WRITE

COLLABORATORS

	TITLE : WRITE	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 31, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1 WRITE **1**

1.1 WRITE.guide . . . 1
1.2 WRITE.guide/Wichtig . . . 2
1.3 WRITE.guide/Autor . . . 3
1.4 WRITE.guide/Danksagungen . . . 3
1.5 WRITE.guide/Einleitung . . . 4
1.6 WRITE.guide/Copyright . . . 5
1.7 WRITE.guide/Demoversion . . . 6
1.8 WRITE.guide/Vollversion . . . 6
1.9 WRITE.guide/Allgemein . . . 8
1.10 WRITE.guide/Registrierung . . . 8
1.11 WRITE.guide/Installation . . . 9
1.12 WRITE.guide/Konzept . . . 11
1.13 WRITE.guide/Das Prinzip . . . 12
1.14 WRITE.guide/Der Ed . . . 12
1.15 WRITE.guide/Syntax . . . 13
1.16 WRITE.guide/allgemeine Syntax . . . 13
1.17 WRITE.guide/Typen . . . 14
1.18 WRITE.guide/Variablen . . . 15
1.19 WRITE.guide/Konstanten . . . 16
1.20 WRITE.guide/ToolTypes . . . 17
1.21 WRITE.guide/Sound . . . 18
1.22 WRITE.guide/Editorfenster . . . 19
1.23 WRITE.guide/Requester . . . 20
1.24 WRITE.guide/Listen und Buffer . . . 21
1.25 WRITE.guide/PublicScreens . . . 22
1.26 WRITE.guide/Konstantenbeschreibung . . . 22
1.27 WRITE.guide/TRUE . . . 23
1.28 WRITE.guide/FALSE . . . 23
1.29 WRITE.guide/LOWER . . . 24

1.30 WRITE.guide/HIGHER . . . 24
1.31 WRITE.guide/EQUAL . . . 24
1.32 WRITE.guide/CURSOR . . . 25
1.33 WRITE.guide/MARKA . . . 25
1.34 WRITE.guide/MARKB . . . 25
1.35 WRITE.guide/SOL . . . 25
1.36 WRITE.guide/EOL . . . 26
1.37 WRITE.guide/SOW . . . 26
1.38 WRITE.guide/EOW . . . 26
1.39 WRITE.guide/SOT . . . 26
1.40 WRITE.guide/EOT . . . 27
1.41 WRITE.guide/SOP . . . 27
1.42 WRITE.guide/EOP . . . 27
1.43 WRITE.guide/SOS . . . 27
1.44 WRITE.guide/MOS . . . 28
1.45 WRITE.guide/EOS . . . 28
1.46 WRITE.guide/Variablebeschreibung . . . 28
1.47 WRITE.guide/_AColor . . . 31
1.48 WRITE.guide/_BColor . . . 32
1.49 WRITE.guide/_CColor . . . 32
1.50 WRITE.guide/_DColor . . . 32
1.51 WRITE.guide/_ReadTabs . . . 33
1.52 WRITE.guide/_WriteTabs . . . 33
1.53 WRITE.guide/_CaseSense . . . 33
1.54 WRITE.guide/_Optimize . . . 33
1.55 WRITE.guide/_XPos . . . 34
1.56 WRITE.guide/_YPos . . . 34
1.57 WRITE.guide/_Changed . . . 34
1.58 WRITE.guide/_FileName . . . 34
1.59 WRITE.guide/_FindString . . . 34
1.60 WRITE.guide/_ReplaceString . . . 35
1.61 WRITE.guide/_Length . . . 35
1.62 WRITE.guide/_RS . . . 35
1.63 WRITE.guide/_RN . . . 35
1.64 WRITE.guide/_WBStarted . . . 36
1.65 WRITE.guide/_LoadTab . . . 36
1.66 WRITE.guide/_SaveTab . . . 36
1.67 WRITE.guide/_Version . . . 37
1.68 WRITE.guide/_ChipMem . . . 37

1.69 WRITE.guide/_FastMem . . . 37
1.70 WRITE.guide/_PublicMem . . . 37
1.71 WRITE.guide/_Wins . . . 37
1.72 WRITE.guide/_Time . . . 38
1.73 WRITE.guide/_Path . . . 38
1.74 WRITE.guide/_FRPattern . . . 38
1.75 WRITE.guide/_PatCase . . . 38
1.76 WRITE.guide/_ConfigPath . . . 39
1.77 WRITE.guide/_ShowSpace . . . 39
1.78 WRITE.guide/_ShowEOL . . . 39
1.79 WRITE.guide/_PatNoCase . . . 39
1.80 WRITE.guide/_CurrentChar . . . 40
1.81 WRITE.guide/_CurrentWord . . . 40
1.82 WRITE.guide/_CurrentLine . . . 40
1.83 WRITE.guide/_TabLength . . . 40
1.84 WRITE.guide/_MaxBuffer . . . 41
1.85 WRITE.guide/_REXX . . . 41
1.86 WRITE.guide/_WordDef . . . 41
1.87 WRITE.guide/_WriteIcon . . . 41
1.88 WRITE.guide/_AppIcon . . . 42
1.89 WRITE.guide/_AppWindow . . . 42
1.90 WRITE.guide/_AppMenu . . . 42
1.91 WRITE.guide/_Cursor . . . 42
1.92 WRITE.guide/_AutoIndent . . . 43
1.93 WRITE.guide/_OverwriteIcon . . . 43
1.94 WRITE.guide/_SleepMode . . . 43
1.95 WRITE.guide/_EditMode . . . 44
1.96 WRITE.guide/_Marked . . . 44
1.97 WRITE.guide/_MarkAx . . . 44
1.98 WRITE.guide/_MarkAy . . . 44
1.99 WRITE.guide/_MarkBx . . . 45
1.100 WRITE.guide/_MarkBy . . . 45
1.101 WRITE.guide/_WinMode . . . 45
1.102 WRITE.guide/_CurrentID . . . 46
1.103 WRITE.guide/_VersionString . . . 46
1.104 WRITE.guide/_File . . . 46
1.105 WRITE.guide/_FilePath . . . 46
1.106 WRITE.guide/_REXXPortName . . . 47
1.107 WRITE.guide/_REG1 . . . 47

1.108 WRITE.guide/_REG2 . . . 47
1.109 WRITE.guide/_Sound . . . 47
1.110 WRITE.guide/_DefaultTool . . . 47
1.111 WRITE.guide/_Language . . . 48
1.112 WRITE.guide/_CurrentConfig . . . 48
1.113 WRITE.guide/_DefaultConfig . . . 48
1.114 WRITE.guide/_AutoFree . . . 48
1.115 WRITE.guide/_WinWidth . . . 49
1.116 WRITE.guide/_WinHeight . . . 49
1.117 WRITE.guide/_WordWrap . . . 49
1.118 WRITE.guide/_RightMargin . . . 50
1.119 WRITE.guide/_ScreenWidth . . . 50
1.120 WRITE.guide/_ScreenHeight . . . 50
1.121 WRITE.guide/_Undo . . . 50
1.122 WRITE.guide/_ScrRelWidth . . . 51
1.123 WRITE.guide/_ScrRelHeight . . . 51
1.124 WRITE.guide/_WordOnly . . . 51
1.125 WRITE.guide/_ReqToMouse . . . 52
1.126 WRITE.guide/_EColor . . . 52
1.127 WRITE.guide/_FColor . . . 52
1.128 WRITE.guide/_AutoFold . . . 52
1.129 WRITE.guide/_FoldStart . . . 53
1.130 WRITE.guide/_FoldEnd . . . 53
1.131 WRITE.guide/_DelFoldMark . . . 53
1.132 WRITE.guide/_Fold . . . 54
1.133 WRITE.guide/_Screenname . . . 54
1.134 WRITE.guide/_PathPath . . . 54
1.135 WRITE.guide/_User . . . 55
1.136 WRITE.guide/_CollectorMem . . . 55
1.137 WRITE.guide/Funktionsbeschreibung . . . 55
1.138 WRITE.guide/NOP . . . 65
1.139 WRITE.guide/Open . . . 65
1.140 WRITE.guide/Save . . . 66
1.141 WRITE.guide/SetBackUp . . . 66
1.142 WRITE.guide/New . . . 67
1.143 WRITE.guide/NewEd . . . 67
1.144 WRITE.guide/QuitEd . . . 68
1.145 WRITE.guide/Window . . . 68
1.146 WRITE.guide/Iconify . . . 69

1.147 WRITE.guide/Hide . . . 70
1.148 WRITE.guide/Silent . . . 70
1.149 WRITE.guide/Font . . . 71
1.150 WRITE.guide/WinArranger . . . 71
1.151 WRITE.guide/SetTitle . . . 72
1.152 WRITE.guide/Screen . . . 73
1.153 WRITE.guide/About . . . 73
1.154 WRITE.guide/Prefs . . . 74
1.155 WRITE.guide/GPrefs . . . 74
1.156 WRITE.guide/HelpFkt . . . 75
1.157 WRITE.guide/WinManager . . . 75
1.158 WRITE.guide/ShowVars . . . 76
1.159 WRITE.guide/ShowFunctions . . . 77
1.160 WRITE.guide/ShowConstants . . . 78
1.161 WRITE.guide/ShowASCII . . . 78
1.162 WRITE.guide/ShowIndex . . . 79
1.163 WRITE.guide/GetString . . . 80
1.164 WRITE.guide/GetNumber . . . 80
1.165 WRITE.guide/GetFindReplace . . . 81
1.166 WRITE.guide/GetFile . . . 81
1.167 WRITE.guide/GetFiles . . . 82
1.168 WRITE.guide/GetFont . . . 83
1.169 WRITE.guide/Ask . . . 83
1.170 WRITE.guide/Message . . . 84
1.171 WRITE.guide/MessageOK . . . 84
1.172 WRITE.guide/Flash . . . 85
1.173 WRITE.guide/Beep . . . 85
1.174 WRITE.guide/ParseBuffer . . . 86
1.175 WRITE.guide/DoBuffer . . . 86
1.176 WRITE.guide/DoString . . . 87
1.177 WRITE.guide/PreparseString . . . 87
1.178 WRITE.guide/SetUserFkt . . . 88
1.179 WRITE.guide/Compare . . . 89
1.180 WRITE.guide/If . . . 89
1.181 WRITE.guide/Break . . . 90
1.182 WRITE.guide/SetError . . . 91
1.183 WRITE.guide/SetVar . . . 91
1.184 WRITE.guide/GetVar . . . 92
1.185 WRITE.guide/GetConst . . . 92

1.186 WRITE.guide/SetEnv . . . 92
1.187 WRITE.guide/GetEnv . . . 93
1.188 WRITE.guide/System . . . 93
1.189 WRITE.guide/SetMark . . . 94
1.190 WRITE.guide/Mark . . . 94
1.191 WRITE.guide/UnMark . . . 95
1.192 WRITE.guide/DeleteBlock . . . 95
1.193 WRITE.guide/CopyBlock . . . 96
1.194 WRITE.guide/InsertBlock . . . 96
1.195 WRITE.guide/DeleteArea . . . 97
1.196 WRITE.guide/CopyArea . . . 98
1.197 WRITE.guide/SaveBuffer . . . 98
1.198 WRITE.guide/LoadBuffer . . . 99
1.199 WRITE.guide/ClearBuffer . . . 99
1.200 WRITE.guide/StrToBuffer . . . 100
1.201 WRITE.guide/BufferToStr . . . 100
1.202 WRITE.guide/ClipToBuffer . . . 100
1.203 WRITE.guide/BufferToClip . . . 101
1.204 WRITE.guide/BlockLeft . . . 101
1.205 WRITE.guide/BlockRight . . . 102
1.206 WRITE.guide/BlockLftAlig . . . 102
1.207 WRITE.guide/BlockRghtAlig . . . 102
1.208 WRITE.guide/BlockCenter . . . 103
1.209 WRITE.guide/CursorUp . . . 103
1.210 WRITE.guide/CursorDown . . . 104
1.211 WRITE.guide/CursorRight . . . 104
1.212 WRITE.guide/CursorLeft . . . 104
1.213 WRITE.guide/NextWord . . . 105
1.214 WRITE.guide/LastWord . . . 105
1.215 WRITE.guide/PageUp . . . 106
1.216 WRITE.guide/PageDown . . . 106
1.217 WRITE.guide/Goto . . . 107
1.218 WRITE.guide/GotoMouse . . . 107
1.219 WRITE.guide/SetTextMark . . . 108
1.220 WRITE.guide/GoTextMark . . . 108
1.221 WRITE.guide/Find . . . 108
1.222 WRITE.guide/Replace . . . 110
1.223 WRITE.guide/ReplaceList . . . 111
1.224 WRITE.guide/FindPattern . . . 111

1.225 WRITE.guide/MatchBracket . . . 112
1.226 WRITE.guide/Return . . . 112
1.227 WRITE.guide/Delete . . . 113
1.228 WRITE.guide/DeleteToEOL . . . 113
1.229 WRITE.guide/DeleteLine . . . 114
1.230 WRITE.guide/UnDelLine . . . 114
1.231 WRITE.guide/BackSpace . . . 115
1.232 WRITE.guide/Tab . . . 115
1.233 WRITE.guide/BackTab . . . 115
1.234 WRITE.guide/UpperBlock . . . 116
1.235 WRITE.guide/LowerBlock . . . 116
1.236 WRITE.guide/WriteChar . . . 117
1.237 WRITE.guide/WriteText . . . 117
1.238 WRITE.guide/Key . . . 118
1.239 WRITE.guide/DoubleKey . . . 118
1.240 WRITE.guide/ClearKeys . . . 119
1.241 WRITE.guide/SetHotKey . . . 119
1.242 WRITE.guide/ClearHotKey . . . 120
1.243 WRITE.guide/Menu . . . 120
1.244 WRITE.guide/Item . . . 121
1.245 WRITE.guide/Sub . . . 121
1.246 WRITE.guide/ItemBar . . . 122
1.247 WRITE.guide/SubBar . . . 123
1.248 WRITE.guide/ClearMenu . . . 123
1.249 WRITE.guide/WaitPointer . . . 124
1.250 WRITE.guide/NormalPointer . . . 124
1.251 WRITE.guide/DoREXX . . . 124
1.252 WRITE.guide/LockWindow . . . 125
1.253 WRITE.guide/NextEd . . . 126
1.254 WRITE.guide/OpenPort . . . 126
1.255 WRITE.guide/ClosePort . . . 127
1.256 WRITE.guide/WaitPort . . . 127
1.257 WRITE.guide/ModifyWin . . . 128
1.258 WRITE.guide/ModifyScreen . . . 128
1.259 WRITE.guide/SetREXXClip . . . 129
1.260 WRITE.guide/GetConfig . . . 129
1.261 WRITE.guide/SetREXXVar . . . 130
1.262 WRITE.guide/GetREXXVar . . . 130
1.263 WRITE.guide/Refresh . . . 130

1.264 WRITE.guide/ChangeConfig . . . 131
1.265 WRITE.guide/MacroRec . . . 131
1.266 WRITE.guide/MacroStop . . . 132
1.267 WRITE.guide/MacroPlay . . . 133
1.268 WRITE.guide/SetMacro . . . 133
1.269 WRITE.guide/ExecuteMacro . . . 134
1.270 WRITE.guide/MacroPannel . . . 134
1.271 WRITE.guide/Undo . . . 135
1.272 WRITE.guide/ClearList . . . 135
1.273 WRITE.guide/AddList . . . 136
1.274 WRITE.guide/RemoveList . . . 136
1.275 WRITE.guide/Push . . . 136
1.276 WRITE.guide/Pop . . . 137
1.277 WRITE.guide/ShowList . . . 137
1.278 WRITE.guide/ListToBuffer . . . 138
1.279 WRITE.guide/BufferToList . . . 138
1.280 WRITE.guide/DoList . . . 139
1.281 WRITE.guide/ListSize . . . 139
1.282 WRITE.guide/GetListEntry . . . 139
1.283 WRITE.guide/FindListEntry . . . 140
1.284 WRITE.guide/Exists . . . 140
1.285 WRITE.guide/Delay . . . 140
1.286 WRITE.guide/GuideHelp . . . 141
1.287 WRITE.guide/VersionCheck . . . 141
1.288 WRITE.guide/Inc . . . 142
1.289 WRITE.guide/Dec . . . 142
1.290 WRITE.guide/RangeCheck . . . 143
1.291 WRITE.guide/RangeRound . . . 143
1.292 WRITE.guide/Begin . . . 144
1.293 WRITE.guide/Close . . . 144
1.294 WRITE.guide/Start . . . 145
1.295 WRITE.guide/Quit . . . 145
1.296 WRITE.guide/Fold . . . 146
1.297 WRITE.guide/UnFold . . . 146
1.298 WRITE.guide/AutoFold . . . 146
1.299 WRITE.guide/ReFold . . . 147
1.300 WRITE.guide/RexxScripts . . . 147
1.301 WRITE.guide/Index . . . 149
1.302 WRITE.guide/Funktionsindex . . . 153
1.303 WRITE.guide/Variableindex . . . 161

Chapter 1

WRITE

1.1 WRITE.guide

Dokumentation für WRITE 3.848 vom 16 August 1994
****

Wichtig
Wichtige Bemerkungen bevor sie WRITE starten

Autor
Adresse des Autors

Danksagungen
Danksagungen an verschiedene Leute

Einleitung
Eine kleine Einleitung

Copyright
Rechtliches über die Benutzung von WRITE

Registrierung
Wie bekomme ich die Vollversion von WRITE ?

Installation
So installieren sie WRITE auf ihrem System

Konzept
Allgemeiner Aufbau des Editors

Syntax
Die Syntax des internen Parsers

ToolTypes
Die ToolsTypes des WRITE-Icons

Sound
Unterstützung von Sounds

Editorfenster
Allgemeines über das Editorfenster

Requester
Allgemeine Features der Requester

Listen und Buffer
Allgemeines über die Verwaltung derselben in WRITE

PublicScreens
WRITE und Public Screens

Konstantenbeschreibung
Beschreibung aller Konstanten

Variablebeschreibung
Beschreibung aller Variablen

Funktionsbeschreibung
Beschreibung aller Funktionen

RexxScripts
Kurze Beschreiben der mitgelieferten Scripts

Index
Das Stichwortverzeichnis

Funktionsindex
Verzeichnis aller Funktionen

Variableindex
Verzeichnis aller Variablen

1.2 WRITE.guide/Wichtig

Einige wichtige Bemerkungen
****

- WRITE läuft nur unter OS 2.0 oder höher !!!

- Lesen sie bitte ausführlich die Anleitung bevor sie WRITE das erste Mal benutzen.Vor allen Dingen die Kapitel
  Copyright
  ,
  Registrierung
  und
  Installation
  .

1.3 WRITE.guide/Autor

```
Autor
*****

Geld,Kritik,Vorschläge,Lob und Tadel in Deutsch oder English an:

Postadresse
               Tim Teulings
               An der Dorndelle 16
               59192 Bergkamen
               DEUTSCHLAND

EMailaddresse
     MausNet :

          Tim Teulings @ UN

     InterNet :

          rael@edge.ping.de

Bankverbindung
          Sparkasse Berkamen - Bönen
          BLZ     : 410 518 45
          KontoNr.: 16186496
          Tim Teulings
```

1.4 WRITE.guide/Danksagungen

```
Danksagungen
************

                                Danke...

Chris Gray
     Für den grandiosen Draco-Compiler mit dem WRITE bis zur Version 2.50
     geschrieben wurde.

A + L
     Für ihren Oberon-Compiler, der das Schreiben von WRITE noch
     einfacher machte.

Lars Hanke
     Für Tips, Ktitik und Beta-Testing.

Thomas Lottermoser
     Für Tips, Kritik und die Literatur.

Thomas Stuff
     Für Tips, Ktitik und Beta-Testing.

Andreas Schulz
```

Für den sagenumwobenden DRACO-Club Deutschland.

COMMODORE
Für diesen schnuckeligen, kleinen Rechner, mit dem man Tag und Nacht viel Spaß haben kann. (Hehe...)

1.5 WRITE.guide/Einleitung

Einleitung
****

Ich besitze den AMIGA nun seit mehreren Jahren. In dieser Zeit bin ich mit vielen Texteditoren in Berührung gekommen. Anfangs das NOTEPAD und ED, später mit verschieden Version von EMACS sowie PD-Editoren wie DME, AZ, DED und JED als auch einige komerziellen Produkte. Mit allen konnte man Texte schreiben, doch auch alle ließen endweder bei der Geschwindigkeit oder beim Bediehnungskomfort zu wünschen übrig. So erschütterten einige durch eine unerträgliche Scrollgeschwindigkeit, andere erschwerten einem die Arbeit durch unerträgliche und unmögliche Tastaturbelegungen und - kombinationen. Wieder andere erschlugen mich durch viele, für einen Texteditor unnötige Funktionen, die zu Lasten der Größe des Editors gingen, oder glänzten durch unzureichende Editiermöglichkeiten. So beschloß ich alle Vorteile in einem Programm zu vereinen (was sonst !) und einen Editor zu schreiben, der allen meinen Anforderungen entsprach.

* WRITE als reiner Texteditor ist dazu gedacht Texte oder Programme zu schreiben.

* Er besitzt alle gängigen Blockoperationen (Cut, Copy, Paste...), sowie umfangreiche Funktionen zum Suchen und Ersetzen von Wörtern etc...

* Seine Speicherverwaltung ist völlig dynamisch. Es können somit beliebig viele Textfenster geöffnet werden, der Text kann, ebenso wie jede Zeile beliebig lang sein. Einzige Begrenzung ist der vorhandene Speicherplatz.

* WRITE wird voll von Intuition unterstützt. D.h. alle Funktionen können über Menüs aufgerufen werden, der Cursor kann mit der Maus gesetzt, Blöcke mit ihr markiert werden.

* Menüs und Tastatur können über eine Textdatei beliebig konfiguriert werden. Auch ist WRITE darüber hinaus weiter programmierbar. WRITE kann somit selbst in Details den Erfordernissen und Wünschen des Benutzers vollständig angepaßt werden.

* WRITE besitzt eine Undo-Funktion, mit der eine beliebige, voreinstellbare Zahl von Textveränderungen wieder rückgängig gemacht werden kann.

* WRITE beherrscht das automatische generieren von Backups. Dabei kann einfach eine Endung an die Datei angehangen, die Datei unbenannt

oder in ein angegebenes Verzeichnis unter einen festen Namen abgespeichert werden.

* Es können unterschiedliche Konfigurationen gleichzeitig benutzt werden. So kann man in einem Fenster mit einer TeX-Konfiguration arbeiten, während gleichzeitig in einem anderen Fenster typische Einstellungen zur Programmierung in C verwendet werden. Gleichzeitig kann man mit der MAILER.CONFIG eine Mail mit Wordwrap etc. schreiben.

* Jede Konfiguration kann auf einem eigenen Screen mit einem eigenen Font in den eigenen Farben arbeiten. WRITE ist in allen seinen Fenstern und Requestern völlig auflösungs- und fontsensitiv. Es wird immer der eingestellte DefaultFont eines Screen verwendet. Dabei kann es sich sogar um einen Proportionalfont handeln.

* WRITE kann Textefalten. Verschachtelte Textfalten sind möglich.

* WRITE läuft unter OS 2.0, OS 2.1 und OS 3.0

* WRITE unterstützt daher viele Features von OS 2.0, wie z.B. AppWindows, -menus und -icons, sowie auch einige Features von 3.0.

* WRITE unterstützt die locale.library ab OS 2.1 und ist dadurch fast völlig lokalisierbar.

* WRITE besitzt eine komfortable AREXX-Schnittstelle und kann so beliebig weiter ausgebaut werden. Über diese Schnittstelle kann man voll auf die über 150 Funktionen und mehr als 80 Variablen zugreifen.

* WRITE beinhaltet einige konfortable AREXX-Scripts für SAS C, DFA, und den OBERON-Compiler der Firma A+L AG. Desweiteren liegen Scripts für umfangreichere Textverarbeitungoperationen sowie einige Beispielscripts für komplexere Funktionen bei.

* WRITE besticht in vielen Funktionen durch seine Schnelligkeit.

* WRITE ist völlig über eine eingebaute GUI vollständig konfigurierbar. Wahlweise kann aber auch direkt in dem Konfigurationsfile editiert werden.

1.6 WRITE.guide/Copyright

```
                Copyright
*********



                 Demoversion
                  Copyright der Demoversion

                 Vollversion
                  Copyright der komerziell vertriebenen Version
```

Allgemein
Allgemeine Hinweise für beide Versionen

1.7 WRITE.guide/Demoversion

Demoversion
==========

* Die Demoversion von WRITE ist Shareware. Sie darf unter folgenden Bedingungen benutzt und weitergereicht werden:

* Eine Diskette, die dieses Demo von WRITE in irgendeiner Form beinhaltet, darf nicht für mehr als 5 DM oder deren Gegenwert in einer ausländischen Währung kopiert, verkauft, weitergereicht werden.

* Fred Fish ist es ausdrücklich erlaubt diese Demoversion in seine Sammlung aufzunehmen.

* Ebenso ist es erlaubt die Demoversion auf Aminet zu packen. Desweiteren darf sie auch auf einer AminetCD erscheinen.

* Das Paket darf nur vollständig weitergereicht werden !

* Dem Benutzer wird es gestattet, WRITE ein bis zwei Monate zu benutzen, dann muß er sich registrieren lassen oder den Gebrauch von WRITE einstellen.

1.8 WRITE.guide/Vollversion

Vollversion
==========

* Diese Version von WRITE fällt ebenfalls unter dem Begriff Shareware. Es gelten jedoch folgende Einschränkungen

* WRITE darf nicht...

  ... in irgendeiner Form weitergereicht werden.

  ... von jemand anders als mir persönlich verkauft oder vertrieben werden.

* Eine vollständige Version ist bei mir unter der oben genannten Adresse für 30DM zu erhalten.

* Alle Limitierungen in der Demoversion sind in der vollständigen Version aufgehoben.

* Jeder, der eine voll funktionsfähige Version von WRITE mehr als einmal benutzt ohne sie offiziell gekauft zu haben, wird aufgefordert, ja sollte sich von seinem Gewissen gezwungen fühlen, diese nachträglich zu erwerben, da in WRITE mittlerweile mehr als vier Jahre Programierarbeit stecken !

* Besitzer der vollständigen Version werden über Updates informiert und können diese für ein weiteres Entgeld erwerben.

* Ich (als der Autor von WRITE) verpflichte mich gravierende Mängel zu beheben.

* WRITE darf nur auf einen einzigen Rechner installiert sein. Er darf z.B. also nicht in Netzwerken oder auf Mehrplatzsystem benutzt werden. Sollte dies jedoch erforderlich sein, so bitte ich um Rücksprache.

Ich möchte hier außerdem ausdrücklich bemerken, daß ich nicht verpflichtet bin an WRITE weiterzuarbeiten. Ich kann jederzeit Teile des Editors ändern, rauschmeißen, neues einfügen. Das Format des Konfigurationsfiles kann sich jedezeit ändern. Kompatibilität zu älteren Versionen kann somit auch auf Grund der Leistungsoptimierung nicht immer gewährleistet werden. Ich kann jeder Zeit Preise und Kaufbedingungen ändern.

Das heißt natürlich nicht, daß ich mich nicht den Wünschen der User beuge. Einige Teile von WRITE wie z.B. die Undofunktion und Folds währen ohne die Wünsche meiner Betatester nicht implementiert worden. Dies heißt eher, daß ich, um den Editor zu verbessern, schlechtere onzepte etc. bereit bin herauszuschmeißen.

Ich mache darauf aufmerksam, daß die Programmiererei nur eines meiner Hobbies ist und ich auch noch nebenbei studiere. Sollte also nicht sofort am nächsten Tag WRITE in der Post liegen, so gilt der klassische Spruch: KEINE PANIK !!!. Einen bis eineinhalb Monate Bearbeitungszeit nehme ich mir in Ausnahmefällen schon heraus. Möglicherweise programmiere ich auch gerade an einer neuen Version mit doppelt so vielen Features... ?

Schließlich bin ich nicht in der Lage nach jedem Bugfix allen Besitzer eine neue Version zuzuschicken (ich habe einfach nicht das Geld !). Sollte es jedoch bei einzelnen Personen zu gravierenden, unumgehbaren Fehlern in hochwichtigen Funktionen kommen, so sollte sich immer etwas machen lassen. Spätestens dann, wenn man mir mit der ausführlichen Fehlerbeschreibung einen Rückantwortbrief beilegt.

Registierte Besitzer erhalten einen Keyfile, mit dem sie in der Lage sind jede neu herausgekommene Version sofort ohne weitere Umstände zu benutzen. Über größere Updates weren alle registrierten Benutzer postalisch/EMail benachrichtigt. Ein Update kann man auch direkt von mir gegen eine Gebühr von 5DM bzw. 10DM für alle Länder außerhalb Europas erhalten.

Die Zeit die ich in WRITE investiere, sowie die Höhe der Registrierungsgebühr hängen entscheidend von der Zahl der Registrierungen ab. Also ... Je mehr sich registrieren lassen ...

1.9 WRITE.guide/Allgemein

```
Allgemein
=========
```

* Ich kann keine Haftung für jegliche Schäden, die direkt oder indirekt durch WRITE enststehen, übernehmen. So bin ich z.B. nicht für versehendlich gelöschte oder durch Absturz verlorengegangende wichtige Dokumente, noch für qualmende Hardware, defekte Software, verwirrte Geister, gescheiterten Ehen etc., die irgendwie berechtigt oder unberechtigt mit WRITE in Verbindung gebracht werden können, verantwortlich zu machen. WRITE ist gut aber nicht perfekt. (noch nicht !)

* Jegliche Änderungen an Teilen dieses Paketes (Docs...) sind verboten !

* Das widerrechtliche Benutzen (klauen) von Teilen dieses Paketes ist ebenfalls verboten !

* Es ist verboten WRITE zu dissassemblieren, decodieren, decompilieren...

* Der Käufer akzeptiert mit der Benutzung von WRITE obige Konditionen.

* Jeder, der gegen diese Bedingungen verstößt, sollte sich bewust sein, daß damit gegebenenfalls eine strafbare Handlung begeht, gegen die ich entsprechend vorgehen werde. Speziell das Vertreiben der Vollversion von WRITE als Raubkopie fällt unter diesen Punkt.

1.10 WRITE.guide/Registrierung

```
                Registrierung
*************
```

Wie bekomme ich den nun die Vollversion von WRITE ?

1. Sie schicken mir offiziell eine Bestellung per Post oder EMail mit ungefähr folgenden Wortlaut

   "Hiermit bestelle ich die neuste Vollversion von WRITE. Ich akzeptiere die in der Anleitung unter den Punkten Copyright und Registrierung genannten Kaufbedingungen und Hinweise."

   Fügen sie dem Brief auf jeden Fall ihre komplette Adresse, wenn möglich auch ihre EMailadresse und ihre Telephonnummer bei. Sollte es zu Rückfragen kommen, ist es sicherlich im ihren Sinne, wenn ich sie so schnell wie möglich erreiche.

Geben sie an, wie viele Versionen sie von WRITE haben wollen.

Geben sie an, für welchen Prozessor die Vollversion kompiliert werden soll. Momentan habe ich die Möglichkeit Code für den 68000 aber auch für 68020 und 68030 zu erzeugen.

Geben sie an, ob sie WRITE sofort bekommen wollen, oder noch gegebenenfalls auf eine neue Version warten wollen.

2. Lassen sie mir das Geld zukommen. Sie können mir das Geld in Form von Bargeld oder auch als Scheck zukommen lassen. Auch eine Überweisung auf das unter
   Autor
   angegebene Konto sowie eine
   postalische Zustellung sind möglich. Naturalien nehme ich nicht an. Im Einzelfall läßt sich aber auch über eine Begleichung durch ein eigenes, sharewarebehaftetes Profukt reden.

   Für Versendung auserhalb Europas bitte ich sie, weitere 5DM zu überweisen !

   Stellen sie auf jeden Fall sicher das ich mit dem Geld auch ihren Namen und Addresse erhalte, damit ich ihnen auch WRITE zustellen kann.

3. Warten sie darauf, daß der Postbote ihnen WRITE in Haus bringt.

4. Klatschen sie in die Hände, freuen sie sich und werden sie sich bewußt, daß sie einen weiteren, entscheidenen Schritt in ihrem Leben getan haben.

5. Ich möchte noch einmal darauf hinweisen, daß wenn sie über einen Netzanschluß (InterNet,Fido,Zerberus,Maus etc.) mir ihre Addresse unbedingt mitteilen sollten. Dies ermöglicht es mir ihnen Updates ggf. zuzuschicken, sie um ihre Meinenung zu Fragen, ihnen bei Problemen zu Helfen etc.

1.11 WRITE.guide/Installation

Installation
****

Folgende Sachen sind unbedingt zu beachten:

* Auf ihren System muß OS 2.0 oder höher installiert sein ! Einige (unwesendliche) Features lassen erst ab 3.0 nutzen.

* Folgende Libraries sollten sich im LIBS:-Verzeichnis befinden. Sie sind Bestandteil des Betriebssystems.

  asl.library
  diskfont.library

commodities.library
iffparse.library (optional)
optional für REXX:

rexxsyslib.library
rexxsupport.library

* Es sollte genügend Speicher vorhanden sein ! Obwohl WRITE speichersparsam programmiert ist, braucht er doch einige 100KByte an Speicher. Doch ist es möglich durch Verzicht auf ein weinig Komfort den Speicherbedarf zu verinngern.

* WRITE braucht recht viel Stack (ca. 20000 Bytes). Sollte dieser nicht vorhanden sein, setzt WRITE ihn selbst auf 30000 Bytes herauf.

* Auf der Diskette, die sie bekommen haben, sollte sich ein Installer-Skript für den Installer, den sie mit ihren Betriebssystemdisketten bekommen haben, befinden. Alles was sie also im Normalfall zu tun haben, ist mit einem Doppelklick auf das entsprechende Icon das automatische Installationsprogramm zu starten und die Fragen ihren Wünschen entsprechend zu beantworten.

  Wollen sie dennoch WRITE manuell installieren, sollten sie wie folgt vorgehen.

  1. Einige Dateien auf der Diskette sind mit lha gepackt und müssen gegebenenfalls entpackt werden. Sollten sie eine Version von Lha besitzen, benutzen sie das Programm im c-Verzeichnis.

  2. Der(Die) Konfigurationsfile(s) sollte(n) auf folgende Weise installiert werden:

     1. Schaffen sie auf ihrer Platte ein neues Verzeichnis mit beliebigem Namen.

     2. Kopieren sie alle Konfigurationsdateien dort hinein.

     3. In der Umgebungsvariable WRITE.CONFIG des Betriebsystems sollte der vollständige Pfad dieses Verzeichnisses stehen. Dies kann dadurch erreicht werden,...

        ... daß man diese mit dem Befehl SetEnv (oder Set für eine COMMODORE-SHELL) manuell nach jedem Bootvorgang setzt.

        ... daß man einen diese Befehle in seine startup-sequence oder noch besser in die user-startup einfügt.

        ... daß man nach Setzen dieser Variable den File WRITE.CONFIG aus dem ENV:-Verzeichniss ins ENVARC:-Verzeichnis kopiert, so daß die Umgebungsvariable automatisch nach jedem Boot-Vorgang gesetzt wird.

     (Letzteres ist meiner Meinung nach die geschickteste Methode !)

WRITE such nun alle Konfigurationsdateien in diesen Verzeichnis, oder, falls die Umgebungsvariable nicht gesetzt wurde, im aktuellen. Zur Referenz des Konfigurationsfiles brauchen sie somit immer nur den Dateinamen ohne Pfad.

3. Kopieren sie die garbagecollector.library ins LIBS:-Verzeichnis.

4. Kopieren sie die Datei WRITE.guide in ein Verzeichnis, wo sie von AmigaGuide bzw. MultiView gefunden wird. Der Guide-File ist für die Online-Hilfe wichtig.

   Es besteht auch die Möglichkeit die Umgebungsvariabel WRITE.guide mit dem kompletten Pfad Des Guide-Files zu setzten.

5. Kopieren sie die .catalog-Files in die entsprechenden Verzeichnisse ihrer Workbench.

6. Kopieren sie die REXX-Scripts in ein Verzeichnis, welches im Suchpfad von REXX liegt.

7. Kopieren sie nun WRITE selbst samt seinem Icon in das von ihnen gewünschte Verzeichnis. Setzen sie die Umgebungsvariable WRITE nach einer der oben beschriebenen Methoden dauerhaft auf den vollständigen Pfad von WRITE (z.B. dh0:WRITE oder C:WRITE). Einige AREXx-Scripts lesen diese Variable aus, um WRITE, wenn noch nicht gestartet, hochzufahren.

* Beenden sie möglichst alle weiteren Programme, oder machen sie einen Reset, starten sie das GarbagePrefs-Programm in der Prefs-Schublade der Installationsdiskette, wählen sie den Menüpunkt Editieren/Werte vorschlagen bestätigen sie mit Speichern und verlassen das Programm.

* Starten sie WRITE.

Schließlich (auch nach der Installation durch das Installationsscript) sollten unbedinngt noch folgende Dinge getan werden :

* Kontrollieren sie, ob die Einstellungen im File STARTUP.CONFIG in in dem Icon von WRITE ihren Wünschen entsprechen. Es kann sein, daß die dort definierten Hotkeys sich mit bereits vorhandenen Hotkeys überschneiden, oder das die eingestelleten Backup-Modi auf nicht existierende Verzeichnisse zeigen.

1.12 WRITE.guide/Konzept

Konzept

****

Das Prinzip

Allgemeine Einleitung

Der Ed
Die interne Reprensentation eines Textes

1.13 WRITE.guide/Das Prinzip

Das Prinzip
==========

Viele Texteditoren sind durch eine feste Tastatur- und Menübelegung recht beschränkt in ihren Möglichkeiten. Stört den Benutzer irgendetwas an der Handhabung so muß er sich daran gewöhnen oder mit einen anderen Texteditor vorlieb nehmen.

Bei der Programmierung von WRITE wurde versucht dem Benutzer Möglichkeiten zur Beeinflussung der Handhabung, des äußeren Erscheinungsbildes, mitzugeben.

Dies wird dadurch erreicht,daß WRITE mittels einer kleinen in den Editor eingebauten Interpretersprache, einer Mischung aus der Batchsprache des Betriebssystems und AREXX, programmierbar ist.

Der Editor führt also direkt keine Tastaturkommandos oder Menüpunkte aus, sondern nur noch Befehlsfolgen, die der Tastatur und Menü beliebig zugewiesen werden können. Der Benutzer kann frei bestimmen, was passiert, wenn diese Taste gedrückt oder jener Menüpunkt ausgewählt wird. Ja selbst die Menüs an sich sind frei definierbar; und sollten die Möglichkeiten des internen Interpreters nicht reichen, so können AREXX-Makros verwendet werden. Jeder interne Befehl ist auch über AREXX erreichbar.

1.14 WRITE.guide/Der Ed

Der Ed
======

Die interne Struktur zum Verwalten eines Textes wird Ed genannt. Für jeden Text, der geladen wird, gibt es einen Ed. Im Prinzip kann man einen Ed mit einem Editorfenster identifizieren (Obwohl nicht jeder Ed ein offenes Fenster besitzen muß).

Jedem Ed kann bei seiner Erschaffung (mittels
NewEd
) ein eigene
Konfigurationsdatei zugewiesen werden. WRITE läd diese Datei, wenn sie nicht schon bereits für einen anderen Ed benutzt wurde, automatisch nach. D.h., startet man WRITE so, daß er keinen Ed öffnet, läd er auch keinen Konfigurationsfile.

Dies klingt alles recht kompliziert. Sie vermuten, daß sie nun vor der ersten Benutzung erst einmal stundenlang die Anleitung studieren müssen. Doch keine Panik! Es liegt ein dokumentierter Beispielfile bei, der die Möglichkeiten von WRITE ausführlich demonstriert. Spielen sie mit diesem ein wenig herum, merken sie sich Dinge, die sie stören, schauen sie sich ihre Realisierung im Konfigurationsfile an, versuchen sie mittels der Dokumentation zu verstehen, was passiert und Überlegen sie sich , wie sich dies anders machen ließe, um dann schließlich den Konfigurationsfile zu ändern. Achten sie dabei darauf, daß sie immer einen lauffähigen Konfigurationsfile parat haben, da WRITE sich mit einem fehlerhaften Konfigurationsfile nicht starten läßt. Sie währen somit nicht in der Lage WRITE zu benutzen, um die Fehler zu korriegieren. Das Beste ist es, die geänderte Stelle zu markieren und sie mit dem Menüpunkt INTERN/Parse block auf ihre syntaktische Richtigkeit zu überprüfen.

1.15 WRITE.guide/Syntax

```
                Die Syntax des internen Interpreters
***********************************


                allgemeine Syntax
                 Allgemeine Syntax

                Typen
                             Die verschieden Typen

                Variablen
                         Variablen

                Konstanten
                        Konstanten
```

1.16 WRITE.guide/allgemeine Syntax

```
                Allgemeine Syntax
=================
```

Der interne Interpreter verarbeitet beliebig lange Befehlsfolgen. Diese bestehen aus einer Liste von Befehlen, im weiteren auch Funktionen genannt, samt ihrer Argumente. Dabei besitzt jeder Befehl eine feste Zahl von Argumenten von jeweils klar definiertem Typ. Argumente dürfen also z.B. nicht einfach weggelassen werden. Ein Beispiel:

```
     Befehl1 Argument1
```

```
     Befehl2 Argument1 Argument2 Argument3
```

Dem interne Parser (im Gegensatz zu AREXX) ist es dabei egal wie die Befehlsfolge formatiert ist. So kann obriges Beispiel auch folgenderweise geschrieben werden:

```
     Befehl1 Argument1 Befehl2
     Argument1 Argument2
     Argument3
```

Der Interpreter arbeitet nun die Befehlsfolge von oben nach unten ab. Doch ist es dies nicht immer sinnvoll. So öffnet der 1. Befehl z.B. ein Fenster, während der 2. in dieses einen Text schreibt. Was würde passieren wenn z.B. das Fenster wegen Speichermangel nicht geöffnet werden kann ? Es würde versucht in ein nicht vorhandenes Fenster eine Text zu schreiben. Deshalb geben einige Befehle einen Fehler zurück Ist ein Fehler aufgetreten, so wird die Abarbeitung der Befehlsfolge sofort abgebrochen.

Eine komplette Auflistung aller Funktionen finden sie im Kapitel

```
                Funktionsbeschreibung
```

1.17 WRITE.guide/Typen

```
                Die verschieden Typen
====================
```

Der Interpreter unterscheidet zwischen 4 verschieden Typen:

1. Zahlen. Zahlen sind vorzeichenlose Dezimalzahlen von 0 bis 214783647. Einige Beispiele:

   ```
        1234 , 4567 , 229874 , 0
   ```

   Zahlen sind zuweisungskompatibel zu Zeichenketten. Ihr Inhalt wird bei einer Zuweisung automatisch in einen String umgewandelt.

2. Strings. Es gibt zwei unterschiedliche Arten

   1. Wörter. Wörter bestehen aus einem optionalen Unterstrich oder Klammeraffen (@), einem Buchstaben und weiteren Buchstaben oder Ziffern. Bei jedem Wort schaut der Interpreter nach, ob es sich bei dem Wort nicht um einen Befehl oder eine Variable handelt. Befehle werden ausgeführt, Variablen durch ihren Inhalt ersetzt. Beispiele:

      ```
           _EinWort , W2345r , _w2swd
      ```

   2. Zeichenketten. Zeichenketten können im Gegensatz zu Wörtern aus belieben Zeichen bestehen. Sie werden durch sogenannte Quotes umschlossen. Quotes können sein: "",(),`'.Variablen können mittels $VariableName$ in die Zeichenkette eingefügt werden. Auch können Zeichenketten mittels folgender Syntax

über mehrere Zeilen verteilt werden:

```
"1. Teil des Strings"+
(2. Teil des Strings)
```

Zeichenketten sind bedingt zuweisungskompatibel zu Zahlen. D.h. steht in dem String eine Zahl und nur diese Zahl, dann kann dieser String auch einer Zahl zugewiesen werden, ansonsten beschwert sich der Parser mit einer entsprechenden Meldung. Beachten sie daß ihr String zwar auch negative Zahlen beinhalten darf, diese jedoch vom Interpreter wiedder in positive Zahlen umgebandelt werden. Auch ist es möglich, daß der String nicht eine Dezimalzahl sondern eine Hexadezimalzahl der Form "4E75H" beinhalten kann.

3. Tags. Tags bestehen aus einer einleitenden,geschweiften Klammer { beliebig vielen Zahlen oder
   Konstanten
   und einer abschließenden,
   geschweiften Klammer }. Leere Tags können und sollten weggelassen werden.

4. Funktionsliste. Funktionlisten bestehen aus einer Befehlsfolge (d.H. einer Reihe von Befehlen sammt ihren Argumenten) und einem abschließenden Semikolon. Beispiel:

```
Befehl1 Argument 1 Befehl2 Argument1 Argument2 Argument3;
```

5. Kommentare. Kommentare fangen mit /* an und hören mit */ auf. Alles, was im Kommentar steht wird vom Parser ignoriert. Kommentare können überall gesetzt werden. Kommentare können auch verschachtelt werden. Folgendes Konstrukt ist deshalb erlaubt :

```
/* Ein paar Sachen zum Debugen

  /* Aufruf 1 */

  blabla 1 2 3

  /* Aufruf 2 */

  blupblup "Eine Wasserblase "

*/
```

1.18 WRITE.guide/Variablen

```
                Variablen
=========
```

Damit der Benutzer nicht nur die internen Funktionen aufrufen, sondern auch auf interne Einstellungen zugreifen und diese verändern kann, gibt es Variablen. Variablen sind vom Typ Zeichenkette oder Zahl. Auch hier gilt : Zahlen sind zuweisungskompatibel zu Zeichenketten, Zeichenketten

bedingt zuweisungskompatibel zu Zahlen. Dies ist mit Vorsicht zu
behandeln. Da der Parser wärend des Parsens nicht feststellen kann, ob
die Variable nun eine Zeichenkette oder eine Zahl beinhaltet, fällt
diese Aufgabe dem Interpreter zu, der dann einfach mit einem Fehler
abbrechen wird.

Möchte man nun z.B. als Parameter einer Funktion statt einer Zahl eine
Variablen angeben, so tätigt man dies einfach durch die Nennung des
Variablenamens. Hat zum Beispiel die Variable x den Inhalt 25 so sind die
Funktionsaufrufe

```
     MacheMitZahl 25
     MacheMitZahl x
```

identisch.

Im Zusammenhang mit Funktionen gibt es zwei spezielle Variablen:
_RS
 und

_RN
. Sie existieren, da einige Funktionen ein Resultat zurückgeben. ↩
   Ist
der Rückgabewert vom Typ Zahl, dann steht er in
_RN
, ist er vom Typ
Zeichenkette, dann steht er in der Variable
_RS
. Auf den Rückgabewert
einer Funktion kann also solange zugegriffen werden, bis er von einer
Funktion selben Rückgabetyps überschrieben wird.

Wie schon in
Typen
 erwähnt können Variablen durch $VariableName$ in
einen String eingefügt werden. So ergibt Message "Zeit : $_Time$" zum
Beispiel einen Requester mit der aktuellen Zeit.

Bitte beachten sie, daß einige Variabeln nur geschrieben/ausgelesen
werden können, wenn der entsprechende Ed/die entsprechende Konfiguration
aktiviert ist. Dies geschieht mit
LockWindow
 und
GetConfig
.

Eine ausführliche Auflistung aller Variablen finden sie im Kapitel

Variablebeschreibung
.

1.19 WRITE.guide/Konstanten

```
                Konstanten
==========

Konstanten wurden hauptsächlich für den Typ Tag eingeführt. Durch
Kombination der beiden ist es möglich, auf äußerst einfache Weise das
Verhalten von Funktionen zu beeinflussen. Konstanten verhalten sie für
den Benutzer genauso wie Variablen. Der einzige Unterschied ist halt der,
das der Inhalt einer Variablen nach dem Start von WRITE immer gleich
bleibt und das sie nicht geändert werden kann. Von REXX aus können
Konstanten mit dem Befehl
                GetConst
                 ausgelesen werden.
```

1.20 WRITE.guide/ToolTypes

```
                ToolTypes
*********

Über die Toltypes können einige wichtige Einstellungnen von WRITE gleich
beim Starten von WRITE getätigt werden.

Beachten sie, das WRITE, im Gegensatz zum üblichen Vorgehen, seine
ToolTypes immer analysiert.  D.H., sowohl beim Start von der Workbench,
als auch beim Start aus einer Shell, werden die ToolTypes gelesen.

Die ToolTypes und ihre Bedeutung :

   * APPMENU Setzt die Variable
               _AppMenu
               .

   * APPICON Setzt die Variable
               _AppMenu
               .

   * APPWIN Setzt die Variable
               _AppWindow
               .

   * SLEEPMODE Setzt die Variable
               _SleepMode
               .

   * SOUND Setzt die Variable
               _Sound
               .

   * STDCONFIG Setzt die Variable
               _DefaultConfig
                  * GUIDEMODE Ist diese Variable FALSE, so öffnet WRITE beim  ↩
                    Hochfahren
     gleich den Hilffile WRITE.guide.  Ist die Variable TRUE, so wird der
```

```
     Guidefile erst exeplizit beim Aufruf der Hilfe geöffnet und
     anschließend wieder geschlossen.

     Die erste Methode ist schneller. Der Guide wird einmal geladen und
     steht dann immer schnell da speicherresident zur Verfügung. Der
     Nachteil ist, daß der Guidefile permanent im Speicher gehalten
     wird. Nach groben Schätzungen müßten dies mindesten 300 - 400 KB
     sein.  Die zweite Methode ist wesendlich langsamer (auf meinen 68020
     dauert es schon ein paar Sekunden bis AmigaGuide da ist) aber halt
     speicherplatzsparend, da der GuideFile bei jeder Anfrage neu geladen
     wird.
```

1.21 WRITE.guide/Sound

```
                Sound
*****

Neben der Tonausgabe mittels des Befehls
                Beep
                 unterstützt WRITE in
bestimmten Situationen das Abspielen von Samples über das Programm Upd
((C) Jonas Petersson).  Dazu müssen in Upd.ids folgende weitere id's
definiert werden, die dann in den entsprechenden Situationen falls
vorhanden abgespielt werden.

   * write_start Wird beim Start von WRITE gespielt.

   * write_ask Wird jedesmal, wenn der
               Ask
               -Requester erscheint, gespielt.

   * write_message Ertönt bei jedem
               Message
               -Requester.

   * write_getstring Für den
               GetString
               -Requester.

   * write_getnumber Entsprechend für den
               GetNumber
               -Requester.

   * write_list Für die List-Requester.

   * write_manager Beim Erscheinen des
               WinManager
               s

   * write_help Wird bei Erscheinen des Help-Requesters gespielt.

   * write_about Ebenso für den
               About
```

```
                -Requester.

   * write_getfile FileRequester.

   * write_getfont FontRequester.

   * write_gprefs Wird beim Öffnen des GlobalPreferences-Requesters
     geöffnet.

   * write_prefs Wird beim Öffnen des Preferences-Requesters geöffnet.

   * write_end Wird bei Verlassen des Editor gespielt.

   * write_macropannel Wird beim Öffnen des
                MacroPannel
                s gestartet.

   * write_getfindreplace Wird beim Öffnen des
                GetFindReplace
                -Requesters
     abgespielt.
```

1.22 WRITE.guide/Editorfenster

```
                Editorfenster
*************
```

Das Editorfenster gliedert sich in drei wesendliche Teile : Die Titelzeile mit einigen wichtigen Informationen über den Text, den Scrollbalken auf der rechten Seite, so wie der eigendliche Editierbereich. Dr Scrollbalken verhält sich wie üblich und bedarf wohl keiner weiteren Erklärung, ähnlich wie der Editierbereich. Im weitern soll auf die Bedeutung der verschieden Mauszeiger und der kryptischen Zeichen in der Titelzeile des Fensters eingegangen werden.

1. Der normale Mauszeiger. Dies ist der Betriebssystemmauszeiger der normalerweise benutzt wird.

2. Der Busy-Mauszeiger, eine kleine Uhr. Dieser Mauszeiger erscheint immer dann, wenn WRITE gerade arbeitet (z.B. beim Abspeichern) und auf die Eingaben des Benutzers nicht reagieren kann. Bitte warten sie dann mit ihren Eingaben bis der normale Mauszeiger wieder erscheint.

3. Der Sleep-Mauszeiger. Dieser Mauszeiger erscheint immer dann, wenn die graphische Ausgabe von WRITE abgechaltet ist. Dies ist sinnvoll, wenn ein Script viele Änderungen am Text vornimmt, da das Abschalten zu einem nicht unwesendlichen Performancegewinn führt. Der Benutzer sollte auch hier alle Eingaben unterlassen.

4. Der Markierungs-Mauszeiger, ein Visier. Dieser Mauszeiger erschein immer dann, wenn man eine Blockmarke gesetzt hat. WRITE erwartet dann, daß man eine zweite Marke setzt, um einen Block zu markieren.

Die Titelzeile gliedert sich ebenfalls in vier Teile.

Im ersten Teil zeigen einige Buchstaben wesendliche Informationen zum Text. Erschein an der ersten Stelle eine M so wurde eine Marke gesetzt, erscheint ein B so wurde eine zweite Marke gesetzt und damit ein Block markiert. Erscheint an der zweiten Stelle ein *, so wurde der Text nach dem letzten Abspeichern oder Laden verändert. Ein R an vierter Stelle bedeutet, das REXX installiert wurde und WRITE REXX-Scripts ausführen kann. Ein C an vierter Stelle bedeutet, daß WRITE beim Suchen zwischen Groß- und Kleinschreibung unterscheidet. Ein S an fünfter Stelle zeigt an, daß sich WRITE im Sleep-Modus befindet. Und ein E an Position 6, zeigt an das der Text editiert werden kann.

Als nächstest folgen drei Zahlen. Diese stehen für die Spalte und die Zeile im Text, in der sich der Cursor befindet, und die Länge des Textes in Zeilen.

Anschließend folgt, wenn die Undo-Funktion aktiviert ist (
_Undo
), in
eckigen Klammern die Zahl der Textänderungen die WRITE sich gemerkt hat und die man rückgängig machen kann.

SAchließlich, als letztes, steht in der Titelzeile der Name des aktuellen Files.

1.23 WRITE.guide/Requester

Requester
****

* Die Rquester von WRITE sind alle fontsensitiv. Das heißt, daß sie sich an die Größe des eingestellten Systemfonts anpassen. So ist selbst bei großen Fontgrößen, wie man sie z.B. als Besitzer einer GraphiKkarte benutzt, ein übersichtliches, lesbares, komfortables Layout gewährleistet.

* Alle Requester unterstützen Shortcuts. Das bedeutet, daß die meisten Gadgettypen nicht nur mittels der Maus, sondern auch über die Tastatur selelktiert werden können. Angezeigt wird der Shortcut durch einen Strich unter dem entsprechenden Buchstaben. Außerdem gibt es für jedes Fenster meist ein Gadget, welches über die Return-Taste selektiert werden kann, und eins, welches man mit der Escape-Taste erreichen kann. Ersteres wird durch angezeigt, daß der Text des Gadgets fettgedruckt, zweiteres dadurch, daß der Text fettgedruckt und kursiv ist. Bitte machen sie sich bei der Belegung der Shortcuts, der Belegung von Return- und Esacapegadgets der Reihenfolge Gedanken, versuchen sie möglich die Logik der Betriebssystemsrequester zu kopieren...

* Man kann für die internen Requesteraufrufe global über die Variable

```
            _ReqToMouse
            , bzw. für die meisten Requester bei extrenen Aufrufen
     über das Tag  @TOMOUSE WRITE so konfigurieren, daß Requester immer
     unter dem Mauszeiger geöffntet werden. Dies ist vorteilhaft für
     Leute mit Tools, die das Fenster unter dem Mauszeiger automatisch
     aktivieren, oder auch für Leute mit großen Screens.
```

1.24 WRITE.guide/Listen und Buffer

```
                Listen und Buffer
*****************

Listen und Buffer werden in WRiTE dynamisch verwaltet. Dies bedeutet,
daß man beliebig viele Listen und Buffer haben kann, deren Größe nur
durch den vorhandenen Speicher begrenzt ist. Alle Listen und Buffer haben
einen Namen, über den sie jederzeit referenziert werden können. Eine
Liste oder Buffer mit einem bestimmten Namen erzeugt man dadurch, daß
man etwas reinschreibt oder löscht.

Ein Eintrag in eine Liste darf maximal 255 Zeichen lang sein. Alles
darüber wird abgeschnitten.

WRITE besitzt intern beim Programmstart schon einige Listen:

   * List-List Eine Liste, in der alle Listen stehen.

   * Buffer-List Die Liste aller Buffer, die momentan existieren.

   * FindHistory Liste aller Suchwörter seit dem Programmstart.

   * ReplaceHistory Liste aller Ersetzungswörter seit dem Programmstart.

   * DeleleteLine-History Liste aller durch
               DeleteLine
                gelöschter Zeilen.

Listen sind sehr leistungsfähig. Ein Beispiel dafür ist das REXX-Script
ListDemo.wrx.

Ein anderes Beispiel, welches eine DeleteWord / UndeleteWord-Funktion als
ein Menüpunt implementiert, soll folgen:


     ITEM "Delete word" "control delete"
       IF
         COMPARE _CURRENTWORD "";
         NOP;
         PUSH "CED-Word-History" _CURRENTWORD
         DELETEAREA @SOW @SOW @EOW @EOW {@SILENT};
       ;

     ITEM "Undelete word" "control alt delete"
       POP "CED-Word-History"
```

```
       WRITETEXT _RS
       ;
```

1.25 WRITE.guide/PublicScreens

```
                Public Screens
**************
```

Wenn nicht anders angeben, öffnet WRITE seine Fenster immer auf der Workbench. Es ist aber auch möglich Fenster auch auf anderen, öffentlichen Screens (Public Screens) zu öffnen. WRITE besitzt dabei keinen eigenen Screenmanager, daß heißt, daß WRITE nicht in der Lage ist selbst Screens zu öffnen, sondern ist darauf angewiesen, daß andere Programme diese Screens für ihn öffnen.

So kann man in dem entsprechenen Konfigurationsfile ein Programm starten, welches einen öffentlichen Screen öffnet, oder über AREXX ein bereits laufendes Programm dazu veranlassen. Anschließend wird dieser Screen über die Funktion

```
                Screen
```

angemeldet. Der Screen wird gelockt, d.h. dem Screenmanager wird nicht erlaubt das Fenster wieder zu schließen. Dieser wird erst wieder freigegeben, wenn man für diese Konfiguration einen anderen Screen anmeldet oder die Konfiguration beendet. WRITE wird also immer, wenn die Konfiguration aktiv ist, alle seine Fenster auf diesem Screen öffnen. Über die UserFunktion 4 , die mittels

```
                SetUserFkt
```

gesetzt werden kann, kann man nun das Screenmanagerprogramm dazu veranlassen, den Screen wieder zu schließen, wenn die Konfiguration freigegeben wird.

1.26 WRITE.guide/Konstantenbeschreibung

```
                Konstantenbeschreibung
**********************
```

```
                TRUE

                FALSE

                LOWER

                HIGHER

                EQUAL
```

Weitere wichtige Konstanten als Platzhalter für Textpositionen.

Sie können immer dann verwand werden, wenn Funktionen Positionsangaben
im Text (in Form von x,y oder auch nur y-Koordinaten) erwarten.

CURSOR

MARKA

MARKB

SOL

EOL

SOW

EOW

SOT

EOT

SOP

EOP

SOS

MOS

EOS

1.27 WRITE.guide/TRUE

TRUE
====

Schreiben : Nein

Beschreibung : Ist immer 1. Dient dazu Variablen, die das Verhalten
eines Schalters haben (0 = AUS, #0 = AN), verständlich einen Wert
zuzuweisen.

1.28 WRITE.guide/FALSE

FALSE
=====

Schreiben : Nein

```
Beschreibung : Ist immer 0.  Dient dazu Variablen, die das Verhalten
eines Schalters haben (0 = AUS, #0 = AN), verständlich einen Wert
zuzuweisen.
```

1.29 WRITE.guide/LOWER

```
                LOWER
=====

Schreiben    : Nein

Beschreibung : Ist immer 1.  Möglicher Rückgabewert der Funktion

                Compare
                .
```

1.30 WRITE.guide/HIGHER

```
                HIGHER
======

Schreiben    : Nein

Beschreibung : Ist immer 2.  Möglicher Rückgabewert der Funktion

                Compare
                .
```

1.31 WRITE.guide/EQUAL

```
                EQUAL
=====

Schreiben    : Nein

Beschreibung : Ist immer 0.  Möglicher Rückgabewert der Funktion

                Compare
                .
```

1.32 WRITE.guide/CURSOR

```
CURSOR
======

Schreiben    : Nein

Beschreibung : Steht die x-, bzw. y-Koordinaten des Cursors.
```

1.33 WRITE.guide/MARKA

```
                MARKA
=====

Schreiben    : Nein

Beschreibung : Steht für die Position der Blockanfangsmarke. Nur gültig,
wenn eine entsprechende marke gesetzt wurde. Siehe auch
                _Marked
```

1.34 WRITE.guide/MARKB

```
                MARKB
=====

Schreiben    : Nein

Beschreibung : Steht für die Position der Blockendemarke. Nur gültig,
wenn eine entsprechende marke gesetzt wurde. Siehe auch
                _Marked
```

1.35 WRITE.guide/SOL

```
                SOL
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten des Zeilenanfanges der
aktuellen Zeile. Identisch mit(1,
                _YPos
                ).
```

1.36 WRITE.guide/EOL

```
                EOL
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten des Zeilenendes der
aktuellen Zeile. Identisch mit(Länge der aktuellen zeile+1,
                _YPos
                ).
```

1.37 WRITE.guide/SOW

```
                SOW
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten des Anfanges des aktuellen
Wortes. Nur gültig, wenn Cursor auf einem Wort steht. Siehe auch

                _CurrentWord
                .
```

1.38 WRITE.guide/EOW

```
                EOW
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten des Endes des aktuellen
Wortes. Nur gültig, wenn Cursor auf einem Wort steht. Siehe auch

                _CurrentWord
                .
```

1.39 WRITE.guide/SOT

```
SOT
===
```

```
Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten des Textanfanges.
Identisch mit (1,1).
```

1.40 WRITE.guide/EOT

```
                EOT
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten des Textendes. Identisch
mit (Länge der letzten Zeile+1,
                _Length
                ).
```

1.41 WRITE.guide/SOP

```
SOP
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten des Anfanges des aktuellen
Paragraphen. Paragraphen sind durch Leerzeilen getrennte Textblöcke.
```

1.42 WRITE.guide/EOP

```
EOP
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten des Endes des aktuellen
Paragraphen. Paragraphen sind durch Leerzeilen getrennte Textblöcke.
```

1.43 WRITE.guide/SOS

```
SOS
===
```

```
Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten der oberen, linken Ecke des
aktuellen Fensters.
```

1.44 WRITE.guide/MOS

```
MOS
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten der Mitte des aktuellen
Fensters.
```

1.45 WRITE.guide/EOS

```
EOS
===

Schreiben    : Nein

Beschreibung : Steht für die x-,y-Koordinaten der unteren, rechten Ecke
des aktuellen Fensters.
```

1.46 WRITE.guide/Variablebeschreibung

```
                Variablebeschreibung
********************


                _AColor

                _BColor

                _CColor

                _DColor

                _ReadTabs

                _WriteTabs

                _CaseSense

                _Optimize
```

```
_XPos

_YPos

_Changed

_FileName

_FindString

_ReplaceString

_Length

_RS

_RN

_WBStarted

_LoadTab

_SaveTab

_Version

_ChipMem

_FastMem

_PublicMem

_Wins

_Time

_Path

_FRPattern

_PatCase

_ConfigPath

_ShowSpace

_ShowEOL

_PatNoCase

_CurrentChar

_CurrentWord

_CurrentLine
```

_TabLength

_MaxBuffer

_REXX

_WordDef

_WriteIcon

_AppIcon

_AppWindow

_AppMenu

_Cursor

_AutoIndent

_OverwriteIcon

_SleepMode

_EditMode

_Marked

_MarkAx

_MarkAy

_MarkBx

_MarkBy

_WinMode

_CurrentID

_VersionString

_File

_FilePath

_REXXPortName

_REG1

_REG2

_Sound

_DefaultTool

_Language

```
_CurrentConfig

_DefaultConfig

_AutoFree

_WinWidth

_WinHeight

_WordWrap

_RightMargin

_ScreenWidth

_ScreenHeight

_Undo

_ScrRelWidth

_ScrRelHeight

_WordOnly

_ReqToMouse

_EColor

_FColor

_AutoFold

_FoldStart

_FoldEnd

_DelFoldMark

_Fold

_Screenname

_PathPath

_User

_CollectorMem
```

1.47 WRITE.guide/_AColor

```
_AColor
=======

Schreiben    : Ja

Beschreibung : Setzt die aktuelle Vordergrundfarbe (Farbe der Schrift)
auf die entsprechende Farbnummer (0..MAXCOLOR-1).  Die Farben werden
anhand der Reihenfolge im Palletterequester von 0 an aufwärts
durchnummeriert.
```

1.48 WRITE.guide/_BColor

```
_BColor
=======

Schreiben    : Ja

Beschreibung : Setzt die aktuelle Hintergrund auf die entsprechende
Farbnummer (0 ... MAXCOLOR-1).  Die Farben werden anhand der Reihenfolge
im Palletterequester von 0 an aufwärts durchnummeriert.
```

1.49 WRITE.guide/_CColor

```
_CColor
=======

Schreiben    : Ja

Beschreibung : Setzt die aktuelle Vordergrundfarbe (Farbe der Schrift)
für markierten Text auf die entsprechende Farbnummer (0..MAXCOLOR-1).
Die Farben werden anhand der Reihenfolge im Palletterequester von 0 an
aufwärts durchnummeriert.
```

1.50 WRITE.guide/_DColor

```
_DColor
=======

Schreiben    : Ja

Beschreibung : Setzt die aktuelle Hintergrundfarbe für markierten Text
auf die entsprechende Farbnummer (0..MAXCOLOR-1).  Die Farben werden
anhand der Reihenfolge im Palletterequester von 0 an aufwärts
durchnummeriert.
```

1.51 WRITE.guide/_ReadTabs

```
_ReadTabs
=========

Schreiben    : Ja

Beschreibung : Variable , die angibt, ob beim Einlesen eines Files
gefundene Tabulatoren in die entsprechende Anzahl von Spaces umgewandelt
werden soll.  0=NEIN, ansonsten JA.
```

1.52 WRITE.guide/_WriteTabs

```
_WriteTabs
==========

Schreiben    : Ja

Beschreibung : Variable , die angibt, ob beim Schreiben eines Files
Spaces in Tabulatoren umgewandelt werden sollen.  Die Files werden
dadurch kürzer.  Doch nicht alle Programme konvertieren Tabulatoren
wieder richtig zurück.  0=NEIN, ansonsten JA.
```

1.53 WRITE.guide/_CaseSense

```
_CaseSense
==========

Schreiben    : Ja

Beschreibung : Variable, die angibt, ob beim Suchen ein Unterschied
zwischen Groß- und Kleinschreibung gemacht werden soll.  0=NEIN,
ansonsten JA.
```

1.54 WRITE.guide/_Optimize

```
_Optimize
=========

Schreiben    : Ja

Beschreibung : Wenn gesetzt, werden beim Abspeichern Spaces hinter dem
letzten Buchstaben einer Zeile gelöscht.  0=NEIN, ansonsten JA.
```

1.55 WRITE.guide/_XPos

```
_XPos
=====

Schreiben    : Nein

Beschreibung : Gibt aktuelle Spalte, in der sich der Cursor befindet, an.
```

1.56 WRITE.guide/_YPos

```
_YPos
=====

Schreiben    : Nein

Beschreibung : Gibt aktuelle Spalte, in der sich der Cursor befindet, an.
```

1.57 WRITE.guide/_Changed

```
_Changed
========

Schreiben    : Ja

Beschreibung : Wenn ungleich 0, ist der Text seit dem letzten Abspeichern
verändert worden.
```

1.58 WRITE.guide/_FileName

```
_FileName
=========

Schreiben    : Ja

Beschreibung : Beinhaltet den Namen des Textes im aktuellen Ed.
```

1.59 WRITE.guide/_FindString

```
                _FindString
===========

Schreiben    : Ja
```

```
Beschreibung : Zeichenfolge, nach der
                Find
                 suchen soll.
```

1.60 WRITE.guide/_ReplaceString

```
                _ReplaceString
==============

Schreiben    : Ja

Beschreibung : Zeichenfolge, die
                Replace
                 für ein Vorkommen von

                _FindString
                 einsetzten soll.
```

1.61 WRITE.guide/_Length

```
_Length
=======

Schreiben    : Nein

Beschreibung : Länge des Textes in Zeilen.
```

1.62 WRITE.guide/_RS

```
_RS
===

Schreiben    : Nein

Beschreibung : Variable, in die Funktionen ein eine Variable vom Typ
Zeichenkette zurückgeben.
```

1.63 WRITE.guide/_RN

```
_RN
===

Schreiben    : Nein

Beschreibung : Nummerischer Rückgabewert von Funktionen.
```

1.64 WRITE.guide/_WBStarted

```
_WBStarted
==========

Schreiben    : Nein

Beschreibung : Ist _WBStarted ungleich null, so ist WRITE von der
Workbench gestartet worden, ansonsten aus einer Shell.
```

1.65 WRITE.guide/_LoadTab

```
                _LoadTab
========

Schreiben    : Ja

Beschreibung : Umrechnungfaktor für die Konvertierung von Tabulatoren zu
Spaces beim Laden eines Textes.  Im Zusammenhang mit
                _SaveTab
                 können
Texte durch Laden und anschließendes Abspeichern von einer
Tabulatorgröße zu einer anderen konvertiert werden.
```

1.66 WRITE.guide/_SaveTab

```
                _SaveTab
========

Schreiben    : Ja

Beschreibung : Umrechnungfaktor für die Konvertierung von Tabulatoren zu
Spaces beim Speichern eines Textes.  Im Zusammenhang mit
                _LoadTab
                 können
Texte durch Laden und anschließendes Abspeichern von einer
Tabulatorgröße zu einer anderen konvertiert werden.
```

1.67 WRITE.guide/_Version

```
_Version
========

Schreiben    : Nein

Beschreibung : Gibt die Kompatibilitäsversion von WRITE in Form einer
Zahl an.
```

1.68 WRITE.guide/_ChipMem

```
_ChipMem
========

Schreiben    : Nein

Beschreibung : Momentan verfügbarer CHIP/Graphik-Speicher.
```

1.69 WRITE.guide/_FastMem

```
_FastMem
========

Schreiben    : Nein

Beschreibung : Momentan verfügbarer FAST-Speicher.
```

1.70 WRITE.guide/_PublicMem

```
_PublicMem
==========

Schreiben    : Nein

Beschreibung : Momentan verfügbarer Gesamt-Speicher.
```

1.71 WRITE.guide/_Wins

```
_Wins
=====

Schreiben    : Nein
```

Beschreibung : Anzahl der Momentan verfügbarer CHIP/Graphik-Speicher.

1.72 WRITE.guide/_Time

_Time
=====

Schreiben : Nein

Beschreibung : Die aktuelle Zeit in Form eines String, der Tagesnamen
Datum und Zeit beinhaltet.

1.73 WRITE.guide/_Path

_Path
=====

Schreiben : Ja

Beschreibung : Die Pfad des Filerequesters.

1.74 WRITE.guide/_FRPattern

_FRPattern
==========

Schreiben : Ja

Beschreibung : Das Auswahlpattern für den Filerequester.

1.75 WRITE.guide/_PatCase

_PatCase
========

Schreiben : Ja

Beschreibung : Pattern für die Funktion
FindPattern
.

1.76 WRITE.guide/_ConfigPath

```
_ConfigPath
===========

Schreiben    : Ja

Beschreibung : Das Verzeichnis, in dem WRITE alle Konfigurationsdateien
sucht.
```

1.77 WRITE.guide/_ShowSpace

```
_ShowSpace
==========

Schreiben    : Ja

Beschreibung : Wenn #0, werden Spaces durch das Zeichen · angezeigt.
```

1.78 WRITE.guide/_ShowEOL

```
_ShowEOL
========

Schreiben    : Ja

Beschreibung : Wenn #0, wird das Ende einer Zeile im Text durch das
Zeichen ¶ angezeigt.
```

1.79 WRITE.guide/_PatNoCase

```
                _PatNoCase
==========

Schreiben    : Ja

Beschreibung : Pattern für die Funktion
                FindPattern
                .
```

1.80 WRITE.guide/_CurrentChar

```
_CurrentChar
============

Schreiben    : Nein

Beschreibung : ASCII-Code des Zeichens unter dem Cursor.  0 wenn kein
Zeichen.
```

1.81 WRITE.guide/_CurrentWord

```
                _CurrentWord
============

Schreiben    : Nein

Beschreibung : Das Wort unter dem Cursor.  Definition eines Wortes ist
abhängig vom Inhalt der Variable
                _WordDef
                .
```

1.82 WRITE.guide/_CurrentLine

```
_CurrentLine
============

Schreiben    : Ja

Beschreibung : Inhalt der aktuellen Zeile.
```

1.83 WRITE.guide/_TabLength

```
                _TabLength
==========

Schreiben    : Ja

Beschreibung : Abstand zwischen Tabulatoren wärend des Editierens.  Wird
von den Funktionen
                Tab
                 und
                BackTab
                 benutzt.
```

1.84 WRITE.guide/_MaxBuffer

```
_MaxBuffer
==========

Schreiben    : Nein

Beschreibung : Zahl der Textbufer minus 1.  Textbuffere gehen also von
0.._MaxBuffer-1.
```

1.85 WRITE.guide/_REXX

```
_REXX
=====

Schreiben    : Nein

Beschreibung : Wenn #0, so ist REXX installiert und es können
REXX-Sripte gestartet  werden.  (WRITE kontrolliert allerdings nur das
Vorhandensein von LIBS:rexxsyslib.library.)
```

1.86 WRITE.guide/_WordDef

```
_WordDef
========

Schreiben    : Ja

Beschreibung : Definiert, was WRITE unter einem Wort zu verstehen hat.
Ist _WordDef=0, dann besteht ein Wort nur aus Buchstaben, d.h.  a-z, A-Z,
_, Umlaute etc.  Ist _WordDef=1, werden Wörter durch sogenannte
Trennzeichen getrennt.  Trennzeichen sind :  , ., ,, ;, : , !, ?, ’,  `,
(, ) und ".  Ist _WordDef=2, so werden Wörter ausschließlich durch
Spaces getrennt.
```

1.87 WRITE.guide/_WriteIcon

```
                _WriteIcon
==========

Schreiben    : Ja

Beschreibung : Wenn #0, so wird beim Abspeichern auch ein entsprechendes
Icon kreiert.  Siehe auch
                _OverwriteIcon
```

1.88 WRITE.guide/_AppIcon

```
                _AppIcon
========

Schreiben    : Ja

Beschreibung : Wenn #0, so wird auf der Workbench ein AppIcon
installiert, über welches der
                WinManager
                 aufgerufen und Texte geladen
werden können.
```

1.89 WRITE.guide/_AppWindow

```
_AppWindow
==========

Schreiben    : Ja

Beschreibung : Wenn #0, so wird jedes Editorfenster auch zu einem
AppWindow.  D.h. mit dem ziehen von Workbenchicons in dieses Fenster,
können diese geladen werden.
```

1.90 WRITE.guide/_AppMenu

```
                _AppMenu
========

Schreiben    : Ja

Beschreibung : Wenn #0, wird im Hilfsmittelmenü der Workbench ein
Menüpunkt installiert, über den man den
                WinManager
                 aufgerufen und
selektierte Texte laden kann.
```

1.91 WRITE.guide/_Cursor

```
_Cursor
=======

Schreiben    : Ja
```

```
Beschreibung : Je nach Wert wird der Cursor auf verschiedene Weise
dargestellt.
```

1.92 WRITE.guide/_AutoIndent

```
                _AutoIndent
===========

Schreiben    : Ja

Beschreibung : AutoIndent heißt, das noch einem
                Return
                 der Cursor
automatisch nicht am Anfang der Zeile, sondern unter dem ersten
Buchstaben der Zeile darüber steht.  Dies ist besonders paktisch, wenn
man mit WRITE Programme schreibt, da hier besonders oft mit
Texteinrückungen gearbeitet wird.  _AutoIndent  sagt WRITE nun, wie
wieviele Leerzeilen er rückwärts suchen soll, um nach dem Indent, dem
Maß der Einrückung, zu suchen.  Beispiel : DerCursor steht an der
Position %.
        Hier steht ein Text !!!


                 %
     1  2
Ist _AutoIndent größer als 2, so steht der Cursor nach einem Return an
der Position 2, ansonsten an der Position 1.  Ist _AutoIndent=0, so ist
dieses Feature abgeschaltet.
```

1.93 WRITE.guide/_OverwriteIcon

```
                _OverwriteIcon
==============

Schreiben    : Ja

Beschreibung : Diese Variable gibt an, ob WRITE, wenn
                _WriteIcon
                #0 ist,
ein bestehendes Icon überschreiben soll.  Ist _OverwriteIcon=0, so wird
nur, wenn nicht bereits vorhanden, in das bestehende Icon WRITE als
DefaultTool eingetragen, ansonsten schreibt WRITE sein eigenes Icon.
```

1.94 WRITE.guide/_SleepMode

```
_SleepMode
==========

Schreiben    : Ja

Beschreibung : Ist diese Variable ungleich null, so wird WRITE nicht
beendet, wenn der letzte Ed geschlossen worden ist.  Dieser Modus wird
empfohlen und sollte immer eingeschaltet sein, wenn es der Speicherplatz
zuläßt.
```

1.95 WRITE.guide/_EditMode

```
_EditMode
=========

Schreiben    : Ja

Beschreibung : Wenn#0, kann der Text auch verändert werden.  Gilt für
alle Fenster !
```

1.96 WRITE.guide/_Marked

```
_Marked
=======

Schreiben    : Nein

Beschreibung : Ist _Marked=0 so ist weder eine Blockmarke gesetzt, noch
ein Block markiert. Ist _Marked=1, so ist eine Blockmarke gesetzt und ist
_Marked=2, so ist ein Block markiert worden.
```

1.97 WRITE.guide/_MarkAx

```
_MarkAx
=======

Schreiben    : Nein

Beschreibung : Spaltennummer der 1. Blockmarke.  0, wenn keine Marke
gesetzt.
```

1.98 WRITE.guide/_MarkAy

```
_MarkAy
=======

Schreiben    : Nein

Beschreibung : Zeilennummer der 1. Blockmarke.  0, wenn keine Marke
gesetzt.
```

1.99 WRITE.guide/_MarkBx

```
_MarkBx
=======

Schreiben    : Nein

Beschreibung : Spaltennummer der 2. Blockmarke.  0, wenn keine Marke
gesetzt.
```

1.100 WRITE.guide/_MarkBy

```
_MarkBy
=======

Schreiben    : Nein

Beschreibung : Zeilennummer der 1. Blockmarke.  0, wenn keine Marke
gesetzt.
```

1.101 WRITE.guide/_WinMode

```
                _WinMode
========

Schreiben    : Ja

Beschreibung : Hier kann ausgelesen weren, ob der aktuelle Ed ein Fenster
geöffnet hat (0) (
                Window
                ), ikonifiziert worden ist(1) (
                Iconify
                ), kein
Fenster offen hat  (2) (
                Hide
                ), oder schließlich ein Fenster offen hat,
aber die graphische Ausgabe unterdrückt wird (3) (
                Silent
```

```
                ).
```

1.102 WRITE.guide/_CurrentID

```
_CurrentID
==========

Schreiben    : Nein

Beschreibung : ID des aktuellen Eds. Ist der ID 0, so ist kein Ed
aktiviert.
```

1.103 WRITE.guide/_VersionString

```
_VersionString
==============

Schreiben    : Nein

Beschreibung : Versionsstring von WRITE.
```

1.104 WRITE.guide/_File

```
_File
=====

Schreiben    : Nein

Beschreibung : Dateiname des aktuellen Textes, ohne kompletten Pfad.
```

1.105 WRITE.guide/_FilePath

```
_FilePath
=========

Schreiben    : Nein

Beschreibung : Directory des aktuellen Textes.
```

1.106 WRITE.guide/_REXXPortName

```
_REXXPortName
=============

Schreiben    : Nein

Beschreibung : Name des globalen REXX-Ports von WRITE.
```

1.107 WRITE.guide/_REG1

```
_REG1
=====

Schreiben    : Ja

Beschreibung : Variable, die für die eigene Verwendung freigestellt ist.
```

1.108 WRITE.guide/_REG2

```
_REG2
=====

Schreiben    : Ja

Beschreibung : Variable, die für die eigene Verwendung freigestellt ist.
```

1.109 WRITE.guide/_Sound

```
                _Sound
======

Schreiben    : Ja

Beschreibung : Wenn #0, so ruft WRITE für einige Requester Upd auf, um
Samples abzuspielen.  Siehe auch
                Sound
                .
```

1.110 WRITE.guide/_DefaultTool

```
_DefaultTool
============

Schreiben    : Ja

Beschreibung : Bestimmt, was als DefaultTool in Icons abgespeichert
werden soll.  Voreigestellt ist WRITE (inclusive kompletten Pfad) selbst.
```

1.111 WRITE.guide/_Language

```
_Language
=========

Schreiben    : Nein

Beschreibung : Hier steht, auf welche Sprache WRITE eingestellt ist.
```

1.112 WRITE.guide/_CurrentConfig

```
                _CurrentConfig
==============

Schreiben    : Nein

Beschreibung : Der Name der aktuellen Konfiguration. Ist er "", so ist
keine Konfiguration aktiviert. Eine Konfiguration läßt sich mit

                GetConfig
                 aktivieren.
```

1.113 WRITE.guide/_DefaultConfig

```
_DefaultConfig
==============

Schreiben    : Ja

Beschreibung : Der Name der Konfiguration, die immer dann benutzt wird,
wenn expliziet eine bestimmte Konfiguration vorgegeben wird.
```

1.114 WRITE.guide/_AutoFree

```
_AutoFree
=========

Schreiben    : Ja

Beschreibung : Ist diese Variable ungleich Null, so wird eine
Konfiguration, die von keinem Ed mehr gebraucht wird, automatisch
freigegeben.
```

1.115 WRITE.guide/_WinWidth

```
_WinWidth
=========

Schreiben    : Nein

Beschreibung : In dieser Variable steht die Breite des aktuellen Fensters
in Zeichen.
```

1.116 WRITE.guide/_WinHeight

```
_WinHeight
==========

Schreiben    : Nein

Beschreibung : In dieser Variable steht die Höhe des aktuellen Fensters
in Zeichen.
```

1.117 WRITE.guide/_WordWrap

```
                _WordWrap
=========

Schreiben    : Ja

Beschreibung : _WordWrap steuert das WordWrapping von WRITE.  Ist
_WordWrap=0, so gibt es kein WordWrap. Ist _WrdWrap=1, so ertönt ein
Beepen, wenn der Text einer Zeile über den in
                _RightMargin
                 angegebenen
Rand geht. Dies ist ähnlich dem Verhalten einer Schreibmaschine.  Ist
_WordWrap=2, so bricht WRITE eine Zeile, wer sie über den Rand geht,
automatisch um. Der Inhalt der Variable
                _AutoIndent
                 wird dabei beachtet.
```

1.118 WRITE.guide/_RightMargin

```
                _RightMargin
============

Schreiben    : Ja

Beschreibung : Gibt den rechten Rand für die WordWrap-Funktion
(
                _WordWrap
                ) an.
```

1.119 WRITE.guide/_ScreenWidth

```
_ScreenWidth
============

Schreiben    : Nein

Beschreibung : Breiten des aktuellen Screens.
```

1.120 WRITE.guide/_ScreenHeight

```
_ScreenHeight
=============

Schreiben    : Nein

Beschreibung : Höhe des aktuellen Screens.
```

1.121 WRITE.guide/_Undo

```
_Undo
=====

Schreiben    : Ja

Beschreibung : Gibt die Größe des Undo-Puffers an. WRITE merkt dich
alle Veränderungen am Text. Undo gibt nun an, wieviele Veränderungen er
sich behalten soll. Ist _Undo=0, so ist die Undo-Funktion ausgeschaltet.
```

1.122 WRITE.guide/_ScrRelWidth

```
                _ScrRelWidth
============

Schreiben    : Ja

Beschreibung : Wird beim Aufruf des Befehls
                Window
                 das Tag  @SCREENREL
übergeben, so beziehen sich alle Größenangaben nicht auf die reale
Screengröße sondern auf den hier angegebenen Wert. Dies ist zum
Beispiel nützlich, wenn ein Script WINDOW 0 0 0 0 zum öffnen eines
bildschrimfüllenden Fenster benutzt. Auf großen Bildschirmen ist dies
aber nicht unbedingt erwünscht. Mit dieser Variablen kann man nun einen
Screen der angegebenen Größe vortäuschen.
```

1.123 WRITE.guide/_ScrRelHeight

```
                _ScrRelHeight
=============

Schreiben    : Ja

Beschreibung : Wird beim Aufruf des Befehls
                Window
                 das Tag  @SCREENREL
übergeben, so beziehen sich alle Größenangaben nicht auf die reale
Screengröße sondern auf den hier angegebenen Wert. Dies ist zum
Beispiel nützlich, wenn ein Script WINDOW 0 0 0 0 zum öffnen eines
bildschrimfüllenden Fenster benutzt. Auf großen Bildschirmen ist dies
aber nicht unbedingt erwünscht. Mit dieser Variablen kann man nun einen
Screen der angegebenen Größe vortäuschen.
```

1.124 WRITE.guide/_WordOnly

```
                _WordOnly
=========

Schreiben    : Ja

Beschreibung : Variable, die angibt, ob beim Suchen nur nach ganzen
Wörtern oder auch nach Wortteilen gesucht werden soll.  0=NEIN, ansonsten
JA. Die Definition eines Wortes hängt von dem Inhalt der Variable

                _WordDef
                 ab.
```

1.125 WRITE.guide/_ReqToMouse

```
                _ReqToMouse
===========

Schreiben    : Ja

Beschreibung : Variable, die angibt, ob bei internen Requesteraufrufen,
diese unter dem Mauszeiger geöffnet werden. Siehe auch
                Requester
```

1.126 WRITE.guide/_EColor

```
_EColor
=======

Schreiben    : Ja

Beschreibung : Eine Falte wird durch zwei horizontale Linien angezeigt.
Durch diese Variable kann man die Farbe der oberen bestimmen.
```

1.127 WRITE.guide/_FColor

```
_FColor
=======

Schreiben    : Ja

Beschreibung : Eine Falte wird durch zwei horizontale Linien angezeigt.
Durch diese Variable kann man die Farbe der unteren bestimmen.
```

1.128 WRITE.guide/_AutoFold

```
                _AutoFold
=========

Schreiben    : Ja

Beschreibung : Folds werden beim Abspeichern im Text durch die
Markierungen
                _FoldStart
                 und
```

```
                _FoldEnd
                 angezeigt. Ist _AutoFold TRUE so
wird nach dem Einladen eines Textes automatisch nach diesen Markierungen
gesucht und diese wieder zu Falten konvertiert.
```

1.129 WRITE.guide/_FoldStart

```
                _FoldStart
==========

Schreiben    : Ja

Beschreibung : Folds werden beim Abspeichern im Text durch die
Markierungen angezeigt. FoldStart zeigt den Anfang einer Falte,
                _FoldEnd
                das Ende einer Falte an. Diese Markierungen können durch den  ↩
                   Benutzer
frei definiert werden, damit gewährleistet werden kann, daß diese
Markierungen z.B. einen Compiler nicht stören.  C Programmierer würden
z.B. /*S*/ und /*E*/ wählen, Oberon Programmierer (*S*) und (*E*),
Leute, die mit TeX arbeiten einfach /%S und %E.
```

1.130 WRITE.guide/_FoldEnd

```
                _FoldEnd
========

Schreiben    : Ja

Beschreibung : Folds werden beim Abspeichern im Text durch die
Markierungen angezeigt.
                _FoldStart
                 zeigt den Anfang einer Falte, _FoldEnd
das Ende einer Falte an. Diese Markierungen können durch den Benutzer
frei definiert werden, damit gewährleistet werden kann, daß diese
Markierungen z.B. einen Compiler nicht stören.  C Programmierer würden
z.B. /*S*/ und /*E*/ wählen, Oberon Programmierer (*S*) und (*E*),
Leute, die mit tex arbeiten einfach /%S und %E.
```

1.131 WRITE.guide/_DelFoldMark

```
                _DelFoldMark
============
```

```
Schreiben    : Ja

Beschreibung : Folds werden beim Abspeichern im Text durch die
Markierungen angezeigt.
                _FoldStart
                 zeigt den Anfang einer Falte
                _FoldEnd
                das Ende einer Falte an. Werden diese Faltenmarkierungen   ↩
                   anschließend
durch das
                _AutoFold
                -Feature der durch die Funktion
                ReFold
                 in Falten
umgewandelt, kann es erwünscht sein, daß die Faltenmakierungen
gelöscht werden. Ist die variable DelFoldMark ungleich TRUE, so geschieht
dies. Nach einem anschließenden Entfalten sind die Markierungen wieder
verschwunden.
```

1.132 WRITE.guide/_Fold

```
_Fold
=====

Schreiben    : Nein

Beschreibung : Ist diese Variable ungleich 0, so handelt es sich bei der
aktuelllen Zeile um eine Falte. Bevor man auf den Inhalt der aktuellen
Zeile zugreifen will, sollte man immer diese Variable kontrollieren.
```

1.133 WRITE.guide/_Screenname

```
_Screenname
===========

Schreiben    : Nein

Beschreibung : Names des Screens, auf den die aktuelle onfiguration ihre
Fenster öffnet.
```

1.134 WRITE.guide/_PathPath

```
_PathPath
=========

Schreiben    : Nein
```

Beschreibung : Der Pfadanteil des Filerequesterpfades.

1.135 WRITE.guide/_User

_User
=====

Schreiben : Nein

Beschreibung : Falls es sich um eine registrierte Version handelt, steht hier der Name des Besitzers.

1.136 WRITE.guide/_CollectorMem

_CollectorMem
=============

Schreiben : Nein

Beschreibung : Größe des Speichers, welcher vom GarbageCollector benutzt wird.

1.137 WRITE.guide/Funktionsbeschreibung

Funktionenbeschreibung
****

Funktionen für Ed's und Fenster

NOP
Diese Funktion macht gar nichts

Open
Laden eines Textes

Save
Speichert einen Text

SetBackUp
Setzt die verschiedenen BackUpmodi

New
Löscht aktuellen Text

```
                NewEd
                          Öffnet einen neuen Ed

                QuitEd
                         Schließt Ed

                Window
                         Öffnet Fenster für Ed

                Iconify
                        Iconfifiziert Ed

                Hide
                           Versteckt Ed

                Silent
                         Unterbindet Graphikausgabe in Fenster

                Font
                           Set neuen Font

                WinArranger
                    Organisiert offene Fenster

                SetTitle
                       Setzt FensterTitel

                Screen
                         Gibt Screen an, auf den Konfig Fenster öffnen soll

Requester


                About
                          Informationen über Version,Autor...

                Prefs
                          Änderung einiger konfigurationslokalen Variablen

                GPrefs
                         Änderung der globalen Variablen

                HelpFkt
                        Hilfsfunktion

                WinManager
                     Erleichtert Umgang mit Eds

                ShowVars
                       Zeigt Variablen

                ShowFunctions
                  Zeigt Funktionen

                ShowConstants
                  Zeigt Konstanten
```

```
ShowASCII
      Zeigt ASCII-Code aller Zeichen

ShowIndex
      Erstellt einen Index zu einem Suchmuster

GetString
      Requester für Stringeingabe

GetNumber
      Requester für Zahleneingabe

GetFindReplace
 Requester für die Eingabe von Suchwörtern

GetFile
        Der Filerequester

GetFiles
       Filerequester mit MultiSelect

GetFont
        Fontrequester

Ask
            Frage Requester mit definierbaren Gadgets

Message
        Zeigt kurz eine Meldung

MessageOK
      Zeigt Nachricht und erwartet Bestätigung

Flash
          Bildschirmblitz

Beep
           Beep
```

Funktionen für den internen Parser und das Betriebssystem

```
ParseBuffer
    Parsen eines Puffers

DoBuffer
       Führt Puffer aus

DoString
       Führt String aus

PreparseString
 Parsed String vor. Variablen etc. werden ersetzt

SetUserFkt
     Set eine UserFunktion
```

```
                Compare
                        Vergleicht zwei Werte

                If
                             Bedingte Abarbeitung

                Break
                          Bedingter Abbruch eine Anweisungsfolge

                SetError
                       Bricht Anweisungsfolge ab

                SetVar
                         Setzt eine Variable

                GetVar
                         Liest Variable

                GetConst
                       Liest Konstante

                SetEnv
                         Setzt Umgebungsvariabel

                GetEnv
                         Liest Umgebungsvariabel

                System
                         Führt DOS-Befehle etc. aus
```

Puffer- und Blockopperationen

```
                SetMark
                        Setzt die Marken für Blockoperationen

                Mark
                           Markieren eines Blockes

                UnMark
                         Löscht die aktuelle Blockmarkierung

                DeleteBlock
                    Löscht den markierten Block

                CopyBlock
                      Kopiert den markierten Block

                InsertBlock
                    Fügt einen Puffer an der Cursorposition ein

                DeleteArea
                     Löscht angebaren Bereich

                CopyArea
                       Copiert angebaren Bereich
```

```
                SaveBuffer
                     Speichert ein Puffer ab

                LoadBuffer
                     Läd einen Puffer

                ClearBuffer
                    Löscht einen Puffer

                BufferToStr
                    Konvertiert Buffer zu String

                StrToBuffer
                    Konvertiert String zu Buffer

                ClipToBuffer
                   Kopiert Text im clipboard-device in einen Puffer

                BufferToClip
                   Kopiert einen Puffer ins clipboard-device

                BlockLeft
                      Verschiebt Block nach Links

                BlockRight
                     Verschiebt Block nach Rechts

                BlockLftAlig
                   Macht Block linksbündig

                BlockRghtAlig
                  Macht Block rechtsbündig

                BlockCenter
                    Zentriert einen Block


Befehle für die Bewegung im Text


                CursorUp
                      Bewegt den Cursor 1 hoch

                CursorDown
                    Bewegt Cursor 1 runter

                CursorRight
                   Bewegt Cursor 1 rechts

                CursorLeft
                    Bewegt Cursor 1 links

                NextWord
                      Springt zum Anfang des nächsten Wortes

                LastWord
                      Springt zum Anfang des letzten Wortes
```

PageUp
Springt eine Seite nach oben

PageDown
Springt eine Seite nach unten

Goto
Springt zu einer angegebenen Zeile/Spaltenposition

GotoMouse
Positioniert Cursor unter der Maus

SetTextMark
Setzt eine TextMarke

GoTextMark
Springt zu einer TextMarke

Suchen und Finden

Find
Springt zum gesuchten Text

Replace
Ersetzt Suchstring durch anderen String

FindPattern
Sucht Text nach einen Muster

ReplaceList
Such und ersetzt den Inhalt einer Liste

MatchBracket
Sucht die entsprechende 2. Klammer

Befehle für die Texteditierung

Return
Zeilenumbruch

Delete
Löscht Zeichen unter dem Cursor

DeleteToEOL
Löscht vom Cursor bis Ende der Zeile

DeleteLine
Löscht ganze Zeile

UnDelLine
Setzt mit DeleteLine gelöschte Zeile wieder ein

BackSpace
Einen nach links und dann löschen

```
                Tab
                          Springt zum nächstem Tab

                BackTab
                       Springt zum letzten Tab

                UpperBlock
                    Wandelt einen Textbereich in Großbuchstaben um

                LowerBlock
                    Wandelt einen Textbereich in Kleinbuchstaben um

                WriteChar
                     Schreibt in Zeichen

                WriteText
                     Schreibt Zeichenkette

Menüs und Tastaturbelegung


                Key
                           Belegt eine Taste mit einer Befehlsfolge

                DoubleKey
                     Belegt eine Doppellick Tastenkombination

                ClearKeys
                     Löscht die Tastaturbelegung

                SetHotKey
                     Definiert einen HotKey mit einer Befehlsfolge

                ClearHotKey
                   Löscht den Hotkey wieder

                Menu
                          Neues Menü

                Item
                          Definiert einen Menüeintrag

                Sub
                           Definiert einen Untermenüeintrag

                ItemBar
                       Macht in einem Menü einen Unterteilungsstrich

                SubBar
                        Macht in einem Untermenü einen Unterteilungsstrich

                ClearMenu
                     Löscht die Menübelegung

                WaitPointer
                   Erzeugt eien Wartemauszeiger
```

NormalPointer
 Normaler Mauszeiger

REXX

DoREXX
        Führt ein REXX-Script aus

LockWindow
    Alle REXX-Komandos gehen an das markierte Fenster

NextEd
        Aktierviert einen bestimmten Ed

OpenPort
      Öffnet einen REXX-Port für ein Fenster

ClosePort
     Schließt diesen

WaitPort
      Wartet bis das Fenster geschlossen wird

ModifyWin
     Verändert Editorfenster

ModifyScreen
  Toggle Screen nach vorne und hinten

SetREXXClip
   Kopiert den Inhalt einer Zeile/Puffer in das REXXClipboard

GetConfig
     Aktiviert eine bestimmte Konfiguration

SetREXXVar
    Setzt eine REXX-Variablen mit den Inhalt einer Zeile etc.

GetREXXVar
    Liest den Inhalt einer REXX-Variablen nach
_RS

Refresh
       Aktuallisiert das Fenster

ChangeConfig
  Ändert die Konfiguration eines bestehenden Eds

MAKROS

MacroRec
      Startet die Aufzeichnung eines Makros

MacroStop

Stopt die Aufzeichnung eines Makros

MacroPlay
Spielt ein Makro ab

SetMacro
Funktionsliste als Makro definieren

ExecuteMacro
Makro Ausführen

MacroPannel
Selektion aller Makros über GadgetPannel

UNDO

Undo
Macht Veränderungen wieder Rückgängig

LISTEN

ClearList
Löscht eine Liste

AddList
Hängt einen Eintrag an eine Liste

RemoveList
Löscht einen Listeneintrag

Push
Schiebt Eintrag auf die erste Position d. Liste

Pop
1. Eintrag nach
_RS
und aus Liste löschen

ShowList
Zeigt Liste/Selektieren aus Liste

ListToBuffer
Kopiert eine Liste in einen Buffer

BufferToList
Macht aus jeder Zeile eines Buffers einen Listeneintrag

DoList
Führt Funktion mit jedem Listenelement aus

ListSize
Gibt die Größe einer Liste zurück

GetListEntry

Gibt bestimmten Eintrag einer Liste zurück

FindListEntry
Such bestimmten Eintrag und gibt dessen position zurück

FILES

Exists
Schaut, ob ein File existiert

Delay
Wartet eine eintstellbare Zeit

HELP

GuideHelp
Ruft AmigaGuide mit bestimmten Stichwort auf

VersionCheck
Kontrolliert, ob WRITE zur angegebenen Version kompatibel ist

Inc
Addiert zu einer Zahl eine weitere Zahl

Dec
Subtrahiert zwei Zahlen voneinander

RangeCheck
Kontrolliert, ob Zahl im angegebenen Interval liegt

RangeRound
Vergrößert, verkleiner Zahl, daß sie im angegebenen Intervall ↩
liegt

Begin
Wird beim Starten v. WRITE ausgeführt

Close
Wird beim Schließen v. WRITE ausgeführt

Start
Wird beim Laden einer Konfiguration ausgeführt

Quit
Wird beim Beenden einer Konfiguration ausgeführt

TEXTFALTEN

Fold
Falten einen bestimmten Textbereich

```
                UnFold
                        Entfaltet alle Falten in einem bestimmten Textbereich

                AutoFold
                      Konvertiert all Faltenmarkierungen zu Falten

                ReFold
                        Schaut, ob Cursor zwischen zwei Faltenmarkierungen steht ↩
                           und konvertiert diese zu einer Falte
```

1.138 WRITE.guide/NOP

```
NOP
===

Aufruf       : NOP

Benötigt     :

Setzt Fehler :

Ergebnisse   :

Beschreibung : NOP steht für 'no operation'.  Diese Funktion macht gar
nichts.

Siehe auch =>
```

1.139 WRITE.guide/Open

```
                Open
====

Aufruf       : Open Dateiname/S Tags/T

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Diese Funktion versucht den Text mit dem angegebenen
Dateinamen in den aktuellen Ed zu laden.  Ist dies nicht möglich, so
bricht dies Funktion mit einer Fehlermeldung ab.  Diese Funktion mach
keine Sicherheitsabfrage, falls der alte Text verändert sein sollte! Die
Fehlermeldung durch Requester kann mittels des Tags  @SILENT unterdrückt
werden. Open gibt dann einfach nur einen Fehler zurück.

Siehe auch =>
```

GetFile
,
Save
,
LoadBuffer
,
SaveBuffer

1.140 WRITE.guide/Save

Save
====

Aufruf : Save Dateiname/S Tags/T

Benötigt : Ed

Setzt Fehler : Ja

Ergebnisse : Keine

Beschreibung : Save speichert den Text im aktuellen Ed unter dem angegebenen Namen ab. Namensangaben wie PRT: zum Drucken sind möglich. WRITE restauriert dabei die ursprünglichen Protection-Flags mit Ausnahme des Archive-Flags (A). Ist in Mode @RAW gesetzt, werden LF nur bei leeren Zeilen abgespeichert. WRITE erzeugt so überlange Zeilen die aus jeweils einem ganzen Absatz bestehen. Dies ist praktisch, falls man den Text anschließend in eine Textverarbeitung einlesen will.

Kann der Text nicht abgespeichert werden, so wird das Abspeichern mit einem Fehler und dem entsprechenden Fehlerrequester abgebochen. Der Requester kann durch das Setzten des Tags @SILENT unterdrückt werden.

Siehe auch =>
GetFile
,
Open
,
LoadBuffer
,
SaveBuffer

1.141 WRITE.guide/SetBackUp

SetBackUp
=========

Aufruf : SetBackUp Mode/N Pattern/S To/S

Benötigt : Nichts

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : SetBackUp stellt die verschiedenen BackUpmodi von WRITE ein. BackUp bedeutet, daß, wenn ein Text unter einem Dateinamen, der bereits existiert, abgespeichert werden soll, von der bereits bestehenden Datei eine Sicherheitskopie gemacht wird. Momentan gibt es 3 Modi :

1. Modus 0 : Entspricht die Datei dem angegebenen DOS-Pattern, so wird sie als angegebe Datei To umbenannt bzw. kopiert.

2. Modus 1 : Entspricht die Datei dem Pattern, so wird die angegebene Endung angehangen.

3. Modus 2 : Entspricht die Datei dem Pattern, so wird sie in das angegebene Directory kopiert.

WRITE vergleicht die Pattern in absteigender Reihenfolge, d.h. 2... 1...0.

Siehe auch =>

1.142 WRITE.guide/New

New
===

Aufruf : New

Benötigt : Ed

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : New löscht den aktuellen Text unwiederruflich. Eine eingebaute Sicherheitsabfrage gibt es dabei nicht. Diese muß programmiert werden. Für ein Beispiel, wie dies zu tun ist, schauen sie bitte in die beigelegten Standardkonfigurationen.

Siehe auch =>

1.143 WRITE.guide/NewEd

NewEd
=====

Aufruf : NewEd Konfiguration/S

```
Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : NewEd öffnet einen neuen Ed mit der angegebenen
Konfiguration. Wird als Konfigurations(file) "" angegeben, so die
Konfiguration aus der Variable
                _DefaultConfig
                 benutzt.  Kann der
Konfigurationfile nicht gefunden werden, oder tritt beim Parsen des selben
ein Fehler auf, so bricht NewEd ab und gibt einen Fehler zurück.

Siehe auch =>
                Der Ed
```

1.144 WRITE.guide/QuitEd

```
                QuitEd
======

Aufruf       : QuitEd

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Die Funktion schließt den aktuellen Ed, den dazugehörigen
Text sowie, wenn geöffnet, das Fenster.

Eine Sicherheitsabfrage gibt es nicht. Zu Realisierung einer
Sicherheitsabfrage schauen sie bitte in einen beliebigen beigelegten
Konfigurationsfile nach.

Siehe auch =>
                NewEd
```

1.145 WRITE.guide/Window

```
                Window
======

Aufruf       : Window dx1/N dy1/N dx2/N dy2/N Tags/T

Benötigt     : Ed
```

```
Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Window öffnet für den aktuellen Ed ein Fenster mit den
angegebenen Maßen.  Dabei steht (x1,y1) für die linke obere Ecke
relativ zur linken oberen Ecke des Bildschirms und (x2/y2) für die
rechte untere Ecke relativ zur rechten unteren Ecke.
     Window  0  0  0  0
öffnet ein Fenster über den ganzen Screen.
     WINDOW 50 50 50 50
öffnet ein Fenster, dessen Seiten jeweils 50 Pixel vom Bildschirmrand
entfernt sind.  Hatte der Ed schon einmal ein geöffnetes Fenster, so
werden dessen Maße benutzt.  Konnte das Fenster nicht geöffnet werden,
so wird ein Fehler zurückgegeben.

Ist in Tags das  @SCREENREL gesetzt, so beziehen sich die Größenangaben
nicht auf die tatsächliche Screengröße sondern auf einen Screen der
bei
                _ScrRelWidth
                 und
                _ScrRelHeight
                 angegebenen Größe. Dies ist
nützlich für übergroße Screen wie man sie bei Grafikkarten besitzt.

Siehe auch =>
                Iconify
                ,
                Hide
                ,
                Silent
```

1.146 WRITE.guide/Iconify

```
                Iconify
=======

Aufruf       : Iconify

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Schließt ein eventuell vorhandenes Fenster und öffnet ein
iconifiziertes Fenster für den aktuellen Ed.  Dieses ist ein kleines
Fenster mit einen Schließsymbol in der rechten, oberen Ecke des Screens.
Betätigt man dieses Schließsymbol, so wird wieder ein Fenster der alten
Größe geöffnet.  Mehere Icons werden untereinander plaziert.  Kann das
Iconfenster nicht geöffnet werden, so wird ein Fehler zurückgegeben.

Siehe auch =>
                Window
```

```
                ,
                Hide
                ,
                Silent
```

1.147 WRITE.guide/Hide

```
                Hide
====

Aufruf       : Hide

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Schließt das Icon oder Fenster des aktuellen Eds.  Der
Text ist nicht verloren, er besitzt nur kein Fenster mehr.  Es gibt
mehrere Möglichkeiten das Fenster wieder zu öffnen :
  1. Über den
                WinManager
                  2. In dem REXX-Script , das in den Hidemodus geschaltet hat,  ↩
                     mittels

                Window
                 oder
                Iconify
                 (Da die Befehle an das aktuelle Fenster gehen !)

  3. Oder ähnlich über den PrivatePort (dessen Namen man über
                OpenPort
                     bekommen hat, oder den man mit dem ID, der von
                NewEd
                 zurückgegeben
     wird, öffnet.

Siehe auch =>
                Window
                ,
                Iconify
                ,
                Silent
```

1.148 WRITE.guide/Silent

```
                Silent
======
```

```
Aufruf       : Silent

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Das Fenster des aktuellen Ed (wenn er eins geöffnet hat)
wird in den Silentmode geschaltet.  D.h., daß alle graphischen Ausgaben
unterdrückt werden.  Einige Funktionen wie z.B.
                Replace
                 erreichen so ein
vielfaches ihrer Geschwindigkeit.

Siehe auch =>
                Window
                ,
                Iconify
                ,
                Hide
```

1.149 WRITE.guide/Font

```
                Font
====

Aufruf       : Font Name/S yGröße/N DiskFont/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Der Text im Editierfenster wird in dem angegebenen Font
dargestellt.  Name und yGröße stehen dabei für den Namen und die Höhe
des Textes.  Ist DiskFont#0, so wird der Font von Diskette geladen,
ansonsten wird er im ROM gesucht.  Kann der Font nicht gefunden
werden,wird ein Fehler zurückgegeben

Siehe auch =>
                GetFont
```

1.150 WRITE.guide/WinArranger

```
                WinArranger
===========

Aufruf       : WinArranger Gewichtung/N Tags/T
```

```
Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : WinArranger versucht alle geöffneten Fenster
übersichtlich, flächendeckend und ohne Überlappungen auf dem Screen zu
plazieren.  Momentan werden zwei Modi unterstützt :

   *      @VERT - Alle offenen Fenster werden untereinander plaziert und
     nach vorne gebracht.

   *      @HORIZ - Alle Fenster werden nebeneinander plaziert und nach
     vorne gebracht.

WinArranger beachtet beim Plazieren der Fenster auch den Inhalt der
Variablen
                _ScrRelWidth
                 und
                _ScrRelHeight
                .

Über Gewichtung kann man einstellen, wie viel das aktuelle Fenster
größer als alle anderen sein soll. Bei dem Wert 2 z.B. ist das atuelle
Fenster genau zweimal so groß wie die anderen Fenster. Eine interessante
Einstellung ist z.B.  WinArranger _Wins.

Siehe auch =>
```

1.151 WRITE.guide/SetTitle

```
SetTitle
========

Aufruf       : SetTitle Titel/S NoHold/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : SetTitle setzt den Fenstertitel aud Titel.  Die Flags und
Positionsangaben bleiben dabei sichtbar.  Ist NoHold=0, so wird der Titel
nach Ausführung der aktuellen Befehlsfolge gelöscht, ansonsten nicht.

Siehe auch =>
```

1.152 WRITE.guide/Screen

```
                Screen
======

Aufruf       : Screen Data/S Titel/S Flags/T

Benötigt     : Konfiguration

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Screen gibt an, auf welchem Screen eine Konfiguration ihre
Fenster öffnen soll.  Momentan werden nur fremde PublicScreens
unterstützt.  Dabei ist Titel der Name des PublicScreens.  In Flags muß
Der neue Screen gilt erst für alle neu geöffneten Fenster der aktuellen
Konfiguration.  Screen sollte deshalb schon bereits im Konfigurationsfile
aufgerufen werden.

Kann Screen auf einen Screen nicht zugreifen, so wird der
"DefaultPublicScreen" benutzt.  Dies ist meißtens die Workbench.
Außerdem gibt Screen dann einen Fehler zurück.

Es sollte hier noch einmal außdrücklich erwähnt werden, daß WRITE sich
völlig an die Gegebenheiten eines beliebigen Screens anpaßt.  Dies sind
vor allem die Auflösung, so wie der eingestellte Zeichensatz des Screens.
Alle Fenster und Requester von WRITE sind fontsensitiv und passen sich
sogar Proportionalfonts problemlos an.

Siehe auch =>
                PublicScreens
```

1.153 WRITE.guide/About

```
                About
=====

Aufruf       : About

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : About öffnet ein kleines Fenster, in dem die
Versionsnummer, mein Name und meine Adresse, sowie ein kleiner Hinweis in
Sachen Raubkopien und dem legalen Besitz von Originalen stehen.

Siehe auch =>
                Copyright
                ,
```

Vollversion
,
Requester

1.154 WRITE.guide/Prefs

Prefs
=====

Aufruf : Prefs

Benötigt : Ed

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : Hier kann man einige konfigurationsspezifische
Einstellungen ändern und abspeichern.

Dabei werden die Einstellung nach einzelnen Themenbereichen aufgeteilt.
Alle Einstellungen in den Unterrequestern werden bei drücken der OK-Taste
übernommen. Drückt man im Preferencerequester Speichern..., so wird
automatisch ein neuer Konfigurationsfile mit dem angegebenen Namen
erzeugt. Durch Drücken von Ende werden die Einstellung nur übernommen,
aber nicht abgespeichert.

Durch das Setzen des Tags @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
Requester
Siehe auch =>
GPrefs
,
Requester

1.155 WRITE.guide/GPrefs

GPrefs
======

Aufruf : GPrefs

Benötigt : Nichts

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : Hier kann man einige globale Einstellungen ändern.

```
Hier gilt das gleiche wie auch für wie für den
                Prefs
                requester :
Drückt man im Preferencerequester Speichern, so wird automatisch eine
neue STARTUP.CONFIG erzeugen. Durch Drücken von Ende werden die
Einstellung nur übernommen, aber nicht abgespeichert.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                Prefs
                ,
                Requester
```

1.156 WRITE.guide/HelpFkt

```
                Help
====

Aufruf       : Help

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Help öffnet ein kleines Fenster mit verschieden Gadgets
über die einige weitere Hilfrequester geöffnet werden können.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                ShowVars
                ,
                ShowFunctions
                ,
                ShowASCII
                ,
                Requester
```

1.157 WRITE.guide/WinManager

```
                WinManager
==========

Aufruf       : WinManager
```

```
Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : WinManager öffnet ein Fenster mit einem Listrequester, in
dem alle geöffneten Eds gezeigt werden. Da bei bedeutet die erste Zahl
der ID, ein anschließendes Sternchen zeigt an, daß der Text verändert
wurde, und ein abschließendes h, i, s gibt an, ob sich der Ed im Hiden-,
Iconify- oder im Silent-Mode befindet. Der Eintrag schließt mit dem
Dateinamen des aktuellen Textes ab.  Mit den darunter angebrachten
Gadgets kann der Requester wieder verlassen, ein neues Fenster geöffnet,
ein selektierter Ed gezeigt, gelöscht , oder, falls kein Ed vorhanden,
WRITE verlassen werden.  Ein Eintrag kann durch anklicken mit der Maus,
oder (ab OS 3.0) über die CursorUp bzw.  CursorDown Tasten selektiert
werden.

Das Verlassen von WRITE geht nur, wenn alle Fenster geschlossen sind.

die Taste Neues Fenster führt bei Selektion die Userfunktion 2
(
                SetUserFkt
                ,
                GPrefs
                ) aus.  Damit läßt sich
konfigurieren, was für ein Fenster (Konfiguration, Größe) geöffnet
werden soll.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                Requester
```

1.158 WRITE.guide/ShowVars

```
                ShowVars
========

Aufruf       : ShowVars

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Öffnet ein Fenster mit einem Listgadget, in dem alle
internen Variablen mit ihren Namen sowie dem aktuellen Wert stehen.

Steht hinter dem Variablennamen ein P, so ist die Variable nur auslesbar,
wenn eine Konfiguration aktiviert ist. Steht hinter dem Namen ein E,
sogar nur, wenn ein Ed aktiviert ist.
```

```
Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                HelpFkt
                ,
                ShowFunctions
                ,
                ShowASCII
                ,
                ShowConstants
                ,

                Requester
                ,
                GetConfig
```

1.159 WRITE.guide/ShowFunctions

```
                ShowFunctions
=============

Aufruf       : ShowFunctions

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Eine Liste mit allen internen Funktionen samt ihrer
Parameter.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Hinter dem Variablennamen stehen einige Buchstaben :

   *     C : Die Funktion kann nur mit aktiver Konfiguration ausgeführt
     werden.

   *     E : Die Funktion kann nur mit aktiven Ed ausgeführt werden.

   *     W : Die Funktion kann nur mit aktivem Fenster ausgeführt werden.

   *     * : Die Funktion ändert den Text.

   *     S : Die Funktion gibt einen String zurück.

   *     N : Die Funktion bit eine Zahl zurück.

   *     ! : Die Funktion gibt gegebenenfalls einen Fehler zurück.
```

```
Siehe auch =>
                HelpFkt
                ,
                ShowVars
                ,
                ShowASCII
                ,
                ShowConstants
                ,
                Requester
```

1.160 WRITE.guide/ShowConstants

```
                ShowConstants
=============

Aufruf       : ShowConstants

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Eine Liste mit allen internen Konstanten.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                HelpFkt
                ,
                ShowVars
                ,
                ShowASCII
                ,
                ShowFunctions
                ,
                Requester
```

1.161 WRITE.guide/ShowASCII

```
                ShowASCII
=========

Aufruf       : ShowASCII

Benötigt     : Nichts

Setzt Fehler : Nein
```

```
Ergebnisse   : Nein

Beschreibung : Öffnet ein Fenster mit einem Listgadget, in dem alle
dastellbaren Zeichen samt ihrer ASCIInummer in dezimaler Schreibweise
stehen.  Diese Nummern können dann z.B. als Parameter für
                WriteChar
                übergeben werden.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                WriteChar
                ,
                HelpFkt
                ,
                ShowFunctions
                ,
                ShowVars
                ,
                ShowConstants
                ,

                Requester
```

1.162 WRITE.guide/ShowIndex

```
                ShowIndex
=========

Aufruf       : ShowIndex Pattern/S Flags/T

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Sucht im gesamten Text nach dem angegebenen DOS-Pattern.
Wenn  @CASE in Flags gesetzt ist, so wird dabei zwischen Groß- und
Kleinschreibung unterschieden. Wenn  @NOCASE gesetzt ist , nicht.
Ansonsten wird die Preferenceseinstellung benutzt.  Anschließend werden
alle Zeilen, in den das Pattern gefunden wurde, in einem Requester mit
ihrer Zeilennummer angezeigt.  Durch Selektion einer Zeile kann dann zu
dieser gesprungen werden.

Für Oberon/Modula2-Programmierer ist zum Beispiel nützlich :
     ShowIndex "#?PROCEDURE#?" 1

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>,
```

```
                Requester
```

1.163 WRITE.guide/GetString

```
                GetString
=========

Aufruf       : GetString Title/S Vorbelegung/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : GetString öffnet einen Requester mit dem angegebenen
Titel und einem Stringgadget mit dem angegebenen Inhalt.  Wird im
Stringgadget RETURN gedrückt oder das Bestätigungsgadget betätigt, so
wird in der Variablen
                _RS
                 der eingegebene String zurückgegeben.
Andernfalls wird ein Fehler zurückgegeben.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                GetNumber
                ,
                Requester
```

1.164 WRITE.guide/GetNumber

```
                GetNumber
=========

Aufruf       : GetNumber Titel/S Vorbelegung/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : GetNumber öffnet einen Requester mit dem angegebenen
Titel und einem Integergadget mit dem angegebenen Inhalt.  Wird im
Integergadget RETURN gedrückt oder das Bestätigungsgadget betätigt, so
wird in der Variablen
                _RN
                 die eingegebene Zahl zurückgegeben.
```

```
Andernfalls wird ein Fehler zurückgegeben.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                GetString
                ,
                Requester
```

1.165 WRITE.guide/GetFindReplace

```
                GetFindReplace
==============

Aufruf       : GetFindReplace FindWord/S ReplaceWord/S Flags/T

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : GetFindReplace öffnet einen Requester in dem kompfortabel
die Variablen _FindString,
                _CaseSense
                 und, falls  @REPLACE in Flags
gesetzt ist, auch die Variable _ReplaceString gesetzt werden können.
Drückt man in einen der beiden StringGadgets die HELP-Taste, so öffnet
sich ein ListRequester, über den man eins der vorher eingebenen Wörter
selektieren kann.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                Find
                ,
                Replace
                ,
                FindPattern
                ,
                Requester
```

1.166 WRITE.guide/GetFile

```
                GetFile
=======

Aufruf       : GetFile Pfad,Pattern/S Typ/T
```

```
Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : GetFile öffnet einen Filerequester mit dem Pfad Pfad
oder, falls dieser leer ist ("") mit dem Pfad, der in der Variable
                _Path
                steht. Es werden dabei nur Files, die dem Suchmuster Pattern  ↩
                   entsprechen,
oder, falls dieser leer ist, dem Muster in der Variablen
                _FRPattern
                .

Ist in Typ  @SAVE gesetzt, so wird ein Save-Filerequester benutzt (Es
wird helle Schrift auf dunklem Hintergrund benutzt).

Wird eine Datei selektiert, so wird der dazugehörige Dateiname ,falls
Pfad leer ist, in der Variablen
                _Path
                 zurückgegeben, ansonsten in
                _RS
                .

Andernfalls wird ein Fehler zurückgegeben.

Siehe auch =>
                GetFiles
```

1.167 WRITE.guide/GetFiles

```
                GetFiles
========

Aufruf       : GetFiles Liste,Pfad,Pattern/S Typ/T

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Gleiches Verhalten wie
                GetFile
                . Nur das hier nicht nur ein
File sondern mehrere seletiert werden können, und diese nicht in
                _Path
                oder
                _RS
                 sondern in der angegebenen Liste eingetragen werden. Diese muß
vorher existieren.
```

```
Siehe auch =>
                GetFile
```

1.168 WRITE.guide/GetFont

```
GetFont
=======

Aufruf       : GetFont

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : GetFont öffnet einen Fontrequester.  Wird ein Font
selektiert, so wird versucht, diesen als neuen Font für die
Textdarstellung zu benutzen. Wird kein Font selektiert, oder kann dieser
nicht geladen dargestellt etc. werden, wird die Abarbeitung abgebrochen.

Siehe auch =>
```

1.169 WRITE.guide/Ask

```
                Ask
===

Aufruf       : Ask Title/S Gadgets/S Flags/T

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Ja

Beschreibung : Ask öffnet einen Requester mit Title als Text und den in
Gadgets beschriebenen Gadgets als mögliche Antwort.  Ist in Flags das
Tag  @VERT gesetzt, so wird rechts neben dem Text die Gadgeliste vertikal
erzeugt, sonst horzontal unter dem Text.

Die Beschreibung der Gadgets folgt folgender Konvention: Die Gadgettitel
werden hintereinander aufgelistet und durch das Zeichen | von einander
getrennt.  Mittels eines Unterstriches im Titel kann der nächste
Buchstabe als Tastaturkürzel markiert werden.  Ein ^ bedeutet, daß
dieses Gadget auch über die Esacape-Taste selectiert werden kann.
Dieses Gadget sollte demnach für einen Abruch oder für das sichere
Verlassen des Requester stehen.  Ein * bedeutet, daß dieses Gadget auch
mit RETURN beantwortet werden kann.  Dieses Gadget sollte also für die
vorgeschlagene Standardbeanwortung stehen.
```

```
Die Gadgets werden von Links nach Rechts von 0 aufwärts durchnummeriert.
Ask gibt in
                _RN
                 die Nummer des selektierten Gadgets zurück.  Ein
Beispiel:
     Ask "Dies ist ein Test !!!" "Echt _super man !*|_Nicht so toll^"

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                Requester
                ,
                Message
                ,
                MessageOK
```

1.170 WRITE.guide/Message

```
                Message
=======

Aufruf       : Message Text/S

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Message öffnet einen Requester mit dem angegebenen Text.
Nach kurzer Zeit verschwindet der Requester wieder.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                MessageOK
                ,
                Requester
```

1.171 WRITE.guide/MessageOK

```
                MessageOK
=========

Aufruf       : MessageOK Text/S

Benötigt     : Nichts
```

```
Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Funktioniert wie
                Message
                , nur das hier die Meldung erst
durch ein Gadget bestätigt werden muß bevor sie verschwindet.

Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                Message
                ,
                Requester
```

1.172 WRITE.guide/Flash

```
                Flash
=====

Aufruf       : Flash

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Läßt den Bildschirm einmal aufblinken.

Siehe auch =>
                Beep
```

1.173 WRITE.guide/Beep

```
                Beep
====

Aufruf       : Beep Deep/N

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Beep erzeugt einen kurzen Ton.  Ist Deep=0, dann ist der
```

```
Ton relativ hoch, sonst tief.

Siehe auch =>
                Flash
```

1.174 WRITE.guide/ParseBuffer

```
                ParseBuffer
===========

Aufruf       : ParseBuffer Nummer/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : ParseBuffer behandelt den Text in dem Puffer mit der
angegebenen Nummer als WRITE-Script und prüft es auf seine syntaktische
Richtigkeit ohne es dabei jedoch auszuführen.  Eventuell auftretende
Fehler werden wie üblich im Textfenster gemeldet.  Sind Fehler
aufgetreten, so wird dementsprechend von der Funktion ein Fehler
zurückgegeben.

Diese Funktion ist sinnvoll, wenn man Teile des Konfigurationsfiles
ändert, und sie anschließend auf ihre Richtigkeit überprüfen will.

Siehe auch =>
                DoBuffer
                ,
                DoString
                ,
                PreparseString
```

1.175 WRITE.guide/DoBuffer

```
                DoBuffer
========

Aufruf       : DoBuffer Nummer/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Führt den angegebenen Puffer als WRITE-Script aus.
Treten dabei Fehler auf, so gibt die Funktion ebenfalls einen Fehler aus.
```

```
Siehe auch =>
                ParseBuffer
                ,
                DoString
                ,
                PreparseString
```

1.176 WRITE.guide/DoString

```
                DoString
========

Aufruf       : DoString String/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Führt den angegebenen String als WRITE-Script aus.
Natürlich gibt auch diese Funktion dabei auftretene Fehler als Fehler
zurück.

Siehe auch =>
                ParseBuffer
                ,
                DoBuffer
                ,
                PreparseString
```

1.177 WRITE.guide/PreparseString

```
                PreparseString
==============

Aufruf       : PreparseString String/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : PreparseBuffer parsed den String, kontrolliert auf
Richtigkeit und ersetzt dabei Variablen durch ihre Inhalte.  Der String
wird in
                _RS
                 zurückgegeben.
```

```
Siehe auch =>
                DoBuffer
                ,
                DoString
                ,
                ParseBuffer
```

1.178 WRITE.guide/SetUserFkt

```
                SetUserFkt
==========

Aufruf       : SetUserFkt Nummer/N Funktionsliste/F

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Userfunktionen sind Stellen im internen Ablauf, wo der
Benutzer konfigurierend eingreifen kann.  Momentan gibt es folgende
Userfunktionen:

0 : Diese Funktion wird für jeden Commandlineparameter einmal aufgerufen.
Der Parameter steht dabei in der Variablen
                _RS
                .  Diese Funktion sollte
ein Fenster öffnen und die angegebene Textdatei laden.

1 : Diese Funktion wird aufgerufen, wenn das Schließ-Gadget eines
Fenster angeklickt wird.  Die Funktion sollte eventuell eine
Sicherheitsabfrage machen und anschließend das Fenster schließen.

2 : Diese Funktion wird bei Selektion des Gadgets Neues Fenster im

                WinManager
                 aufgerufen. Sie sollte ein Editorfenster öffnen.

3 : Diese Funtion wird nicht mehr unterstützt.

4 : Diese Funktion wird aufgerufen, wenn eine Konfiguration aus dem
Speicher befreit wird. Gibt diese Funktion einen Fehler zurück. So wird
die Freigabe abgebrochen. Hat man z.B. im entsprechenen
Konfigurationsfile z.B. einem externen Screenmanager gesagt, daß er
einen Screen für unsere Konfiguration öffnen soll, kann man ihn in
dieser Funktion sagen, daß der Screen nicht mehr gebraucht wird und somit
geschlossen werden kann. Sehen sie hirzu auch bei der Funktion
                Screen
                 und
im Kapitel
                PublicScreens
                 nach. Für diesen Zweck existieren mittlerweile
```

die Befehle
Begin
/
Close
für Konfigurationsfiles und
Start
/
Quit
für die
STARTUP.CONFIG. Bitte diese Befehle nutzen !

Für Beispiele hierzu schauen sie bitte in die Konfigurationsdateien.

Siehe auch =>

1.179 WRITE.guide/Compare

Compare
=======

Aufruf : Compare String1/S String2 /S

Benötigt : Nichts

Setzt Fehler : Nein

Ergebnisse : Ja

Beschreibung : Vergleicht die beiden Strings und gibt je nach Ergebnis
die Werte
LOWER
,
HIGHER
oder
EQUAL
in
_RN
zurück. Da Zahlen bei der
Übergabe automatisch in Zeichenketten umgewandelt werden, kann man mit
dieser Funktion auch Zahlen untereinander als auch Zahlen und
Zeichenketten miteinander vergleichen.

Siehe auch =>
If
,
Break

1.180 WRITE.guide/If

```
                If
==

Aufruf       : If Command/F Then/F Else/F Tags/T

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : If führt die angegebene Funktionsliste Command aus.  Gibt
diese Funktion keinen Fehler zurück und ist
                _RN
                =0, so wird die
Funktionsliste Then ausgeführt, ansonsten die Funktionsliste Else. Ist
das Tag  @CLEARRN gesetzt, so wird
                _RN
                 vor der Ausführung von Command
gelöscht. Dies ist z.B. wichtig, wenn
                _RN
                 ungleich 0 ist, Command
                _RN
                nicht ändert und If dementsprechend nur auf die Rückgabe eines  ↩
                   Fehlers
reagieren soll.

Siehe auch =>
                Break
```

1.181 WRITE.guide/Break

```
                Break
=====

Aufruf       : Break Compare/N String1/S String2/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Break vergleicht String1 und String2 Und stellt, das sie
gleich (
                EQUAL
                ), String1 kleiner als String2 (
                LOWER
                ) oder größer (
                EQUAL
                )
ist.  Ist nun Compare gleich dem Resultat, so gibt Break einen Fehler
zurück und die aktuelle Befehlsfolge wird abgebrochen.  Beispiel :
```

```
     Break EQUAL "Hallo" "Hallo"
     Break LOWER 255 13
Das erste Beispiel bricht ab, das Zweite nicht.

Siehe auch =>
                If
```

1.182 WRITE.guide/SetError

```
                SetError
========

Aufruf       : SetError

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Bricht die anktuelle Anweisungsfolge ab.

Siehe auch =>
                Break
                ,
                If
                ,
                Compare
```

1.183 WRITE.guide/SetVar

```
                SetVar
======

Aufruf       : SetVar Variable/S Wert/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Setzt den Inhalt der einer internen Variable auf den Wert
Wert.  Variable ist dabei der gequotete Name der Variable.  Existiert die
Variable nichts, so wird ein Fehler zurückgegeben.  Beispiel :
     SetEnv "_FileName" "s:startup-sequence"
Setzt den Dateinamen des aktuellen Textes auf s:startup-sequence.

Siehe auch =>
                GetVar
```

1.184 WRITE.guide/GetVar

```
                GetVar
======

Aufruf       : GetVar Variable/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Liest den Wert der Variablen Variable aus und schreibt ihn
als String in die Variable
                _RS
                .  Existiert die angegebene Variable nicht,
so wird ein Fehler zurückgegeben.

Siehe auch =>
                SetVar
```

1.185 WRITE.guide/GetConst

```
                GetConst
========

Aufruf       : GetConst Konstante/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Liest den Wert der Konstante Konstante aus und schreibt
ihn als String in die Variable
                _RS
                .  Existiert die angegebene Konstante
nicht, so wird ein Fehler zurückgegeben.

Siehe auch =>
                SetVar
```

1.186 WRITE.guide/SetEnv

```
SetEnv
======

Aufruf       : SetEnv Variable/S Wert/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Setzt den Wert einer lokalen DOS-Enviromentvariabeln auf
den Wert Wert.

Siehe auch =>
```

1.187 WRITE.guide/GetEnv

```
                GetEnv
======

Aufruf       : GetEnv Variable/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Ließt den Wert der angegebenen Dos-Enviromentvariabeln
aus und schreibt ihn in die Variable
               _RS
               Siehe auch =>
```

1.188 WRITE.guide/System

```
System
======

Aufruf       : System Befehl/S Flags/T

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : System startet einen Befehl (mit Argumenten).  Ist  @ASYNC
```

ind Flags gesetzt, so wird der Befehl im Hintergrund gestartet. D.h., daß mit WRITE während das Programm läuft, weiter gearbeitet werden kann. Ansonsten wartet WRITE darauf, daß das Programm beendet wird. Aus leicht einsichtigen Gründen, kann nur ein Fehler zurückgegeben werden, wenn @ASYNC nicht gesetzt ist.

Siehe auch =>

1.189 WRITE.guide/SetMark

SetMark
=======

Aufruf : SetMark

Benötigt : Ed

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : Setzt den Anfang bzw. das Ende eines Blockes,welcher dann mit Cut, Copy ... bearbeitet werden kann. Setzt mann zum ersten Mal eine Marke, so verändert sich der Mauszeiger in ein Visier (???) und ein M erscheint bei den Flags in der Titelzeile. Desweiteren ist ein zeichengroßes Kästen unter dem Cursor zu sehen. Man ist im Mark-Modus. Dieser bleibt solange erhalten , wie man nicht auf irgendeine Art einen Zeilenumbruch tätigt. Nun kann mit den gleichen Verfahren das Ende des Blockes gewählt werden. Ist das Ende des Blockes vor dem Anfang, so werden Anfangs- und Endmarken automatisch vertauscht. Ist dies getan, so wird der Block als Ganzes farblich hervorgehoben. Der Block ist markiert und bleibt dies auch, bis ein Zeilenumbruch durchgeführt wurde. Nun kann der Block mit den Blockfunktionen bearbeitet werden.

Mit einen Doppelklick mit der linken Maustaste ist es z.B. möglich Marken zu setzen.

Siehe auch =>
UnMark
,
Mark

1.190 WRITE.guide/Mark

Mark
====

Aufruf : Mark VonX,VonY,BisX,BisY/N Tags/T

Benötigt : Ed

```
Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Markiert den angegebenen Block.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                Mark
                ,
                UnMark
                ,
                Konstantenbeschreibung
```

1.191 WRITE.guide/UnMark

```
                UnMark
======

Aufruf       : UnMark

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : UnMark löscht den markierten Block.  Diese Funktion wird
auch aufgerufen, wenn der rechte Mausknopf gedrückt wurde, ohne ein
Menü auszuwählen.

Siehe auch =>
                SetMark
                ,
                Mark
```

1.192 WRITE.guide/DeleteBlock

```
                DeleteBlock
===========

Aufruf       : DeleteBlock

Benötigt     : Ed

Setzt Fehler : Ja
```

```
Ergebnisse   : Nein

Beschreibung : Löscht den markierten Block.  Ist kein Block markiert, so
wird mit einem Fehler abgebrochen.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                CopyBlock
                ,
                InsertBlock
                ,
                CopyArea
                ,
                DeleteArea
```

1.193 WRITE.guide/CopyBlock

```
                CopyBlock
=========

Aufruf       : CopyBlock Puffername/S

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Kopiert den markierten Block in den angegebenen
Zwischenspeicher.  Ist kein Block markiert, so wird mit einem Fehler
abgebrochen.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                DeleteBlock
                ,
                InsertBlock
                ,
                CopyArea
                ,
                DeleteArea
```

1.194 WRITE.guide/InsertBlock

```
                InsertBlock
===========
```

```
Aufruf       : InsertBlock Puffername/S

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Fügt den Inhalt des angegebenen Blocks an der aktuellen
Cursorposition in den Text ein.  Ist der angegebene Block leer, so wird
mit einem Fehler abgebrochen.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                CopyBlock
                ,
                DeleteBlock
                ,
                CopyArea
                ,
                DeleteArea
```

1.195 WRITE.guide/DeleteArea

```
                DeleteArea
==========

Aufruf       : DeleteArea VonX,VonY,BisX,BisY/N

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Genau die gleiche Funtion wie
                DeleteBlock
                , nur das hier
ein beliebiger Block angegeben werden kann.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                CopyBlock
                ,
                InsertBlock
                ,
                CopyArea
                ,
                Konstantenbeschreibung
```

1.196 WRITE.guide/CopyArea

```
                CopyArea
========

Aufruf       : CopyArea Buffername/S VonX,VonY,BisX,BisY/N

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Genau die gleiche Funtion wie
                CopyBlock
                , nur das hier ein
beliebiger Block angegeben werden kann.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                DeleteBlock
                ,
                InsertBlock
                ,
                DeleteArea
                ,
                Konstantenbeschreibung
```

1.197 WRITE.guide/SaveBuffer

```
                SaveBuffer
==========

Aufruf       : SaveBuffer Dateiname/S Puffername/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Speichert den Inhalt des angegebenen Puffers als Datei mit
dem Namen Dateiname ab.  Treten beim Abspeichern Fehler auf, so wird mit
einem Fehler abgebrochen.  Achtung !  Die Einstellungsmöglichkeiten für
Textfiles, wie z.B. die Abspeicherung von Tabs, gelten hier nicht.
Blöcke werden immer im Standard-ADSCII-Format abgespeichert.
```

```
Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                LoadBuffer
```

1.198 WRITE.guide/LoadBuffer

```
                LoadBuffer
==========

Aufruf       : LoadBuffer Dateiname/S Puffername/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Läd die Datei Dateiname in den angegebenen Block.  Treten
beim Laden Fehler auf, so wird mit einem Fehler abgebrochen.  Achtung !
Die Einstellungsmöglichkeiten für Textfiles, wie z.B. die Abspeicherung
von Tabs, gelten hier nicht.  Blöcke werden immer im
Standard-ADSCII-Format abgespeichert.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                SaveBuffer
```

1.199 WRITE.guide/ClearBuffer

```
ClearBuffer
===========

Aufruf       : ClearBuffer Puffername/S

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Löscht den angegebenen Puffer. Ist der Buffer noch nicht
vorhanden wird er erzeugt.

Siehe auch =>
```

1.200 WRITE.guide/StrToBuffer

```
StrToBuffer
===========

Aufruf       : StrToBuffer String/S Puffername/S

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Der angegebene Buffer wird mit dem angegebenen String
belegt.

Siehe auch =>
```

1.201 WRITE.guide/BufferToStr

```
BufferToStr
===========

Aufruf       : BufferToStr Puffername/S

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Ja

Beschreibung : Die maximal ersten 256 Zeichen des angegebenen Buffers
werden in _RS zurückgegeben.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
```

1.202 WRITE.guide/ClipToBuffer

```
                ClipToBuffer
============

Aufruf       : ClipToBuffer Puffername/S Clipboardnummer/N

Benötigt     : iffparse.library

Setzt Fehler : Ja
```

```
Ergebnisse   : Nein

Beschreibung : Kopiert den Inhalt des angebenen Clipboarddevices in eine
internen Puffer.  Für weitere Informationen über die Funktion und die
Arbeitsweise des Clipboarddevices schauen sie bitte in die Handbücher,
die sie mit ihrem Amiga erhalten haben.

Siehe auch =>
                BufferToClip
```

1.203 WRITE.guide/BufferToClip

```
                BufferToClip
============

Aufruf       : BufferToClip Puffername/S Clipboardnummer/N

Benötigt     : iffparse.library

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Kopiert den angegeben internen Puffer in ein
Clipboarddevice.  Für weitere Informationen über die Funktion und die
Arbeitsweise des Clipboarddevices schauen sie bitte in die Handbücher,
die sie mit ihrem Amiga erhalten haben.

Durch das Setzten des Tags  @SILENT können eventuelle Fehlerrequester
unterdrückt werden.

Siehe auch =>
                BufferToClip
```

1.204 WRITE.guide/BlockLeft

```
                BlockLeft
=========

Aufruf       : BlockLeft x/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Verschiebt den markierten Block als Ganzes um x Zeichen
nach links.
```

```
Siehe auch =>
                BlockRight
```

1.205 WRITE.guide/BlockRight

```
                BlockRight
==========

Aufruf       : BlockRight x/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Verschiebt den markierten Block als Ganzes um x Zeichen
nach rechts.

Siehe auch =>
                BlockRight
```

1.206 WRITE.guide/BlockLftAlig

```
                BlockLftAlig
============

Aufruf       : BlockLftAlig Spalte/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Versucht alle Zeilen im markierten Block linksbündig auf
die angegebene Spalte auszurichten.

Siehe auch =>
                BlockRghtAlig
```

1.207 WRITE.guide/BlockRghtAlig

```
BlockRghtAlig
=============

Aufruf       : BlockRghtAlig Spalte/N
```

```
Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Versucht alle Zeilen im markierten Block rechtsbündig auf
die angegebene Spalte auszurichten.

Siehe auch =>
```

1.208 WRITE.guide/BlockCenter

```
BlockCenter
===========

Aufruf       : Diese Funktion ist noch nicht implementiert

Benötigt     :

Setzt Fehler :

Ergebnisse   :

Beschreibung :

Siehe auch =>
```

1.209 WRITE.guide/CursorUp

```
CursorUp
========

Aufruf       : CursorUp Mode/N

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Ist Mode=0, so wird der Cursor einfach eine Zeile nach
oben bewegt.  Ist Mode=1, so wird der Cursor ebenfalls eine Zeile nach
oben bewegt und, falls der Cursor dann hinter dem letzten Zeichen in
dieser Zeile steht, einen Buchstaben hinter dem Ende der Zeile
positioniert.  Steht der Cursor bereits in der ersten Zeile, so wird ein
Fehler zurückgegeben.

Siehe auch =>
```

1.210 WRITE.guide/CursorDown

```
                CursorDown
==========

Aufruf       : CursorDown Mode/N

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Funktioniert wie
                CursorUp
                , nur daß der Cursor eine Zeile
nach unten bewegt wird.

Siehe auch =>
```

1.211 WRITE.guide/CursorRight

```
CursorRight
===========

Aufruf       : CursorRight Mode/N

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Ist Mode=0, so wird der Cursor eine Stelle nach rechts
bewegt.  Ist Mode=1, so wird der Cursor ebenfalls eine Stelle nach rechts
bewegt.  Steht er dann jedoch hinter dem letzten Zeichen der Zeile,
springt er an der Anfang der nächsten.  Existiert diese nicht, so wird
ein Fehler zurückgegeben.

Siehe auch =>
```

1.212 WRITE.guide/CursorLeft

```
CursorLeft
==========

Aufruf       : CursorLeft

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung :  Ist Mode=0, so wird der Cursor eine Stelle nach links
bewegt.  Steht der Cursor bereits in der ersten Spalte, so wird ein Fehler
zurückgegeben.  Ist Mode=1, so wird der Cursor ebenfalls eine Stelle
nach links bewegt.  Steht er jedoch in der ersten Spalte der Zeile,
springt er ans Ende der letzten.  Existiert diese nicht, so wird ein
Fehler zurückgegeben.

Siehe auch =>
```

1.213 WRITE.guide/NextWord

```
                NextWord
========

Aufruf       : NextWord

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Springt zum nächsten Wort.  Die aktuelle Wortdefinition
wird durch den Inhalt der Variable
                _WordDef
                 angegeben.  Schauen sie für
weitere Information bitte dort nach.  Kann keine nächstes Word gefunden
werden, so gibt NextWord einen Fehler zurück.

Siehe auch =>
                LastWord
```

1.214 WRITE.guide/LastWord

```
                LastWord
========

Aufruf       : LastWord
```

```
Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Springt zum nächsten Wort.  Die aktuelle Wortdefinition
wird durch den Inhalt der Variable
                _WordDef
                 angegeben.  Schauen sie für
weitere Information bitte dort nach.  Kann keine vorheriges Wort gefunden
werden, so gibt NextWord einen Fehler zurück.

Siehe auch =>
                NextWord
```

1.215 WRITE.guide/PageUp

```
                PageUp
======

Aufruf       : PageUp Percent/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Blättert Percent Prozent einer dargestellten Seite nach
oben.  Beispiel :
     PageUp 75
Blättert 75% der dargestellten Seite nach oben.  Bei einem Fenster von
z.B. 30 Zeilen sind dies 21 Zeilen.

Siehe auch =>
                PageDown
```

1.216 WRITE.guide/PageDown

```
                PageDown
========

Aufruf       : PageDown Percent/N

Benötigt     : Ed

Setzt Fehler : Nein
```

```
Ergebnisse   : Nein

Beschreibung : Blättert Percent Prozent einer dargestellten Seite nach
unten.  Beispiel :
     PageDown 75
Blättert 75% der dargestellten Seite nach unten.  Bei einem Fenster von
z.B. 30 Zeilen sind dies 21 Zeilen.

Siehe auch =>
                PageUp
```

1.217 WRITE.guide/Goto

```
                Goto
====

Aufruf       : Goto x/N y/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Goto springt, wenn vorhanden zur x. Zeile und y. Spalte.
Dabei werden auch Positionsangaben mittels der Konstanten  @CURSOR bis
Siehe auch =>
                Konstantenbeschreibung
```

1.218 WRITE.guide/GotoMouse

```
                GotoMouse
=========

Aufruf       : GotoMouse Tags/T

Benötigt     : Ed

Setzt Fehler : ja

Ergebnisse   : Nein

Beschreibung : Positioniert den Cursor unter den Mauszeiger. Ist das Tag
über dem Cursor befindet. Dies ist in Zusammenhang mit Doppelklicks
(
                DoubleKey
                 und
                SetMark
                 nützlich.
```

```
Siehe auch =>
```

1.219 WRITE.guide/SetTextMark

```
                SetTextMark
===========

Aufruf       : SetTextMark Nummer/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : SetTextMark setzt in der aktuellen Zeile eine (nämliche
Nummer Nummer) temporäre Marke.  Beachten sie bitte, daß sich WRITE nur
die Zeilennummer merkt, so daß nach Veränderungen im Text die Marke auf
die falsche Stelle zeigt.

Siehe auch =>
                GoTextMark
```

1.220 WRITE.guide/GoTextMark

```
                GoTextMark
==========

Aufruf       : GoTextMark Nummer/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Springt zur angegebenen Textmarke.

Siehe auch =>
                SetTextMark
```

1.221 WRITE.guide/Find

```
                Find
====
```

```
Aufruf       : Find VonX/N VonY/N BisX/N BisY/N Mode/T

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Find sucht ab der aktuellen Cursorposition nach den Inhalt
der Variable
                _FindString
                .  Ist der Inhalt der Variable
                _CaseSense
                ungleich 0, so unterscheidet Find dabei zwischen Groß- und
Kleinschreibung, ansonsten nicht.  Kann das gesuchte Wort nicht gefunden
werden, so wird ein Fehler zurückgegeben.

Ist der Inhalt der Variable
                _WordDef
                 ungleich 0, so sucht Find nur nach
ganzen Wörtern, ansonsten auch nach Wortteilen. Die Definition eines
Wortes hängt von dem Inhalt der Variable
                _WordDef
                 ab.

Kann das gesuchte Wort nicht gefunden werden, so wird ein Fehler
zurückgegeben.

Ist in Mode  @SILENT gesetzt, so wird das Nichtfinden des Wortes nicht
mittels eines Requesters angezeigt.

Mittels der Tags  @CASE,  @NOCASE,  @WORD und  @NOWORD kann der Wert der
Variablen
                _CaseSense
                 und
                _WordDef
                 für diesen Funktionaufruf direkt
geändert werden, ohne daß globale Einstellungen verändert werden.

Über die Variablen VonX/Y und BisX/Y kann der Textbereich angegeben
werden, auf den sich die Suche beziehen soll.  @CURSOR @CURSOR @EOT @EOT
steht z.B. für die Suche von der Cursorposition bis zum Ende des Textes.
 @MARKA @MARKA @MARKB @MARKB für die Suche in dem markierten Block.

Beachten sie, daß, wenn VonX/Y gleich der Cursorposition ist, Find aus
einsichtigen Gründen erst ab den nachfolgenden Buchstaben sucht. Dieses
Verhalten kann durch das Setzten des Tags  @FIRST abgestellt werden.

Wird das Tag  @COUNT gesetzt, so springt Find nicht zum ersten Vorkommnis
sondern die die zahl der gefundenen Wörter im angegebenen Bereich in
                _RN
                zurück.

Siehe auch =>
                Replace
                ,
                FindPattern
```

,
Konstantenbeschreibung

1.222 WRITE.guide/Replace

```
                Replace
=======

Aufruf       : Replace VonX/N VonY/N BisX/N BisY/N Mode/T

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Replace sucht nach dem unter
                Find
                 angegebenen Schema nach
dem angegebenen Wort und ersetzt es durch den Inhalt der Variable

                _ReplaceString
                .

Ist  @NOREQ nicht in Modegesetzt, so wird vorher ein Requester geöffnet,
in dem das Ersetzen explizit bestätigt werden muß.

Ist auch hier in Mode  @SILENT gesetzt, so wird das Nichtfinden des
Wortes nicht mittels eines Requesters angezeigt.

Ist  @ALL gesetzt, so ersetzt die Funktion nicht nur das nächste
Vorkommen, sonndern alle Vorkommen bis des Ende des Textes erreicht ist
oder die Suche abgebrochen wurde.

Ist bei gesetztem  @ALL  @NOREQ nicht in Modegesetzt, so wird bei jedem
gefundenen Wort ein Requester geöffnet, in dem man Replace abbrechen
kann, das aktuelle Wort uberspringen oder ersetzten kann.

Für die Parameter VonX/Y, BisX/Y sowie weitere Tags sehen sie Bitte bei
der Beschreibung der Funktion
                Find
                 nach.

Die Zahl der Ersetzungen wird in _RN zurückgegeben.

Siehe auch =>
                Find
                ,
                FindPattern
                ,
                Konstantenbeschreibung
```

1.223 WRITE.guide/ReplaceList

```
                ReplaceList
===========

Aufruf       : ReplaceList xStart/N yStart/N xEnd/N yEnd/N Liste/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Noch nicht implementiert.

Siehe auch =>
                Konstantenbeschreibung
```

1.224 WRITE.guide/FindPattern

```
                FindPattern
===========

Aufruf       : FindPattern Von,Bis/N Flags/T

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Sucht von der Zeile von bis zur Zeile bis nach dem
DOS-Pattern, welches in der Variable
                _PatCase
                 bzw.
                _PatNoCase
                 angegeben
ist. Dies hängt davon ab, ob  @CASE oder  @NOCASE in Flags angegeben
wird. Wird kein von beiden angegeben, so wird die aktuelle
Preferenceseinstellung benutzt. Durch das setzten von  @SILENT kann die
Meldung des Nichtfindens des Patterns abgeschaltet werden.

Steht der Cursor in der Startzeile, so wird es ab der Position hinter dem
Cursor gesucht.

Siehe auch =>
                Find
                ,
                Replace
                ,
                Konstantenbeschreibung
```

1.225 WRITE.guide/MatchBracket

```
MatchBracket
============

Aufruf       : MatchBracket

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : MatchBracket sucht zu dem Zeichen unter dem Cursor das
konjugierte.  Unterstützt werden folgende Paare : ( ), { }, [ ], < >, «
».  MatchBracket unterstützt dabei auch Verschachtelungen.  Im
folgenden Beispiel springt MatchBracket, wenn der Cursor auf dem ersten
Zeichen steht, zum letzten :
     {
       {
         /* Dies ist ein Test */
         arg[0]:=0;
       }
     }
Desweiteren wird für die folgenden Zeichen, zum nächsten Vorkommen
gesprungen : ', `, ".

Eine interessante Tastaturbelegung ist:


     KEY ")"
       WRITETEXT ")"
       CURSORLEFT 0
       MATCHBRACKET
       DELAY 10
       MATCHBRACKET
       CURSORRIGHT 0
       ;

Was dafür sorgt, daß der Cursor, immer wenn man ) drückt, der Cursor
kurz zu entsprechenden öffnenden Klammer springt.

Siehe auch =>
```

1.226 WRITE.guide/Return

```
                Return
======

Aufruf       : Return

Benötigt     : Ed
```

```
Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Bricht die aktuelle Zeile an der Cursorposition um.
Beispiel :
       sadlkjsadf asf gra htrhtrzrtgre
                         ^
                         Cursor

      wird zu...

       sadlkjsadf asf gra h
     trhtrzrtgre

Siehe auch =>
                _AutoIndent
```

1.227 WRITE.guide/Delete

```
Delete
======

Aufruf       : Delete

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Löscht das Zeichen unter dem Cursor.  Steht der Cursor
hinter dem letzten Zeichen einer Zeile, so wird die nächste Zeile in die
Aktuelle geholt.

Siehe auch =>
```

1.228 WRITE.guide/DeleteToEOL

```
                DeleteToEOL
===========

Aufruf       : DeleteToEOL

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein
```

```
Beschreibung : DeleteToEOL löscht die aktuelle Zeile von der aktuellen
Cursorposition bis zum Ende der Zeile.

Siehe auch =>
                Delete
                ,
                DeleteLine
                ,
                UnDelLine
```

1.229 WRITE.guide/DeleteLine

```
                DeleteLine
==========

Aufruf       : DeleteLine

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : DeleteLine löscht die aktuelle Zeile.  Die zuletzt
gelöschte Zeile kann kann mit
                UnDelLine
                 wieder eingefügt werden.
Desweitern wird sie an den Anfang der Liste DeleteLine-History angefügt.

Siehe auch =>
                Delete
                ,
                DeleteToEOL
                ,
                UnDelLine
```

1.230 WRITE.guide/UnDelLine

```
                UnDelLine
=========

Aufruf       : UnDelLine

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : UnDelLine fügt die zuletzt mit
```

```
                DeleteLine
                 gelöschte
Zeile wieder über der aktuellen ein. Ist das Tag  @NODUP gesetzt, wird
diese Zeile gleichzeitig aus der Liste DeleteLine-History gelöscht.

Siehe auch =>
                Delete
                ,
                DeleteToEOL
                ,
                DeleteLine
```

1.231 WRITE.guide/BackSpace

```
BackSpace
=========
```

1.232 WRITE.guide/Tab

```
                Tab
===

Aufruf       : Tab Mode/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Tab springt zur nächsten Tabulatormarke.  Diese sind
gerade
                _TabLength
                 Zeichen von einander entfehrnt.  Ist Mode=0, so springt
nur der Cursor zum Tabulator.  Ist Mode=1 so der Text unter dem Cursor
bis zum Tabulator bewegt.

Siehe auch =>
                BackTab
```

1.233 WRITE.guide/BackTab

```
                BackTab
=======

Aufruf       : BackTab
```

```
Benötigt     : Ed

Setzt Fehler : Nichts

Ergebnisse   : Nichts

Beschreibung : BackTab springt zur vorherigen Tabulatormarke.  Diese sind
gerade
                _TabLength
                 Zeichen von einander entfehrnt.

Siehe auch =>
```

1.234 WRITE.guide/UpperBlock

```
                UpperBlock
==========

Aufruf       : UpperBlock VonX,VonY,BisX,BisY/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Konvertiert alle Buchstaben im angebenen Textbereich in
Großbuchstaben. Dabei werden auch internationale Zeichen je nach der
aktuellen Ländereinstellung des Betriebssystems berücksichtigt.

Die Positionen werden dabei mittels der Konstanten  @CURSOR bis  @EOT
angegeben.

Siehe auch =>
                LowerBlock
                ,
                Konstantenbeschreibung
```

1.235 WRITE.guide/LowerBlock

```
                LowerBlock
==========

Aufruf       : LowerBlock VonX,VonY,BisX,BisY/N

Benötigt     :

Setzt Fehler :
```

```
Ergebnisse   :

Beschreibung : Konvertiert alle Buchstaben im angegebenen Textbereich zu
Kleinbuchstaben. Dabei werden auch internationale Zeichen je nach der
aktuellen Ländereinstellung des Betriebssystems berücksichtigt.

Die Positionen werden dabei mittels der Konstanten  @CURSOR bis  @EOT
angegeben.

Siehe auch =>
                UpperBlock
                ,
                Konstantenbeschreibung
```

1.236 WRITE.guide/WriteChar

```
                WriteChar
=========

Aufruf       : WriteChar DecimalCode/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : WriteChar schreibt das Zeichen mit dem ASCII-Code
DecimalCode an die aktuelle CursorPosition.  Das Zeichen geht dabei an
die gleiche Routine die auch für die über die Tastatur eingegebenen
Zeichen zuständig ist.  D. h. alle nicht dastellbaren Zeichen, werden
auch nicht gedruckt.  Diese Funktion ist auch völlig unabhängig von der
aktuellen Tastaturbelegung.  Ist A mit einer Funktion belegt, so wird
WriteChar 65 dennoch ein A schreiben.

Siehe auch =>
                WriteText
```

1.237 WRITE.guide/WriteText

```
                WriteText
=========

Aufruf       : WriteText Zeichenkette/S

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein
```

```
Beschreibung : WriteText funktioniert genauso wie
                WriteChar
                .  Nur wird
hier ein ganzer String ausgegeben.  Auch hier weren nicht dastellbare
Zeichen herausgefiltert.  Die Möglichkeit der Angabe von Steuersequencen
in der Zeichenkette besteht also nicht.

Siehe auch =>
                WriteChar
```

1.238 WRITE.guide/Key

```
                Key
===

Aufruf       : Key Definition/S Befehlsfolge/F

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Key belegt eine beliebige Taste mit einer Funktionsfolge.
Bei Definition handelt es sich um eine Tastatur- oder Mausbeschreibung,
wie sie auch für die Commodities des Betriebssystem verwendet wird.
Schauen sie bitte für eine genauere Beschreibung in ihren Handbüchern
unter dem Stichwort Commodities nach.  Erlaubt ist momentan nur der Typ
rawkey.

Siehe auch =>
                DoubleKey
```

1.239 WRITE.guide/DoubleKey

```
                DoubleKey
=========

Aufruf       : DoubleKey Definition1,Definition2/S Befehlsfolge/F

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Diese Funktion arbeitet im Prinzip genauso, nur das hier
die Befehlsfolge nur dann ausgeführt wird, wenn erst die Befehlsfoge 1
und dann im Zeitintervall eines Doppellickes die Definition 2 (die
```

unterschiedlich zu der Definition 1 sein kann) gedrückt wird.

Siehe auch =>
Key

1.240 WRITE.guide/ClearKeys

ClearKeys
=========

Aufruf : ClearKeys

Benötigt : Nichts

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : Löscht die komplette mit
Key
eingegebene
Tastaturbelegung. Diese Funktion sollte mit Vorsicht benutzt werden, da nach ihrem Aufruf ein sinnvolles Arbeiten mit WRITE kaum noch möglich ist.

Siehe auch =>
Key

1.241 WRITE.guide/SetHotKey

SetHotKey
=========

Aufruf : SetHotKey Nummer/N Tastensequenz/S Funktion/F

Benötigt : Nichts

Setzt Fehler : Ja

Ergebnisse : Nein

Beschreibung : WRITE besitzt intern eine Reihe von Hotkeys, die mit einer beliebigen Tastenkombinationsbeschreibung über die commodities.library mit einer beliebigen WRITE-internen Funktion belegt werden können. Da die commodities.library benutzt wird, muß die Tastensequenz den Richtlinien, die auch für alle anderen Commodities gelten, folgen. Für das genaue Format der Tastensequenz schauen sie demnach in ihrer Dokumentation, die sie beim Kauf ihres Computers bekommen haben, nach. Funktion ist eine beliebige Funktionsfolge. Ein bereits bestehender HotKey wird durch eine Redefinition überschrieben.

```
Siehe auch =>
```

1.242 WRITE.guide/ClearHotKey

```
                ClearHotKey
===========

Aufruf       :

Benötigt     :

Setzt Fehler :

Ergebnisse   :

Beschreibung :

Siehe auch =>
                DoBuffer
                ,
                DoString
                ,
                ParseBuffer
```

1.243 WRITE.guide/Menu

```
                Menu
====

Aufruf       : Menu Titel/S

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Menu hängt an die Menüleiste ein neues Menü mit dem
Titel Titel an.

Siehe auch =>
                Item
                ,
                Sub
                ,
                ItemBar
                ,
                SubBar
                ,
```

ClearMenu

1.244 WRITE.guide/Item

Item
====

Aufruf : Item Titel/S ShortCut/S Funktionliste/F

Benötigt : Nichts

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : Item hängt an den zuletzt initialisierten Menü einen Menüpunkt mit dem Titel Titel und dem Shortcut ShortCut an und belegt diesen mit der angegeben Funktionsliste. Diese wird immer dann automatisch ausgeführt, wenn der Menüpunkt selektiert wird. Anstelle eines ShortCut-Buchstabens kann auch eine Kommodities-Tastenbeschreiben angegeben werden. Diese erscheint dann ab OS 3.0 neben dem Menüpunkt. Gleichzeitig wird die entsprechenden Taste mit der Funktion des Menüpunktest belegt.

Wird (ab OS3.0) die Helptaste während der Mauszeiger über einem Menü ist gedrückt, so wird ein Requester geöffnet, der die Funktionsbelegung dieses Menüpunktes zeigt. c

Siehe auch =>
Menu
,
Sub
,
ItemBar
,
SubBar
,
ClearMenu

1.245 WRITE.guide/Sub

Sub
===

Aufruf : Sub Titel/S ShortCut/S Funktionliste/F

Benötigt : Nichts

Setzt Fehler : Nein

```
Ergebnisse   : Nein

Beschreibung : Sub hängt an den zuletzt initialisierten Menüpunkt ein
Untermenü mit dem Titel Titel und dem Shortcut ShortCut an und belegt
diesen mit der angegeben Funktionsliste.  Diese wird immer dann
automatisch ausgeführt, wenn das Untermenü selektiert wird.

Weitere Informationen, siehe
                Item
                Für Menüpunkte gibt es auch eine Onlinehilfe. Siehe
                Item
                .

Siehe auch =>
                Menu
                ,
                Item
                ,
                ItemBar
                ,
                SubBar
                ,
                ClearMenu
```

1.246 WRITE.guide/ItemBar

```
                ItemBar
=======

Aufruf       : Itembar

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Hängt an den letzten Menüpunkt einen nicht selektierbaren
Querstricht an.  Diese Funktion sollte zur optischen Aufteilung eines
Menüs benutzt werden.

Siehe auch =>
                Menu
                ,
                Item
                ,
                Sub
                ,
                SubBar
                ,
                ClearMenu
```

1.247 WRITE.guide/SubBar

```
                SubBar
======

Aufruf       : SubBar

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Hängt an das letzte Untermenü einen nicht selektierbaren
Querstricht an.  Diese Funktion sollte zur optischen Aufteilung eines
Menüs benutzt werden.

Siehe auch =>
                Menu
                ,
                Item
                ,
                Sub
                ,
                ItemBar
                ,
                ClearMenu
```

1.248 WRITE.guide/ClearMenu

```
                ClearMenu
=========

Aufruf       : ClearMenu

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Löscht die komplette eingegebene Menübelegung.  Diese
Funktion sollte mit Vorsicht benutzt werden, da nach ihrem Aufruf ein
sinnvolles Arbeiten mit WRITE kaum noch möglich ist.

Siehe auch =>
                Menu
                ,
                Item
                ,
                Sub
                ,
                ItemBar
```

```
                ,
                SubBar
```

1.249 WRITE.guide/WaitPointer

```
                WaitPointer
===========

Aufruf       : WaitPointer

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Setzt für alle Fenster den Warte-Mauszeiger.  Sinnvoll,
wenn z.B. ein Makro oder REXX-Script etwas länger dauert und man dem
Benutzer zeigen möchte, daß WRITE noch arbeitet und nicht etwa
abgestürzt ist.

Siehe auch =>
                NormalPointer
```

1.250 WRITE.guide/NormalPointer

```
                NormalPointer
=============

Aufruf       : NormalPointer

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Setzt Setzt den Mauszeiger wieder in den Normalzustand.

Siehe auch =>
                WaitPointer
```

1.251 WRITE.guide/DoREXX

```
                DoREXX
======
```

```
Aufruf       : DoREXX DateiName/S Flags/T

Benötigt     : REXX-Libraries im libs:-Verzeichnis. RX-Befehl.

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Führt die angegebe Datei als REXX-Script aus.  Je nach
dem, ob  @ASYNC in Flags gesetzt ist oder nicht, wartet WRITE auf
Beendigung des Scripts oder nicht.  Soll das Script auf auch Kommandos an
WRITE schicken, so sollte  @ASYNC gesetzt (das Script wartet sonst
darauf, daß WRITE die Befehle ausführt, während WRITE darauf wartet
das das Script ausgeführt wird), außerdem sollte man vorher eventuell

                LockWindow
                 aufrufen, da die Befehle immer an das aktive Fenster gehen und
wenn während der Ausführung ein anderes Fenster aktiviert wird, gibt es
ein heilloses Chaos.

Siehe auch =>
                LockWindow
                ,
                OpenPort
                ,
                ClosePort
                ,
                WaitPort
```

1.252 WRITE.guide/LockWindow

```
                LockWindow
==========

Aufruf       : LockWindow ID/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : LockWindow lockt den Ed mit dem zugehörigen ID.  Das
heißt, daß alle Befehle, die über die REXX-Schnittstelle kommen, auf den
gelockten Ed wirken, selbst wenn zwischendurch andere Fenster aktiviert
werden etc.

Ein gelocktes Fenster sollte mit LockWindow 0 vor Verlassen des
REXX-Script wieder entlockt werden, da sonst Scripte, die ein Fenster
nicht explizit locken, alle Nachrichten ebenfalls an diesen Ed schicken.
Ein erneuter Aufruf von LockWindow überschreibt einen alten Lock.  Kann
der angegebene ID nicht gefunden werden, so wird ein Fehler
zurückgegeben.
```

```
Sollen mehrere Script gleichzeitig gestartet werden und diese auf
verschiedene Fenster wirken, so benutzen sie bitte die Funktionen

                OpenPort
                ,
                ClosePort
                ,
                WaitPort
                .

Siehe auch =>
                OpenPort
                ,
                ClosePort
                ,
                WaitPort
```

1.253 WRITE.guide/NextEd

```
NextEd
======

Aufruf       : NextEd ID/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Aktiviert den nächsten Ed, dessen ID nach dem angegebenen
kommt.  Kann kein weiterer ID gefunden werden, so gibt NextEd einen Fehler
zurück.

Siehe auch =>
```

1.254 WRITE.guide/OpenPort

```
                OpenPort
========

Aufruf       : OpenPort ID/N

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : OpenPort öffnet einen persönlichen REXX-Port für den Ed
```

```
mit dem angegebenen ID.  Der Name des Ports wird in
                _RS
                 zurückgegeben.
Kann der Port nicht geöffnet werden (ein Task kann nur eine begrenzte
Zahl von Ports verwalten) so wird ein Fehler zurückgegeben.

Bitte beachten sie das man für einen Ed nicht eines zweiten PrivtaePort
öffnen kann. Ein zweiter Aufruf von OpenPort gibt nur einenFehler
zurück.

Siehe auch =>
                ClosePort
                ,
                WaitPort
```

1.255 WRITE.guide/ClosePort

```
                ClosePort
=========

Aufruf       : ClosePort ID/N

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : ClosePort schließt den mit
                OpenPort
                 geöffneten REXX-Port
wieder.  Kann der Ed mit dem angegebenen ID nicht gefunden werden, oder
existiert kein Port, so wird ein Fehler zurückgegeben.

Siehe auch =>
                OpenPort
                ,
                WaitPort
```

1.256 WRITE.guide/WaitPort

```
                WaitPort
========

Aufruf       : WaitPort ID/N

Benötigt     : Ed

Setzt Fehler : Ja
```

```
Ergebnisse   : Nein

Beschreibung : Wird dieser Befehl von einem REXX-Script aus aufgerufen,
so kehrt er erst zurück wenn der REXX-Port des Ed geschlossen wird.
Dies passiert im algemeinen, wenn der Ed verlassen (d.h. meisten, das
Fenster geschlossen) wird.

Siehe auch =>
                OpenPort
                ,
                ClosePort
```

1.257 WRITE.guide/ModifyWin

```
                ModifyWin
=========

Aufruf       : ModifyWin Mode/N

Benötigt     : Fenster

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Mit ModifyWin läßt sich ein offenes Fenster auf
verschiedene Weise manipulieren.  Mode=0 : Fenster wird gezipt.  Mode=1 :
Fenster wird aktiviert.  Mode=2 : Fenster wird nach vorne gebracht.
Mode=3 : Fenster wird nach hinten gebracht.

Siehe auch =>
                ModifyScreen
```

1.258 WRITE.guide/ModifyScreen

```
                ModifyScreen
============

Aufruf       : ModifyScreen Mode/N

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Mit ModifyScreen läßt sich der Screen, auf welchen alle
Fenster und Requester geöffnet werden, nach vorne und nach hinten
bringen.  Mode=0 : Screen nach vorne.  Mode=1 : Screen nach hinten.
```

```
Siehe auch =>
                ModifyWin
```

1.259 WRITE.guide/SetREXXClip

```
SetREXXClip
===========

Aufruf       : SetREXXClip ClipName/S Typ/N Nummer/N

Benötigt     : Nichts/Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : SetREXXClip kopiert den Inhalt einer Zeile/eines Puffers
in das REXXClipboard unter dem angegebenen Namen, wo sie dann mit
getclip(Name) wider ausgelesen werden kann. Ist Typ=0, so wird die Zeile
mit der angegebenen Nummer gelesen. Ist Typ= 1, so wird der angegebene
Puffer ins REXXClipboard geschrieben. Dabei markieren Linefeets den
Zeilenumbruch. Ist beim Auslesen der Zeile kein Ed aktiviert, oder
existiert die angegebene Zeile/der angegebene Puffer nicht, so wird ein
Fehler zurückgegeben.

Siehe auch =>
```

1.260 WRITE.guide/GetConfig

```
GetConfig
=========

Aufruf       : GetConfig Name/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : GetConfig aktiviert die Konfiguration mit dem angegebenen
Namen, damit auf dessen Einstellungen/Variabeln zurückgegriffen werden
kann. GetConfig läd nicht nocht nicht geladene Konfigurationen nach,
auch kann nicht "" für die Standardkonfiguration angegeben werden.

Siehe auch =>
```

1.261 WRITE.guide/SetREXXVar

```
                SetREXXVar
==========

Aufruf       : SetREXXVar Name/S Which/N Welcher/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Schreibt den Inhalt einiger WRITE-internen Werte direkt in
eine AREXX-Variable. Dadurch wird die 256 Bytes Limtierung des Parsers
von WRITE bei Strings umgangen.

Which=0 schreibt den Inhalt der Zeile Welcher in die angegebene Variable.
Which=1 schreibt den Inhalt des Buffers Welcher in die Variable.

Siehe auch =>
                GetREXXVar
```

1.262 WRITE.guide/GetREXXVar

```
                GetREXXVar
==========

Aufruf       : GetREXXVar Name/S Which/N Welcher/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Im Gegensatz zu
                SetREXXVar
                 wird hier der Inhalt einer
REXX-Variablen in eine WRITE-interne Varaile geschrieben. Die
Übergabeparameter sind die gleichen wie bei
                SetREXXVar
                Siehe auch =>
```

1.263 WRITE.guide/Refresh

```
Refresh
=======
```

```
Aufruf       : Refresh

Benötigt     : Fenster

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Aktuallisiert das EditorFenster, d.h. der sichtbare Text
wird neu geschrieben.

Siehe auch =>
```

1.264 WRITE.guide/ChangeConfig

```
ChangeConfig
============

Aufruf       : ChangeConfig ConfigName/S

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Ändert die Konfiguration eines Eds auf die Konfiguration
ConfigName. Die genannte Konfiguration wird ggf. nachgeladen.  Diese
Funktion ist nützlich, wenn man zum Beispiel mit einer minimalen
Konfiguration arbeitet (Mailer etc.) und die Leistungfähigkeit einer
komplexeren Konfiguration braucht.

Siehe auch =>
```

1.265 WRITE.guide/MacroRec

```
                MacroRec
========

Aufruf       : MacroRec

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : MacroRec startet die Aufzeichnung eines Macros.  D. h.
Alle Komandos nach dem Befehl MacroRec bis ausschließlich dem Befehl
MacroStop werden aufgezeichnet.  Dabei sind ein paar Dinge zu beachten:
```

1. Mausbewegungen, sowie Aktionen die mit Mausdrücken verbunden sind, als auch Bewegungen am Scrollbalken werden nicht aufgezeichnet.

2. Fensterwechsel etc. werden nicht aufgezeichnet.

3. Der MacroSaver speichert nicht den (im Konfigurationsfile angegebenen) Quelltext, sondern den erzeugten Zwischencode. So haben z.B. Variablen den Wert zum Zeitpunkt des Abspeicherns nicht den zum Zeitpunkt der Macroausführung.

4. Macros können nicht rekursiv verschachtelt werden. Erneutes Aufrufen von MacroRec löscht das alte und beginnt ein neues Makro.

Siehe auch =>
MacroStop
,
MacroPlay
,
SetMacro
,
ExecuteMacro
,
MacroPannel

1.266 WRITE.guide/MacroStop

MacroStop
=========

Aufruf : MacroStop

Benötigt : Nichts

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : Stopt die Aufzeichnung eines Makros.

Siehe auch =>
MacroRec
,
MacroPlay
,
SetMacro
,
ExecuteMacro
,
MacroPannel

1.267 WRITE.guide/MacroPlay

```
                MacroPlay
=========

Aufruf       : MacroPlay Anzahl/N

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Führt das aufgezeichnete Makro Anzahl mal aus.

Siehe auch =>
                MacroRec
                ,
                MacroStop
                ,
                SetMacro
                ,
                ExecuteMacro
                ,
                MacroPannel
```

1.268 WRITE.guide/SetMacro

```
                SetMacro
========

Aufruf       : SetMacro Nummer/N Name/S Funktionsliste/F

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : SetMacro weist einem internen Array (0...19) eine
Funktionsliste mit den angegebenen Namen als Referenz zu. Dieses kann
dann anschließen über diesen Namen wieder mit der Funktion
                ExecuteMacro
                aufgerufen oder mittels der Funktion
                MacroPannel
                 selektiert werden.

Siehe auch =>
                MacroRec
                ,
                MacroStop
                ,
                ExecuteMacro
```

```
                ,
                MacroPannel
```

1.269 WRITE.guide/ExecuteMacro

```
                ExecuteMacro
============

Aufruf       : ExecuteMacro Name/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Führt das mittels
                SetMacro
                 definierte Makro mit den Namen
Name aus.

Siehe auch =>
                MacroRec
                ,
                MacroStop
                ,
                SetMacro
                ,
                MacroPannel
```

1.270 WRITE.guide/MacroPannel

```
                MacroPannel
===========

Aufruf       : MacroPannel

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Nach dem Aufruf dieser Funktion öffnet sich ein Requester,
in dem alle mit
                SetMacro
                 definierten Makros namendlich in Form von einem
Gadgetpannel aufgeführt werden. Durch drücken eines Gadgets startet man
das entsprechende Makro. Das MacroPannel kann auch mit einem Doppelklick
der rechten Maustaste geöffnet werden.
```

```
Durch das Setzen des Tags  @TOMOUSE wird der Requester zentriert unter
dem Mauszeiger geöffnet. Siehe dazu auch das Kapitel
                Requester
                Siehe auch =>
                MacroRec
                ,
                MacroStop
                ,
                SetMacro
                ,
                ExecuteMacro
                ,
                Requester
```

1.271 WRITE.guide/Undo

```
                Undo
====

Aufruf       : Undo Count/S

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Mittels dieses Befehles, kann man die angegebene Anzahl
von Textveränderungen rückgängig machen. Beachten sie, daß die
maximale Anzahl der gespeicherten Veränderungen vom Wert der Variablen

                _Undo
                 abhängt.

Siehe auch =>
```

1.272 WRITE.guide/ClearList

```
ClearList
=========

Aufruf       : ClearList List/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein
```

```
Beschreibung : Lösche die angegebene Liste.

Siehe auch =>
```

1.273 WRITE.guide/AddList

```
Addlist
=======

Aufruf       : AddList String/S List/N Tags/T

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Hängt an die angegebene Liste den String String an. Ist
in Tags  @NODUP gesetzt, wird der String nun an die Liste angehängt,
wenn er nicht schon bereits vorhanden ist.

Siehe auch =>
```

1.274 WRITE.guide/RemoveList

```
RemoveList
==========

Aufruf       : RemoveList Name/S Element/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Löscht das n-te Element aus der angegebenen Liste. Die
Elemente weden von 1 bis zur Größe der Liste heraufgezählt.

Siehe auch =>
```

1.275 WRITE.guide/Push

```
Push
====
```

```
Aufruf       : Push Name/S Eintrag/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Fügt den Eintrag an Position 1 der Liste ein.

Siehe auch =>
```

1.276 WRITE.guide/Pop

```
                Pop
===

Aufruf       : Pop Name/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Gibt den Inhalt des ersten Elementes der Liste in
                _RS
                zurück und löscht den Eintrag anschließend.

Siehe auch =>
```

1.277 WRITE.guide/ShowList

```
                ShowList
========

Aufruf       : ShowList Lists/N Zahl/N Tags/T

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Zeigt die angegebene Liste mittels eines Listerequester.
Ist  @Select in Tags gesetzt, so kann einer der Strings selektiert
werden. Ist ein String selektiert worden, so gibt ShowList keinen Fehler
zurück. Der String ist aus der Variable
                _RS
```

auslesbar.

Siehe auch =>

1.278 WRITE.guide/ListToBuffer

```
                ListToBuffer
============

Aufruf       : ListToBuffer List/N Buffer/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Konvertiert die angegebene list zu einem Block.

Ist die Liste leer, so wird ein Fehler zurückgegeben.

Siehe auch =>
                BufferToList
```

1.279 WRITE.guide/BufferToList

```
                BufferToList
============

Aufruf       : BufferToList Buffer/N List /N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Konvertiert den angegebenen Buffer in eine Liste.

Ist der angegebene Buffer leer, so wird ein Fehler zurückgegeben.

Folds werden nicht konvertiert.

Siehe auch =>
                ListToBuffer
```

1.280 WRITE.guide/DoList

```
DoList
======

Aufruf       : DoList

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Noch nicht implementiert.

Siehe auch =>
```

1.281 WRITE.guide/ListSize

```
ListSize
========

Aufruf       : ListSize List/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Gibt die Zahl der Einträge in der angegebenen Liste
zurück.

Siehe auch =>
```

1.282 WRITE.guide/GetListEntry

```
                GetListEntry
============

Aufruf       : GetListEntry List/N Entry/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Die den angegebenen Eintrag in einer Liste in
                _RS
```

zurück.

Siehe auch =>

1.283 WRITE.guide/FindListEntry

```
FindListEntry
=============

Aufruf       : FindListEntry List/N String/S

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja

Beschreibung : Such nach dem Eintrag String und gibt, wenn gefunden,
seine Position in der Liste zurück.

kann der Eintrag nicht gefunden werden, wird ein Fehler zurückgegegen.

Siehe auch =>
```

1.284 WRITE.guide/Exists

```
Exists
======

Aufruf       : Exists FileName/F

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Schaut nach, ob der angegebene File existiert. Wenn nicht,
wird ein Fehler zurückgegeben.  Siehe auch =>
```

1.285 WRITE.guide/Delay

```
Delay
=====

Aufruf       : Delay Time/N
```

Benötigt : Nichts

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : Wartet die angegebene Zahl von 1/50 Sekunden. Auf Grund des Parser/Executeroverheads von WRITE, wartet diese Funtion natürlich nie genau die angegebene Zeit.

Siehe auch =>

1.286 WRITE.guide/GuideHelp

GuideHelp
=========

Aufruf : GuideHelp Keyword/S

Benötigt : Nichts

Setzt Fehler : Nein

Ergebnisse : Nein

Beschreibung : Dieser Befehl versucht die Anleitung im Guide-Format über AmigaGuide bzw. MultiView zu laden und versucht eine Referenz zu dem angegebenen Wort zu finden.

Siehe auch =>

1.287 WRITE.guide/VersionCheck

VersionCheck
============

Aufruf : VersionCheck Version/N Name/S

Benötigt : Nichts

Setzt Fehler : Ja

Ergebnisse : Nein

Beschreibung : Kontrolliert, ob die übergebene Versionsnummer kompatibel zur der aktuellen Versionsnummer von WRITE ist. Ist dem nicht so, wird ein Requester geöffnet, der darauf hinweist, daß das angegebene Programme/Konfiguration etc. möglicherweise nicht kompatibel ist. Der Benutzer kann dann entscheiden, ob er weiter machen, oder abbrechen

```
möchte. Bircht er ab, gibt VersionCheck einen Fehler zurück. Durch
Setzten des Tags  @SILENT kann der Requester unterbunden werden.
VersionChec gibt dann bei unterschiedlichen Versionsnummern gleich einen
Fehler zurück.

Die aktuelle Kompatibilitätsversionsnummer steht auch in der Variablen

                _Version
                .

Siehe auch =>
```

1.288 WRITE.guide/Inc

```
                Inc
===

Aufruf       : Inc Zahl,Wert/N

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Ja

Beschreibung : Erhöht Zahl um Wert. Das Ergebnis wird in
                _RN
                zurückgegeben.

Siehe auch =>
                Dec
                ,
                RangeCheck
                ,
                RangeRound
```

1.289 WRITE.guide/Dec

```
                Dec
===

Aufruf       : Dec Zahl,Wert/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Ja
```

```
Beschreibung : Erniedrigt Zahl um Wert. Ist das Ergebnis kleiner als
null, so wird ein Fehler zurückgegeben, ansonsten enthlt
                _RN
                 das
Ergebnis.

Siehe auch =>
                Inc
                ,
                RangeCheck
                ,
                RangeRound
```

1.290 WRITE.guide/RangeCheck

```
                RangeCheck
==========

Aufruf       : RangeCheck Zahl,Min,Max/N

Benötigt     : Nichts

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Liegt Zahl außerhalb des durch Min und Max angegebenen
Intervalls, so wird ein Fehler zurückgegeben.

Siehe auch =>
                Inc
                ,
                Dec
                ,
                RangeRound
```

1.291 WRITE.guide/RangeRound

```
                RangeRound
==========

Aufruf       : RangeRound Zahl,Min,Max/N

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Ja

Beschreibung : Ist Zahl < Min, so wird Zahl gleich Min gesetzt. Ist Zahl
```

```
> Max, so wird Zahl gleich Max gesetzt. Ansonsten bleibt Zahl
unverändert. Das Ergebnis wird in
                _RN
                 zurückgegeben.

Beispiel : Setzten des Cursor 12 Zeilen niedriger


     INC _YPos 12
     RANGECHECK _RN 1 _Length
     GOTO _xPos _RN

Siehe auch =>
```

1.292 WRITE.guide/Begin

```
                Begin
=====

Aufruf       : Begin Funktionsliste/F

Benötigt     : Konfiguration

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Die angegebene Funtionsliste wird ausgeführt, wenn eine
Konfiguration geladen wird.  Sollte am Anfang der Konfiguration stehen.
Hier können manuell Befehle eingefügt werden, die durch die
Benutzeroberfläche nicht direkt programmiert werden können. (Was nicht
heißt, daß man den Inhalt der Funktionsliste nicht über die
Oberfläche geändert werden kann). Alle manuell eingefügten Befehle
müssen hier eingefügt werden, da sie an anderer Stelle eingefügt, von
WRITE vergessen, und beim nächsten Abspeichern der Konfiguration
vergessen werden.

Siehe auch =>
                Screen
                ,
                PublicScreens
```

1.293 WRITE.guide/Close

```
                Close
=====

Aufruf       : Close Funktionsliste/F
```

```
Benötigt     : Konfiguration

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Hier können Funktionen angegeben werden, die ausgeführt
werden, wenn die Konfiguration oder der Editor beendet werden. Hier kann
z.B. ein Screenmanager dazu bewegt werden, einen Screen zu schließen.

Siehe auch =>
                Screen
                ,
                PublicScreens
```

1.294 WRITE.guide/Start

```
Start
=====

Aufruf       : Start Funktionsliste/F

Benötigt     : Nichts

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung :  Die angegebene Funtionsliste wird ausgeführt, wenn der
Editor gestartet wird.  Sollte am Anfang der STARTUP.CONFIG stehen. Hier
können manuell Befehle eingefügt werden, die durch die
Benutzeroberfläche nicht direkt programmiert werden können. (Was nicht
heißt, daß man den Inhalt der Funktionsliste nicht über die
Oberfläche geändert werden kann). Alle manuell eingefügten Befehle
müssen hier eingefügt werden, da sie an anderer Stelle eingefügt, von
WRITE vergessen, und beim nächsten Abspeichern der Konfiguration
vergessen werden.

Siehe auch =>
```

1.295 WRITE.guide/Quit

```
Quit
====

Aufruf       : Quit Funktionsliste/F

Benötigt     : Nichts

Setzt Fehler : Nichts
```

```
Ergebnisse   : Nichts

Beschreibung : Hier können Funktionen angegeben werden, die ausgeführt
werden, wenn der Editor beendet wird. Hier kann z.B. ein Screenmanager
dazu bewegt werden, einen Screen zu schließen.

Siehe auch =>
```

1.296 WRITE.guide/Fold

```
                Fold
====

Aufruf       : Fold Von/N Bis/N

Benötigt     : Ed

Setzt Fehler : Ja

Ergebnisse   : Nein

Beschreibung : Fold faltet den Text von Zeile Von bis Zeile Bis.
Verschachtelte Falten sind möglich.  Siehe auch =>
                Konstantenbeschreibung
```

1.297 WRITE.guide/UnFold

```
                UnFold
======

Aufruf       : Unfold Von/N Bis/N Ebenen/N

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Entfaltet in dem angegebenen Bereich alle Falten bis zur
angegebenenTiefe.

Siehe auch =>
                Konstantenbeschreibung
```

1.298 WRITE.guide/AutoFold

```
AutoFold
========

Aufruf       : AutoFold

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Such im ganzen Text nach Foldmarkierungen und faltet die
entsprechenen Textteile.

Siehe auch =>
```

1.299 WRITE.guide/ReFold

```
ReFold
======

Aufruf       : ReFold

Benötigt     : Ed

Setzt Fehler : Nein

Ergebnisse   : Nein

Beschreibung : Schaut, ob der Cursor  zwischen zwei Faltmarken steht und
faltet den entsprechenden Block.

Können keine Markierungen gefunden werden, wird ein Fehler
zurückgegeben.

Siehe auch =>

Hier endet die Anleitung...

Viel Spaß...

                                    TIM
```

1.300 WRITE.guide/RexxScripts

```
                RexxScripts
***********

   *     autodoc.wrx Versucht zu dem Wort unter dem Cursor über
```

AmigaGuide bzw. MultiView Informationen zu bekommen.

* ClearUmlauts.wrx Wandelt alle deutschen Umlaute in der Form ä => ae etc. um.

* CloseDown.wrx Schließt alle Fenster nach Sicherheitsabfrage und beendet WRITE.

* define.wrx Ein Script für den SAS C-Compiler. Renumiert #defines (z.B. für Lcale-Files) und sucht zu dem Wort unter dem Cursor modulübergreifend das entsprechenden #define.

* ListeDemo.wrx Demonstriert den Umgang mit den WRITE-internen Funktionen zur Listenverwaltung.

* locale.wrx Diverse Routinen für das Arbeiten mit Locale-Files für den SAS C-Compiler.

* mailer.wrx Startet die Konfiguration Mailer.config. Dieses Script kann z.B. als Editor aus einem NewsReader heraus in der Form rx mailer.wrx gestartet werden. Mailer.config ist schnell, hat wenig Menüs,WordWrap eingeschaltet etc.

* OberonError.wrx Script für den Oberon2-Compiler der Firma A+L. Zeigt die Fehler im Quelltext an.

* open.wrx Läd WRITE, wenn noch nicht gestartet, Öffnet ein Fenster der Standardkonfiguration und läd den übergebenen Text. Es ist nützlich auf diese Funktion einen Alias (z.B. alias ED rx open.wrx) zu legen.

* OpenWrite.wrx Script ähnlich wie open.wrx, nur kann hier die gewünschte Konfiguration mit angegeben werden.

* PrivatePortDemo.wrx Demonstriert den Umgang mit PrivatePorts. Siehe auch
  OpenPort
  ,
  ClosePort
  und
  WaitPort
  .

* Renumber.wrx Renummiert eine eine Reihe untereinanderstehender Zahlen so, daß sie sowohl rechtsbündig als auch aufeinanderfilgend sind.

* ResortIcon.wrx Resortiert iconifizierte Fenster, so daß Lücken zwischen den Fenstern verschwinden.

* sc.wrx Komfortable Fehler-Anzeige für den SAS c_Compiler.

* scman.wrx Zeigt den entsprechende AmigaGuideeintrag zu der SAS-Libraryfunktion unter dem Cursor.

* SetRexxClipDemo.wrx

Demonstriert die Funktion
SetREXXClip
. Mit dieser Funktion ist es
möglich. Zeilen und Buffer beliebiger Länge nach REXX zu
exportieren.

* ShowConfig.wrx Zeigt alle geladenen Konfigurationen.

* templates.wrx Erzeugt automatische Funktionsbeschreibungen /
Autodocsfür / Manpages für den SAS C-Compiler.

* texadr.wrx Script, um Addressen von DFA nach WRITE im TeX-Format
zu importieren.

* View.wrx Ähnlich wie mailer.wrx. Nur wird die die Konfiguration
View.config benutzt. Diese dient zu reinen Darstellung von Texten.
Hier kann z.B. nicht editiert werden.

1.301 WRITE.guide/Index

Index
*****

Adresse
Autor

Allgemeine Syntax
allgemeine Syntax

APPICON
ToolTypes

APPMENU
ToolTypes

APPWIN
ToolTypes

Autor
Autor

Betriebssystem
Installation

Buffer
Listen und Buffer

Copyright
Copyright

Copyright allgemein
Allgemein

Credits
Danksagungen

Danksagungen
Danksagungen

Das Prinzip
Das Prinzip

Demoversion
Demoversion

Der Ed
Der Ed

Einleitung
Einleitung

EMail
Autor

Fehlerreport
Autor

Fred Fish
Demoversion

Funktionen
Funktionsbeschreibung

Geld
Autor

GUIDEMODE
ToolTypes

Haftung
Allgemein

Index
Index

Installation
Installation

Installer
Installation

Kaufkonditionen
Vollversion

KaufPreis
Vollversion

Kommentare
Autor

Konfigurationsfile
Installation

Konstanten
Konstanten

Konstantenbeschreibung
Konstantenbeschreibung

Konzept
Konzept

Kritik
Autor

Libraries
Installation

Listen
Listen und Buffer

MausNet
Autor

Mauszeiger
Editorfenster

Mousepointer
Editorfenster

OS 2.0
Installation

Preis
Vollversion

Public Screens
PublicScreens

Registrierung
Registrierung

Requester
Requester

RexxScripts
RexxScripts

Samples
Sound

Screenmanager
PublicScreens

Scrollbalken
Editorfenster

Scrollgadgets
Editorfenster

Shortcuts
Requester

SLEEPMODE
ToolTypes

Sound
Sound

SOUND
ToolTypes

Speicher
Installation

Stack
Installation

Syntax
Syntax

System
Installation

Tastaturshortcuts
Requester

Titelzeile
Editorfenster

Ton
Sound

ToolTypes
ToolTypes

Upd
Sound

Update
Vollversion

Variablen
Variablen

Variablen
Variablebeschreibung

Vollversion
Vollversion

Warum WRITE ?
Einleitung

Wichtig
Wichtig

Workbench
PublicScreens

1.302 WRITE.guide/Funktionsindex

Funktionsindex
****

About
About

AddList
AddList

Ask
Ask

AutoFold
AutoFold

BackTab
BackTab

Beep
Beep

Begin
Begin

BlockCenter
BlockCenter

BlockLeft
BlockLeft

BlockLftAlig
BlockLftAlig

BlockRghtAlig
BlockRghtAlig

BlockRight
BlockRight

Break
Break

```
BufferToClip
                        BufferToClip

BufferToList
                        BufferToList

BufferToStr
                         BufferToStr

ChangeConfig
                        ChangeConfig

ClearBuffer
                         ClearBuffer

ClearHotKey
                         ClearHotKey

ClearKeys
                           ClearKeys

ClearList
                           ClearList

ClearMenu
                           ClearMenu

CliptoBuffer
                        ClipToBuffer

Close
                               Close

ClosePort
                           ClosePort

Compare
                             Compare

CopyArea
                            CopyArea

CopyBlock
                           CopyBlock

CursorDown
                          CursorDown

CursorLeft
                          CursorLeft

CursorRight
                         CursorRight

CursorUp
                            CursorUp
```

Dec
Dec

Delay
Delay

Delete
Delete

DeleteArea
DeleteArea

DeleteBlock
DeleteBlock

DeleteLine
DeleteLine

DeleteToEOL
DeleteToEOL

DoBuffer
DoBuffer

DoList
DoList

DoREXX
DoREXX

DoString
DoString

DoubleKey
DoubleKey

ExecuteMacro
ExecuteMacro

Exists
Exists

Find
Find

FindListEntry
FindListEntry

FindPattern
FindPattern

Flash
Flash

Fold
Fold

```
Font
                                    Font

GetConfig
                                GetConfig

GetConst
                                 GetConst

GetEnv
                                   GetEnv

GetFile
                                  GetFile

GetFiles
                                 GetFiles

GetFindReplace
                            GetFindReplace

GetFont
                                  GetFont

GetListEntry
                              GetListEntry

GetNumber
                                 GetNumber

GetREXXVar
                               GetREXXVar

GetString
                                 GetString

GetVar
                                    GetVar

GoTextMark
                               GoTextMark

Goto
                                       Goto

GotoMouse
                                 GotoMouse

GPrefs
                                    GPrefs

GuideHelp
                                 GuideHelp

Help
                                       HelpFkt
```

```
Hide
                                        Hide

Iconify
                                    Iconify

If
                                         If

Inc
                                        Inc

InsertBlock
                                 InsertBlock

Item
                                        Item

ItemBar
                                    ItemBar

Key
                                         Key

LastWord
                                   LastWord

ListSize
                                   ListSize

ListToBuffer
                                ListToBuffer

LoadBuffer
                                  LoadBuffer

LockWindow
                                  LockWindow

LowerBlock
                                  LowerBlock

MacroPannel
                                MacroPannel

MacroPlay
                                  MacroPlay

MacroRec
                                   MacroRec

MacroStop
                                  MacroStop

Mark
                                        Mark
```

```
MatchBracket
                        MatchBracket

Menu
                              Menu

Message
                            Message

MessageOK
                         MessageOK

ModifyScreen
                        ModifyScreen

ModifyWin
                         ModifyWin

New
                               New

NewEd
                             NewEd

NextEd
                           NextEd

NextWord
                         NextWord

NOP
                               NOP

NormalPointer
                       NormalPointer

Open
                              Open

OpenPort
                           OpenPort

PageDown
                           PageDown

PageUp
                             PageUp

ParseBuffer
                        ParseBuffer

Pop
                               Pop

Prefs
                             Prefs
```

```
PreparseString
                        PreparseString

Push
                                Push

Quit
                                Quit

QuitEd
                              QuitEd

RangeCheck
                            RangeCheck

RangeRound
                            RangeRound

ReFold
                              ReFold

Refresh
                             Refresh

RemoveList
                            RemoveList

Replace
                             Replace

ReplaceList
                          ReplaceList

Return
                              Return

Save
                                Save

SaveBuffer
                            SaveBuffer

Screen
                              Screen

SetBackUp
                           SetBackUp

SetEnv
                              SetEnv

SetError
                            SetError

SetHotKey
                           SetHotKey
```

```
SetMacro
                                SetMacro

SetMark
                                 SetMark

SetREXXClip
                              SetREXXClip

SetREXXVar
                               SetREXXVar

SetTextMark
                              SetTextMark

SetTitle
                                 SetTitle

SetUserFkt
                               SetUserFkt

SetVar
                                    SetVar

ShowASCII
                                ShowASCII

ShowConstants
                            ShowConstants

ShowFunctions
                            ShowFunctions

ShowIndex
                                ShowIndex

ShowList
                                 ShowList

ShowVars
                                 ShowVars

Silent
                                   Silent

Start
                                    Start

StrToBuffer
                              StrToBuffer

Sub
                                      Sub

SubBar
                                   SubBar
```

System
System

Tab
Tab

UnDelLine
UnDelLine

Undo
Undo

UnFold
UnFold

UnMark
UnMark

UpperBlock
UpperBlock

VersionCheck
VersionCheck

WaitPointer
WaitPointer

WaitPort
WaitPort

WinArranger
WinArranger

Window
Window

WinManager
WinManager

WriteChar
WriteChar

WriteText
WriteText

1.303 WRITE.guide/Variableindex

Variableindex
****

CURSOR

CURSOR

EOL

EOL

EOP

EOP

EOS

EOS

EOT

EOT

EOW

EOW

EQUAL

EQUAL

FALSE

FALSE

HIGHER

HIGHER

LOWER

LOWER

MARKA

MARKA

MARKB

MARKB

MOS

MOS

SOL

SOL

SOP

SOP

SOS

SOS

SOT

SOT

SOW

SOW

TRUE

TRUE

```
_AColor
                                _AColor

_AppIcon
                                _AppIcon

_AppMenu
                                _AppMenu

_AppWindow
                              _AppWindow

_AutoFold
                                _AutoFold

_AutoFree
                                _AutoFree

_AutoIndent
                            _AutoIndent

_BColor
                                _BColor

_CaseSense
                              _CaseSense

_CColor
                                _CColor

_Changed
                                _Changed

_ChipMem
                                _ChipMem

_CollectorMem
                           _CollectorMem

_ConfigPath
                             _ConfigPath

_CurrentChar
                            _CurrentChar

_CurrentConfig
                          _CurrentConfig

_CurrentID
                              _CurrentID

_CurrentLine
                            _CurrentLine

_CurrentWord
                            _CurrentWord
```

```
_Cursor
                                _Cursor

_DColor
                                _DColor

_DefaultConfig
                           _DefaultConfig

_DefaultTool
                             _DefaultTool

_DelFoldMark
                            _DelFoldMark

_EColor
                                _EColor

_EditMode
                              _EditMode

_FastMem
                               _FastMem

_FColor
                                _FColor

_File
                                  _File

_FileName
                              _FileName

_FilePath
                              _FilePath

_FindString
                            _FindString

_Fold
                                  _Fold

_FoldEnd
                               _FoldEnd

_FoldStart
                             _FoldStart

_FRPattern
                              _FRPattern

_Language
                               _Language

_Length
                                _Length
```

```
_LoadTab
                                  _LoadTab
_MarkAx
                                   _MarkAx
_MarkAy
                                   _MarkAy
_MarkBx
                                   _MarkBx
_MarkBy
                                   _MarkBy
_Marked
                                   _Marked
_MaxBuffer
                               _MaxBuffer
_Optimize
                                  _Optimize
_OverwriteIcon
                            _OverwriteIcon
_PatCase
                                  _PatCase
_Path
                                      _Path
_PathPath
                                 _PathPath
_PatNoCase
                               _PatNoCase
_PublicMem
                               _PublicMem
_ReadTabs
                                 _ReadTabs
_REG1
                                      _REG1
_REG2
                                      _REG2
_ReplaceString
                            _ReplaceString
_ReqToMouse
                              _ReqToMouse
```

```
_REXX
                                    _REXX

_REXXPortName
                               _REXXPortName

_RightMargin
                                _RightMargin

_RN
                                      _RN

_RS
                                      _RS

_SaveTab
                                  _SaveTab

_ScreenHeight
                               _ScreenHeight

_Screenname
                                 _Screenname

_ScreenWidth
                                _ScreenWidth

_ScrRelHeight
                               _ScrRelHeight

_ScrRelWidth
                                _ScrRelWidth

_ShowEOL
                                  _ShowEOL

_ShowSpace
                                 _ShowSpace

_SleepMode
                                 _SleepMode

_Sound
                                      _Sound

_TabLength
                                 _TabLength

_Time
                                     _Time

_Undo
                                     _Undo

_User
                                     _User
```

```
_Version
                                _Version

_VersionString
                           _VersionString

_WBStarted
                               _WBStarted

_WinHeight
                               _WinHeight

_WinMode
                                _WinMode

_Wins
                                    _Wins

_WinWidth
                                _WinWidth

_WordDef
                                 _WordDef

_WordOnly
                                _WordOnly

_WordWrap
                                _WordWrap

_WriteIcon
                               _WriteIcon

_WriteTabs
                               _WriteTabs

_XPos
                                    _XPos

_YPos
                                    _YPos
```