different program (e.g. rx), it will have to select a port for communication: use ADDRESS <port name> for this purpose.

Port name

The editor's ARexx port is called "GOLDED.1" if the editor is run only once. The ARexx port of a further editor task would be "GOLDED.2". Select

project/about

if you want to know the current port/screen name. Or use the

QUERY

command (with the HOST argument) from within a script. As long as ↔
 you

use the

QuickStarter

to run GoldED, you usually won't have to deal with

port names different from 'GOLDED.1'.

1.166 Lock a window

Lock a window

Your script has to tell the editor what window is going to be affected - use the

LOCK

command (e.g. 'LOCK CURRENT' to lock the current window). After locking a window, the GUI is locked, too, to prevent the user from disturbing the macro. If you don't use the lock command, your macro might still appear to work perfectly, but it will break under special circumstances (e.g. if the user closes a window while a script is executed). Once you have locked a window successfully (ARexx return code RC is 0), you have to take care of unlocking it again on termination of your script (see

Unlock GUI
).

1.167 Do your Job

Do your Job

You can use any of the editor's

internal commands

within your macro. Please

keep in mind that commands send from ARexx to GoldED are parsed twice: first by the ARexx server while executing the script, second by GoldED using the ReadArgs() function of the dos library. This sometimes screws things up a bit - especially as far as quotes are concerned. Suggestion: Write the lines of your script as if you were talking directly to GoldED: quote strings, command names uppercase (step 1). Then put the lines to be sent to GoldED into single quotation marks to mark them as commands (step 2). Finally double single quotation marks within these lines to prevent ARexx from regarding them as string delimiters (step 3). Example:

```
step 1: REQUEST BODY "Hi, I'm an empty macro"
step 2: 'REQUEST BODY "Hi, I'm an empty macro"'
step 3: 'REQUEST BODY "Hi, I''m an empty macro"'
```

Usually GoldED passes command results to your script using the special ARexx variable RESULT - at least if you have asked for results using OPTIONS RESULTS. Some commands like

QUERY

support specification of a variable name, too, using the VAR/K option. Example: 'QUERY ABSLINE VAR LINE'

No result is returned if a command fails - instead the special variable RC is

set to the error level: 5 = warning, 10 = error, 20 = fatal error. RC would be 0 if a command has been successful. You have to use the OPTION FAILAT command of ARexx to prevent ARexx from stopping execution if RC is not NULL, i.e. to receive RC return codes at all. The special variable RC2 will keep an error text if a command has failed (i.e. if RC is not 0).

1.168 Unlock GUI

Unlock GUI

An ARexx script must
 unlock
 the GUI before it terminates, if a prior call
 to
 Lock
 (see
 Lock a window
) has been successful. It mustn't use unlock if
 a prior attempt to lock has failed. Omitting unlock will leave the editor dead-locked, so take care to unlock the GUI even if your script breaks (maybe due to a syntax error). This can be achieved using the error handling facilities of ARexx (e.g. SIGNAL or OPTION FAILAT). Just have a look at the scripts in the GoldED:ARexx drawer. As a last resort the Unlock macro is provided: simple doubleclick at its icon - all editors will be unlocked (which is a dangerous operation if one of those tasks is processing an ARexx script).

1.169 Internal commands

Internal commands

GoldED offers a set of about 420 commands/options (see
 Command list
),
 supported by all interfaces of GoldED: you may use them in ARexx macros, bind
 them to menu items (see
 config/menus
), attach them to keys
 (
 config/keyboard
) or execute them directly using
 misc/command
 . It is
 possible to combine several functions (see
 multiple commands
). As far as
 arguments are concerned, the DOS rules apply since GoldED uses the ReadArgs
 function of OS2.0 just like most CLI commands: strings containing spaces must

be quoted, option and keywords may be uppercase or lowercase. Command templates/options are described in the same way as CLI commands are described. Example:

```
PRINT FORCE/S,ITALICS/K,ALL/S,LPI/N,CONFIG/K
```

This PRINT command obviously accepts five options: force, italics, all, lpi and config. The option force is a switch (/S): it makes the print command behave in a special way described in this manual *_if_* this option is specified. The second option introduces a keyword (/K) - value pair; e.g. print italics=true. The equality sign may be omitted. Due to ReadArgs() parsing, you will have to use "*" instead of " when embedding quotation marks into strings to prevent the parser from considering a quote as start/end of a string (** results in a single *). Options of type '/K' - just like any other options apart from /A ones - do not have to be specified. If they are specified, a further argument (like true) is expected. Supported arguments depend on the command: If one of the commands below offers an option described as BOOL, it would accept the strings true, false and probably toggle, too. If a command's option is marked as STRING, any text string is accepted as argument (e.g. print config "S:prt.prefs"). The 4th keyword in the example above (lpi) introduces a numerical (/N) argument; example: print lpi=10. The equality sign may be omitted once more. The valid argument range depends on the command (e.g. byte, word, unsigned word or long). Further option types are "\F" (accepts rest of line as string), "\M" (accepts multiple strings) and "\A" (means: this argument must be specified).

1.170 Command list

```
Command list (use: see
internal commands
):
```

API

DPAGE

HELP

MORE

PUSH

TABS

BACK

ENDWORD

HUNTER

MOUSE

QUERY

TASK
BEEP
EXALL
INDENT
NAME
QUIT
TEXT
BIND
EXTRACT
INFO
NEW
REFRESH
TMPLATE
BITS
FDOWN
INSERT
NEXT
REMAP
UJUMP
BLOCK
FILE
KEY
NOTIFY
REPLACE
UNDO
BRACKET
FIND
LAYOUT

OPEN
REQUEST
UNLOCK
CLIP
FIRST
LEFT
PATH
RIGHT
UP
CMD
FIX
LINES
PHRASE
RUN
UPAGE
CODE
FOLD
LOCK
PING
RX
USE
COLON
FORMAT
MACRO
PONG
SAVE
VIEW
CR
FREEZE

MARK
POP
SCREEN
VLEFT
DEL
FUNC
MAXDOWN
PREFS
SET
VRIGHT
DELETE
FUP
MAXUP
PREV
SHIFT
WINDOW
DIR
GOTO
MENUS
PREVEND
SMARTCR
XREF
DJUMP
GREP
MISC
PRINT
SUFFIX
DOWN

GUI
 MODE
 PROJECT
 TAB

1.171 API

API

Description of

internal commands			
	command	option	description
API	ASK/S	open config/API preferences window	
	ADD/K	API client to be launched (STRING)	
	FIND/K	check whether this client is running (STRING)	
	PORT/N	add client's reply port (struct MsgPort *)	
	MASK/N	notify mask (ULONG)	
	CONFIG/K	name of a preset file (STRING)	
	LOAD/S	load preset file	
	SAVE/S	save preset file	

Comment: PORT/N and MASK/N are reserved for use by external API clients (experienced programmers only). API documentation and API examples are available in the GoldED:API drawer.

1.172 BACK

BACK

Description of

internal commands			
	command	option	description
BACK	(no options)	backspace operation	
	SMART/S	backspace over marked word will delete word	

1.173 BEEP

BEEP

Description of

internal commands			
	command	option	description

BEEP	(no options)		audible beep

1.174 BIND

BIND

Description of

internal commands			
	command	option	description

BIND	ASK/S		open keyboard requester (command assignment)
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

1.175 BITS

BITS

Description of

internal commands			
	command	option	description

BITS	ASK/S		open a requester to edit protection bits/comment
	R/K		set readable bit (BOOL)
	W/K		set writeable bit (BOOL)
	D/K		set deletable bit (BOOL)
	S/K		set script bit (BOOL)
	COMMENT/K		set comment (STRING)

1.176 BLOCK

BLOCK

Description of

internal commands			
	command	option	description

BLOCK	UPPER/S		make block uppercase

LOWER/S	make block lowercase
SORT/S	sort block
COPY/S	copy block to cursor position
MOVE/S	move block to cursor position
HIDE/S	hide marker after operation

1.177 BRACKET

BRACKET

Description of

		internal commands	
	command	option	description
BRACKET	MATCH/S		move cursor to matching bracket
	CHECK/S		check use of () in current line
	TWINS/K		bracket type (STRING, default: "()")

1.178 CLIP

CLIP

Description of

		internal commands	
	command	option	description
CLIP	CUT/S		move block to clipboard
	COPY/S		copy block to clipboard
	PASTE/S		insert clipboard contents at cursor position
	VPASTE/S		vertical clipboard paste
	UNIT/N		clipboard unit to use (UBYTE); defaults to 0

1.179 CMD

CMD

Description of

		internal commands	
	command	option	description
CMD	(no options)		open command requester

1.180 CODE

CODE

Description of

	internal commands		
	command	option	description
CODE	SHOW/S		show ASCII code of character under cursor
	SET/N		insert code (UBYTE)
	ASK/S		ask for ASCII code to be inserted
	TABLE/S		open character set table requester
	TOGGLE/S		change case of character under cursor

Comment: The SET option is influenced by current writing mode: in insert mode a character is inserted, in overwrite mode the character under the cursor is overwritten.

1.181 COLON

COLON

Description of

	internal commands		
	command	option	description
COLON	(no options)		insert semicolon and possibly a CR (return)

Comment: Suggested use is mapping to the ;-Key. Useful for C/C++ programmers. The editor tries to figure out whether a CR should be inserted (e.g. no CR is inserted if the semicolon is part of a 'for' statement). Press CTRL simultaneously to disable CR insertion temporarily.

1.182 CR

CR

Description of

	internal commands		
	command	option	description
CR	(no options)		<RETURN> command; splits line at cursor position

Comment: This function is influenced by the current setup (e.g. by the indention mode settings).

1.183 DEL

DEL

Description of

internal commands		
command	option	description
DEL	(no options)	deletes character under cursor

1.184 DELETE

DELETE

Description of

internal commands		
command	option	description
DELETE	WORD/S	delete next word
	EOW/S	delete until end of word
	SMART/S	consider white space settings
	EOL/S	delete until end of line
	LEFT/S	delete until beginning of line
	LINE/S	delete current line
	BLOCK/S	delete block
	COLUMN/S	delete column (see AT/N) from block
	AT/N	column to be deleted (UWORD); defaults to current

Comment: the last 100 deleted words (WORD/S) may be reinserted using

```
INSERT
(ININSERT WORD).
```

1.185 DIR

DIR

Description of

internal commands		
command	option	description
DIR	ASK/S	open requester to set 'current directory'
	NEW/F	set 'current directory' (STRING)

Comment: the current directory is passed to any program run by GoldED (e.g. a shell using misc/shell

). It is used by many of GoldED's internal functions, too.

1.186 DJUMP

DJUMP

Description of

internal commands	command	option	description
-------------------	---------	--------	-------------

DJUMP	(no options)		jump to end of screen / next page
-------	--------------	--	-----------------------------------

Comment: Cursor jumps to last line of screen if placed above that line so far. Jumps to next page otherwise. Compare:

DPAGE

.

1.187 DOWN

DOWN

Description of

internal commands	command	option	description
-------------------	---------	--------	-------------

DOWN	(no options)		move cursor one line down
------	--------------	--	---------------------------

1.188 DPAGE

DPAGE

Description of

internal commands	command	option	description
-------------------	---------	--------	-------------

DPAGE	(no options)		show next page (compare: DJUMP)
-------	--------------	--	--

1.189 ENDWORD

ENDWORD

Description of

internal commands		
command	option	description

ENDWORD	(no options)	move cursor to end of word
---------	--------------	----------------------------

1.190 EXALL

EXALL

Description of

internal commands		
command	option	description

EXALL	(no options)	Examine text
-------	--------------	--------------

Comment: To be used within ARexx macros only. Used to update variables related to text statistics (see

QUERY

).

1.191 EXTRACT

EXTRACT

Description of

internal commands		
command	option	description

EXTRACT	(no options)	Extract file name under cursor
	VAR/K	where to put the result: ARexx variable (STRING)
	LEFT/K	left delimiter(s) (STRING); e.g. "<[("
	RIGHT/K	right delimiter(s) (STRING); e.g. ">]"

Comment: left & right delimiter strings must be of paired and of the same length. Priority is from left to right.

1.192 FDOWN

FDOWN

Description of

internal commands

	command	option	description
FDOWN	(no options)		scroll down in fast mode

1.193 FILE

FILE

Description of

	internal commands		
	command	option	description
FILE	NAME/K		file to delete/search (STRING)
	DELETE/S		delete file
	FORCE/S		don't ask for confirmation
	SEARCH/K		search this path for specified file (STRING)
	VAR/K		where to put the result: ARexx variable (STRING)
	NEWDIR/K		create directory (STRING)

Comment: Delete-protected files are not deleted unless the FORCE mode is used.

1.194 FIND

FIND

Description of

	internal commands		
	command	option	description
FIND	STRING/K		pattern to search for (STRING)
	COUNT/S		count pattern (doesn't affect cursor position)
	PREV/S		jump to previous occurrence
	NEXT/S		jump to next occurrence
	FIRST/S		jump to first occurrence
	ASK/S		open requester
	CASE/K		case (in)sensitive (BOOL)
	QUIET/S		don't complain about missing pattern ('not found')
	WORDS/K		look for whole words only ? (BOOL)

1.195 FIRST

FIRST

Description of

internal commands

	command	option	description
FIRST	(no options)		move to beginning of line (see GOTO)

1.196 FIX

FIX

Beschreibung für

internal commands

Kommando Option Beschreibung

FIX	VAR/K/A		ARexx variable name (STRING)
-----	---------	--	------------------------------

Comment: To be used within macros only. Fixes the contents of the given ARexx string variable to make it "parser-proof" (e.g. handles embedded '''); compare internal commands).

1.197 FOLD

FOLD

Description of

internal commands

command option description

FOLD	OPEN/K		open fold or (ALL/S specified) folds (BOOL)
	ALL/S		consider all lines
	TOGGLE/S		toggle fold (open/close)

1.198 FORMAT

FORMAT

Description of

internal commands

command option description

FORMAT	LINES/S		select current paragraphe for formatting
	MARK/S		select block for formatting
	LEFT/S		make selected area left -aligned
	RIGHT/S		make selected area right-aligned

BLOCK/S	make selected area block-aligned
CENTER/S	center selected area

1.199 FREEZE

FREEZE

Description of

		internal commands		
		command	option	description
FREEZE	CURRENT/S			freeze current window
	ASK/S			ask for text to unfreeze
	SWAP/S			swap current/frozen window
	ADD/M			load file(s) directly to frozen list

1.200 FUNC

FUNC

Description of

		internal commands		
		command	option	description
FUNC	C/S			set mode to C
	BASIC/S			set mode to BASIC
	A68K/S			set mode to Assembler
	PASCAL/S			set mode to Pascal
	AUTODOC/S			set mode to AutoDoc
	STRUCT/S			set mode to C-Header
	SMART/S			automatic mode setting according to file name
	CURRENT/S			extract function name from text (below cursor)
	UNFOLD/K			examine folds (BOOL)

Comment: scans text for structures, functions, ... (depending on selected mode) to make up an index. If no mode is specified (and SMART is not used) the current mode is used.

1.201 FUP

FUP

Description of

		internal commands		
		command	option	description
FUP	(no options)			scroll upwards in fast mode

1.202 GOTO

GOTO

Description of

internal commands			
	command	option	description

GOTO	LINE/N		line number to go to (ULONG: 1, ...)
	COLUMN/N		column to go to (UWORD: 1, ...)
	UNFOLD/K		unfold if necessary ? (BOOL)
	TOP/S		go to first line of text
	BOTTOM/S		go to last line of text
	OTHEREND/S		toggle position
	CHANGE/S		go to last change
	ASK/S		ask for line number to go to
	EOL/S		place cursor over last character of line
	BFIRST/S		go to beginning of block
	BLAST/S		go to end of block
	STEP/N		move cursor left/right (WORD)
	TOF/S		move cursor to first line of screen
	BOF/S		move cursor to last line of screen
	BYTE/N		byte offset to go to (LF's included)
	<p>Comment: line numbers are considered to be absolute if UNFOLD is enabled (folded blocks would count as single line otherwise).</p>		

1.203 GREP

GREP

Description of

internal commands			
	command	option	description

GREP	STRING/K		string to search project files for (STRING)
	ASK/S		ask for string to search for
	CASE/K		case sensitive search ? (BOOL)

1.204 GUI

GUI

Description of

internal commands

	command	option	description
GUI	ASK/S		open GUI configuration window
	CENTER/K		center windows ? (BOOL)
	X/N		vertical scroll border (UWORD: 0, ...)
	Y/N		horizontal scroll border (UWORD: 0, ...)
	OVERWRITE/K		overwrite files without warning ? (BOOL)
	ARRANGE/K		AutoArrange windows ? (BOOL)
	WEIGHT/N		arrange windows: window weight (UWORD 1...4)
	CLOCK/K		clock ? (BOOL)
	FAST/K		fast scrolling ? (BOOL)
	TINYMSG/K		use status bar instead of requesters ? (BOOL)
	SPC/K		white space characters (STRING)
	DATE/K		date format (STRING); requires OS3.0+
	REVERSED/K		(not supported)
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

Comment: the list of white space characters (SPC) may consist of ASCII codes, code ranges or strings, separated by colons. Example: 0-" ",128-160,".,;()". You'll have to ensure that quotation marks actually reach GoldED; see internal commands

The date format string may consist of the following formatting codes (besides normal characters):

```
%a - abbreviated weekday name
%A - weekday name
%b - abbreviated month name
%B - month name
%d - day number with leading 0s
%D - same as "%m/%d/%y"
%e - day number with leading spaces
%j - julian date
%m - month number with leading 0s
%U - week number, taking Sunday as first day of week
%W - week number, taking Monday as first day of week
%w - weekday number
%x - same as "%m/%d/%y"
%y - year (two digits)
%Y - year (four digits)
```

1.205 HELP

HELP

Description of

	command	option	description
HELP	CATALOG/K		set database (STRING)

TOPIC/K node to look for (STRING)

Comment: if no database (i.e. help file) is specified, the menu's database is used (see config/menus).

1.206 HUNTER

HUNTER

Description of

	internal commands		
	command	option	description
HUNTER	ASK/S		open configuration window of file hunter
	CURRENT/S		hunt (i.e. search & open) filename under cursor
	NAME/K		hunt this file (STRING)
	DEEP/K		scan subdirectories ? (BOOL)
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

1.207 INDENT

INDENT

Description of

	internal commands		
	command	option	description
INDENT	ASK/S		open indentation requester
	AUTO/K		set automatic indentation (BOOL)
	SMART/K		set smart indentation (BOOL)
	IN/K		add keyword for smart indentation (STRING)
	OUT/K		add keyword for smart outdentation (STRING)
	CLR/S		clear smart indentation keyword list
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

1.208 INFO

INFO

Description of

	internal commands		
	command	option	description

INFO	VERSION/S	show version
	USER/S	show copyright requester
	TEXT/S	show statistics
	ERROR/S	show last error

1.209 INSERT

INSERT

Description of

		internal commands	
	command	option	description
INSERT	LINE/S		insert a line
	BLOCK/S		insert into block (see the following options)
	COLUMN/S	BLOCK/S:	insert empty column; see AT/N
	STRING/K	BLOCK/S:	insert text; see AT/N (STRING)
	AT/N	BLOCK/S:	column where to insert (UWORD)
	APPEND/S	BLOCK/S:	append text to marked lines
	WORD/S		reinsert deleted word (see
	DELETE)
	PATH/S		ask user for file name to insert

1.210 KEY

KEY

Description of

		internal commands	
	command	option	description
KEY	EVENT/K		input event description (STRING)
	RAW/S		event is a plain character sequence

Comment: see
input events
for details

1.211 LAYOUT

LAYOUT

Description of

		internal commands	
	command	option	description

LAYOUT	LEFT/N	set left margin for formatting (UWORD)
	RIGHT/N	set right margin for formatting (UWORD)
	WRAP/K	set WordWrap (BOOL)
	ASK/S	open requester to set layout
	AUTO/K	use current indentation as left border (BOOL)
	REFORMAT/K	reformat during WordWrap ? (BOOL)
	CONFIG/K	name of a preset file (STRING)
	LOAD/S	load preset file
	SAVE/S	save preset file

1.212 LEFT

LEFT

Description of

internal commands		
command	option	description

LEFT	(no options)	move cursor one position to the left
------	--------------	--------------------------------------

1.213 LINES

LINES

Description of

internal commands		
command	option	description

LINES	JOIN/S	join current line & next line
	SWAP/S	swap current line <-> next line
	DOUBLE/S	double current line

1.214 LOCK

LOCK

Description of

internal commands		
command	option	description

LOCK	CURRENT/S	lock current window
	NAME/K	window to lock (STRING)
	QUIET/S	don't activate window

Comment: To be used within ARexx macros. An UNLOCK command must follow later on to prevent deadlocks (ensure a clean exit even after errors using the SIGNAL/OPTION FAILAT commands of ARexx). Exit your script if the LOCK call fails (error code RC=20). This call doesn't nest: a single UNLOCK unlocks any number of locks. This call moves GoldED's screen to the front.

1.215 MACRO

MACRO

Description of

internal commands		
command	option	description
MACRO	RECORD/S	toggle sequence recording mode
	PLAY/S	play previously recorded sequence
	LOOPS/N	number of playback loops (UWORD); defaults to 1
	ASK/S	ask for number of loops
	FILE/K	sequence file to load/write (STRING)
	SAVE/S	save previously recorded sequence
	LOAD/S	load a sequence

1.216 MARK

MARK

Description of

internal commands		
command	option	description
MARK	HIDE/S	hide mark
	SET/S	set mark
	LINE/S	resolution = lines
	COLUMN/S	resolution = characters
	WORD/S	mark word under cursor
	STRICT/S	only SPC (ASCII 32) is regarded as word delimiter

1.217 MAXDOWN

MAXDOWN

Description of

internal commands		
command	option	description
MAXDOWN	(no options)	move to next quarter of document

1.218 MAXUP

MAXUP

Description of

internal commands		
command	option	description
MAXUP	(no options)	move to previous quarter of document

1.219 MENUS

MENUS

Description of

internal commands		
command	option	description
MENUS	ASK/S	open menu requester
	CONFIG/K	name of a preset file (STRING)
	LOAD/S	load preset file
	SAVE/S	save preset file

1.220 MISC

MISC

Description of

internal commands		
command	option	description
MISC	ASK/S	open misc preferences requester
	AUTOBAK/K	set AutoBak mode (BOOL)
	PERIOD/N	set backup interval (UWORD); units are minutes.
	CONFIRM/K	set confirm-backup mode (BOOL)
	PATH/K	set backup path (STRING)
	INFOS/K	set creation of info files (BOOL)
	AUTOLOAD/K	set AutoLoad mode (BOOL)
	AUTOFOLD/K	set AutoFold mode (BOOL)
	FOLDSTART/K	set fold-start marker (STRING)

FOLDEND/K	set fold-end marker (STRING)
BACKUP/K	set backup creation (BOOL)
CONFIG/K	name of a preset file (STRING)
LOAD/S	load preset file
SAVE/S	save preset file

1.221 MODE

MODE

Description of

	internal commands		
	command	option	description
MODE	INSERT/K		set insert/overwrite mode (BOOL)
	AUTOCASE/K		set AutoCase mode (BOOL)
	AUTOBRACKET/K		set automatic parenthesis check (BOOL)
	NUMPAD/K		enable/disable NumPad assignments (BOOL)

1.222 MORE

MORE

Description of

	internal commands		
	command	option	description
MORE	(no options)		open further window
	SMART/S		open window if current window is not empty

1.223 MOUSE

MOUSE

Description of

	internal commands		
	command	option	description
MOUSE	ASK/S		open mouse preferences requester
	SET/S		move cursor, mark block using the mouse
	LINE/S		mark whole lines only
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

Command: SET/S and LINE/S reserved for mouse configuration

1.224 NAME

NAME

Description of

	internal commands		
	command	option	description
NAME	ASK/S		ask for a new text name
	NEW/F		set new text name (STRING)

1.225 NEW

NEW

Description of

	internal commands		
	command	option	description
NEW	(no option)		clear text; user is asked for confirmation
	FORCE/S		clear text at any rate
	NONAME/S		reset name to 'unnamed'

1.226 NEXT

NEXT

Description of

	internal commands		
	command	option	description
NEXT	(no options)		move cursor to next word within current line

1.227 NOTIFY

NOTIFY

Description of

	internal commands		
	command	option	description
NOTIFY	FILE/K/A		file to be monitored (STRING)
	START/S		start monitoring
	STOP/S		stop monitoring
	CHECK/S		number of write accesses since last check (UWORD)

MACRO/K macro to be executed upon write access (STRING)

Comment: Provides access to the notification mechanism of AmigaDOS. The file name is passed as argument to the macro.

1.228 OPEN

OPEN

Description of

	internal commands		
	command	option	description
OPEN	NAME/M		file(s) to open (STRING or STRINGS)
	FAST/S		use fast loading (no TAB substitution)
	NEW/S		open new window for each file
	AGAIN/S		reload current file
	APPEND/S		append file(s) to current text
	INSERT/S		insert file(s) into current text
	ASK/S		ask for file(s)
	QUIET/S		don't complain about missing files
	PATH/K		default path to be used by file requester (STRING)
	OLDPATH/S		use path of current text as default path
	SMART/S		use current window unless window is not empty
	RAW/S		don't convert TABs to spaces

Comment: returns window handle in ARexx mode. The window handle may be used to activate a window later on (see window).

1.229 PATH

PATH

Description of

	internal commands		
	command	option	description
PATH	ASK/S		open requester to set reference file(s)
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

1.230 PHRASE

PHRASE

Description of

		internal commands		
		command	option	description

PHRASE	CURRENT/S			try to complete current word
	ASK/S			open dictionary requester
	ADD/K			add keyword to dictionary (STRING)
	CLR/S			clear dictionary
	CONFIG/K			name of a preset file (STRING)
	LOAD/S			load preset file
	SAVE/S			save preset file

1.231 PING

PING

Description of

		internal commands		
		command	option	description

PING	SLOT/N			write cursor position to named slot
<p>Comment: GoldED offers ten slots ('bookmarks', 0-9) for each window to be recalled by</p>				
<p>PONG</p>				
<p>. Slot 0 usually is reserved for use within ARexx scripts.</p>				

1.232 PONG

PONG

Description of

		internal commands		
		command	option	description

PONG	SLOT/A/N			recall one of 10 bookmark positions (UWORD 0-9)
<p>Comment: GoldED offers ten bookmarks (0-9) for each window. Slot 0 usually is reserved for use within ARexx scripts (see</p>				
<p>PING</p>				
<p>).</p>				

1.233 POP

POP

Description of

	internal commands		
	command	option	description
POP	(no options)		move line from text to pick/push buffer

Comment: the pick/push buffer can hold up to 50 entries (lines). It is a last-in-first-out buffer.

1.234 PREFS

PREFS

Description of

	internal commands		
	command	option	description
PREFS	CONFIG/K		name of a preferences file (STRING)
	LOAD/S		load preferences file
	SAVE/S		save preferences file
	SPLIT/K		split preferences file: output prefix (STRING)

Comment: SPLIT - splits the current configuration into several files (menu definition file, dictionary, ...). Mainly used during development of GoldED.

1.235 PREV

PREV

Description of

	internal commands		
	command	option	description
PREV	(no options)		move cursor to previous word

1.236 PREVEND

PREVEND

Description of

internal commands		
command	option	description

PREVEND	(no options)	move cursor to end of previous word
---------	--------------	-------------------------------------

1.237 PRINT

PRINT

Description of

internal commands		
command	option	description

PRINT	FORCE/S	don't ask for confirmation
	BLOCK/S	print block
	ALL/S	print complete file
	LPI/N	set lines per inch (UWORD): 0=6 lpi, 1=8 lpi
	LQ/K	set letter quality (BOOL)
	ITALICS/K	set italics printing (BOOL)
	PROP/K	set proportional mode (BOOL)
	DOUBLE/K	set double width mode (BOOL)
	RESET/K	reset printer before output (BOOL)
	ASK/S	open printer configuration requester
	CONFIG/K	name of a preset file (STRING)
	LOAD/S	load preset file
	SAVE/S	save preset file

1.238 PROJECT

PROJECT

Description of

internal commands		
command	option	description

PROJECT	ASK/S	open project requester
	ADD/K	add source file (STRING)
	DEL/N	remove a source file (UWORD: 0, ...)
	CLR/S	clear list of source files
	LIST/N	set list (struct List *)
	CONFIG/K	name of a preset file (STRING)
	LOAD/S	load preset file
	SAVE/S	save preset file

1.239 PUSH

PUSH

Description of

internal commands			
	command	option	description

PUSH	(no options)		insert last line of pick/push buffer into text

1.240 QUERY

QUERY

Description of

internal commands			
	command	option	description

QUERY	NAME/M VAR/K		variable(s) you are interested in (STRING) ARexx variable; where to put the result (STRING).

Comment: Used to query the state of one or more of GoldED's internal variable(s). This function may be used in interactive mode (see

misc/command

). In interactive mode a requester is

used to display the result including the variable name (e.g. LINES=123). If evoked from a script file (after a

LOCK

command or if the VAR/K option is used)), the ←
variable name

will not be part of the result. In interactive mode it will. In ARexx mode you may specify the name of an ARexx variable to put the result into (defaults to RESULT). It is possible to combine several options (e.g. QUERY DIR DOC); the results will be separated by spaces, too. Valid variable names are:

ABAK..... AutoBackups enabled ? (BOOL)
 ABSLINE..... current absolute line number (ULONG)
 ABSLINES..... absolute number of lines (ULONG)
 ACENTER..... CenterWin mode (BOOL)
 AFOLD..... AutoFold mode (BOOL)
 ALEFT..... Layout: use old border ? (BOOL)
 ALOAD..... AutoLoad mode (BOOL)
 ANSI..... number of non-ASCII characters (*)
 ANYCHAR current line not empty ? (BOOL)
 ANYFOLDS..... does text contain folds ? (BOOL)
 ANYTEXT..... any text in current window ? (BOOL)
 ASKBAK..... ask for backups ? (BOOL)
 BACKUP..... create backups ? (BOOL)
 BAKDIR..... backup path (STRING)
 BITS..... default protection bits (ULONG)
 BLOCK..... marker type (UWORD): 0=none 1=lines 2=characters
 BLOCKX..... block start column (UWORD: 1, ...)

BLOCKY..... block start line (ULONG: 1, ...)
BLOCKR..... block end column (UWORD: 1, ...)
BLOCKB..... block end line (ULONG: 1, ...)
BOLD..... bold mode used for printing (BOOL)
BRACKET..... automatic parenthesis check ? (BOOL)
BUFFER..... current line (STRING)
BYTES..... text size (number of bytes) (ULONG) (*)
CAT..... catalog name/language (STRING)
CHKCASE..... automatic case check ? (BOOL)
CODE..... ASCII code of character under cursor (UBYTE)
COLUMN..... current column (UWORD: 1, ...)
COLUMNS..... window width (UWORD)
CON..... window dimension string (STRING)
CURRENT..... pointer to memory area of current line (char *)
DIR..... path used by file requester (STRING)
DOC..... window title = file name (STRING)
DOUBLE..... use doublestrike printing ? (BOOL)
DTABS..... use distinctive TABs ? (BOOL)
ERR..... last error (STRING)
FILE..... name of current text without path (STRING)
FIND..... search pattern (STRING)
FOLDA..... fold start marker (STRING)
FOLDB..... fold end marker (STRING)
FOLDS..... number of folds in text (ULONG) (*)
FONTX..... width of text font (UWORD)
FONTY..... height of text font (UWORD)
FUNC..... mode of QuickFunc list (UWORD: 0, ...)
HANDLE..... window handle of current window (ULONG)
HMI..... horizontal motion index/printer (UWORD)
HOST..... name of ARexx port (STRING)
INBLOCK..... cursor within block (BOOL)
INDENT..... AutoIndention ? (BOOL)
INFOS..... create *.info files ? (BOOL)
INSERT..... insert mode used ? (BOOL)
ITALICS..... italics printing ? (BOOL)
LEFT..... layout: left margin (UWORD)
LEN..... length of current line
LINE..... number of current line; not absolute (ULONG)
LINES..... number of lines; not absolute (ULONG)
LPI..... lines per inch (UWORD: 0 = 6lpi, 1 = 8lpi)
LQ..... letter quality printing ? (BOOL)
MARKED..... any block marked ? (BOOL)
MAXLEN..... length of longest line (UWORD) (*)
MODIFY..... has text been modified ? (BOOL)
PATH..... path of current text (STRING)
PICKED..... number of lines in pick/push buffer (UWORD)
PREVIEW..... current window: preview mode used ? (BOOL)
PROG..... program's name - usually GoldED (STRING)
PRJLIST..... list of project files (struct List *)
READONLY..... is window read-only ? (BOOL)
REM..... (file-)comment of current text (STRING)
REMAP..... character translation file (STRING)
RESET..... reset printer before output ? (BOOL)
RIGHT..... layout: right margin (UWORD)
ROWS..... window height (UWORD)
RPLC..... replace text (STRING)
SCREEN..... screen name (STRING)

```

SCREENW..... screen width (UWORD)
SCREENH..... screen height (UWORD)
SCRMODE..... screen mode ID (ULONG)
SCRTYPE..... screen type: public/custom (UWORD)
SHANGHAI..... shanghai mode set ? (BOOL)
SINDENT..... smart indention enabled ? (BOOL)
SOLID..... solid TABs ? (BOOL)
SPC ..... white space characters (STRING); see
    config/GUI
        STDLINE..... is current line a 'normal' line ? ( ←
            BOOL)
TAB..... TAB distance keyboard (UWORD)
TABFILE..... TAB distance for TAB substitution (UWORD)
TIMER..... backup interval - minutes (UWORD)
TOPLINE..... line number of window's first line (ULONG)
UNFOLD..... unfold during GOTO ? (BOOL)
USECASE..... case sensitive search/replace ? (BOOL)
USER..... user name (STRING)
USER1-USER20.. user variable 1-20 (BOOL)
VER..... version string (STRING)
VERSION..... version ID code (ULONG)
WINDOWS..... number of open text windows (UWORD)
WINH..... window height (UWORD)
WINW..... window width (UWORD)
WORDS..... number of words (ULONG) (*)
WORD..... current word (STRING)
WRAP..... word wrap mode used ? (BOOL)
X..... window's left edge (UWORD)
XPK..... XPK compression mode (STRING)
Y..... window's top edge (UWORD)

(*) : These variables are valid after an
      EXAll
      command only.

```

1.241 QUIT

QUIT

Description of

	internal commands		
	command	option	description
QUIT	(no option)		close current window (ask for confirmation)
	FORCE/S		close current window at any rate
	UNLOAD/S		close window, unload editor (see
	HotKey)

1.242 REFRESH

REFRESH

Description of

internal commands			
	command	option	description
REFRESH	PAGE/S		redraw current text
	LINE/S		redraw current line

1.243 REMAP

REMAP

Description of

internal commands			
	command	option	description
REMAP	TABLE/K		remap text; use this translation file (STRING)
	ASK/S		open remap requester

1.244 REPLACE

REPLACE

Description of

internal commands			
	command	option	description
REPLACE	STRING/K		pattern to look for (STRING)
	BY/K		replacement for pattern (STRING)
	PREV/S		replace previous occurrence of pattern (*)
	NEXT/S		replace next occurrence of pattern only (*)
	ALL/S		replace all occurrences of pattern (*)
	BLOCK/S		replace pattern within marked lines (*)
	ASK/S		open find/replace requester
	CASE/K		set case sensitive mode (BOOL)
	QUIET/S		don't complain about missing pattern ('not found')
	WORDS/K		look for whole words only ? (BOOL)
	CONFIRM/K		ask for confirmation ? (BOOL)

(*) only one of these options may be specified.

1.245 REQUEST

REQUEST

Description of

internal commands		
command	option	description
REQUEST	HIDE/K	turn requesters off (BOOL)
	DEFAULT/K	set default if requesters are off (UWORD)
	BODY/K	body text, lines separated by ' ' (STRING)
	BUTTON/K	text for button(s), separated by ' ' (STRING)
	TITLE/K	requester title (STRING)
	LONG/S	ask for a number
	MIN/N	lower limit for number (WORD)
	MAX/N	upper limit for number/characters (WORD)
	OLD/K	default value (STRING)
	FILE/S	ask for a file
	SAVE/S	put ASL file requester into SAVE mode
	PATH/K	default path if asking for a file (STRING)
	MASK/K	file requester mask (e.g."#?.c") (STRING)
	VAR/K	ARexx variable; where to put the result (STRING)
	STRING/S	ask for a string
	STATUS/K	text to display in status line (STRING)
	KEY/S	ask for key (returns code and qualifier)
	PROBLEM/K	error message to be displayed (STRING)

Comment: Don't use the hide option outside of ARexx macros. Enable requesters before leaving the macro. Turning requesters off is useful to suppress messages like 'pattern not found' (wich might annoy the user during macro execution); you'll be responsible for checking whether an operation was successful by examining the return code RC.

1.246 RIGHT

RIGHT

Description of

internal commands		
command	option	description
RIGHT	(no options)	move cursor one position to the right

1.247 RUN

RUN

Description of

internal commands		
	command	option description
RUN	CMD/K	external program to run (STRING)
	LINE/S	execute current line of text
	PRIO/N	priority to be used (WORD: -3...3)
	STACK/N	stack to be used (ULONG)
	ASYNC/S	run program asynchronously
	OUTPUT/K	output (STRING)
	SHANGHAI/N	temporary shanghai time: seconds (UWORD)
	WAITPORT/K	wait for appearance of this port (STRING)
	SECONDS/N	WAITPORT timeout; defaults to 5 seconds (UWORD)

1.248 RX

RX

Description of

internal commands		
	command	option description
RX	CMD/K	command to be send (STRING)
	SYNC/S	send in synchronous mode (default: asynchronous)
	ASK/S	ask for command
	PORT/K	reciever; defaults to "AREXX" (STRING)
	MACRO/K	macro to execute if PORT is not valid (STRING)
	OUTPUT/K	output (STRING)

Comment: The macro is called with the command string as first argument. Basically same usage as the AmigaDos command RX if you set the port to AREXX: Use single quotation marks to have the command string evaluated by the Arexx server (e.g. rx 'info user'). Use double quotation marks or no quotation marks at all to make the server execute a script (e.g. rx golded:arexx/empty.ged).

1.249 SAVE

SAVE

Description of

internal commands		
	command	option description
SAVE	BLOCK/S	save block only
	ALL/S	save complete file
	NAME/K	set file name to be used for saving (STRING)
	ASK/S	open save-as requester
	EXIT/S	close window if save operation is succesful

CRUNCH/S	compress file (XPK) while saving it
XPKMODE/K	XPK compression mode (STRING, e.g. IMPL)
PASS/K	XPK password (STRING)

1.250 SCREEN

SCREEN

Description of

	internal commands		
	command	option	description
SCREEN	ASK/S		open display mode requester
	USE/K		name of public screen to use (STRING)
	ICONIFY/K		(un)iconify (BOOL)
	FRONT/S		move GoldED's screen to the front
	BACK/S		move GoldED's screen to the back
	CLOSE/S		close screen (wait for HotKey activation)
	NOSIZE/K		no fixed screen dimensions ? (BOOL)
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

1.251 SET

SET

Description of

	internal commands		
	command	option	description
SET	USER/N		user variable to set (UWORD 1..20)
	VALUE/K		new value (BOOL)

1.252 SHIFT

SHIFT

Description of

	internal commands		
	command	option	description
SHIFT	COLUMNS/N		number of columns to shift (UWORD)
	TAB/S		set shifting distance to TAB size
	ASK/S		open requester (left/right shifting)

LEFT/S	shift to the left
RIGHT/S	shift to the right (indent)

1.253 SMARTCR

SMARTCR

Description of

internal commands		
command	option	description
SMARTCR	(no options)	'smart' return (no splitting of line)

1.254 SUFFIX

SUFFIX

Description of

internal commands		
command	option	description
SUFFIX	VAR/K	ARexx variable to be updated (STRING)
	SUFFIX/K	desired suffix (STRING, e.g. ".c")

1.255 TAB

TAB

command	option	description
TAB	BACK/S	perform a backtab (else a normal tab)
	SOLID/K	make it a solid TAB (BOOL)
	FIXED/S	make it a distinctive TAB
	DYNAMIC/S	make it a dynamic TAB (see dynamic TABs)
	REGULAR/S	make it a regular (standard) TAB

Comment: options of this command may be used to change TAB mode temporarily. They do not affect global TAB settings (use

TABS
instead).

1.256 TABS

TABS

Description of

internal commands			
	command	option	description
TABS	FIXED/S		set TAB mode to distinctive TAB's
	REGULAR/S		set TAB mode to regular TAB's
	DYNAMIC/S		set TAB mode to dynamic TAB's
	FILE/N		number of SPC's used for TAB substitution (UWORD)
	KEY/N		TAB distance on screen (UWORD)
	ASK/S		open TAB preferences requester
	SET/N		add distinctive TAB position (UWORD)
	CLR/S		clear all distinctive TAB positions
	SOLID/K		make TABs solid (BOOL)
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

1.257 TASK

TASK

Description of

internal commands			
	command	option	description
TASK	PRI/N		set task priority of GoldED (WORD, -3 to 3)
	DEBUG/K		set debug mode (BOOL)

Comment: in debug mode recieved ARexx commands are printed to standard output.

1.258 TEXT

TEXT

Description of

internal commands			
	command	option	description
TEXT	T/K		text to be inserted at cursor position (STRING)
	VAR/K		variable to be inserted; see
	QUERY		(STRING)
	STAY/S		don't move cursor while inserting text
	CR/S		append linefeed to text

Comment: use "*" within T/K to insert quotation marks (single quotation marks are considered as string delimiters).

1.259 TMLATE

TMLATE

Description of

	internal commands		
	command	option	description

TMLATE	ASK/S		open templates requester
	USE/K		enable/disable templates (BOOL)
	CHECK/S		check word under cursor
	CONFIG/K		name of a preset file (STRING)
	LOAD/S		load preset file
	SAVE/S		save preset file

1.260 UJUMP

UJUMP

Description of

	internal commands		
	command	option	description

UJUMP	(no options)		jump to beginning of screen / previous page

Comment: Cursor jumps to first line of screen if placed below that line so far. Jumps to previous page if placed in line one already. Compare:

UPAGE

.

1.261 UNDO

UNDO

Description of

	internal commands		
	command	option	description

UNDO	(no options)		undo line (restore previous contents)

1.262 UNLOCK

UNLOCK

Description of

internal commands		
command	option	description
UNLOCK	(no option)	unlock GUI after a previously call to LOCK
	DELAY/S	unlock GUI, delay until exit of GoldED
	STICKY/S	unlock GUI, delay until current window is closed

Comment: The DELAY/STICKY options are reserved for use by external applications (e.g. the QuickStarter ED). They provide ways and means to synchronize with GoldED.

1.263 UP

UP

Description of

internal commands		
command	option	description
UP	(no options)	move cursor one line up

1.264 UPAGE

UPAGE

Description of

internal commands		
command	option	description
UPAGE	(no options)	move to previous page (compare UJUMP)

1.265 USE

USE

Description of

	internal commands		
	command	option	description
USE	(no options)		accept current line

Comment: To be used within ARExx macros only. After having written directly to the memory area of the current line (which is *dangerous*) you have to call this function to make GoldED accept your changes. Use QUERY CURRENT (see

QUERY
) to get a pointer to the current line's buffer. It is not possible to change the length of the current line by poking into the line buffer.

1.266 VIEW

VIEW

Description of

	internal commands		
	command	option	description
VIEW	LEFT/S		shift view left
	RIGHT/S		shift view right
	COLUMNS/N		number of columns to shift (UWORD); defaults to 5

Comment:

VLEFT
 and
 VRIGHT
 provide a better performance and thus should be preferred.

1.267 VLEFT

VLEFT

Description of

	internal commands		
	command	option	description
VLEFT	(none)		shift view 5 columns left

1.268 VRIGHT

VRIGHT

Description of

internal commands		
command	option	description
VRIGHT	(none)	shift view 5 columns right

1.269 WINDOW

WINDOW

Description of

internal commands		
command	option	description
WINDOW	MAX/S	blow up current window
	CENTER/S	center current window on screen
	ARRANGE/N	arrange windows (0: vertical, 1: horizontal)
	ZIP/S	zip window
	USE/K	activate named window/file (STRING)
	FORCE/S	load named file if necessary (see USE/K)
	WIDTH/N	resize window width (UWORD)
	HEIGHT/N	resize window height (UWORD)
	X/N	set window's x position (UWORD)
	Y/N	set window's y position (UWORD)
	NEXT/S	activate next window
	PREV/S	activate previous window
	RECOVER/S	redraw window
	HANDLE/N	activate window using its handle (ULONG)
	ORDINAL/N	activate 1st , 2nd, ... window (ULONG: 0, ...)
	QUIET/S	NEXT/USE/ORDINAL: leave window in the background
	SNAP/S	Use current window's dimensions as default size

Comment: A window handle is returned by the
OPEN
function.

You may query the current window's handle, too (see

QUERY
/handle).

1.270 XREF

XREF

Description of

internal commands		
	command	option
		description
XREF	CURRENT/S	find/open reference file related to current word
	PHRASE/K	find/open file related to this phrase (STRING)
	ASK/S	ask for topic
	CHECK/S	just determine whether a reference is available

1.271 Input events

Input events

Input event insertion (suggested by Markus Aretz):

GoldED's

KEY

command may be used to insert "events" (e.g. keystrokes) into intuition's global input stream. Inserting events makes the current application (the active GoldED window) behave as if the user had performed the described action. A key event description string EVENT/K may consist of plain text or plain text mixed with "event descriptions" in angle brackets (e.g. "<shift>"). You would have to specify the RAW/S option if you want to insert plain text containing angle brackets. Examples:

```
KEY EVENT="hello world"
KEY EVENT="hello world<return>"
KEY EVENT="--->" RAW
```

If you do not specify RAW, event descriptions like "<return>" are not treated as plain text but translated into input events (<return> would act as if the return key were pressed). The following expression outlines the format of description strings: <CLASS QUALIFIER(S) KEY>

A) CLASS may be one of the following (assuming <rawkey> if none is given):

```
rawkey ..... this is a keyboard event
rawmouse ..... this is a mouse button event
```

B) QUALIFIER(s) may be one or more of ...

```
shift ..... shift
control ..... ctrl
capslock ..... capslock
alt ..... alt
lcommand ..... left Amiga
rcommand ..... right Amiga
numericpad ..... numeric pad
leftbutton ..... left mouse button
rbutton ..... right mouse button
```

C) KEY may either be a plain character or ...

```
space ..... space
```

```

backspace ..... backspace
tab ..... tab
enter ..... enter
return ..... return
esc ..... esc
del ..... delete
up ..... cursor up
down ..... cursor down
right ..... cursor right
left ..... cursor left
f1 - f10 ..... function key
help ..... help

```

```

Examples: KEY EVENT="<rawkey shift A>"
          KEY EVENT="<rawkey f1>"
          KEY EVENT="<rawkey shift down>"
          KEY EVENT="<rawkey rcommand o>"

```

1.272 GENERAL HINTS

GENERAL HINTS

Never change display mode if you are short of memory - the editor might not be able to reopen screen/windows.

Turn AutoFold (see `config/misc`) OFF if you do not need folding. This will speed up loading since the editor won't have to examine each line after loading a text from disk (looking for fold markers).

Use the same marker strings for beginning/end of folded sections (see `config/misc`) to increase the speed of folding: the editor won't have to look for two different strings. However, using the same strings doesn't allow nested folding.

The backup path should point to the same device as your main text directory, otherwise backup generation is slowed down because file copies have to be used instead of a fast DOS rename (which is not available across devices).

Don't use soft/hardlinks with text files. If a file is renamed by the editor (happens during backup creation) the link doesn't change, i.e. it would point to the backup instead of the new file.

Do not use fast loading if you are short of RAM or for huge files since fast loading requires a huge buffer of exactly the original file's size. Attempting to load a 400 KB file would allocate a 400 KB IO buffer as well as about 450 KB to store the text (however the 400 KB IO buffer is freed immediately after loading). Slow loading is much more efficient in terms of memory consumption: a 16K buffer is required, no matter how large a file is.

The editor itself is not able to read the position of the 'sleep' icon (reading an AppIcon's position is not supported by the OS), so it is not able to remember the icon's position after you have moved it. Nevertheless it is possible to snapshot the preferred position: Open the 'prefs/env-archive/GoldED' directory. Move the 'AppIcon' icon from within that directory to desired location, than snapshot it (icon menu of workbench). Finally move the icon back to its drawer. The editor will read the new position the next time it is evoked.

The Amiga OS supports font/library caching: Fonts or libraries opened by GoldED usually are not removed from memory when GoldED is closed - instead they are marked as 'unused'. The OS is able to flush these resources if required (in low memory situations), so the memory occupied by libs/fonts is not lost (you may verify this by using the 'avail flush' command).

In spite of the fact that GoldED keeps its default configuration file in 'envarc:GoldED' - do not use this directory to store further configuration files: A copy of envarc is written to RAM: during startup; you would waste memory. 'GoldED:' or even 's:' are more appropriate locations.

1.273 CREDITS

CREDITS

* DICE * Reqtools * XPK * ARexxBox * GadToolsBox *

This program has been developed using Matt Dillon's Dice C compiler (v2.07.54R). Most of the requester design has been done using Jan van den Baard's 'GadToolsBox'. Thanx to Nico François for his 'reqtools.library'. Further credits got to the developers of 'xpk.library' - who created a fine (de)compression standard. The ARexx routines of GoldED are somewhat related to source code created by Michael Balzer's ARexxBox (though less sophisticated). Thanks to Stefan Zeiger for Boopsi example source code. And last but not least thanks to Joerg Gutzke, Dario Fava & Thomas Lechner for running Mowgli/Mailway/Tomate BBS (sources of most of the tools above). GUIMake has been developed by Rico Krasowski. Included with kind permission of the author. Finally, I would like to thank these people for their invaluable suggestions, ideas & support: Markus Aretz, Martin Fay, David 'Edi' Göhler, Christian Gottschling, Serge Hammouche, Andreas Harrenberg, Martin Korndörfer, René Laederach, Lieven Lema, Rudolphe Sanderson (French translations) and Stefan Schor !

1.274 HOW TO REGISTER

HOW TO REGISTER

If you are currently using the unregistered version (saving/printing limited to 1000 lines) you may want to register. Registered users receive a keyfile,

providing unrestricted access to the program(s). Please allow up to 8 weeks for delivery. This is the worst case. Average is three weeks. The following sites provide registration services (prices may differ; some manual translations are not available at all sites):

Registration site Germany

Registration site Belgium

Registration site France

1.275 Registration site Germany

Registration site Germany

The German registration site (address: see
how to contact author
) offers

the packages listed below. To register for GoldED, send an EUROcheck or cash. I can not accept ANY OTHER kind of payment than eurochecks or cash with exception of the GoldED Pro/NET distribution. No foreign checks, no postal money orders. Please type your letter. Supply a valid and fully qualified address for shipment, including country name. All offers valid until end of 1994:

GoldED Light (20 DM German orders, 25 DM Europe, 20\$ outside Europe)

Mailing consists of one disk. Includes a keyfile for GoldED (only). The latest versions of the GoldED package as well as standard disk-based documentation is shipped. Please state whether you own a 1.7 MB HD disk drive.

GoldED Pro (35 DM German orders, 40 DM Europe, 30\$ outside Europe)

Mailing consists of one or two disks and a printed manual of about 70 pages. Please state whether you own a 1.7 MB HD disk drive. Includes a single user licence for the latest versions of GoldED & HiSpeed. Please choose the manual translation you would like to receive (French translations are available at the

Registration site France
):

- o English (default)
- o German (if requested)

GoldED Pro/NET (25 DM worldwide)

The net distribution: you'll receive a keyfile for GoldED and HiSpeed upon orders sent to DIETMAR@TOMATE.OCHE.DE (a German domain). Z-Net/Internet access required. Once your keyfile is installed, you can use your currently installed unregistered GoldED/HiSpeed (0.99 or better) without restrictions. Since keyfiles are shipped PGP encrypted, your PGP key has to be part of

your order (pgp -kxa); orders without a PGP key are not processed. Payment is expected to be transferred to the following account within two weeks: Dietmar Eilert, account 5129 92-505, Postbank Köln, BLZ 370 100 50

In generally FD support tools and libraries are not part of the distribution unless we get permission from the authors. We do provide disk(s)/postage, covered by slightly higher prices for international orders. Special conditions are available if you order more than one copy; don't forget to provide a user name/address for each of them (required for registration):

2 packages -20% each
 more (just joking :-) -30% each

1.276 Registration site Belgium

Registration site Belgium

AUGFL offers a Belgian registration site by the services of Lieven Lema, Sint-Amandsstraat 61, 1853 Strombeek, Belgium. Bank Account ASLK 001-1869832-39. He can be joined on Fido 2:292/603.11 or Compuserve 100343,241 or Internet Lieven.Lema@AUGFL.BE. The rates listed below will be applied for customers, the exchange rate for 1 DM = 22 BEF will be applied. Should significant changes occur (for better or for worse) they will be taken into account after consulting. The following packages are available; offers valid until end of 1994:

GoldED Light (650 BEF Europe, AUGFL members: 550 BEF, 20\$ international)

 Mailing consists of one disk. Includes a keyfile for GoldED (only). The latest versions of the GoldED package as well as standard disk-based documentation is shipped. Please state whether you own a 1.7 MB HD disk drive.

GoldED Pro (950 BEF, AUGFL members: 850 BEF, 30\$ international)

 Mailing consists of one or two disks and a printed manual of about 70 pages. Please state whether you own a 1.7 MB HD disk drive. Includes a single user licence for the latest versions of GoldED & HiSpeed. Please choose the manual translation you would like to receive (French translations are available at the

Registration site France
):

- o English (default)
- o German (if requested)
- o Dutch (not yet available)

GoldED Pro/NET (650 BEF international, AUGFL members: 550 BEF)

 The net distribution: you'll receive a keyfile for GoldED and HiSpeed upon orders sent to Lieven.Lema@AUGFL.BE (Internet) or 2:292/603.11 (Fido) or Compuserve 100343,241. Once your keyfile is installed, you can use your currently installed unregistered GoldED/HiSpeed (0.99 or better) without

restrictions. Since keyfiles are shipped PGP encrypted, your PGP key has to be part of your order (pgp -kxa); orders without a PGP key are not processed. Payment is expected to be transferred to the following account within two weeks: Lieven Lema, Sint-Amandsstraat 61, 1853 Strombeek. Bank Account ASLK 001-1869832-39

In generally FD support tools and libraries are not part of the distribution unless we get permission from the authors. We do provide disk(s)/postage, covered by slightly higher prices for international orders. Special conditions are available if you order more than one copy; don't forget to provide a user name/address for each of them (required for registration):

2 packages -20% each
 more (just joking :-) -30% each

1.277 Registration site France

Registration site France

The French company FFD France Festival Distribution offers registered GoldED PRO packages in France. These packages include a printed professional-made French translation of the manual (about 100 pages). French manuals are exclusively distributed by FFD and not available at the other sites. A GoldED PRO package is available for 250 FF TTC; taxes and shipping included. Please send your orders to:

France Festival Distribution
 3, rue Anatole France
 13220 Chateuaneuf Les Martigues
 FRANCE
 Fax: +33.42.76.18.70

1.278 HOW TO GET UPDATES

HOW TO GET UPDATES

The only way to receive updates, whether registered or unregistered, is to call your local BBS and look out for the latest GoldED copy. Registered users receive a keyfile and may thus use demo versions without restrictions. Don't send disks unless you want to get rid of them (don't worry, they are put to a good cause; preferably backups :-). Uploads usually go to Tomate BBS (Aachen, Germany) - this is the main support BBS:

TOMATE (Aachen/Germany); Sysop: Thomas 'Tom' Lechner - SYSOP@TOMATE.OCHE.DE

 +49-(0)2408-7788 (ZyXEL). Editor placed in GoldED file area. Guest access.

MOWGLI (Aachen/Germany); Sysop: Joerg Gutzke

+49-(0)241-405949. The editor is placed in the <files/utilities> area. Fido file request: magic GoldED.

DOOM (Bremen/Germany)

Ports: +49-(0)4223-8355, +49-(0)4223-3256, +49-(0)4223-3313 (ZyXEL). Filearea FILESERVER-AMIGA/SUPPORT/GOLDED.

SUNBURN (Germany/Westfalen-Lippe)

+49-(0)5231-18626 USR DS, ISDN +49-(0)5231-969361, SUPPORT area.

TAURUS ALPHA 1/Austria

Ports: 43-732-611243, 43-732-609032, 43-732-663090 (ZyXEL). Fido FREQ magic GOLDED.

1.279 HOW TO CONTACT AUTHOR

HOW TO CONTACT AUTHOR

Feel free to e-mail or fax bug reports, comments or suggestions. Please do not send normal letters unless you want to register. We aren't able to answer your written questions unless you provide a self-addressed envelope, postage paid (international reply coupons, no foreign stamps). In generally you can reach the support by writing to one of the addresses below. These addresses are not valid for requesting updates (see:

How to get updates
)

Dietmar Eilert
Mies-v-d-Rohe-Str. 31
52074 Aachen (Germany)
Phone: +49-(0)241/81665
FAX: +49-(0)241/81665

Dietmar Eilert
Kampstraße 28
59269 Beckum (Germany)
Phone: +49-(0)2525/7776

E-mail: DIETMAR@TOMATE.OCHE.DE

Please call to find out the current address. Or send your registration to one address, a short note to the other (this may slow down delivery a bit).

1.280 GoldED

APC

FIND/ASCII INSERT

MISC/LINE PUSH

API

FIND/ASCII TABLE

MISC/LINE SWAP
APPICON
FIND/CHARACTER SET
MISC/SEARCH FILE
APPLICATION INTERFACE
FIND/CHECK
MISC/SHELL
APPWINDOWS
FIND/COMPLETE
MISC/SOURCE FILES
AREXX PORT
FIND/COUNT
MISC/STATISTICS
AREXXBOX
FIND/FIND
MISC/UNDO LINE
ARGUMENTS
FIND/FIND NEXT
MODE
AUTOARRANGE
FIND/FIND PREVIOUS
MORE
AUTOBACKUP
FIND/FUNCTIONS
MOUSE
AUTCASE
FIND/INSERT CODE
MOUSE HANDLING

AUTOFOLD

FIND/MATCHING BRACKET

MULTIPLE COMMANDS

AUTOINDENTION

FIND/REFERENCE

MULTISELECT

AUTOLOAD

FIND/REFERENCE...

NAME

BACK

FIND/REPLACE

NEW

BEEP

FIND/REPLACE NEXT

NEXT

BIND

FIND/SHOW CODE

NOTIFY

BITS

FIND/TOGGLE CASE

OPEN

BLOCK

FIRST

OVERWRITE

BLOCK MENU

FIX

PARAGRAPHE VS. BLOCK

BLOCK/APPEND TEXT

FOLD

PARENTHESIS CHECK

BLOCK/BCOPY

FOLDING

PATH

BLOCK/BDELETE

FONT

PHRASE

BLOCK/BMOVE

FORMAT

PING

BLOCK/COLUMN TEXT

FORMATTER

PONG

BLOCK/COPY

FREEZE

POP

BLOCK/CUT

FUNC

PREFS

BLOCK/DELETE COLUMN

FUP

PREV

BLOCK/HIDE MARK

GADTOOLSBOX

PREVEND

BLOCK/INDENT

GENERAL HINTS

PREVIEW

BLOCK/INSERT COLUMN
GETTING STARTED
PRINT
BLOCK/LOWERCASE
GLOBAL SEARCH
PROJECT
BLOCK/MARK
GOTO
PROJECT MENU
BLOCK/PASTE
GREP
PROJECT/ABOUT
BLOCK/PASTE VERTICAL
GUI
PROJECT/APPEND
BLOCK/PRINT
GUIMAKE
PROJECT/BITS
BLOCK/SAVE AS
HELP
PROJECT/CLEAR TEXT
BLOCK/SORT
HELP KEY
PROJECT/CURRENT DIR
BLOCK/UPPERCASE
HOTKEY
PROJECT/INSERT
BOTTOM SLIDER
HOW TO CONTACT AUTHOR

PROJECT/MORE ED
BRACKET
HOW TO FOLD LINES
PROJECT/NEW NAME
CENTERWIN
HOW TO GET UPDATES
PROJECT/OPEN
CHARACTER SET REMAP
HOW TO REGISTER
PROJECT/OPEN FAST
CLIP
HUNTER
PROJECT/OPEN NEW
CLIPBOARD
INDENT
PROJECT/OPEN ORIGINAL
CMD
INDEX
PROJECT/PRINT
CODE
INFO
PROJECT/QUIT & UNLOAD
COLON
INPUT EVENTS
PROJECT/QUIT (WINDOW)
COMMAND LIST
INSERT
PROJECT/SAVE

COMMAND SET EXTENSIONS

INSERTION OF COLUMNS

PROJECT/SAVE & EXIT

CONFIG MENU

INTERNAL COMMANDS

PROJECT/SAVE AS

CONFIG/API

INTRODUCTION

PROJECT/SAVE AS XPK

CONFIG/Dictionary

KEY

PROJECT/USER

CONFIG/Display

KEYBOARD

PUSH

CONFIG/FILE HUNTER

LANGUAGE

QUERY

CONFIG/GUI

LAYOUT

QUICKFUNC

CONFIG/INDENTION

LAYOUT MENU

QUICKREFERENCE

CONFIG/KEYBOARD

LAYOUT/AUTOCASE

QUICKSTARTER

CONFIG/LAYOUT

LAYOUT/BLOCK CENTER

QUIT

CONFIG/LOAD

LAYOUT/BLOCK LEFT

REFRESH

CONFIG/MENUS

LAYOUT/BLOCK LEFT/RIGHT

REGISTRATION SITE BELGIUM

CONFIG/MISC

LAYOUT/BLOCK RIGHT

REGISTRATION SITE FRANCE

CONFIG/MOUSE

LAYOUT/RIGHT-TO-LEFT

REGISTRATION SITE GERMANY

CONFIG/PRINTER

LAYOUT/SECTION BLOCK

REMAP

CONFIG/REFERENCES

LAYOUT/SECTION CENTER

REMOVAL OF COLUMNS

CONFIG/SAVE

LAYOUT/SECTION LEFT

REPLACE

CONFIG/TABS

LAYOUT/SECTION RIGHT

REQTOOLS

CONFIG/TEMPLATES

LAYOUT/SET RIGHT MARGIN

REQUEST

CONTROL MENU

LAYOUT/TEMPLATES ON/OFF

REQUIRED SYSTEM

CONTROL/FOLD ALL

LAYOUT/USE CURRENT MARGIN

RETURN KEY

CONTROL/FREEZE WINDOW

LAYOUT/WORD WRAP ON/OFF

REVERSED

CONTROL/GO TO LINE

LEFT

RIGHT

CONTROL/ICONIFY

LICENCE

RIGHT-TO-LEFT

CONTROL/INSERT

LINES

RUN

CONTROL/NEXT WINDOW

LOAD TWICE

RX

CONTROL/NUMPAD = MOVEMENT

LOCK

SAVE

CONTROL/PREVIEW

LOCK A WINDOW

SCREEN

CONTROL/PREVIOUS WINDOW

MACRO

SCROLL BORDERS

CONTROL/RECALL POSITION

MACRO MENU

SEARCH/REPLACE HISTORY

CONTROL/STORE POSITION

MACRO RECORDING

SELECT A HOST

CONTROL/TO LAST CHANGE

MACROS/EDIT MACRO

SEQUENCES

CONTROL/TOGGLE TAB MODE

MACROS/GUIMAKE

SET

CONTROL/TOP-BOTTOM

MACROS/MACROS C

SHIFT

CONTROL/UNFOLD ALL

MACROS/MACROS OTHERS

SHIFTING

CONTROL/WINDOW ARRANGE

MACROS/PLAY MANY

SMARTCR

CONTROL/WINDOW CENTER

MACROS/RUN TEXT AS MACRO

SMARTINDENTION

CONTROL/WINDOW ENLARGE

MACROS/SEQUENCE LOAD

SPEEDS OF SCROLLING

CONTROL/WINDOW ZIP

MACROS/SEQUENCE PLAY

SPELLCHECKER

CR

MACROS/SEQUENCE RECORD

STARTUP MACRO

CREDITS

MACROS/SEQUENCE SAVE

STATUS BAR

CURSOR KEYS

MAGIC CODES

SUFFIX

DEL

MAIN

TAB

DEL KEY

MARGINS

TAB KEY

DELETE

MARK

TABS

DESCRIPTION OF MENUS

MAXDOWN

TASK

DICE

MAXUP

TEMPLATES

DIR

MENU TREE OF BLOCK MENU

TEXT

DISPLAY MODE

MENU TREE OF CONFIG MENU

TMPLATE

DJUMP

MENU TREE OF CONTROL MENU

UJUMP

DO YOUR JOB

MENU TREE OF FIND MENU

UNDO

DOCK

MENU TREE OF LAYOU MENU

UNLOCK

DOWN

MENU TREE OF MACRO MENU

UNLOCK GUI

DPAGE

MENU TREE OF MISC MENU

UP

DYNAMIC TABS

MENU TREE OF PROJECT MENU

UPAGE

ENDWORD

MENUHELP

USE

EOL WRAP

MENUS

USE ASL

ESC KEY

MISC

USER DEFINED GADGETS

EVENT DEFINITION

MISC MENU

USER VARIABLES

EXALL

MISC/CALCULATOR

VIEW

EXTRACT

MISC/COMMAND

VLEFT

F-KEYS

MISC/FILES

VRIGHT

FAST SCROLLING

MISC/FILTER

WEIGHT

FASTLOAD

MISC/HELP

WHITE SPACE

FDOWN

MISC/HISPEED

WINDOW

FEATURE LIST

MISC/INSERT DATE

WORDWRAP

FILE

MISC/INSERT PATH

XPX

FILE HUNTER

MISC/INSERT TIME

XPX SUPPORT

FIND

MISC/LAST ERROR

XREF

FIND MENU

MISC/LINE DOUBLE

FIND/ASCII INSERT

MISC/LINE PICK
