

Fiasco
Release 1.1

Nils Bandener
Dekanatsgasse 4
D-34369 Hofgeismar
Germany

27.1.96

Inhaltsverzeichnis

1	Rechtliches	9
1.1	Nutzungsbedingungen	9
1.2	Copyright	9
1.3	Giftware	10
1.4	Datei-Liste	10
2	Einleitung	13
2.1	Features	14
2.2	Neuigkeiten	15
3	Der Beginn	17
3.1	Voraussetzungen	17
3.2	Installation	17
3.3	Quick Start	18
4	Grundlagen einer Datenbank	21
4.1	Records	21
4.2	Felder	21
4.3	Maske	22
4.4	Liste	22
4.5	Maskendehnung	23
4.6	Edit-Modi in Fiasco	24
4.6.1	Record Modus	24
4.6.2	Masken Modus	24
5	Erstellen und Arbeiten mit einer Datenbank	25
5.1	Erstellen der Maske	25
5.2	Erstellen und Arbeiten mit Records	26
5.3	Ändern von Feldtypen	27
5.4	Benutzung von Markierungen	28

6	Suchen in einer Datenbank	29
6.1	Suchmuster	30
6.2	Unscharfe Suche	30
6.3	Suchen mit ARexx	31
6.4	Zählen	31
6.5	Ersetzen	31
6.6	Filter	32
7	Alternative Daten-Mechanismen	33
7.1	Relationen	33
7.1.1	Erstellen von Relationen	34
7.1.2	Technische Anmerkungen	35
7.2	Virtuelle Felder	36
8	Import und Export	39
8.1	Struktur von Import/Export-Dateien	39
8.2	Besondere Zeichen	40
8.3	Importieren von Daten	41
8.4	Exportieren von Daten	43
9	Feldtypen	45
9.1	Standard-Attribute	46
9.2	Feldtypen	46
9.2.1	Der String-Feldtyp	46
9.2.2	Der Integer-Feldtyp	47
9.2.3	Der Float-Feldtyp	48
9.2.4	Der Boolean-Feldtyp	49
9.2.5	Der Cycle-Feldtyp	49
9.2.6	Der Slider-Feldtyp	50
9.2.7	Der Datums-Feldtyp	51
9.2.8	Der Zeit-Feldtyp	52
9.2.9	Der Extern-Feldtyp	53
9.2.10	Der Datatypes-Feldtyp	54
9.2.11	Der Text-Feldtyp	55
9.2.12	Der Button-Feldtyp	56
10	Benutzeroberfläche von Fiasco	57
10.1	Service-Fenster	58
10.1.1	Neu	58
10.1.2	Löschen	59
10.1.3	Erster	59
10.1.4	Voriger	59

10.1.5	Nächster	59
10.1.6	Letzter	59
10.1.7	Aktives Projekt	60
10.1.8	Status	60
10.1.9	Feldtyp	60
10.2	Menus	60
10.2.1	Projekt/Neu (Project/New)	60
10.2.2	Projekt/Leeren (Project/Erase)	60
10.2.3	Projekt/Öffnen... (Project/Open)	61
10.2.4	Projekt/Optionen... (Project/Options)	61
10.2.5	Projekt/Statistik... (Project/Statistic)	61
10.2.6	Projekt/Relationen neuladen (Project/Reload Rels)	62
10.2.7	Projekt/Speichern (Project/Save)	62
10.2.8	Projekt/Speichern als... (Project/Save As)	62
10.2.9	Projekt/Importieren... (Project/Import)	62
10.2.10	Projekt/Exportieren... (Project/Export)	63
10.2.11	Projekt/Über Fiasco... (Project/About)	63
10.2.12	Projekt/Beenden (Project/Quit)	63
10.2.13	Record/Hinzufügen (Record/Add Record)	63
10.2.14	Record/Duplizieren (Record/Duplicate Record) . . .	64
10.2.15	Record/Löschen (Record/Delete Record)	64
10.2.16	Record/Alle löschen (Record/Delete all Records) . .	64
10.2.17	Record/Ausschneiden (Record/Cut Record)	64
10.2.18	Record/Kopieren (Record/Copy Record)	65
10.2.19	Record/Einfügen (Record/Paste Record)	65
10.2.20	Record/Voriger (Record/Previous)	65
10.2.21	Record/Nächster (Record/Next)	65
10.2.22	Record/Erster (Record/First Record)	66
10.2.23	Record/Letzter (Record/Last Record)	66
10.2.24	Record/Gehe zu... (Record/Goto)	66
10.2.25	Record/Markiere Record (Record/Mark Record) . .	66
10.2.26	Record/Lösche Markierung (Record/Unmark Record)	67
10.2.27	Record/Markiere alle Records (Record/Mark all Re- cords)	67
10.2.28	Record/Lösche alle Markierungen (Record/Unmark all Records)	67
10.2.29	Record/Markierungen umschalten (Record/Toggle all Marks)	68
10.2.30	Feld/Feldtyp (Field/Fieldtype)	68
10.2.31	Feld/Feld Hinzufügen... (Field/Add Field)	68
10.2.32	Feld/Feld Ändern... (Field/Edit Field)	69
10.2.33	Feld/Feld Duplizieren (Field/Duplicate Field)	69

10.2.34 Feld/Feld Entfernen (Field/Remove Field)	69
10.2.35 Feld/Relation Ändern... (Field/Edit Relation)	69
10.2.36 Feld/Relation entfernen (Field/Remove Relation) . .	70
10.2.37 Feld/Feld Konvertieren... (Field/Convert Field) . . .	70
10.2.38 Liste/Spalte verbergen (List/Hide column)	70
10.2.39 Liste/Spalte sichtbar machen... (List/Show column)	70
10.2.40 Liste/Alle Spalten sichtbar (List/Show all columns)	71
10.2.41 Liste/Liste neu berechnen (List/Recalc List)	71
10.2.42 Vergleichen/Suchen... (Compare/Find)	71
10.2.43 Vergleichen/Weitersuchen (Compare/Find next) . .	71
10.2.44 Suchen/Rückwärts Suchen (Compare/Find previous)	72
10.2.45 Vergleichen/Ersetzen... (Compare/Replace)	72
10.2.46 Vergleichen/Zählen... (Compare/Count)	72
10.2.47 Vergleichen/Sortieren... (Compare/Sort)	73
10.2.48 Vergleichen/Filter Ändern... (Compare/Edit Filter)	73
10.2.49 Vergleichen/Filter An? (Compare/Use Filter?) . . .	73
10.2.50 Vergleichen/Markieren... (Compare/Mark)	73
10.2.51 Vergleichen/Filter zu Markierungen (Compare/Filter to Marks)	74
10.2.52 Vergleichen/Markierungen zu Filter (Compare/Marks to Filter)	74
10.2.53 Kontrolle/Record-Modus (Control/Record Mode) . .	74
10.2.54 Kontrolle/Masken-Modus (Control/Mask Mode) . .	74
10.2.55 Kontrolle/Service-Fenster (Control/ServiceWindow)	75
10.2.56 Kontrolle/Listen-Fenster (Control/ListWindow) . .	75
10.2.57 Kontrolle/ARexx-Debug (Control/ARexx-Debug) . .	75
10.2.58 Einstellungen/Icons Erzeugen? (Settings/Create Icons?)	75
10.2.59 Einstellungen/Backups Erzeugen? (Settings/Create Backups?)	76
10.2.60 Einstellungen/Relationen schreiben? (Settings/Write Relations?)	76
10.2.61 Einstellungen/Relationen aktualisieren? (Settings/Update Rels?)	76
10.2.62 Einstellungen/Als Pattern * benutzen? (Settings/Use * as Pattern?)	76
10.2.63 Einstellungen/Sicherheits-Requester? (Settings/Security- Reqs?)	77
10.2.64 Einstellungen/Auto-Öffnen ServiceWin? (Settings/Auto- Open ServiceWin?)	77
10.2.65 Einstellungen/ServiceWin dynamisch? (Settings/Dynamic ServiceWin?)	77
10.2.66 Einstellungen/Sprechen? (Settings/Talking?)	77
10.2.67 Einstellungen/Anzeige... (Settings/Display)	77

10.2.68	Einstellungen/Editor... (Settings/Editor)	77
10.2.69	Einstellungen/Einstellungen speichern (Settings/Save Settings)	78
10.2.70	Einstellungen/Einstellungen speichern als... (Settings/Save Settings as)	78
10.2.71	Einstellungen/Einstellungen laden... (Settings/Load Settings)	78
10.2.72	Benutzer/Ändern... (User/Edit)	78
10.3	Alle Requester	78
10.3.1	Feld-Requester	79
10.3.2	Feld umwandeln-Requester	80
10.3.3	Such-Requester	80
10.3.4	Ersetzen-Requester	81
10.3.5	Zähl-Requester	82
10.3.6	Sortiere-Requester	82
10.3.7	Filter-Requester	83
10.3.8	Markieren-Requester	84
10.3.9	Benutzermenu-Requester	84
10.3.10	Optionen-Requester	85
10.3.11	Gehe zu-Requester	85
10.3.12	Relations-Requester	86
10.3.13	Spalte anzeigen-Requester	86
10.3.14	Anzeige Optionen-Requester	87
10.3.15	Importieren-Requester	88
10.3.16	Exportieren-Requester	89
11	ARexx	91
11.1	Allgemeines zu ARexx mit Fiasco	92
11.2	ARexx-Kommandos	93
11.2.1	F_AboutReq	93
11.2.2	F_ActivateField	93
11.2.3	F_AddFieldReq	93
11.2.4	F_AddRecord	94
11.2.5	F_ClearProject	94
11.2.6	F_CloseList	94
11.2.7	F_CloseServiceWin	95
11.2.8	F_ConvertField	95
11.2.9	F_CountRecs	95
11.2.10	F_CountReq	96
11.2.11	F_DupRec	96
11.2.12	F_Export	97
11.2.13	F_FilterReq	97

11.2.14 F_FindFirst	97
11.2.15 F_FindNext	99
11.2.16 F_FindPrev	99
11.2.17 F_FindReq	100
11.2.18 F_GetFieldAttributes	100
11.2.19 F_GetFieldCont	101
11.2.20 F_GetProjFullName	102
11.2.21 F_GetProjName	102
11.2.22 F_GetRecNum	103
11.2.23 F_GotoFirstRec	103
11.2.24 F_GotoNextRec	104
11.2.25 F_GotoLastRec	104
11.2.26 F_GotoPrevRec	104
11.2.27 F_GotoRec	105
11.2.28 F_GotoRecReq	105
11.2.29 F_Import	105
11.2.30 F_IsMarked	106
11.2.31 F_IsVirgin	107
11.2.32 F_LoadDTOObject	107
11.2.33 F_Locate	107
11.2.34 F_LockGUI	108
11.2.35 F_MakeVirgin	109
11.2.36 F_MarkAllRecords	110
11.2.37 F_MarkMatch	110
11.2.38 F_MarkRecord	110
11.2.39 F_NewProject	111
11.2.40 F_OpenList	111
11.2.41 F_OpenProject	112
11.2.42 F_OpenProjectReq	112
11.2.43 F_OpenServiceWin	112
11.2.44 F_OptionsReq	113
11.2.45 F_Progress	113
11.2.46 F_Quit	113
11.2.47 F_RemAllRecords	114
11.2.48 F_RemRecord	115
11.2.49 F_RequestChoice	115
11.2.50 F_RequestFile	116
11.2.51 F_ResetStatus	116
11.2.52 F_SaveProject	117
11.2.53 F_SaveProjectReq	117
11.2.54 F_SaveSettings	117
11.2.55 F_SelectProj	118

11.2.56 F_SetFieldCont	118
11.2.57 F_SetMode	119
11.2.58 F_SetSearchField	119
11.2.59 F_SetSearchPat	120
11.2.60 F_SetStatus	120
11.2.61 F_Sort	121
11.2.62 F_SortReq	121
11.2.63 F_ToggleAllMarks	121
11.2.64 F_UnlockGUI	122
11.2.65 F_UnmarkAllRecords	122
11.2.66 F_UnmarkRecord	122
11.2.67 F_UserCommand	123
11.2.68 F_VirtualMode	123
12 Beispiel-Projekte	125
12.1 Adressbuch	125
12.2 Datatypes Demo	125
12.3 Stammbaum	126
12.4 Videos	126
12.5 Bilder-Datenbank	127
12.6 FAQs-Datenbank	127
A Alle Suchmuster	131
B Relations-Checkliste	133
C Implementation der Clipboard-Unterstützung	135
D Bugs	137
E Was es noch zu tun gibt	139

Kapitel 1

Rechtliches

1.1 Nutzungsbedingungen

Das Programm “Fiasco” und zugehörige Daten werden “wie sie sind” zur Verfügung gestellt. Der Autor übernimmt keinerlei Gewährleistung für Fehlerfreiheit, Genauigkeit, Verlässlichkeit oder Freiheit von sonstigen Mängeln der oben genannten Programme und Daten, weder ausdrücklich, noch stillschweigend. In keinem Falle kann ich für irgendwelche Schäden, die von diesem Programm verursacht wurden verantwortlich gemacht werden.

1.2 Copyright

Fiasco ist *nicht* Public Domain. Ich behalte mir jegliche Urheberrechte vor. Fiasco Copyright © 1995-1996 Nils Bandener.

Fiasco darf frei verbreitet werden, solange folgende Bedingungen erfüllt sind:

- Das Programm-Paket muß komplett sein. In der Datei-Liste sind alle Dateien aufgelistet, aus denen Fiasco 1.1 besteht.
- Fiasco darf ohne *schriftliche* Genehmigung des Autors nicht für kommerzielle Zwecke verkauft werden. Dies schließt den Vertrieb von Fiasco zu überhöhten Preisen ein. Es dürfen lediglich Kosten für Medien und Kopieren erhoben werden. Der Vertrieb auf CD-Roms ist gestattet, solange der Preis nicht den Preis der “Fresh Fish”-CD-Rom von Fred Fish nicht übersteigt.

1.3 Giftware

Fiasco ist Giftware, das bedeutet, daß jeder Benutzer von Fiasco mir eine kleine Anerkennung für den Programmieraufwand zukommen lassen sollte. Dies kann Geld, irgendein kleines Geschenk (CD-Roms, Bücher, etc.) oder auch nur eine Postkarte sein (Oder auch nichts, wenn man meint, sein Geld für Besseres gebrauchen zu können; bitte beachten Sie aber, daß ln_Pri davon -110 beträgt ;-).

Meine Adresse:

Nils Bandener
Dekanatsgasse 4
D-34369 Hofgeismar
Germany

1.4 Datei-Liste

Fiasco Release 1.1 besteht aus diesen Dateien:

```
Fiasco_1.1.info
Fiasco_1.1/ARexx.info
Fiasco_1.1/ARexx/age.rexx
Fiasco_1.1/ARexx/age.rexx.info
Fiasco_1.1/ARexx/print.rexx
Fiasco_1.1/ARexx/print.rexx.info
Fiasco_1.1/ARexx/unlockgui.rexx
Fiasco_1.1/ARexx/unlockgui.rexx.info
Fiasco_1.1/Catalogs/Deutsch/fiasco.catalog
Fiasco_1.1/Databases.info
Fiasco_1.1/Databases/Addresses.info
Fiasco_1.1/Databases/Addresses/Addresses.fdb
Fiasco_1.1/Databases/Addresses/Addresses.fdb.info
Fiasco_1.1/Databases/Addresses/Countries.fdb
Fiasco_1.1/Databases/Addresses/Countries.fdb.info
Fiasco_1.1/Databases/DatatypesDemo.info
Fiasco_1.1/Databases/DatatypesDemo/AmigaWorld.ilbm
Fiasco_1.1/Databases/DatatypesDemo/AmigaWorld.ilbm.info
Fiasco_1.1/Databases/DatatypesDemo/DatatypesDemo.fdb
Fiasco_1.1/Databases/DatatypesDemo/DatatypesDemo.fdb.info
Fiasco_1.1/Databases/DatatypesDemo/Hallelujah.8svx
Fiasco_1.1/Databases/FamilyTree.info
```

Fiasco_1.1/Databases/FamilyTree/families.fdb
Fiasco_1.1/Databases/FamilyTree/families.fdb.info
Fiasco_1.1/Databases/FamilyTree/persons.fdb
Fiasco_1.1/Databases/FamilyTree/persons.fdb.info
Fiasco_1.1/Databases/FAQs.info
Fiasco_1.1/Databases/FAQs/FAQS.fdb
Fiasco_1.1/Databases/FAQs/FAQS.fdb.info
Fiasco_1.1/Databases/FAQs/RunMost.rexx
Fiasco_1.1/Databases/FAQs/scantxt.dir.rexx
Fiasco_1.1/Databases/FAQs/searchfaqs.rexx
Fiasco_1.1/Databases/FAQs/showtxt.rexx
Fiasco_1.1/Databases/PD-Disks.info
Fiasco_1.1/Databases/PD-Disks/Disks.fdb
Fiasco_1.1/Databases/PD-Disks/Disks.fdb.info
Fiasco_1.1/Databases/PD-Disks/ReadFish.rexx
Fiasco_1.1/Databases/PD-Disks/ReadFish.rexx.info
Fiasco_1.1/Databases/PictureDatabase.info
Fiasco_1.1/Databases/PictureDatabase/Pictures.fdb
Fiasco_1.1/Databases/PictureDatabase/Pictures.fdb.info
Fiasco_1.1/Databases/PictureDatabase/scandir.rexx
Fiasco_1.1/Databases/PictureDatabase/showscr.rexx
Fiasco_1.1/Databases/Videos.info
Fiasco_1.1/Databases/Videos/CalcLen.rexx
Fiasco_1.1/Databases/Videos/Movies.fdb
Fiasco_1.1/Databases/Videos/Movies.fdb.info
Fiasco_1.1/Databases/Videos/Tapes.fdb
Fiasco_1.1/Databases/Videos/Tapes.fdb.info
Fiasco_1.1/Development.info
Fiasco_1.1/Development/fiasco.cd
Fiasco_1.1/Development/fiasco.cd.info
Fiasco_1.1/Development/fiasco.ct
Fiasco_1.1/Development/fiasco.ct.info
Fiasco_1.1/Documentation.info
Fiasco_1.1/Documentation/Deutsch.info
Fiasco_1.1/Documentation/Deutsch/fiasco.dvi
Fiasco_1.1/Documentation/Deutsch/Fiasco.dvi.info
Fiasco_1.1/Documentation/Deutsch/fiasco.guide
Fiasco_1.1/Documentation/Deutsch/Fiasco.guide.info
Fiasco_1.1/Documentation/English.info
Fiasco_1.1/Documentation/English/Fiasco.dvi
Fiasco_1.1/Documentation/English/Fiasco.dvi.info
Fiasco_1.1/Documentation/English/Fiasco.guide

Fiasco_1.1/Documentation/English/Fiasco.guide.info
Fiasco_1.1/Fiasco
Fiasco_1.1/Fiasco.info
Fiasco_1.1/gtlayout.library
Fiasco_1.1/icons/ARexx.info
Fiasco_1.1/icons/ARexxScript.info
Fiasco_1.1/icons/Databases.info
Fiasco_1.1/icons/Documentation.info
Fiasco_1.1/icons/Drawer.info
Fiasco_1.1/icons/Fiasco.dvi.info
Fiasco_1.1/icons/Fiasco.guide.info
Fiasco_1.1/icons/Fiasco.info
Fiasco_1.1/icons/FiascoProject.info
Fiasco_1.1/icons/XPort.info
Fiasco_1.1/icons/XPortData.info
Fiasco_1.1/Install.info
Fiasco_1.1/Install/Deutsch.info
Fiasco_1.1/Install/English.info
Fiasco_1.1/Install/Install
Fiasco_1.1/Libs/MC68020.info
Fiasco_1.1/Libs/MC68020/gtlayout.library
Fiasco_1.1/XPort.info
Fiasco_1.1/XPort/mpearls_II_findpeals.fxp
Fiasco_1.1/XPort/mpearls_II_findpeals.fxp.info
Fiasco_1.1/XPort/RFF.fxp
Fiasco_1.1/XPort/RFF.fxp.info
Fiasco_1.1/XPort/StdTwist.fxp
Fiasco_1.1/XPort/StdTwist.fxp.info

Kapitel 2

Einleitung

Fiasco ist ein neuer Kandidat unter der Masse von Datenbanken für den Amiga. Ursprünglich wollte ich nur ein kleines Programm schreiben, das einem ein paar Vokabeln abhört, die man vorher eingegeben hat. Später kam dann noch die Möglichkeit hinzu, mehr als nur zwei Felder, also Frage und Antwort, zu erstellen. Ab einem gewissen Entwicklungsstadium war dieses Programm einer Datenbank schon so nahe, daß es nur ein paar Änderungen bedurfte, um daraus schon eine Datenbank zu machen. Seitdem (Januar 95) tüftle ich an diesem Meisterstück der Programmierkunst. ;-)

In den Grundlagen unterscheidet sich Fiasco kaum von anderen Datenbanken. Zwar unterstützt Fiasco keine hierarchischen Datenbankstrukturen (wie z.B. AmigaBase), hat dafür aber einen kleinen Teil für Relationen ¹. Außerdem kann Fiasco über ARexx gesteuert werden. So können außerdem Programme einzelnen Feldern zugeordnet werden.

Die Maske von Fiasco wird nicht durch eine Grafikdatei definiert, sondern mit internen Grafiken erstellt. So können Fiasco-Masken mit jedem nicht-proportionalen Font benutzt werden. Die Feldtypen, die in einer Maske benutzt werden können, sind breit gesät. Am besten gefällt mir der Datatypes-Feldtyp, mit dem man Grafiken, Animationen, Texte usw. direkt in die Maske einbauen kann.

Neben der Maske kann man die Daten in der Liste anzeigen. Diese ist wie die Maske frei konfigurierbar. Die Daten können in der Liste leider noch nicht verändert werden.

Das Suchsystem von Fiasco unterstützt unscharfe Suche und Platzhalter. Unscharfe Suche bedeutet, daß nicht nach Gleichheit, sondern nach

¹Ich habe bisher nur von relationalen Datenbanken gelesen, so kann ich nicht 100%ig garantieren, daß ich das Prinzip richtig implementiert habe. Auf jeden Fall macht Fiasco sowas ähnliches :-)

Ähnlichkeit getestet wird. Die Toleranzschwelle, ab der Fiasco einen Eintrag als ähnlich ansieht, kann frei eingestellt werden. Die Platzhalter ähneln den AmigaDOS- Pattern-Matching-Symbolen; zur Zeit werden `#?` (eine unbestimmte Zahl unbekannter Zeichen) und `?` (ein unbekanntes Zeichen) unterstützt.

Außerdem stehen Sortier-, Filter-, und Zählfunktionen zur Verfügung, die allesamt mit dem Suchsystem verwandt sind und so - kennt man einmal das Suchsystem - keine größeren Probleme darstellen sollten.

2.1 Features

Fiasco kann das alles tun:

- Mehrere Projekte können gleichzeitig im Speicher gehalten werden. Die Anzahl ist nur durch den freien Systemspeicher begrenzt
- Masken verhalten sich genau so wie andere GUIs
- Masken, Listen und Requester sind voll Fontsensitiv.
- Wahre Feldtypen-Vielfalt. Neben String, Integer und Float gibt es noch Cycle, Boolean, Slider, Datum, Zeit, Extern und Datatypes.
- Unterstützung von Datatypes-Feldern. Somit können auch Bilder, Texte usw. direkt in die Maske integriert werden.
- ARexx-Port, mit dem das Programm gesteuert oder erweitert werden kann. Der ARexx-Port wird auch benutzt, um einzelnen Feldern Programme zu- zuordnen.
- Frei konfigurierbares “User-Menu”, von dem CLI-Programme und ARexx- Scripts gestartet werden können.
- Die Suchfunktion erlaubt sowohl Suchmuster als auch “unscharfe Suche”
- Sehr flexibles Listen-Fenster, in dem Spalten versteckt und in der Position und Größe verändert werden können.
- Einfaches Relation-Handling

2.2 Neuigkeiten

Neue Features in Fiasco 1.1:

- Der Typ eines Feldes kann umgewandelt werden. Ein String Feld kann beispielsweise in ein Cycle Feld umgewandelt werden, dessen Labels den alten Strings entsprechen.
- Import/Export von Daten.
- Read Only-Feld Attribut.
- Virtuelle Felder
- Fiasco kann einen eigenen Bildschirm öffnen.
- Fiasco entdeckt sich selbst im System und startet nicht zweimal.
- Niedrigere Speicher-Fragmentierung durch Benutzung von Pools.
- Records können markiert werden.
- Extern- und Datatypesfelder können nun mit Filerequestern verändert werden.
- Datatypesfelder unterstützen “Speichern” und “Laden verzögern”
- Datatypesfelder zeigen Fehlermeldungen in sich selbst an
- Änderungen in der Reihenfolge von Labeln eines Cyclefelds ändern nicht mehr die echten Daten.
- Button Felder
- Relationen sich *viel* schneller!
- Die Requester werden mit der gtlayout.library von Olaf Barthel erzeugt.
- Alte ARexx Kommandos erweitert und neue hinzugefügt.
- Fiasco kann mit Ihnen sprechen!
- Nach Enter aktiviert Fiasco das nächste String-Feld
- Wenn Amiga OS 3.0 verfügbar ist, vergrößert Fiasco die Größe der Filehandle-Buffer, wenn ein Projekt gelesen oder geschrieben wird.
- Das ARexx-Interface liest die Argumente mit `ReadArgs()` der dos.library.

- ARexx-Debug zeigt mehr Informationen an und hat eine Hilfe-Option.
- Anleitung im DVI-Format für gedruckte Handbücher
- Ein Editor kann im Feldrequester aufgerufen werden, um ARexx-Scripts zu ändern.
- Verbessertes Pattern-Matching
- Unterstützung des Clipboards

Behobene Fehler in Fiasco 1.1:

- Usermenu verweigerte es, Einträge in der Liste zu verschieben.
- Schloss workbench.library nicht
- Liste aktualisierte die Hoch/Runter-Scroll-Leiste nicht, wenn Records hinzugefügt wurden.
- Wertete die Font-Informationen für die Maske im Settings-File nicht aus.
- F_GetFieldCont und F_SetFieldCont konnten nicht mit Float, Date, Time, Extern und Datatypes Feldern benutzt werden.
- Löschte Grafik von Feldern manchmal nicht korrekt.
- Stürzte ab, wenn man Masken-Modus aktivierte, während der Modus bereits aktiv war.
- Wenn man im Masken-Modus ein Projekt öffnete, wurde das Feld, das unter dem Cursor lag, nicht aktiviert.
- Wenn man ein Projekt im Masken-Modus aktivierte, wurde das Service-Fenster nicht aktualisiert.
- Der Optionen-Requester aktualisierte das Service-Fenster nicht.

Kapitel 3

Der Beginn

3.1 Voraussetzungen

Die minimalen Anforderungen von Fiasco sind ein Amiga mit OS 2.04 (37.175) und 1 MB RAM. Die optimale Konfiguration sind ein Amiga mit OS 3.x (39.x oder höher), ein 68020 Prozessor, 2 MB RAM und eine Festplatte.

Features und benötigte OS-Versionen:

Lokalisation: Amiga OS 2.1 (38.x)

Screenmode-Requester: Amiga OS 2.1 (38.x)

Online-Hilfe: Amiga OS 3.0 (39.x) oder amigaguide.library v34 aus FD

Datatypes-Felder: Amiga OS 3.0 (39.x)

Schnelles Projektladen: Amiga OS 3.0 (39.x)

Fiasco 1.1 benutzt die gtlayout.library von Olaf Barthel für die GUI. Die Library ist im Archiv von Fiasco enthalten.

Die Memory-Pool Funktionen von Amiga OS 3.0 und Amiga OS 3.1 geben nicht mehr benutzte Puddles nicht frei, bis der Pool gelöscht wird. Benutzen Sie SetPatch 40.16 (bereits bei WB 40.42), um dies zu beheben. Wenn Sie Amiga OS 2.0 oder Amiga OS 2.1 benutzen, brauchen Sie sich nicht darum zu kümmern.

3.2 Installation

Wenn Sie den Commodore Installer haben, klicken Sie einfach zweimal auf das Installer Icon Ihrer Lieblingssprache im Verzeichnis "Install". Sie wer-

den dann instruiert, was sie zu tun haben.

Falls Sie den Commodore Installer nicht besitzen, brauchen sie das “Fiasco” Verzeichnis nur irgendwo hin kopieren. Falls Sie wollen, können Sie die Catalogs nach locale:catalogs kopieren, was aber nicht notwendig ist. Weiterhin können Sie die unbenutzten Sprachen in “Documentation” löschen und die verbleibenden Dateien in die übergeordneten Verzeichnisse schieben. Die Dateien in “Development” und “Install” werden für den normalen Gebrauch nicht benötigt. Mit dieser Konfiguration kann Fiasco laufen. Falls sie einen 68020 besitzen, können sie zusätzlich die gtlayout.library im Hauptverzeichnis von Fiasco durch die gtlayout.library aus dem Verzeichnis libs/68020 ersetzen. Um die Library anderen Programmen zugänglich zu machen, muß sie ins Verzeichnis Libs: kopiert werden.

3.3 Quick Start

Dies sind die wichtigsten Dinge, die man wissen muß, um mit Fiasco zu arbeiten:

- Das Programm startet man am besten über das Programm- oder ein Projekt-Icon.
- Es gibt zwei Arbeits-Modi in Fiasco, einen, in dem man die Records verändern kann, und einen anderen, in dem man die Maske verändern kann. Man steuert sie über **Kontrolle/RecordModus** bzw. **Kontrolle/MaskenModus**.
- Das Service-Fenster erleichtert das Arbeiten mit Fiasco, wenn man sich noch nicht mit Shortcuts u.ä. auskennt. Man öffnet es über **Kontrolle/ServiceFenster**. *Achtung:* Die Funktionen der Gadgets variieren in den Modi.
- Die Liste, die man über **Kontrolle/ListFenster** erreicht, kann verändert werden, indem man in die Titeln der Liste klickt. Klickt man einen Titel einmal an, so wird er aktiviert. Über das Menu **List** kann man nun diverse Sachen mit der Spalte machen. Wenn man am rechten Rand eines Titels klickt, und die Maustaste gedrückt hält, kann man die Breite einer Spalte variieren. Falls man in den restlichen Raum des Titels klickt, und die Maustaste gedrückt hält, kann man die Position der Spalte verändern.
- Bestimmte Einstellungen zum Projekt, z.B. zum Masken-Layout kann man über den Menüpunkt **Projekt/Optionen** einstellen.

- Bei sonstigen Problemen kann man einfach das Menu anwählen und bevor man die rechte Maustaste losläßt, die Help-Taste drücken.

Kapitel 4

Grundlagen einer Datenbank

Eine Datenbank kann mit einer einfachen Kartei verglichen werden. Die meisten Implementationen von Datenbanken benutzen dieses Schema als Grundlage.

In Fiasco besteht ein Datenbank-Projekt aus zwei Komponenten: Auf der einen Seite die Daten, die durch Records aufgeteilt werden. Auf der anderen Seite die Maske, die die Struktur der Daten definiert.

Im Folgenden werden die grundlegenden Allgemeinen und die Fiasco-Spezifischen Prinzipien von Datenbanken beschrieben.

4.1 Records

Records stellen in einer Datenbank die Karteikarten einer Kartei dar. Ein Record ist also eine Sammlung verschiedener Aspekte zu einem Oberbegriff; z.B. zu einer Person Name, Adresse usw. In der Maske wird immer nur ein Record auf einmal angezeigt, In der Liste werden Records als Zeilen dargestellt.

4.2 Felder

Felder legen fest, was für Daten gespeichert werden. In Fiasco werden Felder in der Maske definiert. Felder sind die grundlegenden Elemente der Maske und der Liste.

Fiasco unterstützt mehrere Feldtypen. Mehr Informationen über Feldtypen und deren Features gibt es im Kapitel Feldtypen.

4.3 Maske

Die Maske ist die Darstellungsform, mit der Fiasco am meisten arbeitet. In einer Maske kann im Gegensatz zur Liste nur ein Record gleichzeitig angezeigt werden. Dafür ist diese Anzeige übersichtlicher und geordneter (falls man es richtig gemacht hat ;-), weshalb diese Darstellungsform auch manchmal als “Formular” bezeichnet wird.

Die Maske setzt sich aus sogenannten Feldern zusammen, die verschiedene Typen und somit verschiedene Erscheinungsformen haben.

Bei normalen Amiga-Programmen würde man die “Felder” als Gadgets bezeichnen und de facto sind sie sogar welche. Fiasco benutzt für die Maske intensiv Gadgets der `gadtools.library`, weshalb man sich schnell an die Fiasco-Masken gewöhnen können sollte.

Fiasco-Masken passen sich automatisch an einen beliebigen Zeichensatz an, einzige Voraussetzung ist, daß er nicht proportional ist (z.B. `topaz`, `courier`).

Um eine Maske in Fiasco zu gestalten, muß man über den Menüpunkt `Kontrolle/MaskenModus` in den Masken Modus überwechseln. Nun kann man mit der Maus existierende Felder verschieben oder über das `Felder` Menu verschiedene andere Manipulationen ausführen. Mehr dazu kann im Kapitel 5.1 gefunden werden.

4.4 Liste

Über den Menüpunkt `Kontrolle/ListWindow` kann man ein Fenster mit einer Listendarstellung des aktuellen Projektes öffnen. In der Liste wird ein Record durch eine Zeile repräsentiert. Die Felder des Records sind durch Spalten gekennzeichnet. Die erste Zeile der Liste beinhaltet die IDs der jeweiligen Felder. Falls das Fenster nicht groß genug ist, um die gesamte Liste anzuzeigen, kann man die Scroll-Balken im rechten und unteren Fenster- rand benutzen, um sich durch die Liste hindurchzuarbeiten. Die Zeile des aktuellen Records wird durch eine Unterlegung deutlich gemacht.

Sie können auch mit der Liste Records auswählen. Sie müssen lediglich den jeweiligen Record anklicken und schon wird er aktiviert. Änderungen im Record können Sie jedoch nur in der Maske vornehmen.

Falls ein Filter (siehe Abschnitt 6.6) aktiv ist, werden nur noch die Einträge angezeigt, die mit dem Filter-Muster übereinstimmen.

Normalerweise werden für das Layout der Liste Position und Dimensionen der jeweiligen Masken-Felder für die Liste umgesetzt. Sie haben jedoch auch die Möglichkeit, hier einzugreifen. Die Breite der Spalten können Sie verändern, indem Sie in der Titel-Zeile, in der die Ids angezeigt werden, an der rechten Kante die linke Maustaste zu drücken. Nun erscheint eine Linie an der Mausposition, die Sie nun mit der Maus nach links oder rechts ziehen können. Dort, wo Sie sie fallen lassen, d.h. die linke Maustaste loslassen, befindet sich künftig der rechte Rand des Feldes. Falls Felder hiervon überschritten würden, werden Sie verschoben.

Die Position einer Spalten kann man auch verändern. Dazu klickt man über der Mitte des Spalten-Kopfes und hält die Maustaste gedrückt. Nun kann man die Spalte in der Liste an eine beliebige Stelle verschieben. Läßt man die Maustaste wieder los, so wird die Spalte bestmöglich zwischen den benachbarten Feldern plaziert.

Will man Spalten ganz verschwinden lassen, so macht man das, indem man mit der linken Maustaste den Kopf einer Spalte aktiviert. Nun kann man über den Menüpunkt **Liste/Spalte verbergen** diese Spalte aus der Liste verschwinden lassen. Will man sie wieder haben, muß man **Liste/Spalte anzeigen** auswählen. Daraufhin erscheint ein Requester mit allen Spalten, die über **Hide column** entfernt wurden. Wenn man die gewünschte ausgewählt hat, wird sie möglichst nahe an der alten Position wieder eingefügt.

Der Menüpunkt **Liste/Liste neu berechnen** bewirkt, daß der Aufbau der Liste völlig neu berechnet wird. Man kann ihn mit dem Menüpunkt **Inhalt aufräumen** (Clean up) der Workbench vergleichen. Verborgene Spalten werden jedoch nicht sichtbar gemacht.

4.5 Maskendehnung

Normalerweise sitzen die Felder in einer Fiasco-Maske sich dicht auf der Pelle. Dies ist a) nicht schön anzusehen, b) haben eigentliche alle anderen "normalen" GUIs ein paar Pixel zwischen den Gadgets Platz. Man könnte zwar jeweils eine Zeile zwischen den Feldern frei lassen, dies würde jedoch bald zu Platzproblemen führen. Deswegen gibt Fiasco die Möglichkeit, zwischen den einzelnen Zeilen und Spalten pixelweise Raum zu lassen.

Diese Werte können im Options-Requester in den Gadgets **Stretch X / Y** angegeben werden.

Ein Problem bei dieser Methode die Maske zu "dehnen" ist, daß auch die Felder größer werden als in den Feld-Requestern angegeben wurde. Dies fällt insbesondere in der horizontalen Richtung auf, da die meisten Fiasco-Felder sich nur in diese Richtung ausdehnen. Bei String-Feldern ist es so z.B. möglich, daß der Text, der in eingegeben werden kann nur bis z.T. das

Feld füllen kann, da **Max Chars** geringer als der wahre **Width**-Wert ist. Das größte Problem ergibt sich bei Text-Feldern, die normalerweise nur so breit sind, wie der Text es erfordert. Nun sind sie aber breiter und der Text muß zentriert werden.

Um diese Probleme zu vermeiden, empfehle ich, daß als **X**-Wert Null angegeben wird und in dieser Richtung weiterhin einzelne Spalten als Trennung benutzt werden. In **Y**-Richtung liefert der 4 die besten Ergebnisse.

4.6 Edit-Modi in Fiasco

Fiasco unterteilt seine Arbeit in Modi. Wenn Sie Änderungen in der Maske vornehmen wollen, muß der Masken-Modus aktiv sein. Falls Sie Änderungen in den Record-Daten vornehmen wollen, der Record-Modus.

4.6.1 Record Modus

In diesem Modus kann man Records hinzufügen, löschen und deren Inhalt verändern. Dieser Modus kann über den Menüpunkt **Kontrolle/Record-Modus** aktiviert werden. Wenn der Record Modus aktiv ist, ist das Feld-Typ-Cycle-Gadget im Service-Fenster nicht anwählbar und das Status-Gadget zeigt normalerweise die Nummer des aktiven Records und die Anzahl der Records an (z.B.: 78 / 92).

4.6.2 Masken Modus

In diesem Modus hat man die Möglichkeit, die Maske zu verändern, d.h. Felder neu zu erstellen, zu löschen und zu verschieben. Außerdem können hier Relationen erstellt und verändert werden. Dieser Modus kann über den Menüpunkt **Kontrolle/Masken-Modus** aktiviert werden. Wenn der Masken-Modus aktiv ist, sind die "Tapedeck"-Gadgets im Service-Fenster verschlossen und das Status-Gadget zeigt normalerweise die Koordinaten des Cursors in der Maske an (z.B. X: 10, Y: 5).

Kapitel 5

Erstellen und Arbeiten mit einer Datenbank

Und nun zur Praxis: Wenn Sie eine Datenbank in Fiasco erstellen möchten, müssen Sie zuerst die Maske und dann die Records erstellen. Fiasco erlaubt es Ihnen, in den meisten Fällen die Datenbank auf intuitive Weise zu erstellen.

Im Folgenden wird die Erstellung einer einfachen Datenbank beschrieben.

5.1 Erstellen der Maske

Um eine Maske zu erstellen muß man zuerst in den Masken-Modus, indem man den Menüpunkt **Kontrolle/Masken-Modus** aktiviert. Nun erscheint in der Maske ein Cursor, mit dem man die Position des Feldes festlegen kann, das man erstellen will. Den Typ des Feldes, das erzeugt werden soll, kann man über das Menu **Felder/Typ** festlegen. Die verschiedenen Feldtypen werden im Kapitel **Feldtypen** erklärt. Weiterhin läßt der aktuelle Feldtyp sich noch über das **Service-Fenster** und per **Tastaturshortcuts** einstellen (siehe **Menu**)

Nach diesen Einstellungen kann man mit dem Menüpunkt **Feld/Feld Hinzufügen** ein neues Feld erstellen. Zuerst öffnet sich der **Feldrequester**. Die einzelnen Gadgets im **Feldrequester** sind vom **Feldtyp** abhängig und bei den jeweiligen **Typdokumentationen** (siehe **Abschnitt 9**) erklärt. Es reicht übrigens nicht, nur **Ok** anzuklicken. Bestimmte Attribute, wie z.B. die **ID** müssen vom Benutzer immer neu angegeben werden. Sollten alle Attribute gültig sein, schließt sich der **Requester** und das jeweilige Feld erscheint in

der Maske.

Nachträglich können alle Attribute des Feldes bis auf den Feldtypen an sich (dazu mehr später) geändert werden. Die Position läßt sich einfach mit der Maus verändern. Wenn man ein Feld auswählt und die Maustaste gedrückt hält erscheint einmal ein gefülltes Rechteck, das mit der Maus mitgeht und ein anderes, das die genaue neue Position zeigt. Läßt man die linke Maustaste los, wird das Feld an der neuen Stelle abgelegt, soweit keine anderen Felder im Weg sind.

Der Feldrequester läßt sich auch wieder aufrufen, indem man entweder auf ein Feld doppelt klickt, oder ein Feld aktiviert und den Menüpunkt **Felder/Feld ändern** anwählt. Hier lassen sich wiederum alle Parameter frei verändern. Der ID-Wert sollte jedoch nur mit Vorsicht verändert werden, da andere Fiasco-Projekte oder AREXX-Scripts unter Umständen noch die alten Werte eingestellt haben und die Felder nicht mehr finden. Falls man die maximale Zeichenanzahl von String, Extern oder Datatypes-Feldern verkleinert, wird zuerst geprüft, ob Daten eventuell verloren gehen würden und eine Warnung ausgegeben.

Über den Menüpunkt **Felder/Feld entfernen** lassen sich Felder wieder entfernen. *Vorsicht:* Falls die Option **Sicherheits-Requester** nicht aktiviert ist, werden alle in diesem Feld gespeicherten Daten sofort aus dem RAM gelöscht. Beim nächsten Abspeichern werden auch die auf der Diskette verbliebenen Daten dieses Feldes vernichtet.

Im Options-Requester (siehe Abschnitt 10.3.10) lassen sich weitere Parameter für das aktuelle Projekt, wie Maskendehnung (siehe Abschnitt 4.5), Name des Autors usw. angeben.

Der Menüpunkt **Felder/Relationen ändern** arbeitet ähnlich wie **Feld ändern**. Hier kann man jedoch Parameter für Relationen einstellen. Kapitel 7.1 enthält weitere Informationen über Relationen.

Ist die Maske vollständig kann man wieder in den Record-Modus zurückgehen (Kontrolle/Record-Modus) und die eigentlichen Daten erstellen.

5.2 Erstellen und Arbeiten mit Records

Wenn Sie eine Maske mit ein paar Feldern haben, ist die Zeit gekommen, Records zu erstellen, um Daten zu speichern. Die einfachste Methode, einen Record zu erstellen, ist **Record/Hinzufügen** oder das Äquivalent **Neu** im Service-Fenster. Diese Prozedur erzeugt — wie schon der Name sagt — einen Record und aktiviert diesen. Die Felder im Record werden die Werte enthalten, die im Masken-Modus definiert wurden. Mit der Maus können Sie nun ein Feld aktivieren und den Inhalt verändern.

Ein anderer Weg, Records zu erstellen ist **Record/Duplizieren**. Diese

Funktion erzeugt einen Record, der eine exakte Kopie des vorher aktiven Records ist. Alle **Startwert**-Attribute werden ignoriert!

Wenn Sie einen Record nicht mehr brauchen, können Sie ihn mit **Record/Entfernen** oder **Löschen** im Service-Fenster aus dem Projekt entfernen. Wenn währenddessen **Einstellungen/Sicherheits-Abfragen** aktiv ist, werden Sie noch einmal gewarnt, bevor der Record tatsächlich gelöscht wird.

Wenn Sie durch die erzeugten Records “hindurchblättern” wollen, können Sie die Menüs, das Service-Fenster, die Cursor-Tasten oder das Listen-Fenster benutzen¹. Weil ich denke, daß die GUI-Teile intuitiv benutzt werden können, erkläre ich nur die Cursor-Tasten. Fangen wir mit der “Hoch”-Taste an. Bei Betätigung wird der dem aktuellen Record vorausgehende Record aktiviert. Die “Runter”-Taste aktiviert den nächsten Record. Diese Einteilung entspricht der Struktur des Listen-Fensters². Diese Tasten kombiniert mit der Ctrl-Taste aktivieren die jeweiligen Extrema, also den ersten, bzw. den letzten Record.

5.3 Ändern von Feldtypen

Wenn Sie ein Projekt erstellt haben, kann es nützlich werden, wenn man den Typ eines Felds ändern kann. Diese Situation kann dadurch entstehen, daß manche Inhalte eines Feldes sich in eine andere Richtung als ursprünglich erwartet entwickelt haben. Die Konvertierungs-Funktion kann auch beim Importieren von Dateien nützlich werden. Die Import-Funktion erzeugt nämlich ausschließlich String-Felder, während manche Daten jedoch auch in anderen Feldtypen besser aufgehoben wären.

Um den Konvertierungs-Requester zu öffnen, muß der Masken-Modus aktiv sein. Wenn dies so ist, aktivieren Sie das Feld, das Sie umwandeln möchten und wählen **Feld/Feld Konvertieren** auf. Der Konvertierungs-Requester zeigt die ID des Feldes, den jetzigen Typ und die Typen, in die das Feld umgewandelt werden kann, an. Wenn Sie **Alternatives Formataktivieren**, wandelt die Funktion die Daten in einer anderen, normalerweise abstrakteren Form um. Dies wird nicht von allen Feldtypen unterstützt. Wenn Sie den neuen Typ ausgewählt haben und mit **Ok** weitergehen, wird der Umwandlungs-Vorgang gestartet.

Bitte beachten Sie, daß Fiasco nicht vor irgendwelchem Datenverlust, der durch die Umwandlung entstehen kann (z.B. String zu Integer) warnt. Wenn der neue Feldtyp zusätzliche Attribute benötigt (Der Extern-Typ benötigt beispielsweise ein Programm), wird Fiasco den Feld-Requester danach öffnen. Andere Attribute werden Standardwerte benutzen. Bitte be-

¹Bitte beschweren Sie sich nicht, daß dies zu viel ist ;-)

²Wie der aufmerksame Leser sicherlich sofort erkannt hat

achten Sie auch, daß wenn Sie ein Feld umwandeln und es danach wieder in den ursprünglichen Typ wandeln, die alten Attribute nicht benutzt werden.

Informationen über die Ergebnisse vom Feldumwandeln von einem zum anderen Typ können aus der Feldtypen-Dokumentation bezogen werden.

5.4 Benutzung von Markierungen

Bei der fortgeschrittenen Benutzung von Datenbanken können Markierungen nützlich sein. Eine Markierung ist einfach ein kleiner Hinweis, der für einen Record gesetzt oder nicht gesetzt sein kann. Markierungen könnten mit Boolean oder ähnlichen Feldern simuliert werden, die Markierungen von Fiasco haben jedoch zusätzliche Vorteile. Zum Beispiel kann ein markierter Record einfach in der Liste von Fiasco entdeckt werden, da der Record farblich hervorgehoben wird. In der Maske können markierte Records durch ein “M” im Status-Gadget des Services-Fensters entdeckt werden.

Markierungen können mit **Record/Markiere Record** gesetzt und mit **Record/Lösche Markierung** wieder entfernt werden. Wenn Sie alle Markierungen in einem Projekt löschen wollen, steht Ihnen **Record/Lösche alle Markierungen** zur Verfügung. Um alle Markierungen zu setzen, benutzen Sie **Records/Markiere alle Records**. Wenn Sie alle Markierungen löschen und alle nicht markierten Records markieren wollen, können Sie dies mit **Record/Markierungen umschalten** machen.

Filter (siehe Abschnitt 6.6) sind verwandt mit den Markierungen. Dafür hat Fiasco ein paar zusätzliche Menüpunkte im **Compare** menu. **Vergl./Markieren** öffnen einen Such-Requester, der fürs Markieren von Records benutzt werden kann, die zu einem Muster passen. Um die Markierungen in einen Filter umzuwandeln, kann man **Vergl./Markierungen zu Filter** benutzen. Umgekehrt, zum Umwandeln eines Filters zu Markierungen, steht **Vergl./Filter zu Markierungen** zur Verfügung. Diese Funktion setzt Markierungen bei Records, die durch den Filter durchkommen, und löscht diese bei Records, die ausgefiltert werden.

Markierungen werden in der Fiasco-Projekt-Datei gespeichert und werden so nach erneutem Laden eines Projektes beibehalten.

Kapitel 6

Suchen in einer Datenbank

Die GUI-Schnittstelle zur Standard-Suchfunktion von Fiasco bildet der Such-Requester, den man über den Menüpunkt Vergl./Suchen erreicht. Hier kann man auswählen, auf welches Feld die Suche bezogen sein soll, ein Suchmuster angeben und Parameter zur unscharfen Suche angeben.

Das Suchmuster kann einem gewöhnlichen Eintrag entsprechen. Bei den String-, Integer-, Float-, Date- und Timefeldtypen entspricht dies genau dem Inhalt des eigentlichen Felds. Falls sich die Suche auf ein Boolean-Feld bezieht, so kann man entweder **TRUE** oder **1** eingeben, falls man nach einem aktiven Feld suchen will, oder **FALSE** oder **0**, falls man nach einem inaktiven Feld suchen will. Cycle-Felder lassen als Eingabe entweder eine Auswahlmöglichkeit als Klartext zu, oder die Nummer einer Auswahlmöglichkeit (von 0 aufwärts). Slider-Felder lassen wiederum nur eine Zahl an sich zu. Extern- und Data- typesfelder können nur über den Filenamen gesucht werden. Eine Suche, die sich auf den Inhalt der Dateien bezieht, ist über die eingebaute Suchfunktion nicht möglich.

Mit den Gadgets am unteren Rand kann man die Suche starten. Wenn ein passender Eintrag gefunden wird, wird der entsprechende Record automatisch aktiviert. Die Suche läßt sich hinterher ohne Umweg über den Suchrequester mit den Menüpunkten Vergl./weisersuchen und Vergl./rückwärts suchen fortsetzen.

6.1 Suchmuster

Neben Eingaben im Klartext sind auch bei bestimmten Feldtypen Suchmuster zulässig. Der Stringfeldtyp erlaubt die Benutzung von AmigaDOS-ähnlichen Mustern. ? steht für ein unbekanntes Zeichen. ?iasco würde also nach Aiasco, Biasco, Ciasco, Iiasco, usw. suchen. (????) würde nach Einträgen suchen, die 4 Buchstaben lang sind. #? steht für null oder mehr unbekannte Zeichen. A#? könnte also Amiga, Afrika, A, ABCD bedeuten. ?#? sucht nach allen Einträgen, die nicht leer sind. Wie auch unter AmigaDOS können auch diese Muster “escaped” werden, wenn man nach Einträgen suchen will, die genau den Mustern entsprechen. Um dies zu tun, muß man den jeweiligen Zeichen ein ' (Alt-Ä) voranstellen.

Falls Sie anstatt #? lieber einen * als Suchmuster benutzen wollen, können Sie diesen über den Menüpunkt **Einstellungen/Als Pattern * benutzen** (siehe Abschnitt 10.2.62).

Integer- und Sliderfelder unterstützen auch zusätzliche Muster. Folgende sind möglich: >, <, >=, <=, !=. Das Argument muß nach dem Muster gegeben werden. > sucht nur nach Zahlen größer als x, >= nur nach Zahlen größer oder gleich x, < nur nach Zahlen kleiner als x und <= nur nach Zahlen kleiner oder gleich x. != sucht nur nach Zahlen, die ungleich x sind. Ein Muster wie ==, also “gleich” gibt es nicht, da dies durch den Wert ohne jedes Muster dargestellt wird.

Integer- und Sliderfelder unterstützen auch zusätzliche Muster. Folgende sind möglich: > x, < x, >= x, <= x, != x. ‘x’ steht hier für eine beliebige ganze Zahl. ‘>’ sucht nur nach Zahlen größer als x, ‘>=’ nur nach Zahlen größer oder gleich x, ‘<’ nur nach Zahlen kleiner als x und ‘<=’ nur nach Zahlen kleiner oder gleich x. ‘!=’ sucht nur nach Zahlen, die ungleich x sind. Ein Muster wie ‘==’, also “gleich” gibt es nicht, da dies durch den Wert ohne jedes Muster dargestellt wird.

Die von einem Feldtyp unterstützten Muster sind auch nochmal bei den Dokumentationen der jeweiligen Feldtypen aufgeführt.

Außerdem ist eine Zusammenfassung aller Suchmuster im Anhang verfügbar.

6.2 Unscharfe Suche

Die unscharfe Suche bietet die Möglichkeit, nach Strings zu suchen, wobei nicht nur nach gleichen Einträgen gesucht wird, sondern nach ähnlichen Einträgen. Wie tolerant die Funktion dabei ist, kann man unter **Faktor** einstellen. 0 entspricht der niedrigsten Toleranz, dies gleicht der normalen Suche, ist jedoch etwas langsamer. 100 bedeutet, daß alle Einträge als “ähnlich” angesehen werden. (Besser: “sollte bedeuten”, aber die Funktion ist

bei 100 dennoch *sehr* tolerant)

Die Zähl-Funktion eignet sich hervorragend, um mit der Unscharfen Suche etwas zu experimentieren.

6.3 Suchen mit ARexx

Auch mit ARexx können Suchen durchgeführt werden. Die Kommandos `F_FindNext`, `F_FindPrev` und `F_FindFirst` können dafür benutzt werden. Das Feld und das Muster für die Suche können als Argument an diese Kommandos übergeben werden.

Im Gegensatz zu der GUI-Suchfunktion wird nun jedoch nicht der jeweilige Record aktiviert, sondern die Nummer des Records in der Variablen `Result` zurückgegeben. Mit der kann man nun machen, was man will, z.B. mit dem Kommando `F_GotoRec` (siehe Abschnitt 11.2.27) den Record aktivieren.

Wenn Sie `F_FindFirst`, `F_FindNext` oder `F_FindPrev` ohne Argumente aufrufen, werden die Kommandos die Argumente, die vorher im Such-Requester benutzt wurden. Diese Werte können auch von ARexx aus mit den Kommandos `F_SetSearchPat` (siehe Abschnitt 11.2.59) und `F_SetSearchField` (siehe Abschnitt 11.2.58) gesetzt werden.

Die Dokumentation für `F_FindFirst` enthält ein Beispiel für die Suche mit ARexx.

Über ARexx kann man so auch eine Suchfunktion erstellen, die auch mehrere Felder unterstützt.

6.4 Zählen

Über den Menüpunkt `Vergl./Zählen` öffnet man einen dem Suchrequester sehr ähnlichen Requester, der sich genaugenommen nur durch Titel und die Gadgets am unteren Rand unterscheidet. Hier kann man wiederum Suchmuster, Bezugsfeld und die Toleranz für die unscharfe Suche angeben. Falls man dann `Ok` wählt, werden die Übereinstimmungen gezählt und das Ergebnis angezeigt. Natürlich werden auch hier Muster und Toleranz mit einbezogen und beeinflussen so das Ergebnis. So kann man sich auch mal das Verhalten des Toleranz-Faktors genauer betrachten.

6.5 Ersetzen

`Compare/Replace` erlaubt es, bestimmte Werte durch einen anderen zu ersetzen. Auch hier sind Suchmuster möglich, es wird jedoch immer nur ein

Wert eingesetzt. Das Gadget mit dem Titel **Replacement** nimmt den Wert auf, der neu eingesetzt werden soll. Falls das Gadget **Confirm** aktiviert ist, wird ihnen immer der jeweilige Record gezeigt und Sie werden gefragt, ob sie wirklich den alten Wert ersetzen wollen.

Achtung: Mit einem falschen Muster (z.B. #?) kann man schnell viele Einträge löschen bzw. unbrauchbar machen!

6.6 Filter

Compare/Filter öffnet auch den Such-Requester, in dem man auch wieder die altbekannten Parameter angeben kann bzw. muß. Bestätigt man diesen Request mit **Ok**, so ist die Datenbank hinterher scheinbar geschrumpft. In Wirklichkeit werden jedoch nun nur noch die Records angezeigt, die den Suchparametern entsprechen. Die Funktionen **Records/Nächster**, **Records/Voriger** oder deren Äquivalente verhalten sich nun wie **Weitersuchen** bzw. **Rückwärts suchen**.

Der Filter wird gleich nach dem Bestätigen des Requesters erstellt. Das bedeutet, daß neu hinzugefügte Records auch angezeigt werden, auch wenn sie nicht mit dem Filter übereinstimmen. Um dies zu erreichen muß man einfach den Requester noch mal aufrufen und bestätigen.

Über den Menüpunkt **Vergl./Filter An?** läßt sich der Filter zeitweise aktivieren und deaktivieren.

Kapitel 7

Alternative Daten-Mechanismen

Normalerweise speichert Fiasco die Daten der Felder eines Projektes direkt in der Datei des Projektes. Dies kann jedoch bei bestimmten Arten von Daten sehr ineffizient in Sachen Disketten-Platz sein.

Zur Zeit stellt Fiasco zwei alternative Mechanismen, um Daten zu speichern, zur Verfügung. Relationen lesen die Daten von einem anderen Projekt in ihr Projekt ein. So können mehrere Projekte auf die gleichen Daten zugreifen. Im Kontrast dazu speichern virtuelle Felder ihre Daten nirgendwo! Die Daten werden während des Ladens des Projektes automatisch berechnet.

Bitte beachten Sie, daß diese Mechanismen nur Diskettenspeicherplatz sparen, im RAM benötigen diese Felder genauso viel Platz wie andere Felder.

7.1 Relationen

Relationen sind Felder, deren Inhalt nicht in der Datei des Projektes, aus dem die Relationen stammen, sondern in einer weiteren Projekt-Datei zu finden sind. Um zu wissen, aus welchem Record die Daten entnommen werden sollen, müssen beide Projekte ein Feld mit einem eindeutigen Identifikations-Inhalt haben, das als Schlüssel ("Key") bezeichnet wird.

Diese Methode beugt der Situation vor, daß in vielen verschiedenen Projekten die selben Daten gespeichert sind. So wird geholfen, Disketten- oder Platten-Speicherplatz zu sparen. Weiterhin brauchen Sie nur die Daten

in einem Feld zu ändern, damit alle anderen “verwandten” Felder auch ihren Inhalt ändern.

7.1.1 Erstellen von Relationen

Um in Fiasco Relationen nutzen zu können, muß man zuerst in ein Projekt erzeugen, das die Quelle für ein anderes Projekt darstellt, das dann daraus die Daten liest. Dieses Projekt wird später “dort” genannt, während das Projekt, das die Daten später liest “hier” genannt wird.

In dem Quell-Projekt müssen dann mindestens zwei Felder erstellt werden, eins für die eigentlichen Daten, das andere für den Schlüssel. Das Feld für den Schlüssel sollte vom Typ Integer sein, da dies am schnellsten ist und von Fiasco automatisch mit Schlüsseln versorgt werden kann. Für besondere Fälle erlaubt Fiasco jedoch auch, jeden anderen Feldtypen zu verwenden.

Um die Option zu aktivieren, die dem Schlüssel-Feld automatisch einen eindeutigen Schlüssel zuweist, muß man im Feld-Requester das Gadget **Eindeutiger Schlüssel** aktivieren. Bitte beachten Sie, daß der Schlüssel nur beim Erstellen eines Records erzeugt wird! Falls schon Records existierten, bevor diese Option aktiviert wurde, behalten diese ihren alten, und möglicherweise nicht eindeutigen Inhalt. Außerdem kann man später den Inhalt dieses Feldes beliebig verändern, ohne daß irgendeine Überprüfung vorgenommen wird.

Der Typ des zweiten Feldes ist relativ egal. Man sollte sich nur bei den Datentypen String, Extern und Datatypes das Attribut **Maximale Zeichen** gut merken, da dieses im Feld des zweiten Projektes genau übereinstimmen muß.

Es wäre sinnvoll, wenn dieses Projekt bereits einige Records mit etwas Inhalt enthalten würde, damit nach dem Aktivieren der Relation sofort ein Effekt sichtbar wird.

Dieses Projekt muß jetzt abgespeichert werden und ein Neues muß erstellt werden.

Das zweite Projekt muß wiederum aus zwei Feldern bestehen, die in Typ und bei den Typen String, Extern und Datatypes im Attribut **Maximale Zeichen** übereinstimmen. Das Feld für den Schlüssel sollte jedoch *nicht* wieder **Eindeutiger Schlüssel** aktiviert haben, da dies keinen größeren Sinn machen würde.

Bevor nun die Relation aktiviert werden kann, sollte man das Projekt im selben Verzeichnis wie das erste abspeichern, damit im Relations-Requester keine absoluten, sondern relative – oder in diesem Fall keine Pfade – benutzt werden können.

Um nun die Relation erstellen zu können, muß das Feld, das *nicht* den Schlüssel beinhalten soll, aktivieren und dazu den Relations-Requester auf-

rufen (mit **Felder/Relationen Ändern**) Hier sollte man zuerst den Schlüssel “hier” im Listview oben links auswählen. Danach muß die Relations-Datei mit dem Filerequester-Gadget in der Mitte ausgewählt werden. Der Requester sollte sich bereits im richtigen Verzeichnis befinden. Nachdem die Datei ausgewählt wurde, werden in den unteren beiden Listviews die möglichen Schlüssel und eigentlichen Felder “dort” angezeigt. Hier sollte man auch die richtigen Felder auswählen und anschließend **Ok** anwählen. Hat man alles richtig gemacht werden jetzt die Relationen zu den bereits bestehenden Records geladen.

Eine Relationen-Checkliste, die diese Informationen in komprimierter Form enthält, ist ebenfalls verfügbar.

7.1.2 Technische Anmerkungen

Fiasco 1.1 hat die Zugriffsgeschwindigkeit bei Relationen stark erhöht. Dies konnte durch bestimmte Optimierungen erreicht werden, während der grundlegende Code sich nicht geändert hat. Der Teil, der am längsten dauert, ist die Suche nach passenden Schlüsseln. Fiasco muß für jeden Schlüssel in einem Projekt das andere ganz durchlaufen und die Schlüssel vergleichen. Fiasco 1.1 macht dies schneller, indem Einträge, die bereits gelesen wurden, im Speicher behalten werden (“Caching”). Dies verbraucht jedoch viel Speicher. Wenn der Speicher knapp wird muß Fiasco einige Einträge aus dem RAM rauswerfen. Nun muß Fiasco wieder auf die Disk zurückgreifen, um diese Einträge zu lesen, was den Vorgang wieder verlangsamt.

Die zweite Methode, die Zugriffsgeschwindigkeit zu erhöhen, ist das Markieren von Schlüsseln, die keinen passenden Schlüssel im “there” File haben. Dies ist natürlich nur nützlich, wenn das Projekt “blinde” Schlüssel enthält.

Die dritte Methode würde nicht existieren, wenn die erste nicht existieren würde. Das Problem dieser Methode sind große Dateien. Die Methode benutzt eine unsortierte Liste, in der die Einträge mit der Record-Nummer gespeichert werden. Um einen Record zu finden, muß Fiasco durch die ganze Liste gehen und jede Record-Nummer vergleichen. Wenn diese Liste eine relativ große Anzahl von Einträgen enthält, wird dieser Vorgang sehr langsam (Beispiel: Wenn Sie 1000 Records “hier” haben und 1000 Records “dort”, müssen Sie im schlimmsten Fall 1000x1000 Records untersuchen). Sortieren oder besondere Such-Methoden lösen dieses Problem nicht, da diese selbst einige Zeit in Anspruch nehmen. Fiasco 1.1 merkt sich jetzt bloß die Adresse des Records, an dem Fiasco vorher die Suche aufgehört hat und macht das nächste Mal dort weiter. Dies erhöht die Geschwindigkeit bei Dateien, bei denen die Schlüssel ungefähr gleich anwachsen wie bei ihren “dort”-Projekten. Ich denke (hoffe), daß die meisten Dateien so strukturiert sind. Der schlimmste Fall hat sich jedoch nicht verändert.

Diese Informationen sollten Ihnen einen gewissen Eindruck davon geben, warum ein Projekt die Relationen nicht so schnell lädt wie ein anderes. Hier ist nochmal eine kurze Liste der Faktoren, die das Laden verlangsamen können:

- Wenig RAM
- Keine “blinden” Schlüssel; Alle Schlüssel haben einen passenden im “there”-File
- Schlechte Ordnung der Records

Bitte beachten Sie: Wenn Sie versuchen, Relationen von einem Floppy-Diskettenlaufwerk zu laden, wird dies extrem langsam sein, da Fiasco durch die gesamte Datei “seekt”.

7.2 Virtuelle Felder

Die Daten von virtuellen Feldern werden nicht auf Disk gespeichert, sondern während des Ladens des Projektes berechnet. Wenn Sie ein Feld virtuell machen möchten, sollten sie die **Virtuell** Option im Feld-Requester aktivieren.

Fiasco benutzt zum Berechnen dieser Daten das ARexx-Script des Feldes. Das Script wird für jedes virtuelle Feld in jedem Record aufgerufen.

Da Fiasco in einem besonderen Status ist, sind Sie bei den Kommandos, die Sie aufrufen können, etwas eingeschränkt. Zur Zeit können Sie nur diese Kommandos aufrufen:

- F_SetFieldCont
- F_GetFieldCont
- F_MarkRecord
- F_UnmarkRecord
- F_IsMarked
- F_RequestChoice
- F_RequestFile
- F_GetFieldAttributes
- F_VirtualMode

Wenn sie eins dieser Kommandos aufrufen, das sich auf einen Record bezieht, wird der Record, der gerade von Fiasco bearbeitet wird, der aktive sein (Auf den Sie sich beziehen, wenn Sie das Record-Argument weglassen). Wenn Sie sich auf andere Records beziehen wollen, sollten Sie darauf achten, daß Sie nicht wissen, welche virtuellen Felder bereits durchlaufen wurden. Es ist dennoch garantiert, daß alle normalen Felder und Relationen Ok sind. Wenn Sie `F_GetFieldCont` oder `F_SetFieldCont` benutzen, können Sie nicht wissen, welche anderen virtuellen Felder im aktiven Record bereits bearbeitet wurden.

Das ARexx-Scripts eines virtuellen Feldes wird, wie auch bei normalen Feldern, aufgerufen, wenn der Inhalt des Feldes vom Benutzer geändert wurde. Um herauszufinden, ob Sie sich im normalen oder virtuellen Status befinden, benutzen Sie `F_VirtualMode` (siehe Abschnitt 11.2.68).

Kapitel 8

Import und Export

Die Import und Export-Funktionen von Fiasco machen es möglich, Daten von anderen Datenbank-Programmen in Fiasco zu laden und Daten von Fiasco für andere Programme lesbar zu machen.

Solche Import/Export-Dateien enthalten die Daten kodiert im ASCII-Format. Die Felder oder Records sind durch besondere Zeichen markiert, die in der Import/Export-Funktion von Fiasco frei definiert werden können.

An den Anfänger: Um die Import/Export Funktion von Fiasco nutzen zu können, müssen Sie ein paar Details kennen. Im Folgenden wird die Struktur von Import/Export-Dateien beschrieben. Wenn Sie dieses bereits kennen, brauchen Sie das nicht zu lesen. Der Abschnitt danach beschreibt die besonderen Escape-Sequenzen, die von Fiasco benutzt werden. Obwohl andere Datenbanken ein ähnliches Schema benutzen können, sollten Sie sich dies genau durchlesen, da die gesamte Import/Export-Funktion von Fiasco darauf beruht.

8.1 Struktur von Import/Export-Dateien

Die hier benutzen Namen beziehen sich auf die Import/Export-Requester. Bitte beachten Sie, daß manchen Markierungs-Zeichen leer sein könne. Um die Datei benutzen zu können, müssen Sie mindestens entweder Feld Start/Feld Ende oder Feld Separator und entweder Record Start/Record Ende oder Record Separator angeben.

Die Import-Funktion von anderen Programmen kann jedoch möglicher-

weise auch verwirrt werden, auch wenn diese Regeln eingehalten werden.

Record Start	
Feld Start	
<i>Feld Daten</i>	Inhalt des Feldes in ASCII-Format.
Feld Ende	
Feld Seperator	Trennt zwei Felder, nicht nach dem letzten Feld eines Records.
...	
Feld Start	
<i>Feld Daten</i>	
Feld Ende	
Record Ende	
Record Seperator	Trennt zwei Records, nicht nach dem letzten Record einer Datei.
...	
Record Start	
...	(siehe oben)
Record Ende	
<i>Ende der Datei</i>	

Wenn Sie Erster Record enthält IDs aktivieren, werden die IDs der Felder im ersten Record gespeichert, als ob die Felder wären.

Ein Beispiel einer Import/Export-Datei

Record Start und Record Ende sind leer. Record Seperator ist eine Newline. Feld Start und Feld Ende sind Anführungszeichen. Feld Seperator ist ein Komma. Der erste Record enthält die IDs der Felder. Beachten Sie das leere Feld im letzten Record.

```
"Name","FirstName","Rank","Current"
"Picard","Jean-Luc","Captain","U.S.S. Enterprise"
"Riker","William Thomas","Commander","U.S.S. Enterprise"
"Data","","Lieutenant Cmdr.,""U.S.S. Enterprise"
```

8.2 Besondere Zeichen

Oft können die Zeichen fürs Markieren der Felder und Records nicht als einfacher Text eingegeben werden. Beispielsweise kann man für das Newline-Zeichen nicht einfach die Return-Taste betätigen. Stattdessen muß dieses Zeichen als Escape-Sequenz eingegeben werden. Fiasco unterstützt Escape-Sequenzen ähnlich denen der Programmiersprache "C". Die Escape-Sequenzen

werden von einem `\` eingeleitet. Diese werden unterstützt:

<code>\n</code>	Newline-Zeichen, ASCII 10
<code>\f</code>	Formfeed-Zeichen, ASCII 12
<code>\r</code>	Return-Zeichen, ASCII 13
<code>\t</code>	Horizontaler Tabulator, ASCII 9
<code>\v</code>	Vertikaler Tabulator, ASCII 11
<code>\Nummer</code>	Zeichen mit angegebenem ASCII Code
<code>\Zeichen</code>	Zeichen direkt übernommen

Die letzte Option (`\ + Zeichen`) macht es möglich, Zeichen zu benutzen, die für Escape-Sequenzen reserviert sind (z.B. `\\` für einen `\`).

In der Import-Funktion können auch Zeichen-Klassen angegeben werden. Zeichen-Klassen werden in Fiasco mit einem `#` markiert. Diese werden unterstützt:

<code>#p</code>	Druckbares Zeichen.
<code>#a</code>	Druckbares ASCII-Zeichen ohne internationale Zeichen.
<code>#c</code>	Kontrol-Zeichen. Nicht Druckbar

Export unterstützt das Einfügen einiger zusätzlicher Informationen in der Export-Datei. Diese Kommandos werden durch ein `%` markiert. Diese werden unterstützt:

<code>%f</code>	ID des Feldes.
<code>%r</code>	Nummer des Records

8.3 Importieren von Daten

Der Import-Requester ist die GUI-Schnittstelle zur Import-Funktion von Fiasco. Sie können den Requester mit **Projekt/Importieren** öffnen. Die zu importierende Datei muß in **Datei** angegeben werden. Nachdem Sie dies getan haben müssen Sie die Struktur der Datei im Requester angeben. Wenn Sie die Datei gerade aus einer anderen Datenbank exportiert haben und die Struktur-Parameter immer noch kennen, können Sie sie einfach übernehmen. Wenn dies nicht der Fall ist, können Sie die Datei mit dem **View**-Schalter betrachten. Fiasco wird entweder More oder Multiview starten, um die Datei anzuzeigen. Wenn die Datei eine standardmäßige Struktur hat, sollte es nicht zu schwer sein, die Parameter zu erkennen.

Normalerweise sind **Record Start** und **Record Ende** leer und **Record Separator** ist `\n`. **Feld Start** und **Feld Ende** sind oft leer oder Anführungszeichen (`"`). Gewöhnliche Werte für **Feld Separator** sind ein Komma (`,`) oder ein Tabulator (`\t`).

Zeilen Überspringen definiert die Zeichen, die einen Kommentar *am Beginn einer Zeile* markieren. Dies kann auch benutzt werden, um Formatierungs-Informationen zu überspringen, die Fiasco nicht auswerten kann. Mit **Am Start überspringen** kann man Kommentare oder Ähnliches am Anfang der Datei überspringen. **Max. Felder** kann zum Definieren einer Record-Ende-Markierung benutzt werden, wenn weder **Record Trenner** noch **Record Ende** zur Verfügung stehen.

Aktivieren Sie **Erster Record enthält IDs**, wenn der erste Record der Datei nicht aus echten Daten, sondern den IDs der jeweiligen Felder besteht. Wenn dies aktiv ist, wird Fiasco die IDs benutzen, um entweder neue Felder zu erstellen, oder um die bereits bestehenden Felder zu benutzen.

Die Optionen **Neue Felder anhängen** und **Altes Projekt überschreiben** kontrollieren, ob Sie ein Projekt aktualisieren wollen, oder ein neues erstellen wollen. Wenn Sie ein neues Projekt erstellen wollen, sollten Sie beide Optionen aktivieren.

Wenn Sie die Einstellungen, die Sie gemacht haben, in der Zukunft weiterbenutzen wollen, können Sie diese mit **Save** speichern. Die Einstellungen können dann mit **Load** wieder geladen werden. Die Fiasco-Distribution enthält bereits mehrere vorgefertigte Einstellungen.

Um den Import-Vorgang zu starten, müssen Sie einfach auf **Ok** klicken. *Achtung:* Wenn die Import-Datei zu groß ist oder die Struktur-Parameter falsch sind, kann Fiasco allen freien Speicher an sich reißen. Fiasco hat zwar keine größeren Probleme bei RAM-Mangel, aber andere Programme können Probleme haben.

Wenn alles gut ging, schließt sich der Import-Requester und das neue Projekt wird aktiviert. Sie werden wahrscheinlich zuerst im Masken-Modus das Layout der Maske verändern, das von Import nur sehr einfach angelegt wurde. Wenn Sie **Erster Record enthält IDs** nicht aktiviert hatten, sollten Sie die IDs passend zu den Feldinhalten ändern. Zusätzlich sollten Sie Text-Felder erzeugen, um den existierenden Feldern eine gewisse Beschreibung zukommen zu lassen. Nun haben Sie schon eine schön formatierte Maske. Alle Felder sind bis jetzt jedoch noch String-Felder. Nun sollten Sie untersuchen, ob manche Felder Integer-, Cycle- oder andere Felder sein können. Der Typ dieser Felder kann dann mit der Umwandlungs-Funktion (siehe Abschnitt 5.3) von Fiasco verändert werden. Im Beispiel, das in *Sektion 8.1*, gegeben wurde, kann das Rank-Feld zu einem Cycle-Feld umgewandelt werden.

Wenn Sie diese Schritte erledigt haben, sollten Sie das Projekt unter einem passenden Namen speichern.

8.4 Exportieren von Daten

Das Exportieren von Daten ist aus der Sicht von Fiasco viel weniger kompliziert als das Importieren. Normalerweise können Sie die Standard-Parameter von Fiasco, d.h. kein **Record Start** und **Record Ende**, ein Newline für **Record Trenner**, Anführungszeichen für **Feld Start** und **Feld Ende** und ein Komma für **Feld Trenner**. Wenn Sie diese Parameter benutzen, müssen Sie beachten, daß die Daten keinerlei Anführungsstriche enthalten. Zusätzlich müssen Sie darauf achten, daß das Programm, das die Daten lesen soll diese Konfiguration auch unterstützt.

Wenn Sie **Erster Record** enthält IDs anwählen, wird Fiasco einen zusätzlichen Record am Anfang der Datei erzeugen, der die IDs der Felder statt den eigentlichen Daten enthält. Die Datei wird keine andere Formatierungsinformation enthalten.

Wenn Sie **Nur Markierte Records** anwählen, werden nur die Records geschrieben, bei denen die Markierung gesetzt ist.

Um das Exportieren zu starten, klicken Sie auf **Ok**.

Kapitel 9

Feldtypen

Felder sind die eigentlichen Speicher für Daten. Dazu reichen eigentlich die Grundtypen “String” und “Zahl”. Alle anderen Typen sind mehr oder weniger Modifikationen dieser Grundtypen, die die Arbeit mit der Datenbank erleichtern.

Fiasco unterstützt zur Zeit folgende Feldtypen:

- String
- Integer
- Float
- Boolean
- Cycle
- Slider
- Date
- Time
- Extern
- Datatypes
- Text
- Button

9.1 Standard-Attribute

Diese Attribute werden normalerweise von einem Feldtyp unterstützt:

ID: Hier kann man einen String angeben, der für die Identifikation des Feldes dient. Dieser String darf nur einmal in einem Projekt vorkommen nicht ausgeführt. Im Masken-Modus wird dieser String in den Feldern angezeigt. Weiterhin wird er in der Kopf-Zeile der Liste, beim Suchen, dem Relations-Requester und in ARexx benutzt.

Breite: bestimmt die Breite in Zeichen, mit der das Feld in der Maske erscheint. Dieser Wert ist auch ein Standard-Wert für die Breite der Spalte in der Liste. Dieser Wert kann jedoch später noch verändert werden.

Startwert/Eigener Wert: Hier kann man einen Wert angeben, der beim Neuerstellen eines Records in dieses Feld kopiert wird.

Startwert/Alter Wert: Wenn man einen Record neu erstellt wird der Wert des alten Records übernommen. Wenn **Alter Wert** aktiv ist, wird **Eigener Wert** ignoriert. Dies ist insbesondere beim Eingeben vieler ähnlicher Datensätze nützlich.

Script: Hier läßt sich ein ARexx-Script angeben, das aktiviert wird, wenn ein neuer Record erstellt wird, bzw. der Inhalt des jeweiligen Feldes geändert wird. Abhängig vom Inhalt des Scripts kann der eingestellte Startwert von seinem eigentlichen Sinn abweichen. Auf das Script wird relativ von dem Verzeichnis zugegriffen, in dem das Projekt liegt.

Nur Lesbar: Der Inhalt des Feldes wird in einer zurückgelegten Box angezeigt, die nicht aktiviert oder verändert werden kann.

Virtuell: Der Inhalt des Feldes wird nicht auf Diskette gespeichert, sondern jedesmal, wenn das Projekt geladen wird neu berechnet. Dies wird mit den Startwert-Attributen und dem ARexx-Script-Attribut erledigt. Bitte beachten Sie, daß diese Felder die selbe Menge an RAM wie normale Felder beanspruchen.

Durch Dehnungswerte (siehe Abschnitt 4.5) ungleich 0 können die Dimensions-Attribute verzerrt werden.

9.2 Feldtypen

9.2.1 Der String-Feldtyp

In einem String-Feld lassen sich Zeichenketten begrenzter Länge eingeben.

Neue Attribute:

Maximale Zeichen: bestimmt die maximale Anzahl von Zeichen, die in das Feld eingegeben werden dürfen. Dieses Attribut wirkt sich direkt auf die Größe der Datei aus.

Suchäquivalent:

entspricht dem Feldinhalt.

Unterstützte Suchmuster:

? = Ein unbekanntes Zeichen.

#? = Null oder mehr unbekannte Zeichen.

Umwandlung in ein String-Feld:

Alle Felder können ohne Datenverlust in String-Felder umgewandelt werden. Alternative Format sind in Klammern angegeben, wenn sie unterstützt werden.

Zusätzliche Anmerkungen:

- Integer - Nummer umgewandelt
- Float - Nummer umgewandelt
- Boolean - "Checked" wird TRUE(1), sonst FALSE(0)
- Cycle - Label (Labelnummer) umgewandelt
- Slider - Level umgewandelt
- Date - Datum im Format "DD.MM.[YY]YY" umgewandelt
- Time - Zeit im Format "HH:MM[:SS]" umgewandelt
- Extern - String kopiert
- DTypes - String kopiert

9.2.2 Der Integer-Feldtyp

In einem Integer-Feld lassen sich Ganzzahlen im Bereich von -2.147.483.348 bis 2.147.483.347 eingeben.

Neue Attribute:

Maximale Zeichen: bestimmt die maximale Anzahl von Zeichen, die in das Feld eingegeben werden dürfen.

Startwert/Eindeutiger Schlüssel: legt in diesem Feld bei Neuerstellung eines Records eine - für diese Datenbank - einmalige Identifikationsnummer ab. Bitte beachten Sie, daß diese frei geändert werden kann und so die Eindeutigkeit nicht gewährleistet ist.

Suchäquivalent:

entspricht dem Feldinhalt.

Unterstützte Suchmuster:

>- größer als
<- kleiner als
>=- größer oder gleich
<=- kleiner oder gleich
!= - ungleich

Umwandlung in ein Integer-Feld:

Integer-Felder akzeptieren nur den numerischen Teil der Quelldaten. Wenn die Quelldaten mit einem nicht numerischen Teil beginnen, wird das Feld 0 enthalten.

Zusätzliche Bemerkungen:

Float	-	Ganzzahliger Teil umgewandelt
Boolean	-	“Checked” wird 1, sonst 0
Cycle	-	Nummer des Labels umgewandelt
Slider	-	Level umgewandelt
Date	-	Erstes Datums-Element umgewandelt (Tag)
Time	-	Erstes Zeit-Element umgewandelt (Stunde)

9.2.3 Der Float-Feldtyp

In ein Float-Feld können reelle Zahlen eingegeben werden.

Neue Attribute:

Präzision: Anzahl der darzustellenden Nachkommastellen.

Suchäquivalent:

entspricht dem Feldinhalt

Umwandlung in ein Float-Feld:

Float-Felder akzeptieren nur den numerischen Teil der Quelldaten. Wenn die Quelldaten mit einem nicht numerischen Teil beginnen, wird das Feld 0 enthalten.

Zusätzliche Bemerkungen:

Boolean - "Checked" wird 1.0, sonst 0.0

Cycle - Nummer des Labels umgewandelt

Bemerkung: Die Genauigkeit von Float ist eher gering. So sollte man bei größeren Genauigkeitsansprüchen String anstatt Float benutzen. ARexx kann damit genauso gut rechnen, wenn korrekte Zahlen angegeben werden.

9.2.4 Der Boolean-Feldtyp

Ein Boolean-Feldtyp nimmt Wahrheitswerte auf, d.h. entweder Ja/True oder Nein/False. Ein Boolean-Feld erscheint im Maskenmodus als ein "Checkbox-Gadget".

Geänderte Attribute:

Breite: ist immer 3

Suchäquivalent:

TRUE oder 1 - Gesetzter Haken

FALSE oder 0 - Kein Haken

Umwandlung in ein Boolean-Feld:

Boolean-Felder nehmen alle Zahlen ungleich 0 und TRUE als gesetzt. Alle anderen Werte werden nicht gesetzt.

Unter Amiga OS 2.0 kann das Aussehen dieses Feldes leicht kaputt wirken, da die Grafiken noch nicht skalierbar sind. Unter Amiga OS 3.0 oder höher werden die Grafiken korrekt an die Größe der Schrift angepaßt.

9.2.5 Der Cycle-Feldtyp

Der Cycle-Feldtyp bietet mehrere Auswahlmöglichkeiten aus einer selbst definierten Liste. Der Größte Vorteil beim Cycle-Feldtyp ist die Speichersparnis. Es werden maximal 65536 Auswahlmöglichkeiten unterstützt. (Ich hoffe, daß das reicht ;-)) Ein Cycle-Feld erscheint in der Maske als ein "Cycle-Gadget" (wie der Name schon sagt).

Neue Attribute:

Labels: Eine Liste aller Auswahlmöglichkeiten. Ein Eintrag ist Pflicht, ab zweien wird das Cycle-Feld sinnvoll.

Suchäquivalent:

entweder die Nummer des Labels von Null zählend, oder der Eintrag im Klartext (nicht vertippen!)

Umwandlung in ein Cycle-Feld:

Die Werte werden in Labels umgewandelt. Wenn es gleiche Werte gibt, werden diese das selbe Label benutzen. Daten gehen dabei nicht verloren.

Zusätzliche Bemerkungen:

Boolean - "Gesetzt" wird TRUE(1), sonst FALSE(0)

9.2.6 Der Slider-Feldtyp

Ein Slider-Feld ist eine Abwandlung des Integer-Typs. Mit ihm können Ganzzahlen grafisch dargestellt und verändert werden. Der Bereich der Zahlen ist auf -32.768 bis 32.767 begrenzt, kann jedoch durch mehrere Attribute beeinflusst werden.

Neue Attribute:

Min. Wert: gibt den niedrigsten Wert der dargestellt wird an. Dieser Wert entspricht der Stellung des "Knobs" am linken bzw. am oberen Rand des Feldes.

Max. Wert: gibt den höchsten Wert der dargestellt wird an. Dieser Wert entspricht der Stellung des "Knobs" am rechten bzw. am unteren Rand des Feldes.

Format: ist ein Formatstring im Stil der Programmiersprache C. Er baut sich wie folgt auf:

`%[-][0][ZahlenFeld][.Maximum][l]Format`

- -: Die Zahl wird linksbündig im Feld geschrieben, normal rechts
- 0: Das Feld wird mit Nullen aufgefüllt. also: 1 -> 001
- ZahlenFeld: Die minimale Feldbreite
- Maximum: nur für Strings, nicht wichtig hier

- l: Zeigt an, daß die Zahl 32 bit breit ist, muß hier immer angegeben werden.
- Format:
 - c - Char, das ASCII-Zeichen für die Zahl wird ausgegeben
 - d - Die Zahl wird ausgegeben.
 - u - Die Zahl wird vorzeichenlos ausgegeben.
 - x - Die Zahl wird im Hexadezimal-Format ausgegeben.Es gibt außerdem die Format-Zeichen **b** und **s**. Da diese Adressen als Argumente benötigen, produzieren sie in diesem Fall nur Müll!

Die Formatierung erfolgt mit der Exec-Funktion `RawDoFmt()`.

Max. Format-Länge: gibt die maximale Länge der Format-Ausgabe an, dieser Bereich wird bei der Breite angerechnet, d.h. je größer `MaxFormatLen`, desto kleiner das eigentliche Feld.

Suchäquivalent:

Die eigentliche Zahl.

Unterstützte Suchmuster:

> - größer als
< - kleiner als
>= - größer oder gleich
<= - kleiner oder gleich
!= - ungleich

Umwandlung in ein Slider-Feld:

Integer-Felder akzeptieren nur den numerischen Teil der Quelldaten. Wenn die Quelldaten mit einem nicht numerischen Teil beginnen, wird das Feld 0 enthalten. Nach der Umwandlung sollten sie die `Max./Min.` Attribute überprüfen.

9.2.7 Der Datums-Feldtyp

In ein Datums-Feld kann ein Datum eingegeben werden.

Neue Attribute:

Startwert/Aktuelles Datum: Beim Neuerstellen eines Records wird das aktuelle Datum in dieses Feld eingefügt.

Suchäquivalent:

entspricht dem Feldinhalt

Umwandlung in ein Datums-Feld:

Datum-Felder benötigen die Daten im Format DD.MM.[YYYY]. Die einzelnen Teile müssen Zahlen sein. Wenn die Werte nicht numerisch sind, wird der Teil ??.

	Integer	-	ins erste Element (Tag) umgewandelt
Zusätzliche Bemerkungen:	Float	-	Ganzzahliger Teil wird Tag, Nachkommastellen Monat
	Time	-	Stunde wird Tag

Das Datum wird z.Zt. nur im deutschen Format (TT.MM.YYYY) gelesen und geschrieben. Es wird keine Überprüfung der Werte vorgenommen, so sind z.Zt. Werte wie 65.20.3687 möglich.

9.2.8 Der Zeit-Feldtyp

In ein Zeit-Feld kann eine Uhrzeit eingegeben werden.

Neue Attribute:

Startwert/Aktuelle Zeit: Beim Neuerstellen eines Records wird die aktuelle Zeit in dieses Feld eingefügt.

Suchäquivalent:

entsprechen dem Feldinhalt.

Umwandlung in ein Zeit-Feld:

Zeit-Felder benötigen die Daten im Format HH:MM:SS. Jedes Element muß eine Zahl sein. Wenn ein Element nicht numerisch ist, wird es 0.

Zusätzliche Bemerkungen:

Integer	-	In Stunde umgewandelt
Float	-	Ganzzahliger Teil in Stunde umgewandelt
Date	-	Tag wird Stunde

Die Zeit wird z.Zt. nur mit Sekunden ausgegeben (HH:MM:SS). AM und PM wird nicht unterstützt. Außerdem wird keine Wertüberprüfung vorgenommen, so sind z.Zt. Werte wie 55:66:99 möglich.

9.2.9 Der Extern-Feldtyp

Ein Extern-Feld nimmt einen Dateinamen auf, der auf Wunsch an ein benutzerdefiniertes Programm als Argument weitergeleitet wird. So kann man weitere Daten zu einem Record definieren und abrufen.

Neue Attribute:

Kommando: ist der Name eines Programms, mit dem die Daten verarbeitet werden können. Die Zeichenfolge %s wird bei Aufruf durch den Inhalt des Feldes ersetzt. Läßt man %s weg, so wird kein Argument übergeben. (z.B. C:ED %s)

Stack: Gibt die Größe des Stacks für das Kommando an.

Maximale Zeichen: bestimmt die maximale Länge des Dateinamens. Dieses Attribut wirkt sich direkt auf die Dateigröße aus.

FileReq Gadget: Aktivieren Sie dieses Attribut, um ein weiteres Gadget an der linken Seite des Feldes zu haben, das einen Filerequester öffnet, um den Inhalt zu verändern. Dies hat natürlich nur Sinn, wenn das Kommando Dateinamen als Argumente will.

Suchäquivalent:

entspricht dem Feldinhalt.

Umwandlung in ein Extern-Feld

Alle Feldtypen können in Extern-Felder ohne Verlust von Daten umgewandelt werden. Sie müssen jedoch ein Programm angeben, das diese Daten benutzen kann.

Zusätzliche Bemerkungen:

Boolean - TRUE "Gesetzt" wird TRUE(1), sonst FALSE(0)

Cycle - Label (Labelnummer) umgewandelt

Die Programme werden mit der AmigaDOS-Funktion `System()` aufgerufen. Für Aus- oder Eingabeoperationen wird ein Konsolen-Fenster geöffnet.

9.2.10 Der Datatypes-Feldtyp

Ein Datatypes-Feld ist Extern-Feldern ähnlich. Der Unterschied ist, daß die Datatypes-Library benutzt wird (und somit kann man dieses Feld erst ab OS 3.0 benutzen) und daß die Daten direkt in der Maske angezeigt werden. Durch die Datatypes-Library ist dieses Feld auch universell einsetzbar und frei erweiterbar. Ein “Popup”-Gadget an der unteren linken Seite des Feldes macht es möglich, den Inhalt des Feldes mit einem Filerequester zu verändern. Wenn ein Fehler in Zusammenhang mit dem Feld auftritt, wird dieser Fehler im Feld angezeigt.

Neue Attribute:

Maximale Zeichen: bestimmt die maximale Länge des Dateinamens. Dieses Attribut wirkt sich direkt auf die Dateigröße aus.

Scroll-Leisten: Sollen Scroller zur Verfügung gestellt werden, mit denen der Inhalt des Feldes verschoben werden kann? Ohne Scroller kann immer nur die linke obere Ecke der Daten betrachtet werden. (Dies ist nicht ganz wahr. Manche Datentypen unterstützen das verschieben ihres Inhaltes, wenn man in den Bereich der Anzeige klickt und dabei in die Maus in die Richtung des verborgenen Inhalts zieht. Ein Beispiel ist der Bild Datentyp)

Speichern Gadget: Wenn Sie diese Option aktivieren, werden Sie einen zweiten Schalter unter dem Datatypes-Feld kriegen, der mit einem S markiert sein wird. Wenn Sie diesen anwählen, wird ein Filerequester erscheinen, der Sie auffordert, eine Datei auszuwählen, in dem die gerade angezeigten Daten im IFF-Format gespeichert werden.

Dateinamen anzeigen: Wenn diese Option aktiv ist, wird der Dateiname am unteren Rand des Feldes angezeigt. Wenn sie sie ausschalten, können Sie den Inhalt des Feldes nicht verändern.

Rahmen: Wenn diese Option aktiv ist, wird Fiasco einen Rahmen um das Feld zeichnen. Deaktivieren sie diese Option nicht zu oft, da es dann keine sichtbaren Elemente gibt, die das Feld abgrenzen.

Laden Verzögern: Wenn Sie diese Option aktivieren, wird die Datei nicht sofort geladen, wenn der Record aktiviert wurden. Stattdessen wird der Text Deferred in dem Feld angezeigt werden. Nur wenn Sie das String-Gadget aktivieren und Return drücken, werden die Daten geladen und angezeigt werden.

Sofort Spielen: Wählen Sie diese Option an, um das Spielen der Daten sofort nach dem aktivieren des Records zu starten. Wenn diese Option aktiv ist, darf **Laden Verzögern** nicht aktiv sein. Dieses Attribut hat natürlich keine Auswirkungen, wenn der Datentyp ein Abspielen unterstützt. Zur Zeit unterstützen der Animations- und der Sound-Datentyp abspielen.

Suchäquivalent:

Entspricht dem Dateinamen; Inhaltsbezogene Suche ist nicht möglich.

Umwandlung in ein Datatypes-Feld:

Alle Feldtypen können in Datatypes-Felder ohne Verlust von Daten umgewandelt werden. Sie das Datatypes-System benötigt jedoch korrekte Dateinamen.

Zusätzliche Bemerkungen: Boolean - TRUE "Gesetzt" wird TRUE(1), sonst FALSE(0)
Cycle - Label (Labelnummer) umgewandelt

Der AmigaGuide- und der Animations-Datentyp scheinen mit relativ kleinen Feldern nicht zurechtzukommen.

Weil die Daten bei jedem Wechsel des Records neu geladen werden müssen, kann sich dieser Vorgang verlangsamen. Um dies zu vermeiden, benutzen Sie **Laden Verzögern**!

9.2.11 Der Text-Feldtyp

Der Text-Feldtyp ist eigentlich gar kein echtes Feld, er dient lediglich zur Beschriftung der Maske.

Unterstützte Attribute:

Text: ist der Text, der in die Maske geschrieben wird.

Pen: gibt die Farbe an, in der der Text geschrieben werden soll. Es sind möglich: Normal (normalerweise Schwarz), Highlight (normalerweise Weiß). Die Farben lassen sich über den Palette-Preferences-Editor beeinflussen.

Bold: Ist dieses Attribut aktiv, wird der Text Fett geschrieben.

Italics: Läßt den Text kursiv schreiben.

Underlined: Läßt den Text unterstreichen.

Es werden keine Standard-Attribute unterstützt!

Suchäquivalent:

Nach einem Text-Feld kann aufgrund Inhaltfreiheit nicht gesucht werden.

Umwandlung in ein Text-Feld:

Felder können nicht in Text-Felder umgewandelt werden.

9.2.12 Der Button-Feldtyp

Buttons sind keine echten Felder, sie dienen nur dazu, einen Schalter in die Maske zu setzen, mit dem eine benutzerdefinierbare Aktion gestartet werden kann.

Unterstützte Attribute:

Text: wird im Button angezeigt.

Type: Wählen Sie hier aus, ob der Button ein CLI- oder ein ARexx-Programm ausführen soll. CLI-Programme können normale Programme, Kommandos oder Scripts (mit dem "s"-Attribut) sein. ARexx-Programme müssen ARexx Scripts sein.

Command: Wählen Sie hier das Programm aus, das bei Betätigung des Buttons ausgeführt werden soll.

Stack: Hier können Sie die Stack-Größe für das Programm angeben. Der Standard ist 4096. Wenn Sie zu wenig Stack für ein Programm angeben, wird es abstürzen.

Consolen-Fenster: erlaubt es Ihnen, einen I/O-Strom für das Programm zu definieren. Dies kann ein Konsolen-Fenster (CON:), der Drucker (PRT:), eine einfache Datei, oder, wenn Sie keinerlei Ausgabe wünschen NIL: sein.

Der Button-Feldtyp unterstützt nur das Breiten-Standard Attribut.

Suchäquivalent:

Nach einem Button-Feld kann nicht gesucht werden.

Umwandlung in ein Button-Feld:

Felder können nicht in Button-Felder umgewandelt werden.

Kapitel 10

Benutzeroberfläche von Fiasco

Das einzige, womit Fiasco sich normalerweise nach dem Start präsentiert, ist ein leeres Fenster. Durch Pull-Down-Menüs kann man nun dem Programm seine Wünsche mitteilen. Für diejenigen, die Menüs nicht so mögen, können über den Menüpunkt **Kontrolle/ServiceFenster** ein weiteres Fenster öffnen, mit dem sich die meisten Aktionen auch ohne Menü ausführen lassen. Die dritte Möglichkeit, mit Fiasco zu kommunizieren, sind Tastatur-Shortcuts.

Mit der Maus direkt läßt sich im Masken-Modus der Cursor positionieren, und Felder verschieben. Mit einem Doppelklick auf ein Feld wird der Feld-Requester geöffnet, mit dem sich die Feld-Attribute verändern lassen. Dies entspricht dem Menüpunkt **Feld/Ändern....**

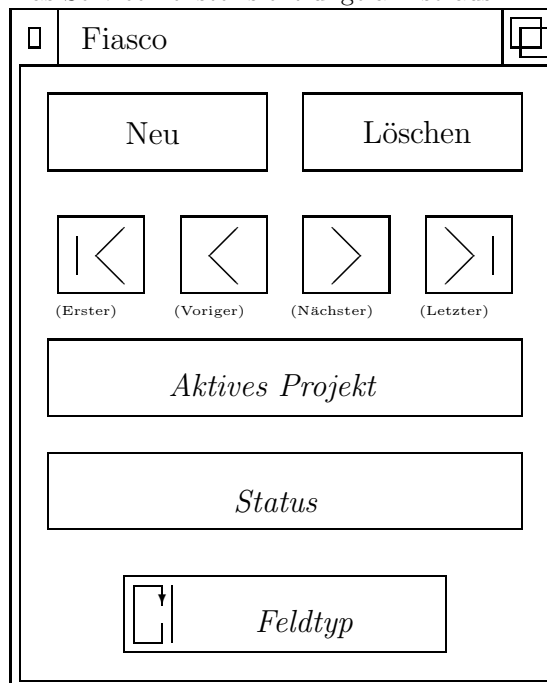
Fiasco unterstützt Menuehelp. Das bedeutet, daß wenn man über einem Menüitem die Help-Taste drückt, eine Beschreibung der Funktion in einem separatem AmigaGuide-Fenster angezeigt wird. (Dieses Feature benötigt die `amigaguide.library`, die Teil des OS seit Release 3.0 ist. Wenn Sie noch 2.0 oder 2.1 benutzen, können Sie die Library von der PD kriegen).

Die Requester, die Fiasco benutzt, haben alle eine grundlegende Struktur. Am unteren Rand befinden sich die Gadgets, mit denen man den Requester bestätigt. Meistens sind dies **Ok**, was die Einstellungen im Requester ausführt und **Cancel**, was die Einstellungen im Requester nicht gültig macht. **Ok** kann mit der Return-Taste abgekürzt werden und **Cancel** kann mit **Esc** abgekürzt werden. Das Closegadget oben links tut das selbe wie **Cancel** oder **Esc**.

10.1 Service-Fenster

Das Service-Fenster läßt sich über den Menüpunkt **Kontrolle/ServiceFenster** öffnen bzw. schließen. Wenn Sie wollen, daß Fiasco bei jedem Programmstart das Service-Fenster automatisch öffnet, brauchen Sie nur **Einstellungen/Auto-Öffnen ServiceWin** (siehe Abschnitt 10.2.64) aktivieren. Wenn Sie **Einstellungen/ServiceWin** dynamisch (siehe Abschnitt 10.2.65) aktivieren sucht Fiasco dann immer nach einem freien Platz auf dem Bildschirm, um das Fenster zu öffnen. Sonst wird die Position des Fensters beim Speichern der Einstellungen benutzt.

Das Service-Fenster sieht ungefähr so aus:



Die Schalter des Service-Fensters sind im Folgenden beschrieben.

10.1.1 Neu

Falls das aktuelle Projekt sich im Record-Modus befindet, wird ein neuer Record erstellt. Ist der Modus der Masken-Modus, wird ein neues Feld des eingestellten Feld-Typs an der aktuellen Cursorposition erstellt.

Äquivalent zu dem Menüpunkt:

Records/Hinzufügen im Record-Modus

bzw.
Felder/Feld hinzufügen... im Masken-Modus

10.1.2 Löschen

Falls das aktuelle Projekt im Record-Modus ist, wird der aktuelle Record gelöscht. Ist der Masken-Modus aktiv, wird das aktive Feld gelöscht (falls keins aktiv ist, wird auch keins gelöscht). *Achtung:* Das Löschen geschieht normaler- weise ohne Sicherheitsabfrage!

Äquivalent zu dem Menüpunkt:
Records/Löschen im Record-Modus
bzw.
Felder/Feld Entfernen im Masken-Modus

10.1.3 Erster

Falls das aktuelle Projekt im Record-Modus ist, wird der Record Nr.1 aktiviert.

Äquivalent zu dem Menüpunkt:
Record/Erster

10.1.4 Voriger

Falls das aktuelle Projekt im Record-Modus ist, wird der dem aktiven Record vorausgehende Record aktiviert.

Äquivalent zu dem Menüpunkt:
Record/Voriger

10.1.5 Nächster

Falls das aktuelle Projekt im Record-Modus ist, wird der dem aktiven Record folgende Record aktiviert.

Äquivalent zu dem Menüpunkt:
Record/Nächster

10.1.6 Letzter

Falls das aktuelle Projekt im Record-Modus ist, wird der letzte Record aktiviert.

Äquivalent zu dem Menüpunkt:
Record/Letzter

10.1.7 Aktives Projekt

Hier wird der Name des aktuellen Projektes angezeigt. Achtung: Sollten sich zwei Projekte nur im Pfad, nicht im Namen voneinander unterscheiden, wird beidesmal dasselbe angezeigt.

Ein anderes Projekt aktiviert man, indem man das jeweilige Fenster aktiviert.

10.1.8 Status

Hier werden Statusinformationen zum aktiven Projekt angezeigt.

Im Record-Modus ist dies:

Nummer des aktiven Records/Anzahl aller Records

Die Zahlen können variieren, falls ein Filter aktiv ist.

Im Mask-Modus wird folgendes angezeigt:

X: *X-Position des Cursors*, Y: *Y-Position des Cursors*

10.1.9 Feldtyp

Hier stellt man ein, was für ein Feldtyp beim nächsten Erzeugen eines Feldes erzeugt wird. Äquivalent mit **Felder/Feldtyp**.

10.2 Menus

Die wichtigste Schnittstelle zu den Funktionen von Fiasco sind die Pull-Down-Menüs der Projekt-Fenster von Fiasco. Die Funktionen aller Menüpunkte (von Links nach Rechts) sind im Anschluß beschrieben.

10.2.1 Projekt/Neu (Project/New)

Shortcut: A N

Erzeugt ein neues Projekt mit dazugehörigem Masken-Fenster. Es enthält dann weder Records noch Felder. Nun können sie entweder eine neue Datenbank erstellen, oder mit **Open** eine bereits existierende laden.

Querverweise: **Open**

10.2.2 Projekt/Leeren (Project/Erase)

Shortcut: A Z

Löscht jegliche Daten aus dem aktuellen Projekt, das Projekt wird hinterher in einem Zustand wie nach dem Programmstart oder **Projekt/Neu** sein. Falls die Daten des Projektes seit dem letzten Abspeichern verändert wurden, wird ein vorher ein Sicherheitsrequester angezeigt.

10.2.3 Projekt/Öffnen... (Project/Open)

Shortcut: A O

Öffnet einen ASL-Requester, mit dem man ein Fiasco-Projekt zum Laden auswählen kann. Die Daten werden im aktiven Fenster angezeigt. Ein eventuell bereits in diesem Fenster geladenes Projekt wird aus dem Speicher entfernt. Um versehentlichen Datenverlust zu vermeiden, werden Sie gefragt, ob sie wirklich laden wollen, falls das Projekt seit dem letzten Speichern verändert wurde.

Wenn Amiga OS 3.0 verfügbar ist, wird Fiasco die Größe des Dateipuffers erhöhen. Dies vergrößert die Ladegeschwindigkeit merkbar.

10.2.4 Projekt/Optionen... (Project/Options)

Shortcut: A \$

Über diesen Menüpunkt können Sie den Options-Requester öffnen, in dem sie projektspezifische Optionen einstellen können. Dies sind:

- Maskendehnung (siehe Abschnitt 4.5)
- Name des Autors sowie Anmerkungen
- Dateiname des Projektes

Der letzte Punkt ermöglicht es, ohne **Save As** aufrufen zu müssen, den Namen des Projektes zu ändern.

10.2.5 Projekt/Statistik... (Project/Statistic)

no Shortcut

Gibt ein paar Informationen zum aktuellen Projekt aus. Beispiel:

Ein Record benötigt ca. 100 Byte RAM.

200 Records dieses Records belegen ca. 19 KByte RAM.

Der Speicher reicht für ca. 2300 weitere Records.

Der Speicherplatzbedarf für ein Projekt mit Feldern an sich wird nicht mitgerechnet.

10.2.6 Projekt/Relationen neuladen (Project/Reload Rels)

Shortcut: A !

Hier kann man Fiasco instruieren, alle Relationen im aktuellen Projekt neu zu laden, wie es bereits beim Laden des Projektes geschehen ist. Dies ist insbesondere für diejenigen nützlich, die den Menüpunkt **Settings/Update Relations?** (siehe Abschnitt 10.2.61) deaktiviert haben, einige Schlüsselworte neu eingegeben bzw. verändert haben und nun die Ergebnisse sehen wollen.

10.2.7 Projekt/Speichern (Project/Save)

Shortcut: A S

Mit Save werden die Daten des aktuellen Projektes unter dem selben Namen gespeichert, mit dem es auch geladen wurde. Falls Sie das Projekt unter einem anderen Namen speichern möchten, wählen sie **Speichern Als** (siehe Abschnitt 10.2.8), oder **Optionen** (siehe Abschnitt 10.2.4) um den Namen zu ändern.

Wenn Amiga OS 3.0 verfügbar ist, wird Fiasco die Größe des Dateipuffers erhöhen. Dies vergrößert die Schreibgeschwindigkeit merkbar.

10.2.8 Projekt/Speichern als... (Project/Save As)

Shortcut: A A

Hier können Sie das aktuelle Projekt unter einem anderen Namen als dem Namen, unter dem Sie es geladen haben, speichern. Der Name wird mit einem Requester der ASL-Library erfragt und nach dem Speichern beibehalten.

Wenn Amiga OS 3.0 verfügbar ist, wird Fiasco die Größe des Dateipuffers erhöhen. Dies vergrößert die Schreibgeschwindigkeit merkbar.

10.2.9 Projekt/Importieren... (Project/Import)

Shortcut: A I

Öffnet den Import-Requester (siehe Abschnitt 10.3.15), die GUI-Schnittstelle der Import-Funktion von Fiasco. Sie können die Import-Funktion benutzen, um Daten aus fremden Datenbank-Programmen in Fiasco einzulesen.

10.2.10 Projekt/Exportieren... (Project/Export)

Shortcut: A E

Öffnet den Export-Requester (siehe Abschnitt 10.3.16), die GUI-Schnittstelle zur Export-Funktion von Fiasco. Die Export-Funktion dient zum Speichern von Fiasco-Daten in einem Format, das von anderen Datenbank-Programmen gelesen werden kann.

10.2.11 Projekt/Über Fiasco... (Project/About)

Shortcut: A ?

Hiermit wird ein kleiner Requester angezeigt, der Auskunft über Version, Copyright und ein paar Systeminternia gibt.

10.2.12 Projekt/Beenden (Project/Quit)

Shortcut: A Q

Dieser Menüpunkt schließt das aktive Projekt. Falls es in der Zwischenzeit verändert und noch nicht gespeichert wurde, werden Sie über einen Requester gefragt, ob sie wirklich dieses Projekt schließen möchten. Falls dies das letzte Projekt war, das Fiasco offen hatte, wird Fiasco beendet.

10.2.13 Record/Hinzufügen (Record/Add Record)

Shortcut: A +

Fügt der Record-Liste des aktuellen Projektes einen neuen Record hinzu. Die einzelnen Felder enthalten nun entweder nichts, oder falls dies im Masken-Modus so eingestellt wurde, einen Startwert. Falls das Listen-Fenster offen ist, erscheint dort auch ein entsprechender Eintrag.

Falls ein Filter aktiv ist, wird der Record automatisch für gültig erklärt, egal was in den einzelnen Feldern steht. Damit neuerstellte Records korrekt in den Filter miteinbezogen werden, sollte man **Vergleichen/Filter Ändern** anwählen und einfach **Ok** anklicken.

Dieser Menüpunkt ist nur im Record-Modus anwählbar.

Querverweise: Record/Löschen

10.2.14 Record/Duplizieren (Record/Duplicate Record)

Shortcut: A 2

Kopiert den aktuellen Record auf Haar genau in einen neuen Record. Jegliche Startwert-Vorgaben werden ignoriert. Selbst ein Feld, das das Attribut Eindeutiger Schlüssel gesetzt hat, enthält denselben Wert wie der alte Record, wodurch die Schlüssel nicht mehr eindeutig sind.

10.2.15 Record/Löschen (Record/Delete Record)

Shortcut: A -

Löscht den aktiven Record und die darin befindlichen Daten. Falls sich noch Relationen in anderen Dateien auf diesen Record beziehen, sind diese ab sofort nicht definiert.

Dieser Menüpunkt ist nur im Record-Modus anwählbar.

Falls der Menüpunkt Settings/Security-Requester? (siehe Abschnitt 10.2.63) aktiv ist, erscheint zuvor noch eine Sicherheitsabfrage, ob dieser Record wirklich gelöscht werden soll.

Querverweise: Record/Hinzufügen, Record/Alle löschen

10.2.16 Record/Alle löschen (Record/Delete all Records)

Shortcut: A

noindentLöscht alle Records des aktiven Projektes. Die Maske wird von dieser Funktion nicht angetastet.

Record/Alle Löschen darf nur im Record-Mode angewählt werden.

Querverweise: Record/Löschen

10.2.17 Record/Ausschneiden (Record/Cut Record)

Shortcut: A X

Kopiert den aktuellen Records ins Clipboard und entfernt den Records aus der Record-Liste des Projektes. Nachdem kann der Record mit Record/Einfügen (siehe Abschnitt 10.2.19) wieder in das Projekt eingefügt werden.

Diese Funktion kann nur im Record-Modus aufgerufen werden.

Querverweise: Record/Kopieren, Record/Einfügen, Sektion Clipboard-Unterstützung von Fiasco

10.2.18 Record/Kopieren (Record/Copy Record)

Shortcut: A C

Kopiert den aktuellen Record ins Clipboard. Danach können Sie mit **Record/Einfügen** den Record wieder in ein Projekt einfügen.

Diese Funktion darf nur im Record-Modus aufgerufen werden.

Querverweise: **Record/Ausschneiden**, **Record/Einfügen**, Sektion **Clipboard-Unterstützung von Fiasco**

10.2.19 Record/Einfügen (Record/Paste Record)

Shortcut: A P

Erzeugt einen neuen Record und fügt den Inhalt des Clipboard in diesen Record ein. Normalerweise sollten Sie **Record/Ausschneiden** (siehe Abschnitt 10.2.17) oder **Record/Kopieren** (siehe Abschnitt 10.2.18) aufrufen, bevor Sie diese Funktion benutzen.

Diese Funktion kann nur im Record-Modus aufgerufen werden.

Querverweise: **Record/Ausschneiden**, **Record/Kopieren**, Sektion **Clipboard-Unterstützung von Fiasco**

10.2.20 Record/Voriger (Record/Previous)

Shortcut: Cursor up

Aktiviert den dem aktuellen Record *vorgehenden* Record. Falls der aktuelle Record der erste in der Liste ist, wird die Anzeige “gebebt”. Bitte beachten sie, daß beim Einsatz von Filtern der nächste vorhergehende Record, der *nicht ausgefiltert* wird, aktiviert wird.

Der Tastatur-Shortcut wurde in Hinblick auf die Darstellung in der Liste gewählt, in der der vorgehende Record über dem aktuellen liegt.

Dieser Menüpunkt ist nur im Record-Modus anwählbar.

Querverweise: **Nächster**, **Erster**, **Letzter**, **Gehe zu**, **Rückwärts suchen**

10.2.21 Record/Nächster (Record/Next)

Shortcut: Cursor down

Aktiviert den Record nach dem *aktuellen* Record. Falls der aktuelle Record der letzte in der Liste ist, wird die Anzeige “gebebt”. Bitte beachten sie, daß beim Einsatz von Filtern der nächste Record, der *nicht ausgefiltert* wird, aktiviert wird.

Der Tastatur-Shortcut wurde in Hinblick auf die Darstellung in der Liste gewählt, in der der nächste Record unter dem aktuellen liegt.

Dieser Menüpunkt ist nur im Record-Modus anwählbar.

Querverweise: Voriger, Erster, Letzter, Gehe zu, Find next

10.2.22 Record/Erster (Record/First Record)

Shortcut: Ctrl Cursor up

Aktiviert den *ersten* Record des aktuellen Projektes. Bitte beachten Sie, daß durch einen Filter dieses Verhalten verändert wird. In so einem Fall wird der erste Eintrag aktiviert, der die Kriterien des Filters *erfüllt*.

Dieser Menüpunkt ist nur im Record-Modus anwählbar.

Querverweise: Nächster, Voriger, Letzter, Gehe zu

10.2.23 Record/Letzter (Record/Last Record)

Shortcut: Ctrl Cursor down

Aktiviert den *letzten* Record des aktuellen Projektes. Bitte beachten Sie, daß durch einen Filter dieses Verhalten verändert wird. In so einem Fall wird der letztmögliche Eintrag aktiviert, der die Kriterien des Filters *erfüllt*.

Dieser Menüpunkt ist nur im Record-Modus anwählbar.

Querverweise: Nächster, Voriger, Erster, Gehe zu

10.2.24 Record/Gehe zu... (Record/Goto)

Shortcut: A G

Öffnet den Goto-Requester mit dem man über die Nummer eines Records diesen erreichen kann. Bitte beachten Sie, daß die Nummern sich durch Einfügen oder Löschen von Records oder durch Filter die Nummern verändern können.

Dieser Menüpunkt ist nur im Record-Modus anwählbar.

Querverweise: Nächster, Voriger, Erster, Letzter

10.2.25 Record/Markiere Record (Record/Mark Record)

Shortcut: A .

Markiert den aktuellen Record. Wenn ein Record markiert ist, wird er in

der Liste hervorgehoben angezeigt und das Zeichen “M” wird im Service-Fenster angezeigt, wenn dieser Record aktiv ist.

Dieser Menüpunkt ist nur im Record-Modus anwählbar.

Querverweise: Lösche Markierung, Markiere alle Records, Lösche alle Markierungen

10.2.26 Record/Lösche Markierung (Record/Unmark Record)

Shortcut: A :

Entfernt die Markierung des aktuellen Records. Der Record wird danach wieder normal dargestellt.

Dieser Menüpunkt kann nur im Record-Modus angewählt werden.

Querverweise: Markiere Record, Markiere alle Records, Lösche alle Markierungen

10.2.27 Record/Markiere alle Records (Record/Mark all Records)

Shortcut: A ,

Markiert alle Records des aktiven Projektes. Bitte beachten Sie, daß die alten Markierungen der Records überschrieben werden.

Dieser Menüpunkt kann nur im Record-Modus angewählt werden.

Querverweise: Markiere Record, Lösche Markierung, Lösche alle Markierungen, Markierungen umschalten

10.2.28 Record/Lösche alle Markierungen (Record/Unmark all Records)

Shortcut: A ;

Löscht die Markierungen aller Records im aktiven Projekt. Bitte beachten Sie, daß die alten Markierungen der Records dabei verlorengehen.

Dieser Menüpunkt kann nur im Record-Modus angewählt werden.

Querverweise: Markiere Record, Lösche alle Markierungen, Markiere alle Records, Markierungen umschalten

10.2.29 Record/Markierungen umschalten (Record/Toggle all Marks)

Kein Shortcut

Schaltet die Markierungen aller Records des aktiven Projektes um. Das heist, daß ein markierter Record ein nicht markierter wird und ein nicht markierter wird ein markierter. Die ursprünglichen Markierungen kann man durch wiederholtes anwählen dieser Funktion wiederherstellen.

Dieser Menupunkt kann nur im Record-Modus angewählt werden.

Querverweise: Markiere Record, Lösche Markierung, Markiere alle Records, Lösche alle Markierungen

10.2.30 Feld/Feldtyp (Field/Fieldtype)

Wählen Sie in diesem Untermenu den aktuellen Feldtypen aus. Dieser wird beim Erstellen von Feldern benutzt. Das Gadget am unteren Rand des Service Fensters hat die selbe Funktion.

Diese Feldtypen sind verfügbar (mit Shortcut):

String	Ctrl S
Integer	Ctrl I
Float	Ctrl F
Boolean	Ctrl B
Cycle	Ctrl C
Slider	Ctrl S
Datum	Ctrl A
Zeit	Ctrl M
Extern	Ctrl E
Datatypes	Ctrl D
Text	Ctrl T
Button	Ctrl U

10.2.31 Feld/Feld Hinzufügen... (Field/Add Field)

Shortcut: Return

Öffnet den Feld-Requester (siehe Abschnitt 10.3.1) für den aktiven Feldtyp und fügt das so erstellte Feld an der Cursorposition ein.

Dieser Menupunkt ist nur im Masken-Modus anwählbar.

Falls sich an der Cursorposition schon ein Feld befindet, läßt sich kein Feld erzeugen.

Bitte beachten sie, daß Return auch Shortcut für Feld Ändern (siehe Abschnitt 10.2.32) ist. Return hat die Funktion Feld Hinzufügen, wenn kein Feld aktiv ist, ansonsten wird Feld Ändern aufgerufen.

Querverweise: Feld Ändern, Relation Ändern, Feld Entfernen

10.2.32 Feld/Feld Ändern... (Field/Edit Field)

Shortcut: Return

Öffnet den für den aktiven Feld-Typ und fügt das so erstellte Feld an der Cursorposition ein. Dieser Menüpunkt ist nur im anwählbar. Falls sich an der Cursorposition schon ein Feld befindet, läßt sich kein Feld erzeugen.

Bitte beachten sie, daß Return auch Shortcut für Feld Ändern ist. Enter hat die Funktion Feld Hinzufügen, wenn kein Feld aktiv ist, ansonsten wird Feld Ändern aufgerufen.

Querverweise: Edit Field... Edit Relations... Remove Gadget

10.2.33 Feld/Feld Duplizieren (Field/Duplicate Field)

Kein Shortcut

Macht eine exakte Kopie des aktiven Feldes. Es wird so nah wie möglich am Original erzeugt. Die ID wird `copy_of_FeldID` sein.

10.2.34 Feld/Feld Entfernen (Field/Remove Field)

Shortcut: Del

Entfernt das aktive Feld aus der Maske und somit auch alle damit verbundenen Daten. Falls sich noch irgendwelche Relationen auf dieses Feld verlassen, werden diese nicht funktionsfähig sein. *Achtung:* Relationen oder ARexx-Scripts werden sich erst beim nächsten Zugriff über diesen Zustand beschweren!

Dieser Menüpunkt ist nur im Masken-Modus anwählbar.

Querverweise: Feld Ändern, Relation Ändern, Feld Ändern

10.2.35 Feld/Relation Ändern... (Field/Edit Relation)

Shortcut: A &

Über diesen Punkt können sie die Relationen (siehe Abschnitt 7.1) für das aktive Feld im Relations-Requester (siehe Abschnitt 10.3.12) einstellen.

Relation Ändern ist nur anwählbar, wenn das Projekt im Masken-Modus ist.

Querverweise: Relationen entfernen

10.2.36 Feld/Relation entfernen (Field/Remove Relation)

Shortcut: A 0

Dieser Menupunkt löscht alle Relations-Informationen des gerade aktiven Feldes. Danach werden die Daten in diesem Feld wieder in die normale Datei geschrieben werden.

Dieser Menupunkt kann nur im Masken-Modus angewählt werden.

10.2.37 Feld/Feld Konvertieren... (Field/Convert Field)

Shortcut: A "

Öffnet den Umwandlungs-Requester (siehe Abschnitt 10.3.2) für das aktive Feld. Mit Feld Konvertieren kann man den Typ eines Feldes auf einfache Weise ändern.

Dieser Menupunkt kann nur im Masken-Modus angewählt werden.

Querverweise: Feld Hinzufügen, Feld Ändern

10.2.38 Liste/Spalte verbergen (List/Hide column)

Shortcut: A [

Über diesen Menupunkt kann man eine Spalte aus dem Listenfenster entfernen. Es wird die aktuelle Spalte, die über den Listenkopf ausgewählt werden kann, entfernt. Die Spalten rechts davon rücken danach auf. Die Spalte kann über Spalte sichtbar machen (siehe Abschnitt 10.2.39) wieder in die Liste eingefügt werden.

Dieser Menupunkt kann sowohl im Record- als auch im Masken-Modus aufgerufen werden, wenn das Listen-Fenster offen ist.

10.2.39 Liste/Spalte sichtbar machen... (List/Show column)

Shortcut: A]

Dieser Menupunkt ermöglicht es, die Spalten, die mit Spalte verbergen aus

der Liste entfernt wurden, wieder in die Liste einzufügen. Fiasco versucht, die in einem Requester ausgewählte Spalte so nahe wie möglich an der alten Position wieder einzufügen.

Dieser Menupunkt kann sowohl im Record- als auch im Masken-Modus aufgerufen werden, wenn das Listen-Fenster offen ist.

10.2.40 Liste/Alle Spalten sichtbar (List/Show all columns)

kein Shortcut

Fügt alle Spalten, die über **Spalte verbergen** aus der Liste entfernt wurden, in einem Schritt wieder ein.

Dieser Menupunkt kann sowohl im Record- als auch im Masken-Modus aufgerufen werden, wenn das Listen-Fenster offen ist.

10.2.41 Liste/Liste neu berechnen (List/Recalc List)

Shortcut: A %

Dieser Menupunkt berechnet alle Positionen und Dimensionen der Spalten in der Liste neu. Verborgene Spalten werden jedoch nicht wieder angezeigt.

Dieser Menupunkt kann mit dem Menupunkt **Inhalt aufräumen** (Clean up) der Workbench verglichen werden.

Dieser Menupunkt kann nur bei offenem Listen-Fenster angewählt werden.

10.2.42 Vergleichen/Suchen... (Compare/Find)

Shortcut: A F

Öffnet den Such-Requester (siehe Abschnitt 10.3.3), in dem man Suchkriterien bestimmen kann.

Dieser Menupunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

Querverweise: Suchrequester, Weitersuchen, rückwärts suchen

10.2.43 Vergleichen/Weitersuchen (Compare/Find next)

Shortcut: A >

Aktiviert den *nächsten* Eintrag, der mit den Suchkriterien übereinstimmt,

die mit dem Such-Requester (siehe Abschnitt 10.3.3) spezifiziert worden sind. Falls kein Eintrag mehr gefunden wird, werden Sie darüber informiert werden.

Dieser Menüpunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

Querverweise: Such-Requester, Suchen, Rückwärts suchen

10.2.44 Suchen/Rückwärts Suchen (Compare/Find previous)

Shortcut: A <

Aktiviert den *vorherigen* Eintrag, der mit den Suchkriterien übereinstimmt, die mit dem Search-Requester (siehe Abschnitt 10.3.3) spezifiziert worden sind. Falls kein Eintrag mehr gefunden wird, werden Sie darüber informiert werden.

Dieser Menüpunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

Querverweise: Such-Requester, Suchen, Weitersuchen

10.2.45 Vergleichen/Ersetzen... (Compare/Replace)

Shortcut: A R

Öffnet den Ersetzen-Requester (siehe Abschnitt 10.3.4), mit dem Parameter für das Ersetzen von Daten angegeben werden können.

Dieser Menüpunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

10.2.46 Vergleichen/Zählen... (Compare/Count)

Shortcut: A #

Öffnet den Count-Requester (siehe Abschnitt 10.3.5), mit dem Sie die Anzahl des Vorliegenden Musters in dem Projekt zählen können.

Dieser Menüpunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

Querverweise: Suchen

10.2.47 Vergleichen/Sortieren... (Compare/Sort)

Shortcut: *A* =

Öffnet den Sortier-Requester (siehe Abschnitt 10.3.6), in dem Sie diverse Sortierkriterien angeben und den Sortiervorgang starten können.

Dieser Menüpunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

10.2.48 Vergleichen/Filter Ändern... (Compare/Edit Filter)

Shortcut: *A*

Öffnet den Filter-Requester (siehe Abschnitt 10.3.7), mit dem man einen Filter erstellen kann.

Dieser Menüpunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

10.2.49 Vergleichen/Filter An? (Compare/Use Filter?)

Shortcut: *A* ‘ (Graue Taste oben links)

Über diesen Menüpunkt kann man den aktuellen Filter ein und ausschalten. Falls noch kein Filter erstellt worden ist, wird der Filter-Requester (siehe Abschnitt 10.3.7) geöffnet.

Dieser Menüpunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

10.2.50 Vergleichen/Markieren... (Compare/Mark)

Shortcut: *A* *K*

Öffnet den Markieren-Requester (siehe Abschnitt 10.3.8), der zum Markieren von Records benutzt werden kann, die auf ein Suchmuster passen. Dies funktioniert ähnlich wie die Erstellung von Filtern.

Existierende Markierungen werden überschrieben; Markierte Records werden demarkiert, wenn sie nicht zu einem Muster passen.

Dieser Menüpunkt ist nur anwählbar, wenn sich das aktuelle Projekt im Record-Modus befindet und mindestens einen Record enthält.

10.2.51 Vergleichen/Filter zu Markierungen (Compare/Filter to Marks)

Kein Shortcut

Wandelt den aktuellen Filter (siehe Abschnitt 6.6) (egal, ob aktiv or inaktiv) in Markierungen um. Records, die auf den Filter passen werden markiert und Records, die ausgefiltert werden, werden demarkiert. Wenn der Filter aktiv ist, wird er deaktiviert.

Querverweise: Filter Ändern, Markierungen zu Filter

10.2.52 Vergleichen/Markierungen zu Filter (Compare/Marks to Filter)

Kein Shortcut

Wandelt die Markierungen des aktiven Projektes in einen Filter (siehe Abschnitt 6.6) um. Jeder markierte Record wird als ein für den Filter passender Record deklariert und jeder Record, der nicht markiert ist wird ausgefiltert. Dieser Filter wird nicht in den Filter-Requester übertragen. Wenn Sie den Filter-Requester öffnen und Ok anklicken, wird der vorige Filter erzeugt.

Querverweise: Filter zu Markierungen

10.2.53 Kontrolle/Record-Modus (Control/Record Mode)

Shortcut: A D

Hiermit können Sie das aktuelle Projekt in den Record-Modus (siehe Abschnitt 4.6.1) versetzen, in dem sie Records und deren Inhalt verändern können. Wenn dieser Modus aktiv ist, wird ein Haken vor den Menüpunkt gesetzt. Das Gegenstück zu diesem Punkt ist Kontrolle/Masken-Modus (siehe Abschnitt 10.2.54).

Querverweise: Record-Modus, Masken-Modus

10.2.54 Kontrolle/Masken-Modus (Control/Mask Mode)

Shortcut: A M

Hiermit können Sie das aktuelle Projekt in den Masken-Modus (siehe Abschnitt 4.6.2)

versetzen, in dem sie die Maske verändern können. Wenn dieser Modus aktiv ist, wird ein Haken vor den Menüpunkt gesetzt. Das Gegenstück zu diesem Punkt ist .

Querverweise: Masken-Modus, Record-Modus

10.2.55 Kontrolle/Service-Fenster (Control/ServiceWindow)

Shortcut: A W

Mit diesem Menüpunkt können sie das Service-Fenster (siehe Abschnitt 10.1) öffnen und schließen. Das Service-Fenster erleichtert die wichtigsten Record- und Maskenoperationen und zeigt Statusmeldungen an. Das Fenster kann auch über das Close-Gadget geschlossen werden.

Das Service-Fenster ist global für alle Projekte zuständig.

10.2.56 Kontrolle/Listen-Fenster (Control/ListWindow)

Shortcut: A L

Mit diesem Menüpunkt können sie das Listen-Fenster (siehe Abschnitt 4.4) öffnen und schließen. Das Fenster kann auch über das Close-Gadget geschlossen werden.

Jedes Projekt kann seine eigene Liste haben.

10.2.57 Kontrolle/ARexx-Debug (Control/ARexx-Debug)

Shortcut: A B

Hier kann man den speziellen Debug-Modus für die ARexx-Schnittstelle (siehe Abschnitt 11) von Fiasco aktivieren. Wenn ARexx-Kommandos von Fiasco Fehler erzeugen, werden zusätzlich Requester angezeigt, die das Problem näher bezeichnen.

10.2.58 Einstellungen/Icons Erzeugen? (Settings/Create Icons?)

Hier wird festgelegt, ob beim Speichern von Fiasco-Projekten Icons erzeugt werden, oder nicht.

10.2.59 Einstellungen/Backups Erzeugen? (Settings/Create Backups?)

Hier wird festgelegt, ob beim Speichern von Fiasco-Projekten von den alten Dateien Backups angelegt werden sollen. Die Backup-Datei wird dann den Namen *Name.bak* haben.

10.2.60 Einstellungen/Relationen schreiben? (Settings/Write Relations?)

Hier kann man einstellen, ob beim Speichern von Projekten auch Relationen in die entsprechenden Dateien, aus denen sie auch gelesen wurden, abgespeichert werden. Ansonsten sind Änderungen, die in solchen Feldern gemacht wurden, verloren. Dieser Menüpunkt sollte nur aktiv sein, wenn auch **Relationen aktualisieren?** (siehe Abschnitt 10.2.61) aktiv ist, oder die Relationen vor dem Speichern mit **Projekt/Relationen neuladen** (siehe Abschnitt 10.2.6) aktualisiert wurden, ansonsten könnten im Projekt, in dem die Relationen im Klartext stehen, manche Einträge durch Leerstrings oder falsche Einträge ersetzt werden.

10.2.61 Einstellungen/Relationen aktualisieren? (Settings/Update Rels?)

Hier kann man steuern, ob bei Eingabe eines neuen Schlüsselwortes in ein Relations-Feld die Daten sofort erneuert werden sollen. Dazu sind Diskettenzugriffe nötig, die für Menschen, die keine Festplatte haben, leicht nervig werden können. Falls man diesen Menüpunkt deaktiviert, sollte man auch **Relationen schreiben?** (siehe Abschnitt 10.2.60) deaktivieren, damit keine falschen Daten in ein anderes Projekt geschrieben werden. Um die Änderungen so zu betrachten, kann man die Relationen in einem Abwasch über **Projekt/Relationen neuladen** (siehe Abschnitt 10.2.6) aktualisieren.

10.2.62 Einstellungen/Als Pattern * benutzen? (Settings/Use * as Pattern?)

Aktivieren Sie diesen Punkt, wenn Sie die Unterstützung des Sterns als ein Suchmuster aktivieren wollen. Der * hat dann die selbe Bedeutung wie #?.

10.2.63 Einstellungen/Sicherheits-Requester? (Settings/Security-Reqs?)

Hier kann man einstellen, ob vor dem Löschen von Feldern oder Records noch eine Bestätigung benötigt wird, um Datenverlust zu vermeiden.

10.2.64 Einstellungen/Auto-Öffnen ServiceWin? (Settings/Auto-Open ServiceWin?)

Hier kann man kontrollieren, ob beim Programmstart das Service-Fenster (siehe Abschnitt 10.1) automatisch geöffnet werden soll.

10.2.65 Einstellungen/ServiceWin dynamisch? (Settings/Dynamic ServiceWin?)

Hier kann man kontrollieren, ob beim Öffnen des Service-Fensters eine freie Stelle gesucht werden soll, oder ob feste Koordinaten benutzt werden sollen. Falls dieser Menüpunkt nicht aktiviert ist, speichert Fiasco beim schreiben der Einstellungen die aktuelle Position des Fensters und aktiviert diese beim Neustart wieder.

10.2.66 Einstellungen/Sprechen? (Settings/Talking?)

Aktivieren Sie diesen Menüpunkt, wenn Sie wollen, daß Fiasco bestimmte Mitteilungen “spricht”.

10.2.67 Einstellungen/Anzeige... (Settings/Display)

Kein Shortcut

Öffnen den Anzeige-Requester (siehe Abschnitt 10.3.14), der zum Spezifizieren von Anzeige-Optionen benutzt werden kann. Sie können hier angeben, ob Fiasco auf einem eigenen Screen oder einem öffentlichen Screen laufen soll. Weiterhin ist hier die Auswahlmöglichkeit für Schriften. Dies ersetzt den Menüpunkt *Einstellungen/Schrift auswählen* von Fiasco 1.0.

10.2.68 Einstellungen/Editor... (Settings/Editor)

Kein Shortcut

Öffnet einen Requester, in dem Sie auswählen können, was für einen Editor Fiasco aufrufen soll, wenn Sie den Script Ändern-Schalter in einem der Feldrequester anklicken.

10.2.69 Einstellungen/Einstellungen speichern (Settings/Save Settings)

Speichert die Einstellungen unter “env:fiasco.prefs” und “envarc:fiasco.prefs”. Sie sind somit auch nach einem Reboot aktiv.

10.2.70 Einstellungen/Einstellungen speichern als... (Settings/Save Settings as)

Öffnet einen ASL-Requester, in dem man eine Datei auswählen kann, in der die Einstellungen gespeichert werden sollen. Wenn Sie die Datei in “env:” speichern, werden die Einstellungen einen Neustart des Systems nicht überstehen. Wenn Sie die Einstellungen in “envarc:” speichern, werden diese erst nach einem Neustart aktiv, da Fiasco nur in “env:” nach seinen Einstellungen sucht.

10.2.71 Einstellungen/Einstellungen laden... (Settings/Load Settings)

Öffnet einen ASL-Requester, in dem man eine Datei auswählen kann, aus der die Einstellungen geladen werden sollen. Sie werden danach benutzt. Um sie dauerhaft benutzen zu können, sollte man sie mit Einstellungen speichern in “envarc:” und “env:” sichern.

10.2.72 Benutzer/Ändern... (User/Edit)

Shortcut: **A U**

Öffnet den UserMenu-Requester (siehe Abschnitt 10.3.9), in dem man das Usermenu verändern kann.

10.3 Alle Requester

Requesters (manchmal auch Dialogfenster genannt) werden von Fiasco benutzt, um Informationen zu bekommen, die für bestimmte Operationen notwendig sind. Normalerweise werde Requester nach dem Aufrufen eines Menüpunktes von Fiasco erzeugt. Sogenannte EasyRequester, mit denen

Fiasco eine einfache Auswahl verlangt, werden hier nicht erklärt, da diese meistens leicht verständlich sind und meistens in den Bereichen der jeweiligen Funktion beschrieben sind.

Die meisten Requester können auch mit der Tastatur kontrolliert werden. Die Shortcuts, die durch einen unterstrichenen Buchstaben gekennzeichnet werden, sind normalerweise einzelne Zeichen ohne Zusatz Tasten (wie Ctrl, etc.)

Die Gadgets am unteren Rand eines Requesters sind normalerweise für das Bestätigen oder sonstiges Fortschreiten in einer Funktion gedacht. Normalerweise ist das linke Gadget eine positive Antwort (Ok), während das rechte eine negative Antwort darstellt (Cancel). Das positive Gadget, das auch eine zusätzliche Umrandung besitzt, hat **Enter** als Shortcut. Das negative Gadget hat **Escape** als Shortcut.

10.3.1 Feld-Requester

Der Feld-Requester erlaubt das Manipulieren der Attribute jedes Feldes. Da jeder Feldtyp andere Attribute hat, unterscheidet sich auch der Feld-Requester von Feldtyp zu Feldtyp. Die Gadgets, die im Feld-Requester dargestellt werden entsprechen genau den Feld-Attributen, die bei den Felddokumentationen definiert sind.

Man wird mit dem Feldrequester konfrontiert, wenn man **Fields/Add Field** oder **Fields/Edit Field** anwählt oder auf einem Feld doppelklickt.

Wenn man den Requester mit **Ok** bestätigt, werden alle Werte auf Gültigkeit überprüft. Falls ein Wert nicht benutzt werden kann, wird man mit einem Requester darauf aufmerksam gemacht.

Hier eine kleine Zusammenfassung aller Bedingungen:(Vorausgesetzt, daß diese Attribute existieren)

- Eine eindeutige ID muß angegeben sein.
- Max. Zeichen muß ≥ 0 sein.
- Breite muß ≥ 2 sein.

Wenn Dimensionswerte aufgrund weiterer Felder, die im Weg sind, nicht benutzt werden können, erscheint ein anderer Requester, der einem die Möglichkeiten **Schieben**, **Quetschen** und **Abbrechen** anbietet. **Cancel** tut gar nichts, außer zum Feld-Requester zurückzukehren. **Quetschen** verkleinert das Feld soweit, daß es paßt. **Schieben** verschiebt das Feld so weit, daß es genug Platz hat. **Schieben** kann nicht immer angewandt werden.

Falls man ein bereits bestehendes Feld verändert, daß die Daten in String-Form speichert und **Max. Zeichen** unterstützt (also String, Datatypes und Extern) und **Max. Zeichen** so weit verkleinert hat, daß bestehende Einträge nicht mehr hinein passen würden, werden Sie gefragt, ob sie diese wirklich auf die neue Länge verkleinern möchten.

10.3.2 Feld umwandeln-Requester

Die Umwandlungs-Requester kann zum Ändern des Types eines Felds benutzt werden, ohne ein ARexx-Script dafür erstellen zu müssen.

Feld ID: Dieses Text-Gadget zeigt die ID des Feldes an, dessen Typ geändert werden soll. Bitte überprüfen Sie hier, ob sie **Feld umwandeln** für das richtige Feld aufgerufen haben.

Alter Typ: Zeigt den aktuellen Typ des Feldes an.

Neuer Typ: Wählen Sie hier den neuen Typ des Feldes aus. Bitte beachten Sie, daß bei bestimmten Änderungen Daten verloren gehen können. Lesen Sie die Feld-Dokumentationen für mehr Informationen über dieses Thema.

Alternatives Format: Aktivieren Sie diese Option, um ein Ausgabe-Format zu erhalten, das von dem normalen Format abweicht. Die Feld-Dokumentationen enthalten mehr Informationen darüber, ob dieses Gadget Auswirkungen hat und welche.

Ok: Startet den Umwandlungs-Vorgang und schließt danach den Requester.

Abbrechen: Schließt den Requester.

10.3.3 Such-Requester

Feld: Hier können Sie das Feld wählen, in dem gesucht werden soll. In dem Listview werden die IDs der Felder angezeigt, Textfelder werden nicht angezeigt. Es kann immer nur ein Feld ausgewählt werden.

Muster: In dieses String-Gadget können sie ein Suchmuster eingeben. Entweder ein einfacher Wert, oder ein Wert mit Mustern. Bitte beachten Sie, daß dieser Wert später auch in den Count und Replace-Requestern benutzt wird.

Unschärfe Suche/Activ: Falls sie eine unscharfe Suche (siehe Abschnitt 6.2) durchführen wollen, muß dieses Gadget aktiviert sein.

Unschärfe Suche/Faktor: Hier können Sie die Toleranz der unscharfen Suchfunktion einstellen. 0 entspricht, daß die Funktion nur exakt übereinstimmende Einträge akzeptiert, 100 entspricht fast allen Einträgen.

Nächstes: Bestätigt den Requester und sucht nach dem nächsten Eintrag, der mit den oben angegebenen Bestimmungen übereinstimmt.

Erstes: Bestätigt den Requester und sucht nach dem ersten Eintrag.

Voriges: Bestätigt den Requester und sucht rückwärts nach dem nächsten Eintrag.

Abbrechen: Schließt den Requester ohne weitere Aktion.

10.3.4 Ersetzen-Requester

Falls Sie sich bereits mit dem Find-Requester auskennen, sollten Sie sich hier *sehr* schnell zurechtfinden.

Feld: Hier können Sie das Feld wählen, in dem gesucht werden soll. In dem Listview werden die IDs der Felder angezeigt, Textfelder werden nicht angezeigt. Es kann immer nur ein Feld ausgewählt werden.

Muster: In dieses String-Gadget können sie ein Suchmuster eingeben. Entweder ein einfacher Wert, oder ein Wert mit Mustern. Bitte beachten Sie, da dieser Wert später auch im Suchrequester benutzt wird, und umgekehrt.

Ersatz: Enter here a value, which will be copied in the matching entries. No patterns are possible. Hier können Sie den Wert eingeben, der in die Einträge eingesetzt werden soll, die vorher mit **Pattern** übereinstimmten. Hier sind keine Muster möglich.

Bestätigen: Wenn Sie wünschen, daß Sie jedesmal gefragt werden, ob ein Eintrag wirklich ersetzt werden soll, sollten sie dieses Gadget aktivieren.

Unschärfe Suche/Aktiv: Falls sie eine unscharfe Suche durchführen wollen, muß dieses Gadget aktiviert sein.

Unschärfe Suche/Faktor: You can control here the tolerance of the blurred search. 0 searches only for exactly matching entries, 100 searches for almost all entries. Hier können Sie die Toleranz der unscharfen Suchfunktion einstellen. 0 entspricht, daß die Funktion keinerlei Einträge akzeptiert, 100 entspricht fast allen Einträgen.

10.3.5 Zähl-Requester

Dieser Requester erlaubt es Ihnen, die Records zu zählen, die einem bestimmten Suchkriterium entsprechen. Mehr zum Zählen im Suchabschnitt(6.4)

Der Requester ist über den Menüpunkt **Vergleichen/Zählen** (siehe Abschnitt 10.2.46) erreichbar.

Falls Sie sich bereits mit dem Such-Requester auskennen, sollten Sie sich hier *sehr* schnell zurechtfinden.

Feld: Hier können Sie das Feld wählen, in dem gesucht werden soll. In dem Listview werden die IDs der Felder angezeigt, Textfelder werden nicht angezeigt. Es kann immer nur ein Feld ausgewählt werden.

Muster: In dieses String-Gadget können sie ein Suchmuster eingeben. Entweder ein einfacher Wert, oder ein Wert mit Mustern. Bitte beachten Sie, daß dieser Wert auch später im Suchrequester benutzt wird und umgekehrt.

Unschärfe Suche/Aktiv: Falls sie eine unscharfe Suche durchführen wollen, muß dieses Gadget aktiviert sein.

Unschärfe Suche/Faktor: Hier können Sie die Toleranz der unscharfen Suchfunktion einstellen. 0 entspricht, daß die Funktion keinerlei Einträge akzeptiert, 100 entspricht fast allen Einträgen.

Ok: Bestätigt den Requester und zählt, wie oft das Suchmuster gefunden wird. Dies wird zum Schluß angezeigt.

Abbrechen: Schließt den Requester ohne jede weitere Aktion.

10.3.6 Sortiere-Requester

Im Sortier-Requester kann man ein Fiasco-Projekt alphabetisch bzw. numerisch sortieren. Er ist über den Menüpunkt **Vergleichen/Sortieren** erreichbar.

Sortieren nach: Hier wird die Liste aller Felder angezeigt. Aus diesen sollte man ein Feld auswählen, nach dem die Suche orientiert wird.

Absteigend: Aktivieren Sie diese Option, damit Fiasco die Daten "rückwärts" sortiert, also von großen zu kleinen Werten (z.B. Z, Y, X, ..., C, B, A)

Ok: bestätigt den Requester und beginnt den Sortiervorgang. Der vorher aktuelle Record wird beibehalten, die Nummer des Records wird sich jedoch höchstwahrscheinlich ändern.

Abbrechen: schließt den Requester ohne irgendetwas anderes noch zu tun.

10.3.7 Filter-Requester

Filter (siehe Abschnitt 6.6) bieten die Möglichkeit, eine Übersicht über eine bestimmte Gruppe von Records zu schaffen. Ein Filter erzeugt den Eindruck, als ob ein Projekt nur aus Records bestünde, die mit einem bestimmten Suchkriterium übereinstimmen. Filter werden nicht während des normalen Programmablaufs ausgewertet, sondern immer nur auf einmal erstellt. Dies bedeutet, daß Records, die während ein Filter aktiv ist hinzugefügt werden, auch angezeigt werden, wenn das Suchkriterium nicht übereinstimmt. Das selbe gilt für veränderte Records. Filter (siehe Abschnitt 6.6) offer the possibility to create an overview over a group of records. A filter creates the impression of a database, which consists only of the matching records. Filter are not created during the normal program functions, but it is created immediately after confirming the filter requester. That means, records which will be added to project, during a filter is active, will be displayed regardless of their contents. The same rules for changes records.

Der Filter-Requester ist über den Menüpunkt **Vergleichen/Filter Ändern** erreichbar.

Falls Sie sich bereits mit dem Such-Requester auskennen, sollten Sie sich hier *sehr* schnell zurechtfinden.

Feld: Hier können Sie das Feld wählen, in dem gesucht werden soll. In dem Listview werden die IDs der Felder angezeigt, Textfelder werden nicht angezeigt. Es kann immer nur ein Feld ausgewählt werden.

Muster: In dieses String-Gadget können sie ein Suchmuster eingeben. Entweder ein einfacher Wert, oder ein Wert mit Mustern.

Unschärfe Suche/Aktiv: Falls der Filter unscharf sein soll, muß dieses Gadget aktiviert sein.

Unschärfe Suche/Faktor: You can control here the tolerance of the blurred search. 0 searches only for exactly matching entries, 100 searches for almost all entries. Hier können Sie die Toleranz der unscharfen Suchfunktion einstellen. 0 entspricht, daß die Funktion keinerlei Einträge akzeptiert, 100 entspricht fast allen Einträgen.

Ok: bestätigt den Requester und erzeugt den Filter. Später scheint das Projekt nur noch aus Records zu bestehen, die den Bedingungen ent-

Cancel: schließt den Requester ohne jede weitere Aktion.

10.3.8 Markieren-Requester

Die Markieren-Funktion von Fiasco gibt Ihnen die Möglichkeit, sich einen besonderen Record zu merken. Der Markieren-Requester hat den Sinn, alle Records zu markieren, die auf ein angegebenes Muster passen. Dieser Requester ist stark verwandt mit dem Filter- und den Such-Requestern.

Die Markieren-Requester kann mit **Compare/Mark** geöffnet werden.

Feld: Hier können Sie das Feld wählen, in dem gesucht werden soll. In dem Listview werden die IDs der Felder angezeigt, Textfelder werden nicht angezeigt. Es kann immer nur ein Feld ausgewählt werden.

Muster: In dieses String-Gadget können sie ein Suchmuster eingeben. Entweder ein einfacher Wert, oder ein Wert mit Mustern.

Unschärfe Suche/Aktiv: Wenn sie unscharfe Suche anwenden wollen, müssen Sie dieses Gadget aktivieren.

Unschärfe Suche/Faktor: Hier können Sie die Toleranz der unscharfen Suchfunktion einstellen. 0 entspricht, daß die Funktion keinerlei Einträge akzeptiert, 100 entspricht fast allen Einträgen.

Ok: bestätigt den Requester und markiert die Records. Die alten Markierungen gehen dabei verloren!

Cancel: Schließt den Requester ohne weitere Aktion.

10.3.9 Benutzermenu-Requester

Fiasco bietet die Möglichkeit, eigene Menupunkte zu definieren und mit speziellen Funktionen zu belegen. Dabei ist die Wahl zwischen Name und Kommando, das das Item ausführen soll, frei wählbar. Die in diesem Requester definierten Menupunkte sind neben dem "normalen" Weg mit der Maus über die F-Tasten erreichbar. F1 bis F10 entsprechen den ersten zehn Menupunkten. Shift und F1 bis F10 entsprechen dann den Menupunkten 11 bis 20. Falls irgendjemand mehr als 20 Menupunkte definieren sollte, muß sich dann damit abfinden, die verbleibenden Menupunkte mit der Maus aufrufen zu müssen.

Weiterhin lassen sich nicht mehr als 63 Menupunkte definieren. Tut man es doch, wird ein Requester erscheinen, der darauf hinweist, daß die Anzahl der Menupunkte auf 63 zurückgeschraubt wurde.

Die Menupunkte lassen sich über **Einstellungen/Einstellungen Speichern** (siehe Abschnitt 10.2.69) dauerhaft sichern.

Items: Hier wird eine Liste aller bereits bestehenden Menu-Items dargestellt. Mit **Neu** kann man weitere Items hinzufügen, mit **Lösch.** das aktive Item entfernen und mit den **</>**-Buttons kann man die Position des aktuellen Items in der Liste verändern.

Type: Hier kann man einstellen, ob das aktuelle Item ein Programm oder ein ARexx-Script ausführen soll.

Command: Hier kann man das Programm bzw. ARexx-Script eingeben, das ausgeführt werden soll.

10.3.10 Optionen-Requester

Der Options-Requester dient dazu, einige Einstellungen speziell für das aktuelle Projekt zu machen. Er ist über den Menüpunkt **Projekt/Optionen** oder dem ARexx-Befehl **F.OptionsReq** erreichbar.

Name: Über dieses Gadget kann man den Datei-Namen des Projektes verändern. Unter diesem Namen wird das Projekt in Zukunft gespeichert werden, alle Verzeichnis-Relativen Operationen werden ab dann dieses Verzeichnis als aktuelles Verzeichnis haben.

Autor: Hier kann sich der Ersteller des Projektes verewigen. Der Name wird sich dann am Anfang der Projekt-Datei finden. Ansonsten wird der Inhalt ignoriert.

Bemerkungen: Noch ein Gadget zur freien Benutzung. Hier kann man irgendwelche Notizen, z.B. einen Versions-String (mit **\$VER:** am Anfang) eintragen. Dieser String wird noch vor dem Autor abgelegt. Und wiederum wird der Inhalt sonst ignoriert.

Dehnung X / Y: Hier kann man einen Wert angeben, der zur Breite bzw. der Höhe des Cursors hinzuaddiert wird. Dies bewirkt eine Dehnung der Make in X-Richtung bzw. Y-Richtung. More dazu in Sektion 4.5.

10.3.11 Gehe zu-Requester

Der Goto-Requester ist einer der übersichtlichsten Requester in Fiasco überhaupt. Er ist über den Menüpunkt **Record/Gehe zu** erreichbar und bietet die Möglichkeit, über die Record-Nummer diesen Record zu aktivieren.

Bitte beachten Sie, daß Filter die Record Nummern beeinflussen.

Gehe zu: Nimmt die Nummer des Records auf.

Ok: Bestätigt den Requester und springt zum Record mit der Nummer.

Cancel: Ist mir gerade entfallen... %-)

10.3.12 Relations-Requester

Dies ist die Schaltzentrale für die Relationen (siehe Abschnitt 7.1) in Fiasco. Dieser Requester ist über **Felder/Relation Ändern** erreichbar.

Schlüssel hier: In diesem Listview muß man den Schlüssel in dem aktuellen Projekt auswählen.

eigentl. Feld hier: Hier wird die ID des Feldes, dessen Relationen gerade bearbeitet werden, angezeigt.

Schlüssel dort: Hier muß man das Feld in dem Projekt, das unter **Relations-Datei** angegeben wurde, auswählen, das den eindeutigen Schlüssel beinhaltet. Da die Typen der beiden Schlüssel übereinstimmen müssen, werden hier nur die Felder angezeigt, die zu dem Schlüssel unter **Schlüssel hier** passen.

eigentl. Feld dort: Hier muß man das Feld in dem Projekt, das unter **Relations-Datei** angegeben wurde, auswählen, das den Partner zu **eigentl. Feld hier** darstellt. Aus diesem Feld werden dann die Daten, die in **eigentl. Feld hier** eingetragen werden, bezogen. Auch hier werden nur die Felder angezeigt, die zu **eigentl. Feld hier** passen (Typ und Max. Zeichen müssen gleich sein).

Relations-Datei: Hier muß man die Projekt-Datei angeben, in der die passenden Informationen stehen. Die Datei kann relativ zum Verzeichnis des aktuellen Projektes angegeben werden.

Ok: Bestätigt den Requester und versucht die Relationen zu laden. Falls dabei ein Fehler auftritt, wird der Requester mit einer entsprechenden Mitteilung wieder aktiviert, ansonsten wird das Haupt-Fenster aktiviert.

Abbrechen: Schließt den Requester ohne weitere Aktion.

10.3.13 Spalte anzeigen-Requester

Mit diesem Requester, der über **Liste/Spalte anzeigen** erreichbar ist, kann man Listen-Spalten, die vorher mit **Liste/Spalte verbergen** verborgen wurden, wieder sichtbar machen. Die Spalten werden dann ihrem alten Platz möglichst nahe in die Liste eingefügt.

Feld: Hier werden alle verborgenen Spalten angezeigt, aus denen man eine auswählen sollte.

Ok: Bestätigt den Requester und berechnet die Liste neu, so daß die Spalte wieder zu sehen ist.

Abbrechen: Schließt den Requester ohne weitere Aktion.

10.3.14 Anzeige Optionen-Requester

Diese Requester kontrolliert die Anzeige-Elemente von Fiasco. Sie können hier einen eigenen Screen für Fiasco öffnen und die Schriften für diesen Screen und für die Maske auswählen. (Fiasco 1.0 hatte den Menüpunkt Schrift auswählen für diesen Zweck)

Bildschirm

Bildschirmtyp: Wählen Sie hier aus, ob Sie einen Public Screen oder einen eigenen Screen benutzen möchten.

PubScreen Name: Geben Sie hier den Namen des Public Screens an, auf dem Fiasco seine Fenster öffnen soll. Fiasco benutzt diese Angabe nur, wenn Bildschirmtyp auf PublicScreen steht. Wenn Sie dieses Gadget leer lassen, wird Fiasco den Standard Public Screen benutzen (normalerweise die Workbench).

Bildschirm-Modus: Hier können Sie den Anzeige-Modus für den eigenen Screen auswählen. Wenn Sie auf das Pubup-Gadget klicken, öffnet sich ein ASL Screenmode-Requester. Dies setzt voraus, daß eine asl.library mit Version 38 oder höher installiert ist.

Bildschirm Font: Diese Gadget kontrolliert, ob Sie eine eigene Schrift oder den Font des Workbench-Screens für einen eigenen Screen benutzen möchten. Die Schrift der Workbench wird durch den Font-Voreinsteller festgelegt.

Eigener Font: Falls Sie einen eigenen Font für den eigenen Bildschirm benutzen möchten, können Sie diesen hier auswählen.

Masken Font

Masken Font: Dieses Gadget kontrolliert, ob sie einen eigenen Font oder den System-Default-Font für die Maske benutzen möchten. Der System-Default-Font wird durch den Font-Voreinsteller kontrolliert.

Eigener Font: Hier können Sie eine eigene Schrift für die Maske auswählen. Die Schrift darf nicht proportional sein, d.h. alle Buchstaben müssen die gleiche Breite haben.

Ok: Bestätigt den Requester und baut die Anzeige von Fiasco neu auf.

Abbrechen: Schließt den Requester ohne weitere Aktion.

10.3.15 Importieren-Requester

Der Importieren-Requester ist die GUI-Schnittstelle zur Import-Funktion (siehe Abschnitt 8) von Fiasco. Durch Import ist Fiasco in der Lage, Daten von anderen Datenbank-Programmen zu lesen. Meistens kann dies nicht direkt geschehen. Die fremde Datenbank muß dann die Daten "exportieren". Sie können verschiedene Parameter für das Importieren angeben, wodurch Sie in der Lage sein sollten, fast alle Import-/Export-Formate in Fiasco einlesen zu können.

Die Fiasco-Distribution enthält mehrere fertige Import-Formate, die mit dem **Laden**-Schalter am unteren Rand des Import-Requesters eingelesen werden können.

Die Werte, die in die Gadgets des Import-Requesters eingegeben werden können, sind im Import/Export-Kapitel dieses Dokuments beschrieben.

Datei: Geben Sie hier die Datei an, die die zu importierenden Daten enthält. Sie können den Auswahl-Schalter an der rechten Seite benutzen, um die Datei mit einem ASL-Requester auszuwählen.

Anzeigen: Klicken Sie hier, wenn Sie den Inhalt der Datei anzeigen wollen. Fiasco wird asynchron More oder MultiView, falls dies verfügbar ist, starten.

Records/Start: Geben Sie hier die Anfangs-Zeichen für Records ein. Standard-Wert: Leer.

Records/Ende: Geben Sie hier die End-Zeichen für Records ein. Standard-Wert: Leer.

Records/Tenner: Geben Sie hier die Zeichen zwischen zwei Records ein. Standard-Wert: \n.

Felder/Start: Geben Sie hier die Zeichen an, mit denen Felder beginnen. Standard-Wert: ".

Felder/Ende: Geben Sie hier die Zeichen an, mit denen Felder enden. Standard-Wert: ".

Felder/Trenner: Geben Sie hier die Zeichen zwischen zwei Feldern an.
Standard-Wert: \t.

Verschiedenes/Zeilen überspringen: Geben Sie hier die Start-Zeichen für Kommentare an. Standard-Wert: Leer.

Verschiedenes/Am Start überspringen: Geben Sie hier die Anzahl von Zeilen an, die am Anfang übersprungen werden sollen. Standard-Wert: 0.

Verschiedenes/Maximale Felder: Geben Sie hier die maximale Anzahl von Feldern eines Records an. Dieser Wert kann auch benutzt werden, wenn es keine Zeichen gibt, die das Ende eines Records markieren. Standard-Wert: 100.

Optionen/Erster Record enthält IDs: Aktivieren Sie dieses Gadget, wenn der erste Record der Datei die IDs der Felder enthält. Fiasco wird dann diese IDs statt generierter IDs benutzen.

Optionen/Neue Felder anhängen: Aktivieren Sie diese Option, wenn Sie wollen, daß Fiasco neue Felder für die Daten erzeugt und nicht die bereits bestehenden benutzt. Wenn Sie ein völlig leeres Projekt haben, sollten Sie diese Option aktivieren.

Optionen/Altes Projekt überschreiben: Entfernt die alten Daten im aktuellen Projekt-Fenster. Wenn Sie diese Option nicht aktivieren, werden Ihre Daten irgendwie an das existierende Projekt angehängt.

Ok: Startet den Import-Vorgang. Bitte beachten Sie, daß der Speicher aufgrund von falschen Struktur-Parameters oder zu großen Dateien knapp werden kann. Programme, die Probleme mit wenig Speicher haben, sollten während dieses Vorganges nicht laufen.

Sichern: Speichert die aktuellen Einstellungen in eine ausgewählten Datei.

Laden: Liest Einstellungen aus einer ausgewählten Datei und aktiviert diese im Requester.

Cancel: Schließt den Requester ohne weitere Aktion.

10.3.16 Exportieren-Requester

Die Export-Funktion gibt die Möglichkeit, Daten, die mit Fiasco erstellt wurden, anderen Datenbank-Programmen zugänglich zu machen. Normalerweise können fremde Datenbanken die Daten von Fiasco nicht lesen, da

sie das Format der Dateien nicht kennen. Der Import/Export-Abschnitt dieses Dokumentes enthält mehr Informationen über diesen Mechanismus.

Datei: Geben Sie hier den Namen der Datei an, in die die Daten geschrieben werden sollen. Falls eine Datei mit diesem Namen bereits bestehen sollte, wird sie überschrieben.

Records/Start: Geben Sie hier die Start-Zeichen für Records an. Standard-Wert: Leer.

Records/Ende: Geben Sie hier die End-Zeichen für Records an. Standard-Wert: Leer.

Records/Trenner: Geben Sie hier die Zeichen zwischen zwei Record an. Standard-Wert: \n.

Felder/Start: Geben Sie hier die Zeichen an, mit denen Felder beginnen sollen. Standard-Wert: “.

Felder/Ende: Geben Sie hier die Zeichen an, mit denen Felder enden sollen. Standard-Wert: “.

Felder/Trenner: Geben Sie hier die Zeichen zwischen zwei Feldern an. Standard-Wert: \t.

Optionen/Erster Record enthält IDs: Aktivieren Sie dieses Gadget, wenn Sie wollen, daß Fiasco die Feld-IDs des ersten Record schreibt.

Optionen/Nur markierte Records: Aktivieren Sie dieses Gadget, wenn Sie wollen, daß Fiasco nur Records schreibt, die markiert sind.

Ok: Klicken Sie hier, um den Export-Vorgang zu starten.

Sichern: Speichert die Struktur-Parameter in einer ausgewählten Datei.

Laden: Läd die Struktur-Parameter aus einer ausgewählten Datei.

Abbrechen: Schließt den Requester ohne weitere Aktion.

Kapitel 11

ARexx

ARexx ist eine Macro-Sprache mit der Fähigkeit, Programme untereinander kommunizieren zu lassen. ARexx wurde von William S. Hawes entwickelt und befindet sich im Lieferumfang von Amiga OS 2.0 oder höher.

Der ARexx-Port von Fiasco kann entweder aus einem Script unabhängig vom Programm angesprochen werden, oder Fiasco ruft selber Scripts auf. Dies geschieht Beispielsweise, wenn bei einem Feld ein Script (siehe Abschnitt 9.1) angegeben wurde und der Inhalt dieses Feldes geändert wurde. Dieses Script hat dann die Möglichkeit auf die Änderung zu reagieren und bestimmte Werte anzupassen oder irgendwas anderes zu tun.

Damit ein ARexx-Script mit Fiasco in Verbindung treten kann, muß ARexx am Anfang instruiert werden, nach Fiascos ARexx-Port zu suchen. Dies geschieht mit **Address FIASCO**.

Fast alle Operationen, die über die GUI von Fiasco ausgeführt werden können, können auch mit ARexx-Scripts erledigt werden. Außerdem kann Fiasco mit ARexx fast beliebig an Funktionen erweitert werden. Viele ARexx Kommandos tun genau das selbe wie die Menüpunkte. Das heißt, daß sie unter Umständen einen Requester öffnen. Es ist aber meistens möglich, dies zu umgehen. Dies wird später einmal behoben werden. Weiterhin gibt es Befehle, die immer einen Requester öffnen. Dies mag zwar für Scripts auch manchmal nützlich sein, ist jedoch primär implementiert wurden, um Fiasco eine Art zweites Menu wie Iconleisten zu geben. Ich habe bereits mit ToolManager-Docks experimentiert, doch für schnelles Umherblättern in einer Datenbank reagiert das ganze noch zu träge.

11.1 Allgemeines zu ARexx mit Fiasco

Ein Fiasco-Kommando gibt normalerweise in der Variable RC 0 zurück, wenn alles glatt gelaufen ist. Falls ein Kommando nicht die Gegebenheiten, die es benötigt, vorfindet, wird RC auf 5 gesetzt. Fehler wie falsche Parameter werden mit RC = 10 belohnt, fatalere Fehler kriegen sogar RC = 20. Parameter werden normalerweise durch Leerzeichen begrenzt. Wenn einzelne Parameter Leerzeichen enthalten sollen, reicht es *nicht*, diese einfach in Anführungszeichen einzuschließen, da ARexx alle Anführungszeichen schluckt. Um dies zu vermeiden, sollten Sie die Anführungszeichen in andere Anführungszeichen einschließen. (z.B. `F_Open ' Test Datei'`) Sie müssen die einfachen Anführungszeichen in der äußeren Position angeben, da Fiasco nur doppelte Anführungszeichen verarbeiten kann. Beachten Sie, daß sie keine Variablen in den Anführungszeichen benutzen können. Schließen Sie dann die äußeren Anführungszeichen, sagen dann den Variablennamen und öffnen dann wieder die Anführungszeichen, um eine Variable in ein Argument einzufügen. Diese Prozeduren sind für Argumente, die den /F-Modifier haben, nicht nötig. Wenn ein Kommando einen Wert zurückliefert steht dieser normalerweise in der Variablen RESULT. Um RESULT zu benutzen, muß man die Zeile `OPTIONS RESULTS` am Anfang eines ARexx-Scripts einfügen.

Die Fehlersuche in ARexx-Scripts ist etwas problematisch. Scripts, die durch das User-Menu oder durch Felder aktiviert wurde, können keine Ausgaben machen, so werden jegliche Fehlermeldungen verschluckt. Zum Testen von ARexx- Scripts empfiehlt es sich daher, sie vorher von der Shell aus zu testen (mit `rx Scriptname`; Fiasco muß natürlich dabei mit den passenden Daten bereitstehen). Um mehr über die Gründe zu erfahren, warum ein an Fiasco gerichtetes Kommando fehlerhaft arbeitete, kann man im Kontrolle-Menu den Menüpunkt **ARexx-Debug** aktivieren. Dies bewirkt, daß jedesmal, wenn Fiasco einen Fehlercode zurückgeben würde, ein Requester erscheint, in dem das Problem beschrieben wird. Hier hat man auch die Möglichkeit, dem Script vorzugaukeln, das Kommando wäre erfolgreich (Ignorieren-Gadget), indem einfach 0 in RC zurückgegeben wird. Will man, daß der Fehlercode normal weitgegeben wird, sollte man **Fortsetzen** anwählen. Eine weitere Möglichkeit ist **Hilfe**, die den Requester nicht schließt, sondern den Hilfs-Text für das Kommando anzeigt, das den Fehler verursachte.

Die Dokumentation zu den Kommandos ist im Stil an die Amiga OS-Autodocs angelehnt. Synopsis legt eine Aufrufschablone fest.

11.2 ARexx-Kommandos

11.2.1 F_AboutReq

Name: F_AboutReq – Öffne der “About”-Requester

Synopsis: F_AboutReq

Funktion: Macht genau das selbe wie Projekt/Über Fiasco (siehe Abschnitt 10.2.11).

Argumente: none

Ergebnisse: none

11.2.2 F_ActivateField

Name: F_ActivateField – Aktiviere ein Feld in der GUI

Synopsis: F_ActivateField Feld/A
rc = Erfolg

Funktion: Aktivier das Feld mit der angegebenen ID in der Maske. Nur Felder, die als String/Longint-Gadgets erscheinen können aktiviert werden. Wenn das Projekt bzw. das Fenster nicht aktiv ist, kann das Feld nicht aktiviert werden. Dieses Kommando kann nur im Masken-Modus aufgerufen werden.

Argumente: Feld - ID des zu aktivierenden Feldes

Ergebnisse: rc = 0, wenn das Feld aktiviert wurde

See also: intuition.library/ActivateGadget()

11.2.3 F_AddFieldReq

Name: F_AddFieldReq – Öffne den Feld Hinzufügen-Requester

Synopsis: F_AddFieldReq

Funktion: Dieses Kommando macht genau das selbe wie Field/Add Field (siehe Abschnitt 10.2.31). Es darf nur im Masken-Modus aufgerufen werden.

Argumente:

Ergebnisse:

Siehe auch:

11.2.4 F_AddRecord

Name: F_AddRecord – Füge einen neuen Record hinzu.

Synopsis: F_AddRecord

Funktion: Fügt einen neuen Record nach dem aktuellen ein. Dieser Record wird dann auch automatisch der aktuelle.

Diese Funktion kann nur im Record-Modus aufgerufen werden.

Argumente: keine

Ergebnisse: keine

Siehe auch: F_RemRecord, Records/Hinzufügen

11.2.5 F_ClearProject

Name: F_ClearProject – leere das aktive Projekt

Synopsis: F_ClearProject Force/S

Funktion: Löscht alle Daten im aktuellen Projekt, so daß es sich hinterher in einem Zustand wie gerade über Projekt/Neu neu geöffnete Projekte befindet. Wenn Sie nicht Force angeben, entspricht dieses Kommando dem Menüpunkt Projekt/Leeren, so ist es also auch möglich, daß ein Requester erzeugt wird, in dem gefragt wird, ob das Projekt vorher noch gespeichert werden soll. Um dies zu umgehen, sollte Force angegeben. Um herauszufinden, ob das Projekt nicht gespeichert ist, können Sie F_IsVirgin (siehe Abschnitt 11.2.31) benutzen.

Argumente: Force – Alle Warnungen unterdrücken

Ergebnisse: keine

Siehe auch: Projekt/Leeren, F_IsVirgin, F_MakeVirgin

11.2.6 F_CloseList

Name: F_CloseList – schließe das Listen-Fenster

Synopsis: F_CloseList

Funktion: Schließt das Listen-Fenster. Falls das Fenster nicht offen sein sollte, passiert gar nichts. Diese Kommando entspricht dem Deaktivieren des Menüpunktes Kontrolle/Listen-Fenster.

Argumente:

Ergebnisse:

Siehe auch: Kontrolle/Listen-Fenster

11.2.7 F_CloseServiceWin

Name: F_CloseServiceWin – schließe das Service-Fenster

Synopsis: F_CloseServiceWin

Funktion: Schließt das Service-Window. Falls es zum Zeitpunkt des Aufrufes nicht offen ist, tut F_CloseServiceWin gar nichts.

Argumente: keine

Ergebnisse: keine

Siehe auch: F_OpenServiceWin

11.2.8 F_ConvertField

Name: F_ConvertField – ändere den Typ eines Feldes

Synopsis: F_ConvertField Field/A,NewType/A,AltFormat/S

Funktion: Ändert den Typ des angegebenen Feldes. Sie können keine Text- oder Button-Felder konvertieren. Kann nur im Masken-Modus aufgerufen werden.

Argumente: Field - ID des Feldes

NewType - Neuer Typ des Feldes (z.B. String)

AltFormat - Angeben, falls Sie ein alternatives Format wünschen

Ergebnisse: keine

Siehe auch: Abschnitt Felder umwandeln

11.2.9 F_CountRecs

Name: F_CountRecs – zähle die Records

Synopsis: F_CountRecs

Result = *Anzahl_der_Records*

Funktion: Zählt die Records, aus denen das aktuelle Projekt besteht. Kann nur im Record-Modus aufgerufen werden.

Argumente: keine

Ergebnisse: Anzahl_der_Records - The number of records, may be zero. Note, that Filter influence this value. Die momentane Anzahl der Records im aktuellen Projekt, kann 0 sein. Bitte beachten Sie, daß Filter diesen Wert beeinflussen.

Siehe auch:

11.2.10 F_CountReq

Name: F_CountReq – Öffne den Count-Requester

Synopsis: F_CountReq

Funktion: Öffnet den Count-Requester, in dem der Benutzer die Anzahl der Übereinstimmungen des aktuellen Projektes mit einem Suchmuster bestimmen kann. Dieses Kommando macht genau dasselbe wie Compare/Count (siehe Abschnitt 10.2.46). Dieses Kommando kann nur im Record-Modus aufgerufen werden.

Argumente:

Ergebnisse:

Siehe auch: Compare/Count

11.2.11 F_DupRec

Name: F_DupRec – Klone den aktiven Record

Synopsis: F_DupRec

Function: Dieses Kommando dupliziert exakt den aktuellen Record. Sämtliche Startwert-Attribute werden ignoriert. Diese Funktion tut exakt das selbe wie der Menüpunkt Record/Duplizieren. Falls kein Record aktiv sein sollte, passiert nichts. F_DupRec kann nur im Record-Modus aufgerufen werden.

Argumente:

Ergebnisse:

Siehe auch: Records/Duplizieren

11.2.12 F_Export

Name: F_Export – exportiere Daten aus Fiasco

Synopsis: F_Export File/A,RecStart/K,RecEnd/K,RecSep/K,FieldStart/K,
FieldEnd/K,FieldSep/K,FirstRecIDs/K,MarkedOnly/S
rc = *Erfolg*

Funktion: Ruft die Export-Funktion von Fiasco auf. Der Import/Export-Abschnitt enthält mehr Informationen über das Exportieren. Wenn Sie ein Argument nicht angeben, wird es leer sein.

Argumente: File - Datei zu schreiben
RecStart,RecEnd,RecSep,FieldStart,FieldEnd,FieldSep - Struktur-Parameter
FirstRecIDs - Erster Record soll die Feld-IDs enthalten
MarkedOnly - Exportiere nur markierte Records

Ergebnisse: rc = 0, wenn alles gutging.

See also: F_Import, Abschnitt Import/Export

11.2.13 F_FilterReq

Name: F_FilterReq – Öffne den Filter-Requester (siehe Abschnitt 10.3.7)

Synopsis: F_FilterReq

Funktion: Macht genau das selbe wie Compare/Filter (siehe Abschnitt 10.2.48).
Darf nur im Record-Modus aufgerufen werden.

Argumente:

Ergebnisse:

Siehe auch: Compare/Filter

11.2.14 F_FindFirst

Name: F_FindFirst – Suche nach einem Muster

Synopsis: F_FindFirst Field,Blur/K,Pattern/F
Result = *Number_of_Record*

Funktion: Sucht vorwärts nach der ersten Übereinstimmung mit dem Suchmuster, das entweder mit `F_SetSearchPat` (siehe Abschnitt 11.2.59) oder den Argumenten angegeben wurde. Wenn `rc = 0` ist, enthält `Result` die Nummer des gefundenen Records. Dieser kann dann mit `F_GotoRec` aktiviert werden. Falls keine Übereinstimmung gefunden wurde, oder noch kein Suchmuster angegeben wurde, ist `rc = 5`.

Argumente: `Field` - ID des zu durchsuchenden Feldes

`Blur` - Faktor für unscharfe Suche. Wenn angegeben, aktiv.

`Pattern` - Suchmuster.

Wenn Sie `Field` oder `Pattern` nicht angeben, werden die Werte benutzt, die vorher im Such-Requester angegeben wurden oder mit `F_SetSearchPat` bzw. `F_SetSearchField` eingestellt wurden.

Ergebnisse: `rc = 0`: `result` = Nummer der Übereinstimmung.
`rc = 5`: nichts mehr gefunden.

Beispiel: `/* Find-Example.rexx */`

```
options results
address FIASCO

count = 0

F_FindFirst "Test" "?#?" /* Irgendein Feldname
                          * Suche nach allen Records, die
                          * mind. ein Zeichen beinhalten */

do while rc = 0           /* Suche solange, bis nichts mehr
                          * gefunden wird */

    F_GotoRec Result      /* Aktiviere den gefundenen Record */

    count = count + 1

    F_FindNext "Test" "?#?" /* Suche nach dem nächsten */

end

/* All records done */
```

Siehe auch:

11.2.15 F_FindNext

Name: F_FindNext – Suche nach einem Muster

Synopsis: F_FindNext Field,Blur/K,Pattern/F
Result = *Number_of_next_Record*

Funktion: Sucht vorwärts nach der nächsten Übereinstimmung mit dem Suchmuster, das entweder mit F_SetSearchPat (siehe Abschnitt 11.2.59) oder den Argumenten angegeben wurde. Wenn rc = 0 ist, enthält Result die Nummer des gefundenen Records. Dieser kann dann mit F_GotoRec aktiviert werden. Falls keine Übereinstimmung gefunden wurde, oder noch kein Suchmuster angegeben wurde, ist rc = 5.

Bemerkung: Der aktuelle Record wird bei der Suche mit F_FindNext und F_FindPrev nicht mit einbezogen. Will man also ein Programm schreiben, in dem das ganze Projekt durchsucht wird, muß man zuerst F_FindFirst (siehe Abschnitt 11.2.14) und dann F_FindNext benutzen.

Argumente: Field - ID des zu durchsuchenden Feldes
Blur - Faktor für unscharfe Suche. Wenn angegeben, aktiv.
Pattern - Suchmuster
Wenn Sie Field oder Pattern nicht angeben, werden die Werte benutzt, die vorher im Such-Requester angegeben wurden oder mit F_SetSearchPat bzw. F_SetSearchField eingestellt wurden.

Ergebnisse: Wenn rc = 0, result = Recordnummer der Übereinstimmung.
Wenn rc = 5, nichts mehr gefunden oder kein Muster.

Beispiel: siehe F_FindFirst

Siehe auch:

11.2.16 F_FindPrev

Name: F_FindPrev – Suche nach einem Muster rückwärts

Synopsis: F_FindPrev Field,Blur/K,Pattern/F
Result = *Number_of_prev_Record*

Funktion: Sucht rückwärts nach der nächsten Übereinstimmung mit dem Suchmuster, das entweder mit F_SetSearchPat (siehe Abschnitt 11.2.59) oder den Argumenten angegeben wurde. Wenn rc = 0 ist, enthält

Result die Nummer des gefundenen Records. Dieser kann dann mit F_GotoRec aktiviert werden. Falls keine Übereinstimmung gefunden wurde, oder noch kein Suchmuster angegeben wurde, ist rc = 5.

Argumente: Field - ID des zu durchsuchenden Feldes

Blur - Faktor für unscharfe Suche. Wenn angegeben, aktiv.

Pattern - Suchmuster.

Wenn Sie Field oder Pattern nicht angeben, werden die Werte benutzt, die vorher im Such-Requester angegeben wurden oder mit F_SetSearchPat bzw. F_SetSearchField eingestellt wurden.

Ergebnisse: Wenn rc = 0, result = Recordnummer der Übereinstimmung
Wenn rc = 5, nichts mehr gefunden oder kein Muster

Bemerkung: Innerhalb von ARexx-Script ist F_FindPrev relativ unpraktisch. Bessere Ergebnisse lassen sich mit Kombinationen von F_FindFirst (siehe Abschnitt 11.2.14) und F_FindNext erzielen.

Siehe auch:

11.2.17 F_FindReq

Name: F_FindReq – Öffne den Such-Requester

Synopsis: F_FindReq

Funktion: Öffnet den Such-Requester (siehe Abschnitt 10.3.3), entspricht exakt dem Menüpunkt *Vergleichen/Suchen* (siehe Abschnitt 10.2.42). Dieses Kommando kann nur im Record-Modus aufgerufen werden.

Argumente: keine

Ergebnisse: keine

Siehe auch:

11.2.18 F_GetFieldAttributes

Name: F_GetFieldAttributes – Lese die Attribute eines Feldes

Synopsis: F_GetFieldAttributes Field/A,X/S,Y/S,W=Width/S,H=Height/S,
Rexx/S,Type/S,ListX/S,ListW/S,MaxChars/S,InitCont/S,OwnInit/S,
Labels/K/N,Commands/S,Stack/S
rc = *Success*
Result = *Attribute_Value*

Funktion: Liest ein Attribut des angegebenen Feldes. Der Wert des Attributes wird dann in **Result** zurückgegeben. Nicht jeder Feldtyp unterstützt alle Attribute; Wenn ein Typ ein Attribut nicht unterstützt wird **rc** ungleich 0 zurückgegeben. Sie dürfen nur *ein* Attribut angeben, wenn Sie dieses Kommando aufrufen. Dieses Kommando kann sowohl im Masken- als auch im Record-Modus aufgerufen werden.

Dieses Kommando kann auch im virtuellen Status aufgerufen werden.

Argumente: Field - ID eines Feldes. Immer benötigt.

X - Ich möchte die obere Ecke des Feldes in Cursor-Einheiten wissen.

Y - Linke Ecke des Feldes in Cursor-Einheiten

W - Breite des Feldes in Cursor-Einheiten

H - Höhe des Feldes in Cursor-Einheiten

Rexx - Name des ARexx-Scripts des Feldes

Type - Typ eines Feldes (z.B. String, Integer, etc.)

ListX - Linke Ecke eines Feldes in der Liste, -1 wenn verborgen

ListW - Breite eines Feldes in der Liste, -1 wenn verborgen

MaxChars - Max. Zeichen Attribut

InitCont - Startwert-Attribut. Eins von own, old, key OwnInit - Eigener Startwert

Labels - Gibt das Label des angegebenen Nummer zurück

Command - Kommando-Attribut des Feldes

Stack - Stack-Attribut des Feldes

Ergebnisse: **rc** - Null, wenn erfolgreich.

Result - Enthält gewünschtes Attribut, wenn **rc** = 0

Siehe auch: Feld Dokumentation

11.2.19 F_GetFieldCont

Name: F_GetFieldCont – Lese den Inhalt eines Feldes

Synopsis: F_GetFieldCont Field/A

rc = Success

result = Content

Funktion: Liest den Inhalt des Feldes mit der angegebenen ID im aktuellen Record aus und liefert ihn in **result** zurück.

Kann nur im Record-Modus aufgerufen werden.

Dieses Kommando kann auch im virtuellen Status aufgerufen werden.

Argumente: FieldId - Id des Feldes

Ergebnisse: rc = 0 - alles Ok, result enthält den Inhalt

rc = 5 - kein Record aktiv

rc = 10 - Argument fehlt oder ID nicht gültig.

result - enthält den aktuellen Inhalt des Feldes, falls rc = 0 ist.

Das Format des Inhaltes:

String - der String selbst.

Integer - die Zahl selbst.

Float - die Fließkommazahl.

Slider - der Wert des Sliders.

Cycle - die Nummer des aktiven Eintrages.

Date - das Datum im Format TT.MM.[JJ]JJ.

Time - die Zeit im Format HH:MM:SS.

Extern - der String selbst.

Datotyp.- der String selbst.

Siehe auch:

11.2.20 F_GetProjFullName

Name: F_GetProjFullName – Lese den Dateinamen des aktuellen Projekts

Synopsis: F_GetProjFullName

Result = *Name*

Funktion: Gibt den Dateinamen des aktuellen Projektes incl. Pfad zurück.

Bemerkung: Falls der Pfad nicht absolut ist, ist er relativ zum aktuellen Verzeichnis von Fiasco.

Argumente:

Ergebnisse: Name - Name des Projektes incl. Pfad.

Siehe auch: F_GetProjName

11.2.21 F_GetProjName

Name: F_GetProjName – lese den Dateinamen des akt. Projektes

Synopsis: F_GetProjName

Result = *Filename*

Funktion: Gibt den Dateinamen ohne Pfad des aktuellen Projektes zurück.
Dieser Wert kann für `F_SelectProj` (siehe Abschnitt 11.2.55) benutzt werden.

Argumente: keine

Ergebnisse: Result - Name der Datei des aktuellen Projektes ohne Pfad
Es muß nicht unbedingt eine Datei mit diesem Namen existieren, z.B. wenn er über Options nachträglich geändert wurde.

Siehe auch: `F_GetProjFullName`

11.2.22 `F_GetRecNum`

Name: `F_GetRecNum` – Lese die Nummer des akt. Records.

Synopsis: `F_GetRecNum`
Result = *Number_of_record*

Funktion: Gibt die Nummer des aktuellen Records in Result zurück. Kann z.B. benutzt werden, um einen Anfangszustand zwischenzuspeichern und am Schluß mit `F_GotoRec` (siehe Abschnitt 11.2.27) wiederherzustellen.

Argumente: keine

Ergebnisse: Result = Nummer des Records. Bitte beachten Sie, daß Filter u.ä. Record Nummern beeinflussen können.

Siehe auch:

11.2.23 `F_GotoFirstRec`

Name: `F_GotoFirstRec` – aktiviere den ersten Record

Synopsis: `F_GotoFirstRec`

Funktion: Aktiviert den ersten Record des aktuellen Projekts. Falls keine Records existieren, passiert nichts.
Äquivalent mit dem Menüpunkt `Records/Erster` (siehe Abschnitt 10.2.22).
Kann nur im Record-Modus aufgerufen werden.

Argumente: keine

Ergebnisse: keine

Siehe auch:

11.2.24 F_GotoNextRec

Name: F_GotoNextRec – aktiviere den nächsten Record

Synopsis: F_GotoNextRec

Funktion: Aktiviert den Record, der dem aktuellen folgt. Falls der aktuelle der Letzte ist oder keine Records existieren, passiert nichts.

Äquivalent mit dem Menüpunkt **Record/Nächster** (siehe Abschnitt 10.2.21).
Kann nur im Record-Modus aufgerufen werden.

Argumente: keine

Ergebnisse: keine

Siehe auch:

11.2.25 F_GotoLastRec

Name: F_GotoLastRec – aktiviere den letzten Record

Synopsis: F_GotoLastRec

Funktion: Aktiviert den letzten Record im aktuellen Projekt

Äquivalent mit dem Menüpunkt **Record/Letzter** (siehe Abschnitt 10.2.23).
Kann nur im Record-Modus aufgerufen werden.

Argumente:

Ergebnisse:

Siehe auch:

11.2.26 F_GotoPrevRec

Name: F_GotoPrevRec – aktiviere den vorherigen Record

Synopsis: F_GotoPrevRec

Funktion: Aktiviert den dem aktuellen Record vorhergehenden Record.
Falls der aktuelle der erste Record ist oder keine Records existieren,
passiert nichts.

Äquivalent mit dem Menüpunkt **Record/Voriger** (siehe Abschnitt 10.2.20).
Kann nur im Record-Modus aufgerufen werden.

Argumente: keine

Ergebnisse: keine

Siehe auch:

11.2.27 F_GotoRec

Name: F_GotoRec – aktiviere einen bestimmten Record

Synopsis: F_GotoRec Record/A/N

Funktion: Aktiviere den Record, dessen Nummer als Argument übergeben wurde. Falls die Nummer falsch war, tue gar nichts.

Argumente: Record - Die Nummer des Records. Bitte beachten Sie, daß sich durch Sortiervorgänge und Hinzufügen oder Entfernen von Records und Filtern die Nummern der Records verändern können.

Ergebnisse:

Siehe auch:

11.2.28 F_GotoRecReq

Name: F_GotoRecReq – öffne den Goto-Requester (siehe Abschnitt 10.3.11)

Synopsis: F_GotoRecReq

Funktion: Dieses Kommando fordert den Benutzer auf, die Nummer eines Records anzugeben, der daraufhin aktiviert wird.

Äquivalent mit dem Menüpunkt Record/Gehe zu (siehe Abschnitt 10.2.24).
Kann nur im Record-Modus aufgerufen werden.

Argumente: keine

Ergebnisse: keine

Siehe auch: F_GotoRec Record/Gehe zu

11.2.29 F_Import

Name: F_Import – Importiere Daten

Synopsis: F_Import File/A,RecStart/K,RecEnd/K,RecSep/K,FieldStart/K,
FieldEnd/K,FieldSep/K,SkipLines/k,StartLine/N/K,FirstRecIDs/S,
AppendFields/S
rc = *Success*

Funktion: Ruft die Import-Funktion von Fiasco auf. Die angegebene Datei wird in das aktuelle Projekt mit den angegebenen Parametern importiert werden. Für mehr Informationen über Import und Export siehe Abschnitt 8. Sie können auch die Escape-Sequenzen von Fiasco benutzen. Wenn Sie einen Parameter nicht angeben, wird dieser leer sein.

Argumente: File - Name der Datei
 RecStart, RecEnd, RecSep, FieldStart, FieldEnd, FieldSep - die Struktur-Zeichen
 SkipLines - Kommentar-Beginn
 StartLine - Länge des Start-Kommentars
 FirstRecIDs - Erster Record enthält IDs
 AppendFields - Neue Felder anhängen

Ergebnisse: rc = 0, wenn alles gut ging.

Notes: Die Option Altes Projekt überschreiben des Import-Requesters ist nicht direkt unterstützt. Sie müssen dies mit F_ClearProject (siehe Abschnitt 11.2.5) emulieren.

Siehe auch: F_Export, Kapitel Import und Export

11.2.30 F_IsMarked

Name: F_IsMarked – Ist der Record markiert?

Synopsis: F_IsMarked Record/N
 rc = *IsMarked*

Funktion: Schaut nach, ob der aktuelle bzw. der angegebene Record markiert ist. Wenn er nicht markiert ist, wird in rc 5 zurückgegeben.

Dieses Kommando kann auch im virtuellen Status aufgerufen werden.

Argumente: Record - Nummer des Records. Wenn nicht angegeben, wird der aktuelle Record benutzt.

Ergebnisse: rc = 0: Record markiert, = 5: Record nicht markiert, > 5: Fehler

Siehe auch:

11.2.31 F_IsVirgin

Name: F_IsVirgin – Ist das Projekt unverändert?

Synopsis: F_IsVirgin
rc = Is_Virgin

Funktion: Testet, ob das aktuelle Projekt unverändert ist, d.h. ein Sicherheits-Requester bei Quit, Erase, Load o.ä. erscheinen würde.

Argumente: keine

Ergebnisse: rc = 0 - Unverändert
rc = 5 - Verändert

Siehe auch: F_MakeVirgin

11.2.32 F_LoadDTObject

Name: F_LoadDTObject – Lade den Inhalt eines Datatypes-Feldes

Synopsis: F_LoadDTObject Field/A

Funktion: Läd den Inhalt eines Datatypes-Feldes, das “verzögert” ist.

Argumente: Field - ID des Datatypes-Feldes

Ergebnisse: Der Inhalt wird geladen

Siehe auch:

11.2.33 F_Locate

Name: F_Locate – Setze den Cursor

Synopsis: F_Locate X/A/N,Y/A/N

Funktion: Setzt den Cursor an die angegebene Position. An dieser wird die nächste Masken-Operation stattfinden. Diese Funktion kann nur im Masken-Modus aufgerufen werden.

Argumente: X - X-Koordinate
Y - Y-Koordinate

Ergebnisse:

Bugs: Ist z.Zt. nicht besonders nützlich, da noch keine direkten Kommandos zum Verändern der Maske zur Verfügung stehen.

Siehe auch:

11.2.34 F_LockGUI

Name: F_LockGUI – Verschließe die GUI.

Synopsis: F_LockGUI

Funktion: Macht die GUI von Fiasco unverfügbar für den Benutzer. Der Maus-Zeiger wird als eine “Wartuhr” erscheinen. Nachdem die GUI verschlossen wurde, kann das AREXX-Script laufen, ohne daß die Gefahr besteht, daß das Script vom Benutzer beeinflußt wird. Bevor das Script endet, muß F_UnlockGUI aufgerufen werden, um die Kontrolle zum Benutzer zurückzugeben. F_LockGUI und F_UnlockGUI können verschachtelt werden.

Argumente: keine

Ergebnisse: keine

Note: Achten Sie darauf, daß Ihre Scripts die GUI auf jeden Fall vor dem Beenden wieder entriegeln. Benutzen Sie Signal-Kommandos, um Fehler oder Abbrüche abzufangen. Zum Beispiel:

```
/* test.rexx */

address FIASCO
options results

signal on syntax
signal on halt

F_LockGUI      /* Verschließe die GUI */

/* Ihr Programm */

F_UnlockGUI    /* Entriegele die GUI */

exit          /* Und Beende */

Syntax:
Halt:

F_UnlockGUI
```

```
exit
```

Wenn ein Script Fiasco doch verschlossen zurücklassen sollte, kann das folgende Script benutzt werden, das auch als ARexx/UnlockGUI.rexx verfügbar ist:

```
/*
 * Fiasco wird sich einmal beschweren,
 * wenn ARexx-Debug aktiv ist
 */

address FIASCO

do forever

    F_UnlockGUI

    if rc ~= 0 then break

end
```

Siehe auch: F_UnlockGUI

11.2.35 F_MakeVirgin

Name: F_MakeVirgin – Gebe vor, daß ein Projekt unverändert ist

Synopsis: F_MakeVirgin

Funktion: “Macht” das aktuelle Projekt unverändert. Dies bewirkt, daß bei bestimmten Operationen (Erase, Load, Quit,...) kein Sicherheits-Requester geöffnet wird.

Argumente: keine

Ergebnisse: Ein Projekt, das glaubt, daß es seit dem letzten Speichern nicht mehr verändert wurde

Bemerkung: Die ARexx-Kommandos von Fiasco 1.1 haben direkte Argumente, um diese Warnungen zu unterdrücken. Deshalb hat diese Funktion keine echte Bedeutung. Am besten vermeiden Sie es gänzlich, diese Funktion zu benutzen, um den Benutzer nicht zu verwirren.

Siehe auch: F_IsVirgin

11.2.36 F_MarkAllRecords

Name: F_MarkAllRecords – Markiere alle Records

Synopsis: F_MarkAllRecords

Funktion: Markiert alle Records im aktiven Projekt. Die Records werden in der Liste hervorgehoben angezeigt werden. Macht exakt das selbe wie Record/Alle Markieren (siehe Abschnitt 10.2.27).

Argumente:

Ergebnisse:

Siehe auch: F_UnmarkAllRecords F_MarkRecord

11.2.37 F_MarkMatch

Name: F_MarkMatch – Markiere die Records, die auf ein Muster passen

Synopsis: F_MarkMatch Field/A, Blur/K, Pattern/F/A

Funktion: Markiert alle Records, die auf das gegebene Muster passen. Arbeitet ähnlich zum Filter. F_MarkMatch löscht die Marken der Records, die nicht passen.

Argumente: Field – Die ID des Feldes, das untersucht werden soll
Blur – Der Faktor für unscharfe Suche. Nur angeben, wenn Sie es benutzen wollen.
Pattern – Das Suchmuster

Ergebnisse:

Siehe auch: F_MarkRecord F_ToggleAllMarks

11.2.38 F_MarkRecord

Name: F_MarkRecord – Markiere einen Record

Synopsis: F_MarkRecord Record/N

Funktion: Markiert einen Record im aktiven Projekt. Er wird in der Liste hervorgehoben angezeigt werden.

Argumente: Record/N – Optional, wenn angegeben wird der Record, der durch seine Nummer angegeben wird, markiert. Ansonsten wird der aktive Record markiert.

Dieses Kommando kann auch im virtuellen Status aufgerufen werden.

Ergebnisse:

Siehe auch: F_UnmarkRecord F_MarkAllRecords

11.2.39 F_NewProject

Name: F_NewProject – Öffne ein neues Projekt-Fenster

Synopsis: F_NewProject

Funktion: Öffnet ein neues Projekt, d.h. ein neues Fenster und aktiviert dieses. Anschließend können neue Felder erstellt werden oder eine Datenbank geladen werden. Macht exakt dasselbe wie Projekt/Neu (siehe Abschnitt 10.2.1).

Argumente: keine

Ergebnisse: keine

Bugs: Sollte Fehlerfall signalisieren.

Siehe auch:

11.2.40 F_OpenList

Name: F_OpenList – Öffne das Listen-Fenster

Synopsis: F_OpenList

Funktion: Öffnet das Listen-Fenster, in dem der Benutzer einen Überblick über die Records des Projektes bekommt. Falls das Listen-Fenster bereits offen sein sollte, passiert gar nichts. Dieses Kommando entspricht dem Aktivieren des Menüpunktes Kontrolle/Listen-Fenster (siehe Abschnitt 10.2.56).

Argumente:

Ergebnisse:

Siehe auch: Kontrolle/Listen-Fenster F_CloseList

11.2.41 F_OpenProject

Name: F_OpenProject – Lade ein Projekt

Synopsis: F_OpenProject File/A
rc = *Success*

Funktion: Versucht ein Fiasco-Projekt von Diskette ins aktuelle Fenster zu laden. Die aktuellen Daten werden ohne Sicherheitsabfrage gelöscht.

Argumente: Name - Dateiname des Projektes

Ergebnisse: rc = 0, wenn alles glatt ging,
= 10, wenn Argument fehlt oder Datei nicht geladen werden kann.

Siehe auch: F_OpenProjectReq

11.2.42 F_OpenProjectReq

Name: F_OpenProjectReq – Öffne den OpenProject-ASL-Requester

Synopsis: F_OpenProjectReq

Funktion: Öffnet einen File-Requester, mit dem der Benutzer eine Fiasco-Datenbank zum Laden auswählen kann. Falls das alte Projekt zwischenzeitlich geändert wurde, wird ein Sicherheitsrequester erzeugt, der fragt, ob man das alte Projekt wirklich aus dem Speicher werfen will. Entspricht dem Menüpunkt Projekt/Öffnen (siehe Abschnitt 10.2.3).

Argumente: keine

Ergebnisse: keine

Bemerkung: Der Benutzer kann den Request abgebrochen haben.

Siehe auch: F_OpenProject Project/Open

11.2.43 F_OpenServiceWin

Name: F_OpenServiceWin – Öffne das Service-Fenster

Synopsis: F_OpenServiceWin

Funktion: Öffnet das Service-Fenster (siehe Abschnitt 10.1), falls es noch nicht offen ist.

Argumente: keine

Ergebnisse: keine

Siehe auch: F_CloseServiceWin

11.2.44 F_OptionsReq

Name: F_OptionsReq – Öffne den Options-Requester (siehe Abschnitt 10.3.10)

Synopsis: F_OptionsReq

Funktion: Öffnet den Options-Requester (siehe Abschnitt 10.3.10), mit dem der Benutzer bestimmte projektspezifische Einstellungen bestimmen kann. Entspricht dem Menüpunkt Projekt/Optionen.

Argumente: keine

Ergebnisse: keine

Siehe auch:

11.2.45 F_Progress

Name: F_Progress – gebe dem Benutzer einen Hinweis für die Dauer einer Operation

Synopsis: F_Progress Done/A/N, Max/A/N

Funktion: Zeigt eine hübsche Fortschritts-Leiste im Service-Fenster an, wie sie von Sortieren oder Öffnen bekannt ist. Sie sollten das Status-Gadget mit F_ResetStatus (siehe Abschnitt 11.2.51) zurücksetzen wenn die Operation fertig ist.

Argumente: Done – Die Anzahl von Daten-Einheiten, die bereits durchlaufen wurden.
Max – Die Anzahl aller Daten-Einheiten.

Ergebnisse:

Siehe auch: F_SetStatus

11.2.46 F_Quit

Name: F_Quit – schließe das aktive Projekt

Synopsis: F_Quit Force/S

Funktion: Schließt das aktive Projekt. Wenn Sie nicht **Force** angeben, entspricht dieses Kommando dem Menüpunkt **Projekt/Beenden**, so ist es also auch möglich, daß ein Requester erzeugt wird, in dem gefragt wird, ob das Projekt vorher noch gespeichert werden soll. Um dies zu umgehen, sollte **Force** angeben. Um herauszufinden, ob das Projekt nicht gespeichert ist, können Sie **F_IsVirgin** (siehe Abschnitt 11.2.31) benutzen.

Argumente: **Force** – Alle Warnungen unterdrücken.

Ergebnisse: keine

Notes: Wenn das aktuelle Projekt geschlossen wird, wird ein anderes aktiviert oder, falls sonst keins mehr offen war, das Programm beendet. Ein ARexx-Script sollte sich nicht auf die Reihenfolge verlassen, in der die Projekte aktiviert werden.

Siehe auch:

11.2.47 **F_RemAllRecords**

Name: **F_RemAllRecords** – Lösche alle Records eines Projektes

Synopsis: **F_RemAllRecords Force/S**

Funktion: Entfernt alle Records aus dem aktuellen Projekt. Die Maske bleibt unangetastet. Wenn Sie nicht **Force** angeben, entspricht dieses Kommando dem Menüpunkt **Record/Alle Löschen**, so ist es also auch möglich, daß ein Requester erzeugt wird, in dem gefragt wird, ob das Projekt vorher noch gespeichert werden soll. Um dies zu umgehen, sollte **Force** angeben. Um herauszufinden, ob das Projekt nicht gespeichert ist, können Sie **F_IsVirgin** (siehe Abschnitt 11.2.31) benutzen.

Dieses Kommando kann nur im Record-Modus aufgerufen werden.

Argumente: **Force** - Alle Warnungen unterdrücken.

Ergebnisse: Ein Projekt ohne Records.

Siehe auch:

11.2.48 F_RemRecord

Name: F_RemRecord – Entferne den aktiven Record

Synopsis: F_RemRecord Force/S

Funktion: Entfernt den aktuellen Record und aktiviert den nächsten. Wenn Sie den Force-Parameter nicht angeben, macht dieses Kommando exakt das selbe wie Record/Löschen. Das bedeutet, daß ein Requester erzeugt werden kann, der Sie vor dem Entfernen warnt. Um dies zu unterdrücken, geben Sie Force an.

Diese Funktion kann nur im Record-Modus aufgerufen werden.

Argumente: Force – Alle Warnungen unterdrücken.

Ergebnisse: keine

Siehe auch: F_AddRecord, Record/Löschen

11.2.49 F_RequestChoice

Name: F_RequestChoice – Eine Auswahl anfordern

Synopsis: F_RequestChoice Body/A,Gadgets/A,Title/K
result = *Selection*

Funktion: Erzeugt einen Intuition Easy-Requester mit den angegebenen Parametern. Arbeitet sehr ähnlich zum CLI-Kommando RequestChoice. Die Unterschiede: Leicht andere Parameter, erzeugt den Requester auf Fiascos Screen.

Dieses Kommando kann auch im virtuellen Status aufgerufen werden.

Argumente: Body - Haupt-Text des Requesters.

Gadgets - Gadgets am unteren Rand des Requesters. Jedes Gadget muß durch einen — getrennt werden.

Title - Titel des Requesters.

Ergebnisse: result - Nummer des ausgewählten Gadgets, 0 für das rechte.

Siehe auch:

11.2.50 F_RequestFile

Name: F_RequestFile – Eine Datei abfragen

Synopsis: F_RequestFile File,Pattern/K,Title/K,Savemode/S,Drawersonly/S,
Noicons/S
rc = *Success*
result = *SelectedFile*

Funktion: Erzeugt einen ASL-Filerequester. Arbeitet sehr ähnlich zu dem CLI-Kommando RequestFile. Die Unterschiede: Leicht andere Parameter, erzeugt den Requester auf Fiascos Screen.

Dieses Kommando kann auch im virtuellen Status aufgerufen werden.

Argumente: File - Startwert für Datei (incl. Pfad)

Pattern - Startwert für Pattern

Title - Titel für Requester

Savemode - Aktiviert Savemode: Schwarzer Hintergrund, keine Auswahl via Doppelklick

Drawersonly - Zeigt nur Verzeichnisse an

Noicons - Filtert Icons aus

Ergebnisse: rc = 0, wenn der Benutzer eine Datei ausgewählt hat, ansonsten hat der Benutzer abgebrochen.

result = Ausgewählte Datei, wenn rc = 0

Siehe auch:

11.2.51 F_ResetStatus

Name: F_ResetStatus – Setze das Status-Gadget zurück

Synopsis: F_ResetStatus

Funktion: Setzt das Status-Gadget des Service-Fensters auf den normalen Inhalt. Dies ist RecordNummer / AlleRecords im Record-Modus oder X / Y im Masken-Modus. Sie sollten dieses Kommando benutzen, um die Status-Informationen, die mit F_SetStatus (siehe Abschnitt 11.2.60) eingestellt wurden, zurückzusetzen.

Argumente:

Ergebnisse:

Siehe auch:

11.2.52 F_SaveProject

Name: F_SaveProject – Speichert das aktive Projekt

Synopsis: F_SaveProject

Funktion: Speichert das aktive Projekt unter dem aktuellen Namen auf Disk. Entspricht dem Menüpunkt **Projekt/Speichern** (siehe Abschnitt 10.2.7).

Argumente: keine

Ergebnisse: keine

Bugs: Informiert das Script nicht über Fehlerzustände.

Siehe auch: F_SaveProjectReq

11.2.53 F_SaveProjectReq

Name: F_SaveProjectReq – Öffne Requester und speichere

Synopsis: F_SaveProjectReq

Funktion: Öffnet einen FileRequester, mit dem der Benutzer einen Namen angeben kann, unter dem das aktuelle Projekt anschließend gespeichert wird. Entspricht dem Menüpunkt **Projekt/Speichern Als** (siehe Abschnitt 10.2.8).

Argumente: keine

Ergebnisse: keine

Bemerkung: Da der Benutzer im Requester auch **Abbrechen** anwählen kann, ist es möglich, daß das Projekt nicht gespeichert wird.

Siehe auch: F_SaveProject

11.2.54 F_SaveSettings

Name: F_SaveSettings – Sichere die aktiven Programm-Einstellungen

Synopsis: F_SaveSettings

Funktion: Sichert die aktuellen Programm-Einstellungen (in den Menüs **Settings** und **User**). Wenn Fiasco später neu gestartet wird, werden diese Einstellungen sofort gültig. Dieses Kommando entspricht dem Menüpunkt **Settings/Save Settings**.

Argumente: keine

Ergebnisse: keine

Siehe auch: Settings/Save Settings

11.2.55 F_SelectProj

Name: F_SelectProj – aktiviere ein bereits geladenes Projekt

Synopsis: F_SelectProj Name/A
rc = *Success*

Funktion: Aktiviert ein Projekt, das sich bereits im Speicher befindet. Zur Identifikation dient der Dateiname, ohne Pfad! Dies kann der Name, der durch F_GetProjName (siehe Abschnitt 11.2.21) gewonnen wurde sein. Alle nachfolgenden Kommandos beziehen sich dann auf das aktuelle Projekt.

Argumente: Name - Name des Projektes ohne Pfad.

Ergebnisse: rc = 5, falls Projekt bereits aktiv.
= 10, falls es dieses Projekt nicht gibt oder Argument fehlt.

Siehe auch: F_GetProjName

11.2.56 F_SetFieldCont

Name: F_SetFieldCont – Ändere den Inhalt eines Feldes

Synopsis: F_SetFieldCont Field/A,Cont/A/F
rc = *Success*

Funktion: Setzt den Inhalt des Feldes mit der angegebenen ID im aktuellen Record auf das Argument Inhalt.

Kann nur im Record-Modus aufgerufen werden.

Dieses Kommando kann auch im virtuellen Status aufgerufen werden.

Argumente: Field - Identifikationsname des Feldes

Cont - Neuer Inhalt für das Feld. Dieses Argument nimmt die gesamte restliche Kommandozeile ein, inklusive Leerzeichen. Die Interpretation dieses Arguments hängt von Feldtypen ab:

String - wird direkt übernommen

Integer - Zahlen werden direkt übernommen, anderes ergibt 0

Float - entspricht Integer

Boolean - 1 oder TRUE = gesetzt, 0 oder FALSE = nicht gesetzt
 Slider - Zahl wird übernommen, zu große bzw. zu kleine werden angepasst.
 Cycle - Zahl oder Name des Eintrags wird übernommen
 Date - Datum im Format DD.MM.[YY]YY wird übernommen.
 Time - Zeit im Format HH:MM:SS wird übernommen.
 Extern - wird direkt übernommen
 Datat. - wird direkt übernommen

Ergebnisse: rc = 0 - kein Fehler
 rc = 5 - kein Record aktiv
 rc = 10 - Fehlendes Argument oder ID ist falsch

Siehe auch:

11.2.57 F_SetMode

Name: F_SetMode – Wähle den Bearbeitungs-Modus

Synopsis: F_SetMode Mask/S, Records/S
 rc = *Success*

Funktion: Aktiviert den angegebenen Modus für das aktive Projekt.

Argumente: Mask - aktiviert den Masken-Modus.
 Records - aktiviert den Record-Modus.
 Mask und Records dürfen nicht gleichzeitig angegeben werden.

Ergebnisse: rc = 0 - kein Fehler
 = 5 - das Projekt war bereits im gewünschten Modus
 = 10 - Fehlendes oder falsches Argument

Siehe auch:

11.2.58 F_SetSearchField

Name: F_SetSearchField – Setze das Suchfeld

Synopsis: F_SetSearchField Field/A
 rc = *Success*

Funktion: Setzt das aktuelle Bezugsfeld für die Suche auf das Feld, das als Argument angegeben wurde.

Argumente: Field - Die Id des Feldes, in dem gesucht werden soll.

Ergebnisse: Gibt 10 zurück wenn das Argument fehlt, oder die Id unbekannt ist, ansonsten 0.

Bemerkung: Sie benötigen diese Funktion nicht für einfaches Suchen, da Fiasco 1.1 es ermöglicht, diese Parameter direkt an die Suchfunktionen weiterzugeben.

Siehe auch: F_SetSearchPat

11.2.59 F_SetSearchPat

Name: F_SetSearchPat – Setze das Suchmuster

Synopsis: F_SetSearchPat Pattern/A/F
rc = *Success*

Funktion: Setzt das Suchmuster für das aktive Projekt. Anschließend kann, falls mit F_SetSearchField auch ein Suchfeld eingestellt wurde, mit F_FindFirst und F_FindNext nach dem Muster gesucht werden. Der Wert wird im Suchrequester benutzt.

Argumente: Pattern - Ein String, nach dem gesucht werden soll.

Ergebnisse: rc = 0 - alles Ok.
rc = 10 - Argument fehlt
rc = 20 - interner Systemfehler (Speichermangel, etc.)

Bemerkung: Sie benötigen diese Funktion nicht für einfaches Suchen, da Fiasco 1.1 es ermöglicht, diese Parameter direkt an die Suchfunktionen weiterzugeben.

Siehe auch:

11.2.60 F_SetStatus

Name: F_SetStatus – Zeige einen Status-Text an

Synopsis: F_SetStatus String/A

Funktion: Zeigt den übergebenen String im Status-Gadget des Service-Fensters (siehe Abschnitt 10.1) an. Wenn das Service-Fenster nicht offen ist, passiert nichts.

Argumente: String – Der anzuzeigende String

Ergebnisse:

Siehe auch: F_ResetStatus

11.2.61 F_Sort

Name: F_Sort – Sortiere die Records

Synopsis: F_Sort Field/A,Descending/S

Funktion: Sortiert die Records des aktiven Projektes nach alphabetischer oder entsprechender Reihenfolge des Inhalts eines Feldes.

Argumente: Field – Das Feld, nach dem sortiert werden soll
Descending – Angeben, falls rückwärts sortiert werden soll.

Ergebnisse:

Siehe auch: F_SortReq, Compare/Sort

11.2.62 F_SortReq

Name: F_SortReq – Öffne den Sortier-Requester

Synopsis: F_SortReq

Funktion: Öffnet den Sort-Requester, in dem der Benutzer die Möglichkeit hat, das Projekt nach bestimmten Kriterien zu sortieren. Dieses Kommando macht genau das selbe wie der Menüpunkt **Vergleichen/Sortieren**.

Kann nur im Record-Modus aufgerufen werden.

Argumente:

Ergebnisse:

Siehe auch: F_Sort

11.2.63 F_ToggleAllMarks

Name: F_ToggleAllMarks – schalte die Markierungen um.

Synopsis: F_ToggleAllMarks

Funktion: Löscht die Markierungen bei Records, die markiert sind und setzt die Markierungen bei Records, die nicht markiert sind. Macht genau das selbe wie **Record/Markierungen umschalten** (siehe Abschnitt 10.2.29).

Argumente:

Ergebnisse:

Siehe auch: F_MarkRecord

11.2.64 F_UnlockGUI

Name: F_UnlockGUI – Entriegle die GUI von Fiasco

Synopsis: F_UnlockGUI
rc = Success

Funktion: Schließe die GUI von Fiasco auf, die vorher durch F_LockGUI (siehe Abschnitt 11.2.34) verschlossen wurde. Der Benutzer hat dann wieder Zugriff auf Fiasco. F_LockGUI und F_UnlockGUI können verschachtelt werden.

Argumente: keine

Ergebnisse: rc ungleich 0, wenn nicht verriegelt.

Siehe auch: F_LockGUI

11.2.65 F_UnmarkAllRecords

Name: F_UnmarkAllRecords – Lösche die Markierungen aller Records

Synopsis: F_UnmarkRecord

Funktion: Löscht alle Markierungen der Records im aktiven Projekt. In der Liste werden sie normal angezeigt. Macht genau das selbe wie Record/Alle Markierungen löschen (siehe Abschnitt 10.2.28).

Argumente:

Ergebnisse:

Siehe auch: F_UnmarkRecord F_MarkAllRecords

11.2.66 F_UnmarkRecord

Name: F_UnmarkRecord – Lösche die Markierung eines Records

Synopsis: F_UnmarkRecord Record/N

Funktion: Löscht die Markierung eines Records. Macht genau das selbe wie Record/Markierung löschen (siehe Abschnitt 10.2.26).

Dieses Kommando kann auch im virtuellen Status aufgerufen werden.

Argumente: Record/N – Optional: Record-Nummer des Records, dessen Markierung gelöscht werden soll. Wenn nicht angegeben, wird die Markierung des aktiven Records gelöscht.

Ergebnisse:

Siehe auch: F_UnmarkAllRecords F_MarkRecord

11.2.67 F_UserCommand

Name: F_UserCommand – Ruft ein Benutzer-Definiertes Kommando auf.

Synopsis: F_UserCommand Command/N/A

Funktion: Ruft ein Kommando auf, das durch den Benutzer im “User”-Menu definiert wurde.

Argumente: Kommando-Nummer - Nummer des Kommandos: 0 = erstes Menu-Item, 1 = zweites, usw. Falls eine Kommando-Nummer nicht existiert, wird sie ignoriert.

Ergebnisse: keine

Note: Dieses Kommando sollte nur für das Implementieren einer Icon Leiste oder Ähnlichem benutzt werden. Benutzen sie es nicht in normalen Scripts, da Sie nicht wissen können, was hinter dem Kommando steckt.

Siehe auch:

11.2.68 F_VirtualMode

Name: F_VirtualMode – Ist Fiasco im virtuellen Status?

Synopsis: F_VirtualMode rc = *Virtual*

Funktion: Testet, ob das laufende Script von Fiasco im virtuellen Status oder im normalen Status aufgerufen wurde.

Argumente:

Ergebnisse: $rc = 0$: virtueller status
 $rc \neq 0$: normaler status

Siehe auch: Abschnitt Virtuelle Felder

Kapitel 12

Beispiel-Projekte

Im Verzeichnis Databases liegen mehrere Fiasco-Projekte, die z.T. auch so für eigene Zwecke weiterverwendet werden können. Hier folgen die Beschreibungen dieser Projekte:

12.1 Adressbuch

Das Adress-Projekt kann als Adressbuch benutzt werden, in dem Name, Adresse, Anrede usw. eingestellt werden kann. Das Feld `Country` wird über eine Relation mit der Datei `country.fdb` und dem Länderkürzel `SCountry` als Schlüssel kontrolliert. So braucht man nur noch das Länderkürzel einzugeben und schon wird das richtige Land herausgesucht.

Die Felder für Telefon, Fax und Postleitzahl sind String-Felder, da hier auch Sonderzeichen wie “/” oder eine “0” am Anfang beachtet werden müssen.

Eine weitere Idee wäre, den Namen der Stadt nach der Postleitzahl über eine Relation zu laden.

12.2 Datatypes Demo

Dieses Projekt ist eine einfache Demonstration des Datatypes-Feldtyps der die `datatypes.library` benötigt und so erst ab OS 3.0 verfügbar ist. In der Maske gibt es drei Felder, in denen alle Daten angezeigt werden können, für die die entsprechenden Datatypes-Treiber vorhanden sind.

Zwei Felder haben Scrollbalken unten und rechts, damit man die angezeigte Datei frei bewegen kann. Im Stringgadget unter der Anzeige befindet

sich der Name der Datei. Mit dem Gadget, das mit einem Pfeil nach unten markiert ist, kann dieser Name auch mit einem Filerequester verändert werden.

Ein Feld hat ein weiteres Gadget, das mit einem ‘S’ markiert ist. Dieses Feld kann zum Speichern der Daten verwandt werden. Fiasco öffnet einen Filerequester, in dem die Zielfeldauswahl gewählt werden kann und speichert die Daten dann im IFF-Format.

Das Feld oben rechts hat das ‘sofort spielen’-Attribut, das die Daten sofort nach dem Laden abspielt — natürlich nur wenn sie abspielbar sind.

Da die Daten in einer externen Datei liegen und bei jedem Record-Wechsel nachgeladen werden müssen, kann es dabei zu leichten Verzögerungen kommen.

12.3 Stammbaum

Der Stammbaum besteht aus den Projekten “persons.fdb” und “families.fdb”. In “persons.fdb” müssen alle Personen, die im Stammbaum vorkommen sollen, aufgeführt werden. Hier können auch Geschlecht, Beruf, Geburtsdatum, usw. eingetragen werden.

Diese Daten werden dann von “families.fdb” über Relationen benutzt, um Namen von Ehepartnern, Kindern zu holen. Weiterhin sind hier noch Datums- und String-Felder für Heirat und Scheidung. Durch die intensive Nutzung von Relationen gibt es in diesem Projekt nur noch 10 “echte” Felder, die auf Disk abgelegt werden. Die restlichen 12 werden aus “persons.fdb” geladen.

12.4 Videos

In der Video-Datenbank können Homevideos komfortabel verwaltet werden. Die Datenbank besteht aus zwei Projekten: “movies.fdb” und “tapes.fdb”. In “movies” werden Daten zu einzelnen Filmen gespeichert, u.a. Genre, Regie, Darsteller und Jahr. Über das Feld “Kassette” wird jeder Film einer Kassette zugeordnet, die in “tapes” zu finden ist. Hier sind Marke und Länge der Kassette definiert. Über ein ARexx-Script wird dann die restliche freie Zeit jeder Kassette berechnet.

Dies funktioniert z.Zt. übrigens nur, wenn beide Projekte gleichzeitig geladen sind!

12.5 Bilder-Datenbank

Diese Datenbank benutzt den Datatypes-Feldtyp und ist so nur benutzbar, wenn Sie Amiga OS 3.0 oder höher haben. Das Datatypes-Feld benutzt das "Verzögern"-Attribut; wenn Sie das Bild laden wollen, müssen Sie in das String-Gadget des Feldes klicken und Return drücken.

Das String-Feld unter dem Datatypes-Feld nimmt eine Beschreibung des Bildes auf.

Der Schalter "Scan directory" kann zum einlesen eines beliebigen Verzeichnisses in Fiasco benutzt werden. "Show on screen" zeigt das Bild auf einem eigenen Bildschirm an. Dieser Schalter arbeitet z.Zt. nur, wenn Sie absolute Pfade für die Dateien benutzen.

12.6 FAQs-Datenbank

Diese Datenbank verwaltet Textdateien. Zum Anzeigen der Dateien wird das Programm Most von Uwe Röhm benutzt. Wenn Sie ein anderes Textanzeigeprogramm bevorzugen, und dieses ARexx unterstützt, können Sie die ARexx-Scripts entsprechend anpassen.

Das String-Feld am oberen Rand der Maske nimmt den Namen der Datei auf. Darunter befinden sich drei Button-Felder, die zum Kontrollieren der Datenbank benutzt werden können. "Scan dir" liest ein auswählbares Verzeichnis in die Datenbank ein. "View" zeigt die gerade aktive Textdatei an. Der komplexeste Schalter ist "Search". Hiermit können Sie alle Dateien in der Datenbank nach einem String durchsuchen. Wenn Sie den Button anklicken, öffnet sich ein Fenster, in dem sie einen Suchstring eingeben können. Dann wird nach dem Record gefragt, bei dem die Suche beginnen soll. Wenn Sie einfach Enter drücken, beginnt die Suche beim ersten Record. Dann werden Sie gefragt, ob sie alle Records durchsuchen wollen oder nur die markierten. Danach haben Sie noch die Möglichkeit, die Ergebnisse der Suche in einer Datei zu speichern. Die letzte Option ist "Interactive Searching". Wenn dies aktiv ist, werden Sie jedesmal, wenn der String gefunden wurde, gefragt, ob Sie die Datei an der Fundstelle betrachten möchten.

Wenn Sie die Suche zwischendurch abbrechen möchten, müssen sie Ctrl-C drücken.

Anhang

Anhang A

Alle Suchmuster

#?	String, Extern, Datatypes	Ein unbekannter String mit unbekannter Länge
?	String, Extern, Datatypes	Ein unbekanntes Zeichen
> x	Integer, Slider	Eine Zahl, die größer als x ist
< x	Integer, Slider	Eine Zahl, die kleiner als x ist
>= x	Integer, Slider	Eine Zahl, die größer oder gleich x ist
<= x	Integer, Slider	Eine Zahl, die kleiner oder gleich x ist
!= x	Integer, Slider	Eine Zahl, die ungleich x ist

Detaillierte Beschreibungen sind in den Felddokumentationen.

Anhang B

Relations-Checkliste

- Erzeuge Schlüssel-Feld “dort”. Eventuell “eindeutiger Schlüssel” aktivieren.
- Erzeuge eigentliches Feld “dort”. Bei String, Extern oder Datatypes “Max. Zeichen” merken.
- Projekt speichern.
- Erzeuge Schlüssel-Feld “hier”. Muß den selben Typ wie “dort” haben.
- Erzeuge eigentliches Feld “hier”. Muß den selben Typ wie “dort” haben. Bei String, Extern oder Datatypes muß “Max. Zeichen” gleich sein.
- Projekt speichern.
- Öffne Relations-Requester für eigentliches Feld “hier”.
- Schlüssel “hier” auswählen.
- Relations-Datei auswählen
- Schlüssel und eigentliches Feld “dort” auswählen. Wenn das gewünschte Feld nicht angezeigt wird, Typ und bei String, Extern oder Datatypes Max. Zeichen überprüfen.
- Ok anwählen

Anhang C

Implementation der Clipboard-Unterstützung

Die Menüpunkte Record/Ausschneiden, Record/Kopieren und Record/Einfügen benutzen das Clipboard, um die Daten temporär zu speichern. Das Clipboard des Amiga OS soll eine Schnittstelle für verschiedene Programme sein, um bestimmte Daten auszutauschen. Um dies sicherzustellen darf das Clipboard nur IFF-Daten enthalten.

Fiasco benutzt Einheit 0 des Clipboards und speichert seine Daten in IFF-FTXT-Dateien in einem besonderen Format. Jedes Feld hat einen eigenen Chunk. In diesem Chunk ist der Feldinhalt im ASCII-Format abgelegt.

Die Reihenfolge hängt mit der internen Feld-Liste von Fiasco zusammen. Fiasco benutzt auch diese Reihenfolge, um herauszufinden, welche Daten zu welchem Feld gehören.

Mit den meisten anderen Programmen können Sie so strukturierte IFF-FTXT Dateien nicht erzeugen. Das Einfügen in andere Programme ist besser unterstützt. Zum Beispiel fügt das Conclip-Programm die Daten korrekt ein, während MultiView nur den ersten Chunk anzeigt.

Anhang D

Bugs

Falls Sie irgendwelche Bugs oder Fehlfunktionen in Fiasco entdecken, wenden Sie sich bitte an mich. Bitte beschreiben Sie den Fehler detailliert, d.h. geben Sie an, wie sich der Fehler ausdrückt, und geben Sie eventuell im Hintergrund laufende Programme und die Computer-Konfiguration an.

Diese Fehler sind z.Zt. bekannt:

- Unter Kick 37.x flackert der Fensterrahmen des List-Fensters in höchst merkwürdiger Weise, wenn Menuoperationen ausgeführt wurden, und das Fenster aktiv war.
- ARexx scheint Probleme mit Dateinamen zu haben, die Leerzeichen enthalten. Der Name wird dann immer nur bis zum Leerzeichen interpretiert. Da Fiasco vorher den Dateinamen zum vollen Pfad expandiert, wirkt sich dies auch auf den vollen Pfad aus.
- Scheint manchmal (nicht immer!) etwas Speicher nicht freizugeben.

Anhang E

Was es noch zu tun gibt

Natürlich ist an Fiasco noch lange nicht alles perfekt. Hier ist eine Liste aller Dinge, die ich noch irgendwann ändern oder hinzufügen will. Wenn Sie auch Ideen haben, wie man Fiasco noch verbessern können, schreiben sie mir!

- Besseres Scrolling im Masken-Fenster. Durch die Benutzung von Gadgets der GadTools-Library müssen alle Gadgets neu erstellt werden, wenn sie bewegt werden müssen. Ich plane, die benutzen GadTools-Gadgets in Zukunft zu emulieren, wobei der Benutzer möglichst wenig merken sollte. (Tools wie CycleToMenu würden dabei jedoch nicht mehr funktionieren) Außerdem könnte ich mir dieses sch**ß GZZ-Fenster sparen.
- Neue Suchfunktion.
- Sortieren sollte sich auf mehrere Felder beziehen und sollte schneller werden.
- Neue ARexx-Kommandos: ReadRecord und WriteRecord. Sollten alle Feldinhalte eines Records durch ARexx-Variablen mit der Feld-Id als Inhalt lesen bzw. schreiben können.
- Neue Suchmuster.
- Verbergen von Feldern in der Maske.
- Druckfunktion; Reports
- Mehrzeilige Stringfelder

- Umgekehrte Logik beim Suchen/Zählen/Sortieren
- AppWindows für Datatypes und Extern-Felder
- “Packen” von Projekten, indem nach unbenutzten Feldern gesucht wird, und benutzte auf die benutzte Länge verkleinert werden.
- Überprüfen, ob ein ähnlicher Record bereits vorhanden ist (automatisch)
- Möglichkeit, die Reihenfolge anzugeben, wie die Felder nach Return aktiviert werden.
- Status vom Listen-Fenster speichern.
- Fiasco sollte nicht die ganze Datei lesen müssen. (Um Speicher zu sparen)
- Iconify von Projecten
- ARexx-Kommandos erweitern, damit sie sich auch auf andere Records als dem aktiven beziehen können.
- Trennbalken für die Maske
- Service-Fenster verbirgt die unbenutzten Gadget statt sie zu verschließen
- Listen-Feldtyp mit “Neu” und “Löschen”
- “ARexx-Hook” für das erweitern von Relationen
- Bessere Unterstützung des Masken-Modus von ARexx

Index

- ' , 30
- ? , 30
- #? , 30
- Ändern der Spaltenposition, 23
- Öffnen-Menupunkt, 60
- Über Fiasco-Menupunkt, 63

- Absteigend, 82
- Alle Records löschen-Menupunkt, 64
- Alle Spalten sichtbar-Menupunkt, 71
- Als Pattern * benutzen-Menupunkt, 76
- Altes Projekt überschreiben, 106
- AmigaGuide, 17
- Anführungszeichen und ARexx, 92
- Anzeige Optionen-Requester, 87
- Anzeige-Menupunkt, 77
- ARexx, 91
 - Anführungszeichen, 92
 - Debug, 92
 - Fehlersuche, 92
 - Suchen mit, 31
- ARexx-Debug-Menupunkt, 75
- ASCII, 39, 51
- Attribute
 - Script, 91
- Auswahl, 49
- Auto-Öffnen ServiceWin-Menupunkt, 76

- Backslash, 41

- Backups, 75
- Backups Erzeugen-Menupunkt, 75
- Beenden-Menupunkt, 63
- Bemerkungen, 85
- Benutzermenü Ändern-Menupunkt, 78
- Benutzermenü-Requester, 84
- Besonderen Zeichen in Im-Export, 40
- Boolean, 49
- Button, 56

- C, 40
- Checkbox, 49
- Clean up, 23
- Cursor, 57, 107
- Cycle, 49

- Datatypes
 - Animation, 55
 - Schnelle Record-Wechsel, 55
 - Scrollen, 54
 - Sofort Spielen, 55
 - Sound, 55
- datatypes, 54
- Daten Struktur, 21
- Datum, 51
- Dehnung, 23
- Diskettenlaufwerke, 36
- Dort Projekt, 34

- Editor-Menupunkt, 77
- Einstellungen laden-Menupunkt, 78

- Einstellungen Speichern als-Menupunkt, 78
- Einstellungen Speichern-Menupunkt, 77
- Einstrichige Anführungszeichen, 92
- Ersetzen-Menupunkt, 72
- Ersetzen-Requester, 81
- Erster Record-Menupunkt, 66
- Escape Sequenzen in Im-Export, 40
- Export, 39
 - Benötigte Markierungs-Zeichen, 39
 - Struktur von Dateien, 39
- Exportieren
 - Requester, 89
- Exportieren-Menupunkt, 62
- Extern, 53
- F-Tasten, 84
- F_AboutReq, 93
- F_ActivateField, 93
- F_AddFieldReq, 93
- F_AddRecord, 94
- F_ClearProject, 94
- F_CloseList, 94
- F_CloseServiceWin, 95
- F_ConvertField, 95
- F_CountRecs, 95
- F_CountReq, 96
- F_DupRec, 96
- F_Export, 97
- F_FilterReq, 97
- F_FindFirst, 97
- F_FindNext, 99
- F_FindPrev, 99
- F_FindReq, 100
- F_GetFieldAttributes, 100
- F_GetFieldCont, 101
- F_GetProjFullName, 102
- F_GetProjName, 102
- F_GetRecNum, 103
- F_GotoFirstRec, 103
- F_GotoLastRec, 104
- F_GotoNextRec, 104
- F_GotoPrevRec, 104
- F_GotoRec, 105
- F_GotoRecReq, 105
- F_Import, 105
- F_IsMarked, 106
- F_IsVirgin, 107
- F_LoadDTObject, 107
- F_Locate, 107
- F_LockGUI, 108
- F_MakeVirgin, 109
- F_MarkAllRecords, 110
- F_MarkMatch, 110
- F_MarkRecord, 110
- F_NewProject, 111
- F_OpenList, 111
- F_OpenProject, 112
- F_OpenProjectReq, 112
- F_OpenServiceWin, 112
- F_OptionsReq, 113
- F_Progress, 113
- F_Quit, 113
- F_RemAllRecords, 114
- F_RemRecord, 115
- F_RequestChoice, 115
- F_RequestFile, 116
- F_ResetStatus, 116
- F_SaveProject, 117
- F_SaveProjectReq, 117
- F_SaveSettings, 117
- F_SelectProj, 118
- F_SetFieldCont, 118
- F_SetMode, 119
- F_SetSearchField, 119
- F_SetSearchPat, 120
- F_SetStatus, 120
- F_Sort, 121
- F_SortReq, 121
- F_ToggleAllMarks, 121
- F_UnlockGUI, 122

- F_UnmarkAllRecords, 122
- F_UnmarkRecord, 122
- F_UserCommand, 123
- Faktor, 30
- False, 49
- Feld Ändern-Menupunkt, 69
- Feld Duplizieren-Menupunkt, 69
- Feld Entfernen-Menupunkt, 69
- Feld Hinzufügen-Menupunkt, 68
- Feld Konvertieren-Menupunkt, 70
- Feld umwandeln-Requester, 79
- Feld-Requester, 79
- Felder, 21
 - ARexx, 46
 - Attribute, 79
 - Boolean, 49
 - Breite, 46
 - Button, 56
 - Cycle, 49
 - Datum, 51
 - Extern, 53
 - Float, 48
 - Gültigkeit von Attributen, 79
 - Identifikation, 46
 - Integer, 47
 - Quetschen, 79
 - Schieben, 79
 - Slider, 50
 - Standardwert, 46
 - Startwert, 46
 - String, 46
 - Text, 55
 - Typen ändern, 27
 - Umwandeln, 27
 - Verschieben, 57
 - Virtuell, 46
 - Zeit, 52
- Feldtyp-Menupunkt, 68
- fields
 - datatypes, 54
- Filter
 - Markierungen, 28
 - Ändern-Menupunkt, 73
 - An-Menuitem, 73
 - zu Markierungen-Menupunkt, 73
 - Requester, 83
- Float, 48
- Floppy-Diskettenlaufwerke, 36
- Fonts, 22
- Formatstring, 50
- Fremde Daten, 39
- gadgets, 22
- gadtools.library, 22
- Ganzzahlen, 47
- Gehe zu Record-Menupunkt, 66
- Gehe zu Record-Requester, 85
- Giftware, 10
- gtdisplay.library, 17
- GUI, 57
- Hawes, William S., 91
- Hier Projekt, 34
- Hierarchische Strukturen, 13
- Hilfe, 17, 57
- Icons, 75
- Icons Erzeugen-Menupunkt, 75
- IFF, 54
- Import, 39
 - Benötigte Markierungs-Zeichen, 39
 - Struktur von Dateien, 39
- Importieren
 - Requester, 88
- Importieren-Menupunkt, 62
- Inhalt aufräumen, 23
- Integer, 47
- Kopiere Record-Menupunkt, 64
- Lösche alle Markierungen-Menupunkt, 67
- Lösche Markierung-Menupunkt, 67

- Löschen-Gadget, 59
- Leeren-Menupunkt, 60
- Letzter Record-Menupunkt, 66
- Liste, 22
 - Auswählen von Records, 22
 - Feld IDs, 22
 - Layout, 23
 - Markierungen, 28
 - Verbergen von Spalten, 23
 - Verschieben von Spalten, 23
- Liste neu berechnen-Menupunkt, 71
- Listen-Fenster-Menupunkt, 75
- Lokalisation, 17
- Markiere alle Records-Menupunkt, 67
- Markiere Record-Menupunkt, 66
- Markieren-Menupunkt, 73
- Markieren-Requester, 84
- Markierungen, 28
 - Filter, 28
- Markierungen umschalten-Menupunkt, 67
- Markierungen zu Filter-Menupunkt, 74
- Markierungs-Zeichen, 39
- Maske, 22
 - Dehnung, 23
- Masken Modus, 24
- Masken-Modus-Menupunkt, 74
- Maus, 57
- Menuhelp, 57
- Nächster Record-Menupunkt, 65
- Name eines Autors, 85
- narrator.device, 77
- Neu Gadget, 58
- Neu-Menupunkt, 60
- Online-Hilfe, 17
- Optionen-Menupunkt, 61
- Optionen-Requester, 85
- Piktogramme, 75
- Projekt
 - mit ARexx aktivieren, 118
- Projekt Datei
 - Größe, 47, 53, 54
- Projekte
 - Aktives, 59
- Quetschen, 79
- Rückwärts Suchen-Menupunkt, 72
- RAM
 - benötigtes, 61
- RawDoFmt() , 51
- Record ausschneiden-Menupunkt, 64
- Record Duplizieren-Menuitem, 63
- Record einfügen-Menupunkt, 65
- Record hinzufügen-Menupunkt, 63
- Record löschen-Menupunkt, 64
- Record Modus, 24
- Record-Modus-Menupunkt, 74
- Records, 21
 - Auswahl in der Liste, 22
 - Duplizieren, 27
 - Erstellen, 26
 - Klonen, 27
 - Kopieren, 27
- Reelle Zahlen, 48
- Relation Ändern-Menupunkt, 69
- Relation entfernen-Menupunkt, 70
- Relationen, 33, 34
 - Aktualisieren, 62, 76
 - Dort, 34
 - Hier, 34
 - Zugriffsgeschwindigkeit, 35
- Relationen aktualisieren-Menupunkt, 76
- Relationen neuladen-Menupunkt, 61

- Relationen Schreiben-Menupunkt, 76
- Relations-Requester, 86
- RESULT, 92
- Schalter, 56
- Schieben, 79
- Schlüssel, 34
- Schriften, 22
- Screenmode-Requester, 17, 87
- Service-Fenster, 58, 113
 - M, 28
 - Markierungen, 28
- Service-Fenster-Menupunkt, 75
- ServiceWin dynamisch-Menupunkt, 77
- Sicherheits-Abfragen, 27
- Sicherheits-Requester-Menupunkt, 76
- Sicherheitskopien, 75
- Slider, 50
- Sortier-Requester, 82
- Sortieren-Menupunkt, 72
- Spalte anzeigen-Requester, 86
- Spalte sichtbar machen-Menupunkt, 70
- Spalte verbergen-Menupunkt, 70
- Sparen von Diskettenspeicherplatz, 33
- Speicheranforderungen, 61
- Speichermangel, 35, 42
- Speichern als-Menupunkt, 62
- Speichern-Menupunkt, 62
- Sprechen, 77
- Sprechen-Menupunkt, 77
- Statistik, 61
- Statistik-Menupunkt, 61
- String, 46
- Struktur von Import/Export-Dateien, 39
- Such-Requester, 80
 - ARexx, 31
- Suchen mit mehreren Feldern, 31
- Suchen-Menupunkt, 71
- Text, 55
- True, 49
- Verschieben von Feldern, 57
- Virtuelle Felder, 46
- Voriger Record-Menupunkt, 65
- Wahrheitswerte, 49
- Warteuhr, 108
- Weitersuchen-Menupunkt, 71
- Zähl-Requester, 81
- Zählen von Übereinstimmungen, 31
- Zählen-Menupunkt, 72
- Zeichen-Klassen, 41
- Zeichenketten, 46
- Zeit, 52