fiasco

COLLABORATORS

	TITLE : fiasco		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

fiasco

1.1 Fiasco.guide

```
                 Fiasco  RELEASE 1.0

 by Nils Bandener
```

```
                    Giftware

                    my Address

                    Bugs

                    ToDo
```

1.2 Introduction

Fiasco is yet another Database for the Amiga. Originally I just wanted to write a simple Program, which tests one`s vocabulary in English or Latin. Later I implemented the possibility to define more than two fields (answer and question). From then on the program developed more and more and became very similar to a database-program. I had only to make minor changes and there it was! Now Fiasco is grown to a powerful Program with many features.

Basically Fiasco does not differ much from other database-programs. Fiasco does not support hierarchical structures (like AmigaBase), but supports relations (honestly: I have only read about relational Databases, yet, I can not guarantee, that Fiasco is really relational. Fiasco does at least something quite similar :-). Fiasco has also an ARexx interface, which can be used for controlling Fiasco from other Programs or for assigning ARexx scripts to Fields.

The Mask of Fiasco is not defined by a graphic-file, but is created using internal Images and any non-proportional font. Fiasco provides a lot of Fieldtypes. My personal favorite is the Datatypes fieldtype, which can be used to display graphics, animations, texts etc. directly in the Mask .

There is a second way to display the data: The List . The List is much like the Mask fully configurable. However, you cannot use the List for modifying data.

The searchsystem of Fiasco supports "blurred" search and patterns. "Blurred" search means, that Fiasco tests for similarity and not for equality. The threshold, from which Fiasco detects an entry as "similar" may be freely adjusted.

Besides there are sort-, filter- and countfuctions, which are related to the Searchsystem.

(P.S.: Please be lenient, if this text sounds sometimes a bit broken. My English is not as good as it should be. But I work on it ;-)

1.3 Requirements

Fiasco requires an Amiga with at least 1 MB RAM (512 KB could work, too) and OS 2.04 (37.175). Starting with OS 2.1 (38.x) Fiasco is localized. The Datatypes fieldtype requires the datatypes.library, which is available since Amiga OS 3.0 (39.x). The online help requires the amigaguide.library v34 or better, which is included in the Workbench since 3.0. Users of OS 2.0 or 2.1 can get the library from many FD-sites. Fiasco has no special processor-requirements, but a plain 68000 has to do heavy work, if you use Fiasco with several GZZ windows open.

1.4 Installation

```
    Simply doubleclick on the install icon of your preferred language in the
install drawer. You will be instructed what to do.

    If you don`t own the Commodore Installer, you may simply drag the Fiasco
drawer somewhere you want. You may copy the catalogs to locale:catalogs, but
they will work at this place, too. You may delete the unused languages in
"Documentation" and "Examples" and drag the remaining files in the parent
drawers. The files in "Development" and "Install" are not required for normal
work with Fiasco and may be deleted, too.
```

1.5 GZZ windows

If you activated this item, all windows will be opened in the so-called "Gimme-Zero-Zero"-Mode. These windows have a separate layer for the window bars. This prevents gadtools from overwriting them. This mode required a lot of system resources and will slow down small Amigas very quickly.

1.6 features

```
   Fiasco is able to do this:

o  Several projects may be in RAM at the same time. The number of these
   projects is only limited by the RAM.

o  Masks can be uses like any other GUI.

o  Masks, lists and requesters are fully font sensitive.

o  Many fieldtypes: String, Integer, Float, Cycle, Boolean, Slider, Date,
   Time, Extern and Datatypes.

o  Datatypes fields can be used to display graphics etc. directly in the
   mask.

o  ARexx interface for external control and scripts for fields.

o  Freely configurable "User-Menu", which can be used to spawn CLI- and ARexx-
   Programs.

o  Searching allows "blurred" search and patterns.

o  Very flexible list, which supports hiding and resizing of Entries

o  Easy relation handling
```

1.7 giftware

Fiasco is Giftware, that means, that every User of Fiasco may appreciate the work I have done for Fiasco with some gift. This may be money or any other little thing (CD-Roms, Books, etc.) or simply a postcard (or nothing, if you think you could use your money for something better).

My Address:

Nils Bandener

Dekanatsgasse 4

D-34369 Hofgeismar

Germany

1.8 disclaimer

Disclaimer

The Program "Fiasco" and associated files, in the following called Fiasco, are provided "as is". No representations or warranties are made with respect to accuracy, reliability, correctness of Fiasco, either expressed or implied. In no case I am responsible for any damages caused by this software.

Fiasco is not Public Domain. I reserve all rights.
Fiasco Copyright © 1995 Nils Bandener.

Fiasco may be redistributed, as long these conditions are true:

- The program package must be complete. Fiasco consists in Release 1.0 of the files listed
  here
  .

- Fiasco may not be distributed for commercial purposes without a written permission by the author.
  This includes the distribution of Fiasco for too high prices.
  You may only charge a small fee for media and copying.
  The distribution on CD-Roms is allowed, if the price of the CD-Rom is not higher than the price of the "Fresh Fish" CD-Roms of Fred Fish.

1.9 filelist

```
Fiasco_1.0 (dir)
     Development (dir)
       fiasco.cd                       fiasco.cd.info
       fiasco.ct                       fiasco.ct.info
       Fiasco_NewStrings               Fiasco_NewStrings.info
     Databases (dir)
          English (dir)
               PD-Disks (dir)
                 Disks.fdb                       Disks.fdb.info
                 myrequestfile                   ReadFish.rexx
                 ReadFish.rexx.info
               FamilyTree (dir)
                 families.fdb                    families.fdb.info
                 persons.fdb                     persons.fdb.info
               Addresses (dir)
                 Addresses.fdb                   Addresses.fdb.info
                 Countries.fdb                   Countries.fdb.info
               Videos (dir)
                 CalcLen.rexx                    Films.fdb
                 Films.fdb.info                  Tapes.fdb
                 Tapes.fdb.info
               DatatypesDemo (dir)
                 AmigaWorld.ilbm                 AmigaWorld.ilbm.info
                 DatatypesDemo.fdb               DatatypesDemo.fdb.info
                 Hallelujah.8svx
            Addresses.info                   DatatypesDemo.info
            FamilyTree.info                  PD-Disks.info
            Videos.info
          Deutsch (dir)
               PD-Disks (dir)
                 Disks.fdb                        Disks.fdb.info
                 myrequestfile                    ReadFish.rexx
                 ReadFish.rexx.info
               FamilyTree (dir)
                 families.fdb                     families.fdb.info
                 persons.fdb                      persons.fdb.info
               Addresses (dir)
                 Addresses.fdb                    Addresses.fdb.info
                 Countries.fdb                    Countries.fdb.info
               Videos (dir)
                 CalcLen.rexx                     Films.fdb
                 Films.fdb.info                   Tapes.fdb
                 Tapes.fdb.info
               DatatypesDemo (dir)
                 AmigaWorld.ilbm                  AmigaWorld.ilbm.info
                 DatatypesDemo.fdb                DatatypesDemo.fdb.info
                 Hallelujah.8svx
            Addresses.info                   DatatypesDemo.info
            FamilyTree.info                  PD-Disks.info
            Videos.info
       Deutsch.info                     English.info
     Catalogs (dir)
          Deutsch (dir)
            fiasco.catalog
     Documentation (dir)
          Deutsch (dir)
            fiasco.guide                     fiasco.guide.info
          English (dir)
            fiasco.guide                     fiasco.guide.info
       Deutsch.info                     English.info
     Install (dir)
       Deutsch.info                     English.info
       Install
  Databases.info                   Development.info
  Documentation.info               Fiasco
  Fiasco.info                      Install.info
```

1.10 Quick Start

```
                These are the most important things, which you have to know while ↩
                   working
with Fiasco:


 o The program may be started over the Program- or Projecticon

 o There are two working-modes: In the record mode you may edit records,
   search for them etc. The mask mode allows adding or changing of fields.
   The modes are controlled over the menus
                Control/RecordMode
```

```
                 and

                Control/MaskMode
                 o The
                Service-Window
                 makes certain Operations easier, especially
   if you don`t know exactly about shortcuts of the menus. You may open it
   over
                Control/ServiceWin
                .
   ATTENTION: The functions of the gadgets differ in the two modes.

 o The
                List
                , which can be opened with
                Control/ListWin
                , may
   be changed by clicking in the titles of the list. Clicking one time
   activates the column. Over the menu "List" you can do several things with
   this column. If you click at the right border of a title, you can size the
   column. The other space can be used to drag the Column to any other place
   in the list.

 o Certain project options may be changed over the menuitem
                Project/Options
                 o If you have any problems, you may press the help key while  ↩
                    browsing
   through the menu.
```

1.11 Creating a Database

A database consists of two components: On the one hand the data, divided in records. On the other hand the mask, which defines the structure of the data.

To be able to create a mask, you first have to activate the mask mode (Control/MaskMode). After activating this mode, a cursor appears in the mask. With the mouse or the cursor keys you can choose, where the next operation in the mask takes effect. Before creating a new field you have to choose the type of the new field. You can do that using the "Field/Type" menu or using the lowest gadget in the service window.

After these operations you may use Field/Add Field to create a new field. At first, the field requester appears. The gadgets in the requester depend on the supported attributes of the active field type. They are described in the type documentation for each field. It is not sufficient to click on Ok without any other action. Certain attributes, like the Id have to be specified by the User. Fiasco won`t close the requester, if the are any invalid settings in it. After proceeding, the field will appear in the mask.

You may change all attributes in a later operation, with the exception of the fieldtype itself. The position may be changed by mousedragging. The fieldrequester may be opened by doubleclicking on the field or by choosing Field/Edit Field . You should take care, if you want to change the field id. Other Fiasco projects or ARexx scripts, which want to access this field, won`t find it after the change. If you change the value "max chars" of string, extern or datatypes fields, you will be informed, whether you could loose data.

With Field/Remove Field you are able to delete Fields. Attention: If Settings/Security-Requester is not active, all Data in this Field will be immediately freed. While saving the project the data on disk will be also erased.

In the options requester you may specify further parameters for the current project, like mask stretching , name of the author, etc.

Field/Edit Relations works similar to Edit Field . With this menuitem you are able to control relations of this field.

If you have completed the mask, you may return to RecordMode. With Record/Add Record records can be created now.

1.12 Searching in a database

The GUI interface to the Fiasco search function is the search requester . The search requester can be accessed using Compare/Find . You may select, what field will be searched for what pattern. Here are also the controls for the "blurred" search. The search pattern correspondents to a normal entry. If the a boolean field has to be searched, "TRUE" correspondents to a selected field and "FALSE" to a deselected field. Cycle fields take the number of the label (counting from zero) or the real text of the label. Slider-Fields only allow the value. Extern and datatypes fields can only be searched by filename. You cannot search (with the builtin function) for the contents of the files.

The gadgets at the bottom of the requester start the search. If a matching entry is found, the record will be activated. You may search further using the menuitems Compare/Find next and Compare/Find previous

Patterns

blurred Search

Searching with ARexx

Counting

Replacing

Filter

1.13 patterns

Besides the plain inputs, you may use patterns. String fields support the use of patterns, similar to AmigaDOS (not the real, because the "blurred Search" is not compatible with them). ? is equal to one unknown character. "?iasco" would match "Aiasco", "Biasco", "Ciasco", "1iasco", "@iasco", etc. "????" would match entries, which are 4 chars long. #? stands for an unknown number of unknown characters. "A#?" would match for example "Amiga", "Africa", "A" or "ABCD". "?#?" searches for all non-empty entries. Currently, only one "#?" in one pattern is supported. Similar to AmigaDOS, these chars may be "escaped", if you want to search for entries, which contain these special characters. For escaping such a character, you have to write an "’" before it.

Integer and slider fields support also patterns. These are possible: > x, < x, >= x, <= x, != x. ">" only search for numbers greater than x, ">=" only for numbers greater or equal x, "<" only for numbers less than x, "<=" only for numbers less or equal x. "!=" searches only for numbers not equal x. There is not pattern like "==" (equal), because this is represented by the plain number.

The patterns supported by one fieldtype are also documented in the fieldtypes documentation

A summary of all patterns is also available.

1.14 blurred search

"Blurred" search (sounds a little bit strange, if you have a better word for this, write me!) offers you the possibility, to search for entries, which are similar to a pattern. This way you can search for entries, even if you don`t know the exact spelling. The tolerance of the function may be set under "factor". 0 matches only entries, which are exactly equal. 100 matches nearly all entries.

The count function is very good for experimenting with "blurred" search.

1.15 Searching with ARexx

You may also do searches with ARexx. The command F_SetSearchPat sets the pattern, F_SetSearchField sets the field to search. You may search now with F_FindNext or F_FindPrev search for matches. Opposed to the GUI search function, the record won`t be activated. Only the number of it will be returned in RESULT. This number can be used now for example with F_GotoRec .

See the documentation for F_FindFirst for an example on searching with ARexx.

The settings made by F_SetSearchPat and F_SetSearchField will also be used in the search requester.

You may using ARexx create a search function, which supports several fields.

1.16 count

With Compare/Count you can open a requester similar to the search requester. You have also to specify pattern, field and tolerance. If you select "Ok", the matches will be counted. This way you can collect experiences with the "blurred search .

1.17 replace

Compare/Replace is the replace function, with which you are able to replace certain values with others. Here are also patterns possible, but only one value will be inserted. The gadget "Replacement" takes the value, which has to be inserted. If you select the gadget "Confirm", you will be asked for each record, if you really want to replace the value. The record will be displayed while you are asked.

Attention: You can quickly destroy important data with a bad pattern (for example: #?)!!!

1.18 Filter

The Filter function in Fiasco makes it possible to reduce the displayed records to only these records, which match with a pattern. With Compare/Filter you may open the filter requester , which has the same structure as the search requester. If you proceed with "Ok", only these records will be displayed, which match with the specified pattern.

With Record/Next and Record/Previous you may browse through the records. The list also displays only matching records. You may temporarily disable the filter using Compare/Filter On? .

If you create new records while the filter is active, the records will be displayed, regardless if they would match the filter or not. If you change the contents of a existing record, it will be also displayed furthermore. If you want to update the filter, you have to call the filter requester and select "Ok".

1.19 All Searchpatterns

```
                    Pattern  Supported field types      function

--no pattern--   all types            exact matching.

    #?
                String
                ,
                Extern
                ,
                Datatypes
                   An unknown string with undefined length.
                                        May be only used one time in a text.
```

```
?
            String
            ,
            Extern
            ,
            Datatypes
               An unknown character.

> x
            Integer
            ,
            Slider
                         A number, which is greater than x.

< x
            Integer
            ,
            Slider
                         A number, which is less than x.

>= x
            Integer
            ,
            Slider
                         A number, which is greater or equal x.

<= x
            Integer
            ,
            Slider
                         A number, which is less or equal x.

!=
            Integer
            ,
            Slider
                         A number, which is not equal x.


Detailed descriptions are available with the
            field documentations
            .
```

1.20 Relations

(Warning: I never have worked with another database program, which supports relations. I don`t know, whiter my implementation of relations is really the original Idea, but: It works! [my implementation])

Relations are fields, which have not stored their contents in the file of the project of the relations, but in another project file. An additional field with a key is required, which is used to identify the record, from which the data should be taken.

This mechanism prevents the situation, that in many different projects the same data are stored. So it helps to save disk space. Furthermore you have to change only the contents of one field in one of the projects, and all other corresponding fields will recognize that change, too.

For using relations in Fiasco, you have to create a project, which will be the source for another project, which will read the data from there. The source project will be called later "there" and the project, which will read from it will be called "here". (I have searched for good names for a long time. However, these names are only correct from the view, which you will have, when you create the relations, but they have no double meaning, like "source" and "destination", which I used before) You have to create in the "there" project at least two fields, one for the data and one for the key. The field for the key should be an integer field. This is the fastest method. However, it is possible to use all other field types as keys, too.

You may use the special field attribute "gimme unique key", if you want to get automatically a key, whenever you create a new record. Note that the key is only created when you create a new record. If you activate this attribute later, the already existing entries will keep their old value. If you change the contents of such a field, the change will be left without any checking.

It is up to you to change the type of the second field. If you create fields, which store strings (string, extern and datatypes), you should remember the "max chars" value, because you also have to use this value in the second project.

If you want to see any effect after activating the relation, you should create a few records with some content at first.

Now it is time to save the project and to create a new one.

The second project must contain again two fields, which have to match in type and "max chars" if you use string, extern or datatypes. The key field should NOT use "unique key", because you should decide freely, what key do you want to use.

Before you activate the relation, the project should be save in the directory, in which the other project has been saved, in order to be able to use relative and not absolute paths.

Now you can open the relation requester for the field, which is supposed not to contain the key (Field/Edit Relations . Primary, you should select the key "here" in the listview in the upper left edge of the window. After that you should select the other project with the filerequester gadget in the middle of the requester. Now you can select the key and the real field "there". Proceed then with "Ok". If everything works correctly, the requester is closed and the relations will be loaded. Otherwise, a requester will inform you of any failure.

A relation checklist , which contains the information in a compressed form, is also available.

1.21 Relation Checklist

```
 o create key field "there". Optionally activate "unique key".

 o create real field "there". In case of string, extern or datatypes,
   remember "max chars".
 o save project.

 o create key field "here". Must be the same type as "there".

 o create real field "here". Must be the same type as "there". In the case of
   string, extern or datatypes, "max chars" must be equal.
 o save project.

 o open relation requester for real field "here".

 o select key "here"

 o select relation file

 o select key and real field "there". If the correct field is not displayed,
   check type and in case of string, extern or datatypes "max chars".

 o select "Ok"
```

1.22 Fieldtypes

Fields are the stores for data. There are only two basic types ↩ : "string" and "number". All other types are more or less modifications of these types, which make the work with the database easier.

Fiasco supports the following types:

String

Integer

Float

Boolean

Cycle

Slider

Date

Time

Extern

Datatypes

Text

1.23 Standard Attributes

These attributes are normally supported by a field type:

- Id
  This string serves for identification of a field. It is displayed in the mask mode in the fields, in the list header, in the search and related requesters and in the relation requester. You have to use it also in ARexx scripts, if you want to access a field from there.
  This string should be unique in the current project.

- Width
  defines the displayed field width in characters.

- Init Cont
  you may specify a value here, which will be used while creating a new record.

- Use old value
  If you create a new record, the value, which has been used in the old record, will be used in the new record again.

- Script
  You may specify a ARexx script here, which will be called, when a new record is created, or the contents of a field has been changed. It is possible, that "init cont" has not the effect, which has been set in the requester, if the script changes the contents of the field.

By using mask stretching it is possible, that the Dimension-Attributes are changed slightly.

1.24 String fieldtype

A string field takes strings with limited length.

Apart from the standard attributes , this type supports the following attributes:

- Max Chars
  determines, how many chars may be typed in this field. This attribute has direct effect on the size of the project file.

Search equivalent:

correspondents to the content.

Supported search patterns:

- ? = One unknown character.
- #? = No or more unknown characters.

1.25 Integer fieldtype

You may enter integer numbers in the range from -2,147,483,348 to 2,147,483,347 in an integer field.

Apart from the standard attributes , this type supports the following attributes:

- Max Chars
  determines the maximum length of a number in chars.

- Gimme unique Key
  puts in this field whenever a new record is created, a (in this database) unique number. This Attribute is mutually exclusive to "init cont" and "use

```
  old value".

Search equivalent:

  is equal with the field content.

Supported search patterns:

  >  - greater than
  <  - less than
  >= - greater or equal
  <= - less or equal
  != - not equal
```

1.26 Float fieldtype

You may enter real number in a float field.

Apart from the standard attributes, this type supports the following attributes:

- Precision
  Number of digits after the point.

Search equivalent:

  is equal to the field content

Note: The precision of the float field type is not very high. It is recommended to use string fields instead. ARexx is also able to make mathematical operations with string fields, if they contain only numerical characters.

1.27 Boolean fieldtype

A Boolean field can only contain two values: "True" or "False". It appears in the mask as a "checkbox gadget".

These standard attributes are not supported:

- Width (always 3)

Search equivalent:

```
  TRUE or 1  - checked field
  FALSE or 0 - unchecked field
```

Under Amiga OS 2.x this field can look a bit strange, because the images are not scalable. Starting with OS 3.0 the size of it is adjusted to the font size.

1.28 Cycle fieldtype

Cycle fields have several choices from a freely definable list. It helps to save memory. There is a maximum of 65536 choices. (I hope, thats enough ;-)
A cycle field appears in the mask as a "Cycle gadget" (like the name implies).

Apart from the standard attributes, this type supports the following attributes:

- Labels
  A list of all choices. There must be at least one entry, two entries give
  the cycle field a meaning.

"Init Cont" is a number, which correspondents to the labels counting from zero.

Search equivalent:

  the number of the label like "init cont" or the entry itself (write correctly!)

1.29 Slider fieldtype

A slider is related to a integer field. It can be used to display integer number graphically. The range of the numbers goes from -32,768 to 32,767, but may be influenced by several attributes.

Apart from the standard attributes, this type supports the following attributes:

- Min. Value
  defines the lowest value, which is displayed. It correspondents to the
  position of the "knob" at the left or at the upper end of the field.

- Max. Value

defines the highest value, which is displayed. It correspondents to the position of the "knob" at the right or at the lower end of the field.

- Format

  is a formatstring in style of the programming language "C".
  The syntax:

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

  - -: The number is left aligned, normally right
  - 0: The field is padded with zeroes. e.g.: 1 -> 001
  - Field: The minimal field width
  - Maximum: only for strings, no meaning here.
  - l: Says, that the number is 32 bit. This is here always the case.
  - Format:

    %b - BCPL-String, no meaning here.
    %s - C-String, no meaning here.
    %c - Char, the ASCII character for the number is displayed.
    %d - The number is displayed.
    %u - The unsigned number is displayed.
    %x - The number is displayed in hexadecimal format.

  The formatting is done with the exec-function RawDoFmt().

- MaxFormatLen

  the maximum length of the format. This region is in the width region. That means that a higher MaxFormatLen makes the field itself smaller.

Search equivalent:

The number itself.

Supported search patterns:

```
>  - greater than
<  - less than
>= - greater or equal
<= - less or equal
!= - not equal
```

1.30 Date fieldtype

You may enter a date in a date field.

Apart from the
standard attributes
, this type supports the following
attributes:

- use current Date

  When a new record is created, the current date is copied in this field.

Search equivalent:

  is equal to the contents

Currently, the date is only displayed and read in german format (DD.MM.YYYY). No verification of the values is made, this makes values like 65.20.3687 possible.

1.31 Time fieldtype

You may enter a time in a time field.

Apart from the standard attributes, this type supports the following attributes:

- use current Time
  when you create a new record, the current time will be copied in this field.

Search equivalent:

  is equal to the content.

Currently, the time is only displayed with seconds (HH:MM:SS). AM and PM are not supported. No verification of the values is done, that means, that values like 55:66:99 are possible.

1.32 Extern fieldtype

A extern field takes a filename, which will be used on request as argument for a user defined program. This makes it possible to define additional data for a record.

Apart from the standard attributes, this type supports the following attributes:

- Command
  is the name of a program, which is capable to use these files.
  The characters "%s" are replaced with the content of the field.
  If you don‘t use "%s", no arguments will be submitted. (e.g. "C:ED %s")

- Stack
  defines the stack size for a command.

- Max Chars
  defines the maximum length of a filename in chars. This attribute has

direct effect on the size of the project file.

Search equivalent:

is equal to the file name; searching for the contents of the file is not possible.

The programs will be called using the AmigaDOS function System(). A console window will be opened for I/O operations.

1.33 Datatypes fieldtype

A datatypes field is similar to a extern field. The difference is the use of the datatypes.library. This is the reason, why you can use these fields only starting with Amiga OS 3.0. The major advantage is, that the data will be displayed directly in the mask. This field is universal usable and freely extensible.

Apart from the standard attributes , this type supports the following attributes:

- Max Chars
  defines the maximal length of the filename. This attribute has direct effect on the size of the project file.

- Scrollbars
  Determines, if scrollbars shall be created at the bottom and at the right border of the field. Without a scrollbar, you can view the only the upper left of a file. (That is not completely true. Some datatypes scroll their display, if you click in their area and drag the mouse in the direction of the hidden part. One datatype, which supports that is the picture datatype)

Searchequivalent::

Is equal to the filename; You cannot search the content.

The AmigaGuide- and the Animation-Datatype seem to have some problems with Fiasco.

The changing of records gets slower, because the data have to be loaded each time.

1.34 Text fieldtype

Text fields are no real fields, they only serve to put text in the mask.

These attributes are supported:

- Text
  will be written in the mask.

- Pen
  the color, the text shall be written. Normal is normally black and
  Highlight is normally white.

- Bold
  Makes the text bold.

- Italics
  makes the text italic.

- Underlined
  underlines the text.

No
standard attributes
are supported.

search equivalent:

You cannot search for a text field

1.35 Graphic user interface

Fiasco opens normally after a start only an empty window. You can work in
it using pull down menus. The people, who don‘t like pull down menus, may
open an additional window using
Control/ServiceWindow
.
This window makes the most important operations accessible via a mouseclick.
The third way to make operations are keyboard shortcuts.

Menus and shortcuts

Service window

Requester

The mouse can be used in the mask mode to set the cursor and to drag
fields. After a doubleclick on a field, its
field requester
will be opened
(like
Field/Edit field...
).

Fiasco supports Menuhelp. That means, that if you press the help key, while you browse through the menus, a short description will be displayed in an AmigaGuide window (This feature requires amigaguide.library, which is part of the OS since 3.0, if you use older versions, you may get it from the PD).

The requesters used by Fiasco have a standard structure. The gadgets at the bottom are for responding. Normally, the left one is a positive response, while the right one is negative. The close gadget of the window is equal to the negative response. Nearly all gadgets in the requesters may be accessed using the keyboard. The positive response may be activated using the Return key, the negative may be activated with "Esc".

1.36 Service Window

The service window may be opened or closed with
Control/ServiceWindow
.
It contains these gadgets:

```
 Add

 Del

|<

 <

>

>|

 <Filename>

  <Status>

<Fieldtype>
```

1.37 Add

If the current project is in record mode, a new record will be created.
If the mask mode is active, a new field will be created.

Equivalent to:

Records/Add
resp.

Fields/Add field...

1.38 Delete

```
                    If the current project is in record mode, the current record  ↩
                       will be
removed. If mask mode is active, the current field will be remove. Attention:
This will normally happen without any security request!

Equivalent to:

                Records/Remove
                resp.

                Fields/Remove Field
```

1.39 |<

```
                    If the current project is in record mode, the first record  ↩
                       will be
activated.

Equivalent to:

                Records/First
```

1.40 <

```
                    If the current project is in record mode, the previous record  ↩
                       will be
activated.

Equivalent to:

                Records/Previous
```

1.41 >

```
                    If the current project is in record mode, the next record will ↩
                        be
activated.

Equivalent to:

                Records/Next
```

1.42 >|

```
                    If the current project is in record mode, the last record will ↩
                        be
activated.

Equivalent to:

                Records/Last
```

1.43 Active project

```
    The name of the current project is displayed here. If two projects only
differ in the path and not in the name, the same name will be displayed.

    You may activate another project activating the window of a project.
```

1.44 Status

```
                    Here are status informations displayed.

In the record mode:
<number of active record>/<number of records>
A
                Filter
                 may change these numbers.

In the mask mode:
X: <X position of cursor>, Y: <Y position of cursors>
```

1.45 Fieldtype

```
    You may select the fieldtype, which will be used while creating the next
new field, here. Equivalent to "Fields/Field Type"
```

1.46 Mask

The mask is the way to display data, which is used mostly by Fiasco. A mask can display only one Record, as opposed to the List . The advantage of the mask is the clarity of the display.

The mask is built up of so called fields, which have several types and images. A complete description of all field types is here !

If you use normal Amiga programs, you would call these fields "gadgets". Fiasco uses as fields also gadgets (from the gadtools.library).

Fiasco masks adjust automatically to any non-proportional font. For example, topaz and courier are non-proportional.

To create a mask in Fiasco, you have to be in the mask mode. You may change the position of existing fields using the mouse or make other changes with the "Field" menu. More on this topic here .

1.47 list

Control/ListWindow opens a window, which displays the records in a list. The records are represented by lines, while the fields of a record are represented by columns. The first line of a list shows the ids of each field. If the window is not big enough to display the whole list, you may use the scrollbars in the right and in the bottom border of the window to scroll through the list. The line of the current record is marked using a backfill .

You can select records using the list. Click simply on the line of a record. Changes in the record can only made in the mask.

If a Filter is active, the list displays only the matching records.

The layout of the list is normally made automatically. Positions and dimensions of the fields in the mask will be used to determine the dimensions in the list. However, you may change the position and the with of each column in the list. To change the width, you have to click in the header-line at the right corner of a column. One line appears, which shows the actual with of the column. Now you may drag the line using the mouse. The place, where you drop the line (that means, you release the mouse button), will be new right border of the column. Columns, which are overlapped by the column, will be shifted to the right.

The position of a column may be changed, too. Click over the middle of the column-header. You may drag now the column in the list. The column will be inserted near as possible at the place, where you drop it.

If you want to hide columns entirely, you should try List/Hide column . This menuitem hides the column, which is selected. You select a column, clicking one time in the header of the column. The header will then appear highlighted. If you want to get a column back, you have to select List/Show column . A requester will appear, which displays all columns, which have been hidden. If you select one and click on "Ok", the column will be inserted as near as possible at the old place.

List/Recalc list calculates the positions and dimensions of all columns again. You can compare it with "Clean up" of the Workbench. Columns, which have been hidden, are kept hidden.

1.48 Records

Records are the file cards of a database. That means, a record is a collection of several points for one main item (e.g. for a person name, address, etc.). The mask is only able to display one record at the same time. The list displays several records as lines.

1.49 Fillpens

If you use Amiga OS 3.0 or higher, you may manipulate these colors with the Palette Preferences. The "FILLPEN", which is used for filling the background, may be changed with "Active Window Title Bars". The "FILLTEXTPEN", which is used for Text, which is rendered over the "FILLPEN", may be changed with "Active Window Titles".

If you use OS 2.x, "FILLPEN" is normally color 3 (Blue) and "FILLTEXTPEN" color 1 (black). There are some Pubscreen utilities, which support changing of pens on Publicscreens.

The text fieldtype uses an additional pen. The "HIGHLIGHTTEXTPEN" is used, if you activate in the field requester Highlight.

1.50 Stretching of the mask

Normally, the Fields in a Fiasco Mask are placed very tight. This is not very nice, and all other "normal" GUIs leave a few pixels between the gadgets. It would be possible to place one empty line between the fields, but this wastes quickly much place. For this reason, Fiasco makes it possible to leave a few pixels between the gadgets.

These values may be specified in the options requester under Stretch X and Stretch Y.

The owl stretching (ehhhmm -- mask stretching %-) makes fields bigger, than specified in the field requesters. This is evident in the lines, because most Fiasco fields only expand to this directions. String fields may be bigger than the number of chars they can hold. The biggest problem are text fields, because the with of them is normally only the required. The stretching makes them wider and the Text has to be centered.

You should specify as X value only zero to avoid these problems and use one column as separator. In Y direction this value 4 is the best.

1.51 Record Mode

You may add, delete or edit records in this mode. It may be activated with Control/Record Mode . The opposite of the record mode is the Mask Mode

1.52 Mask Mode

This mode makes you able to edit the mask, that means you may create new fields, delete some or change their position or attributes. Relations may also created and changed here. This mode may be activated with Control/Mask Mode . This opposite of the mask mode is the record mode .

1.53 Menus

```
                Fiasco has these pull down menus:
(from left to the right; menus, which are marked with a ’/’, may be
activated or deactivated)



Name                       Keyboardshortcut

Project

              New
               A N

              Erase
               A Z

              Open...
               A O

              Options...
               A $

              Statistic...

              Reload Relations
               A !

              Save
               A S

              Save as...
               A A

              About...
               A ?

              Quit
               A Q

Records
```

```
              Add Record
               A +

              Duplicate Record
               A 2

              Remove Record
               A -

              Previous
               Cursor Up

              Next
               Cursor Down

              First Record
               Ctrl Cursor Up

              Last Record
               Ctrl Cursort Down

              Goto...
               A G

  Fields

              Field Type        »
                       String       Ctrl S
       Integer      Ctrl I
       Float        Ctrl F
       Boolean      Ctrl B
       Cycle        Ctrl C
       Slider       Ctrl S
       Date         Ctrl A
       Time         Ctrl M
       Extern       Ctrl E
       Datatypes    Ctrl D
       Text         Ctrl T

              Add Field...
               Enter

              Edit Field...
               Enter

              Edit Relations...
               A &

              Remove Field
               Del

  List

              Hide column
               A [
              Show column...
               A ]

              Show all colums

              Recalc List
               A %

Compare

              Find...
               A F

              Find next
               A >

              Find previous
               A <

              Replace...
               A R

              Count...
               A #

              Sort...
               A =

              Edit Filter...
               A ~

              / Use Filter?
               A `

Control

              / Edit Records
               A D

              / Edit Mask
               A M

              / ServiceWindow
               A W

              / ListWindow
               A L

              / ARexx-Debug
               A B

Settings

              / Create Icons?

              / Create Backups?
```

/ Write Rels?

/ Update Rels?

/ GZZ-Windows?

/ Security-Reqs?

/ AutoOpen SerWin?

/ Dynamic SerWin?

Choose Font...

Save Settings

Save Settings as...

Load Settings...
User

Edit...
A U

1.54 Project/New

Shortcut: A N

Creates a new project with a mask window. It contains no records or fields. You may create a new database or
Open
a new database.

See also::

Open...

1.55 Project/Erase

Shortcut: A Z

Erases all data in the current project, the project will be in a status like after a
Project/New
.
If data have changed since last saving, you will be asked before the data will be erased.

1.56 Project/Open...

Shortcut: A O

Opens a ASL requester and loads the selected Fiasco project in the current window. If there are any unsaved data, you will be asked, whether you want to save them first.

1.57 Project/Options...

Shortcut: A $

This menuitem opens the options requester , which can be used for editing projectspecific options. That are:

- Mask stretching
- Name of author and annotations
- filename of project

The last point makes it possible to change to name of the project without to have to call Save as .

1.58 Project/Statistic...

no Shortcut

Shows a few informations for the current project. Example:

```
One Record requires about 100 Byte RAM.
200 Records of this project need about 19 KByte RAM.
There is space for about 2300 more Records.
```

The memory required for the project and the basic fields is not included.

1.59 Project/Reload Rels

Shortcut: A !

This item reloads all relations in the current project, like it happened while loading the project. This is particularly useful, if you have deactivated Settings/Update Relations? and changed some keys and want to see the result.

1.60 Project/Save

Shortcut: A S

Save writes the data of the current project under the same name to disk. If you want to save the project under a different name, you have to use

Save as
or
Options
to change the name
and then Save.

1.61 Project/Save As...

Shortcut: A A

Here you may save the current project under a new name. The name will be requested using an ASL requester and will be kept after saving.

1.62 Project/About...

Shortcut: A ?

This item shows a small requester, which gives informations about version, copyright and some system internal data.

1.63 Project/Quit

Shortcut: A Q

This item closes the current project. If it has been changed and has not been saved yet, you will be asked, whether you want to do this.
If this project is the last one, which Fiasco has currently open, Fiasco will exit.

1.64 Records/Add Record

Shortcut: A +

Adds a new
record
to the record list of the current project. Each
Field contains then the "init cont", which is normally nothing. If the list is open, a new line will be inserted.

If a
Filter
is active, the new record will be declared valid automatically. If you want that new Records are filtered correctly, you will have to select
Compare/Edit Filter
and simply click on "Ok".

This menuitem may only be selected in the
record mode
See also:

Records/Remove Record

1.65 Records/Duplicate Record

Shortcut: A 2

Creates an exact copy of the current record. All "init" values will be ignored. Even a field with "gimme unique Key" will contain the old value. That means, that two records with the same "unique" Key will exist.

1.66 Records/Remove Record

Shortcut: A -

Removes the current record and the data in it. If the are relations, which search for a key, which was defined in this record, they will find in the future nothing.

This menuitem may be selected only in the
record mode
.
If you have selected
Setting/Security-Requester?
, you will be asked before proceeding.

See also:

Record/Add Record

1.67 Records/Previous

Shortcut: Cursor up

Activates the record, which is the predecessor of the current. If the current record is the first one, the display will beep.

Please note, that
Filters
change this item. Then the
previous matching record will be activated.

The keyboardshortcut correspondents to the structure of the list, which displays the previous record over the current.

This menuitem may be only selected, if the
record mode
is active.

See also:

Next

First

Last

Goto

Find previous

1.68 Records/Next

Shortcut: Cursor down

Activates the record after the current record. If the current record is the last in the list, the display will "beep". Please note, that
Filters
change
the behavior of this item. In the case of an active filter, the next matching record will be searched.

The keyboardshortcut correspondents to the structure of the list, which displays the next record under the current.

This menuitem may be only selected, if the
record mode
is active.

See also:

Previous

First

Last

Goto

Find next

1.69 Records/First Record

Shortcut: Ctrl Cursor up

Activates the first record of the current project. In the case of an active

Filter
, for the first matching record will be searched.

This item is only in the
record mode
selectable.

See also:

Next

Previous

Last

Goto

1.70 Records/Last Record

Shortcut: Control Cursor down

Activates the last record of the current project. In the case of an active

Filter
, for the last matching record will be searched.

This item may be only selected in the
record mode
See also:

Next

Previous

First

Goto

1.71 Records/Goto...

Shortcut: A G

Opens the
goto requester
, which can be used to activate a
record using it‘s number.
Please note, that the record number may be changed by adding or deleting
records or by using
filters
.

This item can only be selected in the
record mode
.

See also:

Next

Previous

First

Last

1.72 Fieldtype

Select in this submenu the current fieldtype. It will be used, ↩
if you
create fields. The gadget at the bottom of the
service window
has the
same function.

These fieldtypes are available (with Shortcut):

String
Ctrl S

Integer
Ctrl I

Float
Ctrl F

Boolean
Ctrl B

Cycle
Ctrl C

Slider

Ctrl S

Date
Ctrl A

Time
Ctrl M

Extern
Ctrl E

Datatypes
Ctrl D

Text
Ctrl T

1.73 Fields/Add Field...

Shortcut: Return

Opens the
field requester
for the current field type and inserts
the created field at the current cursorposition

This item can only be selected in the
mask mode
.

If there is already a field at the current cursor position, nothing will be done.

Please note, that Return is also shortcut for
Edit Field
. Enter
creates a new field, if no field is currently active, otherwise, it opens the requester for editing the current field.

See also:

Edit Field...

Edit Relations...

Remove Gadget

1.74 Fields/Remove Field

Shortcut: Del

Removes the current field. All data in this field will be lost, too. If there are any relations or ARexx scripts, which refer to this field, they will complain not until they are activated.

This item can only be selected in the
mask mode
.

See also:

Edit Field...

Edit Relations...

Add Field

1.75 Fields/Edit Field...

Shortcut: Return

Opens the
field requester
for the current Field. The
field requester can be used to change several attributes of the field. If certain changes would cause loose of data (e.g. changing "max chars" of a string field to a low number), you will be informed about the problem and have to possibility to cancel the change.
Field types may not be changed this way.

Please note, that Return is also a shortcut for
Add Field
. Return
calls "Add Field", if no field is active, and otherwise "Edit Field".

This item can only be selected in the
mask mode
.

See also:

Add Field

Edit Relations

1.76 Fields/Edit Relations...

Shortcut: A &

This item opens the
relation requester
, which adds a
relation
to the current field.

This item can only be selected in the mask mode .

1.77 List/Hide column

Shortcut: A [

Hides a activated column of the List . You activate a column by clicking in the topmost line of the list, which contains the field ids. After hiding a column, the columns at the right side of it will be shifted to the left.
The column may be made visible again by using Show column .

This item may only be selected, if the list window is open.

1.78 List/Show column...

Shortcut: A]

This item makes the columns, which have been hidden with Hide column , visible again. It tries to place the columns as near as possible at their old positions.

This item may only be selected, if the list window is open.

1.79 List/Show all columns

no shortcut

Makes all columns, which have been hidden using hide column , visible again.

This item may only be selected, if the list window is open.

1.80 List/Recalc List

Shortcut: A %

This menuitem calculates all positions and dimensions of the columns in the list . Hidden columns are not displayed again.

This item can be compared with "Clean up" of the Workbench.

This item may only be selected, if the list window is open.

1.81 Compare/Find...

Shortcut: A F

Opens the search requester , which can be used to define search criteria.

This item is only selectable, if the record mode is active and if the current project contains at least one record.

See also:

search requester

Find next

Find previous

1.82 Compare/Find next

Shortcut: A >

Activates the nextrecord, which matches with the searchcreterions, which have been specified using the search requester . If no matching record is found, you will be informed about that.

This item is only selectable, if the record mode is active and if the current project contains at least one record.

See also:

Search requester

Find...

Find previous

1.83 Compare/Find previous

Shortcut: A <

Activates the previous record, which matches with the searchcriterions, which have been specified with the search requester.
If no record is found, you will be informed about that.

This item is only selectable, if the record mode is active and if the current project contains at least one record.

See also:

Search requester

Find...

Find next

1.84 Compare/Replace...

Shortcut: A R

Opens the replace requester, which can be used for replacing data.

This item is only selectable, if the record mode is active and if the current project contains at least one record.

1.85 Compare/Count...

Shortcut: A #

Opens the count requester

, which can be used to determine the number of the records matching with the specified pattern.

This item is only selectable, if the record mode is active and if the current project contains at least one record.

See also:

Find...

1.86 Compare/Sort...

Shortcut: A =

Opens the sort requester, which may be used to sort the records of the current project.

This item is only selectable, if the record mode is active and if the current project contains at least one record.

1.87 Compare/Edit Filter...

Shortcut: A ~

Opens the filter requester, which can be used to create Filters.

This item is only selectable, if the record mode is active and if the current project contains at least one record.

1.88 Compare/Use Filter?

Shortcut: A ` (Gray key at the upper left)

This item can be used to switch the filter on or off. If no filter has been

created yet, the
filter requester
will be opened.

This item is only selectable, if the
record mode
is
active and if the current project contains at least one record.

1.89 Control/Record Mode

Shortcut: A D

This item switches the current project to
record mode
, in which records
and the contents of records can be changed. If this mode is active , a
checkmark will be set at the left side of the item.

See also:

Record mode

Mask mode

1.90 Control/Edit Mask

Shortcut: A M

This item switches the current project to
mask mode
, in which the
mask
can be changed. If this mode is active , a checkmark will be set at the
left side of the item.

See also:

Mask mode

Record mode

1.91 ServiceWindow

Shortcut: A W

This item controls the

service window
. If it is
checked, the service window is open. The service window makes the most important record- and mask-operations easier and displays some status informations.

The service window serves globally for all projects.

1.92 ListWindow

Shortcut: A L

This item controls the
list window
, if it is checked, the list
is open.

Each project may have it‘s own list window.

1.93 ARexx-Debug

Shortcut: A B

This activates a special Debug-Mode of Fiasco for the
ARexx interface
.
If Fiasco commands fail, an additional request will appear, which describes the problem more extensively.

1.94 Settings/Create Icons?

If this item is checked, Fiasco will create icons while saving projects.

1.95 Settings/Create Backups?

This item determines, whether Fiasco creates backups of old projects while saving new projects. The backup file will be named <oldname>.bak.

1.96 Settings/Write Relations?

If this item is checked, Fiasco will also write relations back in their "there" projects. Otherwise, changes made in these fields will be lost. This item should be only active, if Update Relations? is also active, or if you call Project/Reload Rels before saving. Otherwise you risk, that data in the "there" project will be overwritten by invalid data in some fields of the "here" project.

1.97 Settings/Update Rels?

This item determines, whether relations are updated immediately after the input of a new key. This requires disk accesses, which may become annoying if you do not own a harddisk. If you deactivate this item, you should also deactivate Write relations?, because there may be invalid data in the project, which would be written into the "there" file. If you want to see the changes, you can update the relations using Project/Reload Rels.

1.98 GZZ Windows?

If this item is checked, Fiasco will open it`s mask windows (and only these) are opened in the so called "GimmeZeroZero" Mode. This prevents Intuition from rendering it`s gadgets over the window borders. Small Amigas may get slowed down by this mode.

1.99 Settings/Security-Reqs?

If this item is checked, Fiasco will request you again before deleting any fields or records. This can prevent erroneous deleting and loss of data.

1.100 Settings/Auto-Open ServiceWin?

If this item is checked, the service window will be opened on every program start automatically.

1.101 Settings/Dynamic ServiceWin?

If this item is checked, the service window will be opened at an free place. Otherwise, fixed coordinates will be used.

1.102 Settings/Choose Font...

Opens a font requester for selecting a font, which will be used for rendering of the text in the mask and the list .

1.103 Settings/Save Settings

Saves the current settings under "env:fiasco.prefs" and "envarc:fiasco.prefs". They "survive" reboots.

1.104 Settings/Save Settings as...

Saves the settings in a file, which has been specified with an ASL requester. If you save the file in "env:", the settings won‘t survive an reboot. If you save them only in "envarc:", they will get first active after rebooting, because Fiasco searches for it‘s current settings only in "env:".

1.105 Settings/Load Settings...

Loads a specified settings file and uses them. To use them also after reboots, you should select Save Settings to write them to "env:" and "envarc:".

1.106 User/Edit...

Shortcut: A U

Opens the
Usermenu requester
, which can be used to
define Usermenus.

1.107 All requesters

Edit Field

Find

Replace

Count

Sort

Filter

Usermenu

Options

Goto

Edit Relation

Show column

1.108 Field requester

The field requester can be used to change the attributes of a field.
Each fieldtype has a different field requester, because the gadgets of the
field requester represent the supported attributes of each fieldtype. The
supported attributes are listed with the
field documentations
.

The field requester will show up, if you call
Add Field
,
Edit Field
or make a doubleclick on a field.

If you proceed with Ok, all values will be checked for validity.

If one value cannot be used by Fiasco, a requester will explain the problem.

A small summary of the conditions: (presumed, that these attributes exist)
- There must be an Id.
- "MaxChars" must be >= 1.
- Width must be >= 3.

If dimensionvalues cannot be used, because other fields are too near at the field, another requester appears, which has the gadgets "Shift", "Squeeze" and "Cancel". Cancel does nothing, it only returns to the field requester. "Squeeze" makes the field fitting in the space. "Shift" shifts the field to the left, to make it fitting. It is not always possible to "Shift".

If you change a already existing field, which stores it‘s contents in strings and supports "MaxChars" (currently String , Extern and Datatypes), an additional control is implemented. If you change "MaxChars" to a value, which does not allow to keep all strings in their original length (that means, some strings are longer), you will be asked, whether you really want to cut these strings or to keep the old value.

1.109 Search requester

Field

Select here the field, which will be searched. The listview displays only the ids of fields, textfields are not displayed. Only one field can be selected.

Pattern

Enter here the pattern to search. It may be a simple value or one with Patterns . This value will be also used in the count and replace requesters.

blurred

If you want to use blurred search , you will have to activate this gadget.

factor

You can control here the tolerance of the blurred search. 0 searches only for exactly matching entries, 100 searches for almost all entries.

Next (Return)

Proceeds with searching for the next matching entry and activates it.

First

```
    Searches for the first matching entry.

  Previous
    Searches backwards for the next matching entry.

  Cancel (Esc)
    Closes the requester without any further action.
```

1.110 Replace requester

```
                                If you already know the
                search requester
                , you should
have no problems with this one.

  Field
    Select here the field, which will be searched. The listview displays
    only the ids of fields, textfields are not displayed. Only one field
    can be selected.

  Pattern
    Enter here the pattern to search. It may be a simple value or one with

                Patterns
                . This value will be also used in the count
    and replace requesters.

  Replacement
    Enter here a value, which will be inserted in the matching entries.
    No patterns are possible.

  Confirm
    If you want to be asked for every replacing operation, you should
    select this gadget.

  blurred
    If you want to use
                blurred search
                , you will have
    to activate this gadget.

  factor
    You can control here the tolerance of the blurred search. 0 searches
    only for exactly matching entries, 100 searches for almost all entries.
```

1.111 Count requester

```
                    This requester makes it possible to count records, which match ↩
                        with a
patterns. More on counting
```

here You can open this requester using Compare/Count .

If you already know the search requester , you should have no problems with this one.

Field
Select here the field, which will be searched. The listview displays only the ids of fields, textfields are not displayed. Only one field can be selected.

Pattern
Enter here the pattern to search. It may be a simple value or one with Patterns . This value will be also used in the search and replace requesters.

blurred
If you want to use blurred search , you will have to activate this gadget.

factor
You can control here the tolerance of the blurred search. 0 searches only for exactly matching entries, 100 searches for almost all entries.

Ok (Return)
proceeds and counts the matching records. The number will be displayed at the end.

Cancel (Escape)
closes the requester without any further action.

1.112 Sort requester

You can sort a Fiasco project in the sort requester in respect to one field. It can be opened using Compare/Sort .

Sort by
A list of all fields is displayed here. You should select a field here, which will be used as orientation while searching.

Ok (Return)
begins with searching. The previously active record will be kept active, but it is highly probable, that the number of the record will

change.

Cancel (Escape)
Closes the requester without any further action.

1.113 Filter requester

Filter 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.

The filter requester may be reached using Compare/Edit Filter -.

If you already know the search requester , you should have no problems with this one.

Field
Select here the field, which will be searched. The listview displays only the ids of fields, textfields are not displayed. Only one field can be selected.

Pattern
Enter here the pattern to search. It may be a simple value or one with Patterns .

Replacement
Enter here a value, which will be inserted in the matching entries. No patterns are possible.

Confirm
If you want to be asked for every replacing operation, you should select this gadget.

blurred
If you want to use blurred search , you will have to activate this gadget.

factor
You can control here the tolerance of the blurred search. 0 searches only for exactly matching entries, 100 searches for almost all entries.

Ok (Return)
creates the filter. The project will seem, as if would consist only of matching records.

Cancel (Escape)
closes the requester without any further action.

1.114 Usermenu requester

Fiasco has the ability to create own menuitems and to put CLI programs or ARexx scripts behind them. The defined items may be selected apart from the classic way with the mouse with the F-Keys. F1 to F10 correspondent to the first ten items, Shift and F1 to F10 correspondent to the items 11 to 20. If you want to define more than 20 items, you will have to select the additional items with the mouse.

Furthermore, Intuition limits the number of definable items to 63. If you try to define more items, they will be cut.

The items may be saved using
Settings/Save Settings
.

Items
This is the list of all existing menu items. With "+" you can add one, with "-" you can delete one and "<"/">" serve to change the position of the item.

Type
Select here, whether you want to call a program or a ARexx script.

Command
Select here the program or the ARexx script respectively, which shall be executed.

1.115 Option requester

The option requester contains settings, which concern the project. It may be opened using the menuitem
Project/Options
or the
ARexx command
F_OptionsReq
.

Name
Here you can change the filename of the project. The project will be

saved under this name in future. After saving, all direction-relative operations will use the new directory.

Athor

You can use this field to enter your own Name! It will be stored at the beginning of the project file.

Annotations

Yet another gadget for free use. You may store any notes here, for example a version string (with $VER: at the beginning). It will be written to the project file just before the author.

strech X Y

These values are added to the with or height of the cursor. The effect of this operation is a stretching of the mask in X- or Y-direction. More on stretching here.

1.116 Goto requester

The goto requester is one of the clearest requesters in Fiasco at all. It may be opened with Record/Goto and offers the possibility to activate a record using it‘s number.

Please note, that Filters change the record numbers.

go to

Takes the number of the record.

Ok (Return)

Proceeds and activates the record with the number.

Cancel (Esc)

Oh sorry, i just forgot... %-)

1.117 Relation requester

This is the main interface for relation handling in Fiasco. It may be opened using Field/Edit Relations .

Key here

Select here the key in the current project.

```
  Real here
    displays the Id of the field, whose relations are just now edited.

  Related File
    Select here the project file relative to the directory of the current
    project, which contains the informations.

  Key there
    Select here the field of the project, which has been specified under
    "Related File", which contains the Key. This listview displays only
    fields, which can contain the Key (with same type).

  Real there
    Select here the field of the project, which has been specified under
    "Related File", which is supposed to be the counterpart of "Real here".
    This field is used to read the Data, which will be displayed in "Real
    here". This listview displays also only fields, which look as if they
    could contain the data (type and max chars must be equal).

  Ok (Return)
    Proceeds with loading the relations. If any errors occur while loading,
    the requester will be activated again, otherwise it will return to the
    main window.

  Cancel (Esc)
    closes the requester without any further action.
```

1.118 Show column requester

This requester, which may be reached with List/Show column, displays the currently hidden columns in the list. If you select one and click on Ok, the column will be inserted in the list at it‘s old position.

```
  Field
    All hidden columns are displayed here. Select here the column, which
    you want to be revealed.

  Ok (Return)
    Inserts the column and redisplays the list.

  Cancel (Esc)
    Closes the requester.
```

1.119 arexx

ARexx is a macro programming language, which is capable to connect different programs. ARexx has been developed by William S. Hawes and is part of the system software since OS 2.0.

The ARexx port of Fiasco may be accessed externally out of a script, or ARexx scripts can be called by Fiasco. For example. this happens if you specify in the field attribute
Script
an ARexx script
and you change the contents of a field. These scripts may react on the change and can adjust the value of another field or can do something else.

To be able to communicate with Fiasco, you have to add the line "Address Fiasco" to the script.

Nearly all operations, which can be used with the GUI of Fiasco, can be used with the ARexx commands. Additional, the functions of Fiasco may be extended with ARexx. There are many ARexx commands, which do exactly the same as their GUI "brothers". That means, that certain commands may open a requester under certain conditions. It is often possible to circumvent this problem. It will be fixed sometimes in the future. There are also commands, which open a requester in any case. This may be also useful for scripts, but has been primary implemented to give Fiasco a second menu (Iconbars). I have experimented already with ToolManager-Docks. Unfortunately, this was too slow for fast browsing in a database.

1.120 ARexx and Fiasco in general

A Fiasco command returns in the case of success in RC 0. If a command had problems, because it‘s environment was not proper, 5 is returned. More serious errors, like missing arguments, return 10. Fatal errors return 20. Parameter are separated by white spaces. They may only contain spaces, if they are enclosed in quotation marks (", ’ is not supported by Fiasco). If a command returns a value, this is stored in RESULT. To use RESULT, you have to put an "OPTIONS RESULTS" at the beginning of a script.

The debugging of ARexx scripts is a bit problematic. Scripts, which have been activated using the user menu or fields, have no output stream. All errormessages will be swallowed. If you want to test ARexx scripts, you should run the scripts from the shell (using rx <filename>; Fiasco must be in the correct status). To get more information, why a command failed, which has been sent to Fiasco, you should activate the item
Control/ARexx Debug

.
Fiasco will show a requester with a explanation for the reason of the error. The script won't continue until the requester has been closed. You have two choices there: "Continue" returns the correct error code, "Ignore Error" returns 0 as RC, which looks like the command has succeeded.

The style of the documentation of the commands is similar to the Amiga OS Autodocs. Synopsis defines a template.

1.121 Index of all ARexx commands

F_AboutReq

F_AddFieldReq

F_AddRecord

F_ClearProject

F_CloseServiceWin

F_CloseList

F_CountRecs

F_CountReq

F_DupRec

F_FilterReq

F_FindFirst

F_FindNext

F_FindPrev

F_FindReq

F_GetFieldCont

F_GetProjName

F_GetProjFullName

```
F_GetRecNum

F_GotoFirstRec

F_GotoLastRec

F_GotoNextRec

F_GotoPrevRec

F_GotoRec

F_GotoRecReq

F_IsVirgin

F_Locate

F_MakeVirgin

F_NewProject

F_OpenServiceWin

F_OpenList

F_OpenProject

F_OpenProjectReq

F_OptionsReq

F_Quit

F_RemRecord

F_SaveProject

F_SaveProjectReq

F_SaveSettings

F_SetDefFontReq

F_SetFieldCont

F_SetMode

F_SetSearchPat

F_SetSearchField

F_SortReq

F_SelectProj
```

F_UserCommand

1.122 Index of all GUI dependant ARexx commands

F_AboutReq

F_AddFieldReq

F_AddRecord

F_ClearProject

F_CloseServiceWin

F_CloseList

F_CountReq

F_DupRec

F_FilterReq

F_FindNext
 *

F_FindPrev
 *

F_FindReq

F_GotoFirstRec

F_GotoLastRec

F_GotoNextRec

F_GotoPrevRec

F_GotoRec

F_GotoRecReq

F_NewProject

F_OpenServiceWin

F_OpenList

F_OpenProjectReq

F_OptionsReq

F_Quit

```
                F_RemRecord

                F_SaveProject

                F_SaveProjectReq

                F_SaveSettings

                F_SetDefFontReq

                F_SetMode
                  *

                F_SortReq

                F_UserCommand
                  *

* -- These commands don‘t correspondent exactly to the GUI functions.
     If you write a few more lines, you should be able to rebuild these
     functions.
```

1.123 Index of all GUI independent ARexx commands

```
                F_AddRecord

                F_ClearProject
                  *

                F_CloseServiceWin

                F_CloseList

                F_CountRecs

                F_DupRec

                F_FindFirst

                F_FindNext

                F_FindPrev

                F_GetFieldCont

                F_GetProjName

                F_GetProjFullName

                F_GetRecNum

                F_GotoFirstRec
                F_GotoLastRec

                F_GotoNextRec

                F_GotoPrevRec

                F_GotoRec

                F_IsVirgin

                F_Locate

                F_MakeVirgin

                F_NewProject

                F_OpenServiceWin

                F_OpenList

                F_OpenProject

                F_Quit
                  *

                F_RemRecord
                  *

                F_SaveProject

                F_SaveSettings

                F_SetFieldCont

                F_SetMode

                F_SetSearchPat

                F_SetSearchField

                F_SelectProj

                F_UserCommand
                  *

* -- These commands may open a requester under certain conditions.
     Please refer to the documentation on how to circumvent these things.
```

1.124 f_openservicewin

```
                    NAME
        F_OpenServiceWin -- Open the service window

    SYNOPSIS
        F_OpenServiceWin
    FUNCTION
        Opens the
                Service-Window
                , if it is not already open.

    INPUTS
        none

    RESULTS
        none

    SEE ALSO

                F_CloseServiceWin
```

1.125 f_closeservicewin

```
                    NAME
        F_CloseServiceWin -- close the service window

    SYNOPSIS
        F_CloseServiceWin

    FUNCTION
        Closes the service window. If the window is not open, nothing
        happens.

    INPUTS
        none

    RESULTS
        none

    SEE ALSO

                F_OpenServiceWin
```

1.126 f_newproject

```
                    NAME
        F_NewProject -- Open a new project window

    SYNOPSIS
        F_NewProject

    FUNCTION
        Opens a new project. A new window is opened and activated.
        It is then entirely empty.
        Does exactly the same as
```

```
                Project/New
                .

    INPUTS
        none

    RESULTS
        none

    BUGS
        Should claim on error.

    SEE ALSO
```

1.127 f_openprojectreq

```
                    NAME
        F_OpenProjectReq -- Open the "Open Project" ASL requester

    SYNOPSIS
        F_OpenProjectReq

    FUNCTION
        Does exactly the same as
                Project/Open
                .
        I`m too lazy to write this here again.  :-)

    INPUTS
        none

    RESULTS
        none

    NOTE
        The user may have canceled the request

    SEE ALSO

                F_OpenProject

                Project/Open
```

1.128 f_openproject

```
                    NAME
        F_OpenProject -- Load a project

    SYNOPSIS
        F_OpenProject <Name>
        rc = Success

    FUNCTION
        Tries to read a fiasco project into the current project window.
        The data, which are currently in the window will be freed WITHOUT
        ANY REQUEST.

    INPUTS
        Name - Filename of the project

    RESULTS
        rc = 0, if everything went Ok,
           = 10, if argument is missing or file cannot be loaded.

    SEE ALSO

                F_OpenProjectReq
```

1.129 f_saveproject

```
                    NAME
        F_SaveProject -- Save the current project

    SYNOPSIS
        F_SaveProject

    FUNCTION
        Save the current project under the old name on disk.
        Does exactly the same as
                Project/Save
                .

    INPUTS
        none

    RESULTS
        none

    BUGS
        Does not inform the program about errors.

    SEE ALSO

                F_SaveProjectReq
```

1.130 f_saveprojectreq

```
                    NAME
        F_SaveProjectReq -- Open filereq and save project under new name
```

```
SYNOPSIS
    F_SaveProjectReq

FUNCTION
    Does exactly the same as
            Project/Save As...
                INPUTS
    none

RESULTS
    none

NOTE
    The user may have canceled the request

SEE ALSO

            F_SaveProject
```

1.131 f_aboutreq

```
                NAME
    F_AboutReq -- Open the "About" requester

SYNOPSIS
    F_AboutReq

FUNCTION
    Does exactly the same as
            Project/About
            .

INPUTS
    none

RESULTS
    none

SEE ALSO
```

1.132 f_optionsreq

```
                NAME
    F_OptionsReq -- Open the
            options requester
             for the current project

SYNOPSIS
    F_OptionsReq
```

```
FUNCTION
    Does exactly the same as
            Project/Options
                INPUTS
    none

RESULTS
    none

SEE ALSO
```

1.133 f_setdeffontreq

```
                NAME
    F_SetDefFontReq -- Open the fontrequester

SYNOPSIS
    F_SetDefFontReq

FUNCTION
    Does exactly the same as
            Settings/Choose Font
            .

INPUTS

RESULTS

SEE ALSO
```

1.134 f_findfirst

```
                NAME
    F_FindFirst -- Search for a pattern

SYNOPSIS
    F_FindFirst
    Result = Number_of_Record

FUNCTION
    Searches for the first matching with the pattern, which has been
    set with
            F_SetSearchPat
            . If rc is equal zero, Result is
    equal to the number of the found record. This may be accessed using

            F_GotoRec
            . If nothing is found, 5 is returned.

INPUTS
```

```
        none

    RESULTS
        rc = 0: result = Number of matching record.
        rc = 5: nothing found or no pattern.

    EXAMPLE
        /* Find-Example.rexx */
        options results
        address Fiasco

        count = 0

        F_SetSearchField "TestFeld" /* a Field
                                     * (Should be string in this example) */
        F_SetSearchPat "?#?"        /* search for all records,
                                     * which contain at least one
                                     * character */
        F_FindFirst                 /* search for the first record */

        do while rc = 0             /* Countinue searching until
                                     * nothing is found */
           F_GotoRec Result         /* activate the found record */

           count = count + 1

           F_FindNext               /* search for next */

        end

        /* All records done */

    SEE ALSO
```

1.135 f_findnext

```
                NAME
        F_FindNext -- Seach for a pattern.

    SYNOPSIS
        F_FindNext
        Result = Number_of_next_Record

    FUNCTION
        Searches for the next matching with the pattern, which has been set
        with
                F_SetSearchPat
                . If it succeeds (rc = 0), Result contains
        the number of the found record. This may be activated using
                F_GotoRec
                .

    NOTE
        The active Record is not searched by F_FindNext and F_FindPrev. If
```

you want to write a program, which searches all records, you have to call at first
F_FindFirst
and then F_FindNext.

INPUTS
none

RESULTS
If rc = 0, result = recordnumber of next matching.
If rc = 5, nothing found or no pattern

EXAMPLE
see
F_FindFirst
SEE ALSO

1.136 f_findprev

NAME
F_FindPrev -- Search for a pattern backwards.

SYNOPSIS
F_FindPrev
Result = Number_of_prev_Record

FUNCTION
Searches for the previous matching with the pattern, which has been set using
F_SetSearchPat
. If it succeeds
(rc = 0), Result will contain the number of the found record. This may be activated using
F_GotoRec
.

INPUTS
none

RESULTS
if rc = 0, result contains the recordnumber of previous matching
if rc = 5, nothing found or no pattern

NOTE
F_FindPrev is not very handy in ARexx scripts. You should use combinations of
F_FindFirst
and
F_FindNext
instead.

SEE ALSO

1.137 f_findreq

```
                NAME
    F_FindReq -- open the search requester

SYNOPSIS
    F_FindReq

FUNCTION
    Opens the
            search requester
            . This command does
    exactly the same a
            Compare/Find
            .
    The command may be only called in
            record mode
            .

INPUTS
    none

RESULTS
    none

SEE ALSO
```

1.138 f_setsearchpat

```
                NAME
    F_SetSearchPat -- Set the pattern to search for

SYNOPSIS
    F_SetSearchPat <searchpattern>
    rc = Success

FUNCTION
    Sets the searchpattern for the active project. If you also have
    specified a search field using
            F_SetSearchField
            , you may
    search for matching entries using
            F_FindFirst
             and
            F_FindNext
            .

INPUTS
    searchpattern - a string to search for

RESULTS
    rc = 0 -> everything ok.
    rc = 10 -> no argument
```

```
        rc = 20 -> systemfailure (no memory, etc.)

    SEE ALSO
```

1.139 f_setsearchfield

```
                NAME
        F_SetSearchField -- Set the field to search

    SYNOPSIS
        F_SetSearchField <FieldId>
        rc = Success

    FUNCTION
        Sets the field, which will be searched by
                F_FindFirst
                 and
                F_FindNext
                .

    INPUTS
        FieldId - The Id of the field, which shall be searched.

    RESULTS
        returns 10 in rc, if the argument is missing or the id is unknown.
        otherwise 0 is returned.

    SEE ALSO

                F_SetSearchPat
```

1.140 f_addfieldreq

```
                NAME
        F_AddFieldReq -- open the
                add field requester
                    SYNOPSIS
        F_AddFieldReq

    FUNCTION
        This command does exactly the same as
                Field/Add Field
                .
        It may be only called in
                mask mode
                .

    INPUTS

    RESULTS
    SEE ALSO
```

1.141 f_gotoprevrec

```
                NAME
        F_GotoPrevRec -- activate the previous record.

    SYNOPSIS
        F_GotoPrevRec

    FUNCTION
        Activates the record, which precedes the active record. If the active
        record is the first record, nothing will happen.

        Equivalent with
                Record/Previous
                .
        May be only called in
                record mode
                .

    INPUTS
        none

    RESULTS
        none

    SEE ALSO
```

1.142 f_gotonextrec

```
                NAME
        F_GotoNextRec -- activate the next record

    SYNOPSIS
        F_GotoNextRec

    FUNCTION
        Activates the record after the active one. If the active record is
        the last record or the current project contains no records, nothing
        will happen.

        Equivalent with
                Record/Next
                .
        May be only called in
                record mode
                .
    INPUTS
        none

    RESULTS
        none

    SEE ALSO
```

1.143 f_gotofirstrec

```
                    NAME
        F_GotoFirstRec -- activate the first record

    SYNOPSIS
        F_GotoFirstRec

    FUNCTION
        activates the first record. If the current project does not contain
        any records, nothing will happen.

        Equivalent with
                Record/First
                .
        May be only called in
                record mode
                .

    INPUTS
        none

    RESULTS
        none

    SEE ALSO
```

1.144 f_gotolastrec

```
                    NAME
        F_GotoLastRec -- activate the last record

    SYNOPSIS
        F_GotoLastRec

    FUNCTION
        Activates the last record. If the current project does not contain
        any records, nothing will happen.

        Equivalent with
                Record/Last
                .
        May be only called in
                record mode
                .

    INPUTS

    RESULTS

    SEE ALSO
```

1.145 f_quit

```
                        NAME
        F_Quit -- close the current project

    SYNOPSIS
        F_Quit

    FUNCTION
        Closes the current project. This command does exactly the same as

                Project/Quit
                . That means, it is possible, that a
        requester opens, which ask you whether you want to save the current
        project before proceeding or cancel.

        To prevent this situation, the command
                F_MakeVirgin
                 is
        provided. This function says Fiasco, that the project is not changed.

    INPUTS
        none

    RESULTS
        none

    NOTES
        If the current project is closed, another project will be activated,
        or, if there is no other project, Fiasco will be shut down. A ARexx
        script should not rely on the order, in which the next project will
        be activated.

    SEE ALSO
```

1.146 f_gotorec

```
    NAME
        F_GotoRec -- activate a record.
    SYNOPSIS
        F_GotoRec <RecordNumber>

    FUNCTION
        Activate the record, whose number has been given as arg.
        If the number was invalid, do nothing.

    INPUTS
        RecordNumber - The number of the record. Please note, that sorting,
        adding or removing records or filters may change the record numbers.

    RESULTS

    SEE ALSO
```

1.147 f_gotorecreq

```
                NAME
        F_GotoRecReq -- open the
                Goto-Requester
                    SYNOPSIS
        F_GotoRecReq

    FUNCTION
        Does exactly the same as
                Records/Goto
                .
        May be only called in
                record mode
                .

    INPUTS
        none

    RESULTS
        none

    SEE ALSO

                F_GotoRec
                       Menu
                Record/Goto
```

1.148 f_setfieldcont

```
                NAME
        F_SetFieldCont -- Change the content of a field.

    SYNOPSIS
        F_SetFieldCont <FieldId> <NewContent>
        rc = Success
```

```
FUNCTION
    Sets the content of the specified field in the active record to
    the specified content.

    May be only called in
            record mode
            .

INPUTS
    FieldId - Identificationname of the field
    NewContent - NewContent of the Field. This Arg takes the whole input
                 inclusive spaces. The Interpretation of this arg depends
                 on the fieldtype:
                 String  - is copied directly
                 Integer - Numbers are read directly, other things are 0
                 Float   - dto.
                 Boolean - 1 or TRUE = selected, 0 or FALSE = not selected
                 Slider  - Number is read. Bad numbers will be adjusted.
                 Cycle   - Number or name of label is taken.

RESULTS
    rc = 0  - no error
    rc = 5  - no record is active
    rc = 10 - missing arg or bad FieldId

BUGS

SEE ALSO
```

1.149 f_getfieldcont

```
NAME
    F_GetFieldCont -- Read the content of a field

SYNOPSIS
    F_GetFieldCont <FieldId>
    rc = Success
    result = Content

FUNCTION
    Reads the content of the specified Field in the active record and
    returns it in result.

    May be only called in record mode.

INPUTS
    FieldId - Id of Field

RESULTS
    rc = 0  - everything Ok, result contains the content ( %-/ )
    rc = 5  - no record active
    rc = 10 - arg missing, or unknown id.
```

```
        result  - is equal to the current content of the field, if rc = 0.
            The format:
            String  - the text itself.
            Integer - the number itself.
            Float   - not yet supported.
            Slider  - the value of the slider.
            Cycle   - the number of the active label.

    SEE ALSO
```

1.150 f_addrecord

```
                    NAME
        F_AddRecord -- Add a new record.

    SYNOPSIS
        F_AddRecord

    FUNCTION
        Add to the current project a new record. This record will get active.

        This function may only be called in record mode.

    INPUTS
        none

    RESULTS
        none

    SEE ALSO

                F_RemRecord

                Record/Add Record
```

1.151 f_remrecord

```
                    NAME
        F_RemRecord -- Delete the active record

    SYNOPSIS
        F_RemRecord

    FUNCTION
        Removes the active record and activates the next.

        This function may only be called in record mode.

    INPUTS
        none
    RESULTS
        none

    SEE ALSO

                F_AddRecord

                Record/Remove Record
```

1.152 f_setmode

```
                    NAME
        F_SetMode -- Select the editing mode

    SYNOPSIS
        F_SetMode <Mode>
        rc = Success

    FUNCTION
        Activates the specified mode for the current project.

    INPUTS
        Modus - "Mask" activates the
                mask mode
                .
                "Records" activates the
                record mode
                .

    RESULTS
        rc = 0  - no error
           = 5  - the project was already in the specified mode
           = 10 - missing or bad arg

    SEE ALSO
```

1.153 f_locate

```
    NAME
        F_Locate -- Locate the Cursor

    SYNOPSIS
        F_Locate <X> <Y>

    FUNCTION
        Sets the cursor at the given position. At this place the next mask
        operation will happen.
        May be only called in mask mode.

    INPUTS
        X - X-Coordinate
        Y - Y-Coordinate

    RESULTS

    BUGS
        Currently not particularly useful, because there are no direct
        commands for manipulating the mask.

    SEE ALSO
```

1.154 f_countrecs

```
    NAME
        F_CountRecs -- count the records

    SYNOPSIS
        F_CountRecs
        result = <Number_of_Records>

    FUNCTION
        Counts the records, which are currently in the current project.
        May be only called in record mode.

    INPUTS
        none

    RESULTS
        Number_of_Records - The number of records, may be zero. Note, that
            Filter influence this value.

    SEE ALSO
```

1.155 f_sortreq

```
                    NAME
        F_SortReq -- Open the sort requester

    SYNOPSIS
        F_SortReq

    FUNCTION
        Does exactly the same as
                Compare/Sort
                .
        May be only called in record mode.

    INPUTS

    RESULTS

    SEE ALSO
       "Compare/Sort..."
```

1.156 f_countreq

```
                NAME
    F_CountReq -- Open the count requester

SYNOPSIS
    F_CountReq

FUNCTION
    Does exactly the same as
            Compare/Count
            .
    May be only called in record mode.

INPUTS

RESULTS

SEE ALSO
   "Compare/Count..."
```

1.157 f_openlist

```
                NAME
    F_OpenList -- Open the
            list window
                SYNOPSIS
    F_OpenList

FUNCTION
    This command is equal to activating the menuitem
            Control/List
            .

INPUTS

RESULTS

SEE ALSO
   "Control/List"

            F_CloseList
```

1.158 f_closelist

```
                    NAME
        F_CloseList -- close the
                list window
                    SYNOPSIS
        F_CloseList

    FUNCTION
        This command is equal to deactivating the menuitem
                Control/List
                .

    INPUTS

    RESULTS

    SEE ALSO
       "Control/List"
```

1.159 f_filterreq

```
                    NAME
        F_FilterReq -- open the
                filter requester
                    SYNOPSIS
        F_FilterReq

    FUNCTION
        Does exactly the same as
                Compare/Filter
                .
        May be only called in record mode.

    INPUTS

    RESULTS

    SEE ALSO
       "Compare/Filter..."
```

1.160 f_duprec

```
                    NAME
        F_DupRec -- Clone the active record

    SYNOPSIS
        F_DupRec

    FUNCTION
        This command duplicated the active record exactly. All "init"
        attributes like "Init Cont", "unique Id", "use current time" are
        ignored. This command does exactly the same as
                Record/Dup Record
                .
        May be only called in record mode.

    INPUTS

    RESULTS

    SEE ALSO
        "Records/DupRecord"
```

1.161 f_getrecnum

```
                NAME
    F_GetRecNum -- Get the number of the current record.

SYNOPSIS
    F_GetRecNum
    Result = number of record

FUNCTION
    Returns the number of the active record in result.
    May be used to save the initial status of the project and to restore
    it at the end using
            F_GotoRec
            .

INPUTS
    none

RESULTS
    Result = Number of the record. Note that filters and other operations
             may change the record numbers.

SEE ALSO
```

1.162 f_savesettings

```
                NAME
    F_SaveSettings -- Save the current program settings.

SYNOPSIS
    F_SaveSettings

FUNCTION
    Does exactly the same as
            Settings/Save Settings
                INPUTS
    none

RESULTS
    none

SEE ALSO
    "Settings/Save Settings"
```

1.163 f_clearproject

```
            NAME
    F_ClearProject

SYNOPSIS
    F_ClearProject -- clear the active project

FUNCTION
    Deletes all data in the current project. It will be in a state much
    like after a "New".
    Does exactly the same as
            Project/Erase
            . Because of
    that, it is possible that a requester opens. To prevent this, clear
    the "Changed flag" using
            F_MakeVirgin
            .

INPUTS
    none

RESULTS
    none

SEE ALSO
    "Project/Erase"

            F_IsVirgin

            F_MakeVirgin
```

1.164 f_getprojname

```
            NAME
    F_GetProjName -- read the filename of the current project

SYNOPSIS
    F_GetProjName
    Result = <filename>

FUNCTION
    Returns the filename of the current project without path.
    This value may be used for
```

```
            F_SelectProj
            .

    INPUTS
        none

    RESULTS
        Result - Filename of the current project without path.
                 A file must not necessarily exist. This is possible, if the
                 name has been changed using Options and the project has not
                 been saved.

    SEE ALSO

            F_GetProjFullName
```

1.165 f_getprojfullname

```
                NAME
        F_GetProjFullName -- get the name of the current project

    SYNOPSIS
        F_GetProjFullName
        Result = <Name>

    FUNCTION
        Returns the filename of the current project incl. path.

    NOTE
        The path is relative to the current directory of Fiasco.

    INPUTS

    RESULTS
        Name - Name of project incl. path.

    SEE ALSO

            F_GetProjName
```

1.166 f_selectproj

```
                NAME
        F_SelectProj -- activate an already load project

    SYNOPSIS
        F_SelectProj <FileName>
        rc = <Success>

    FUNCTION
        Activates a project, which stays already in memory. The filename
        without path is used to identify the project. This may be the name, which
        has been obtained using
                F_GetProjName
                .
        All following commands refer to the new project.

    INPUTS
        Name - Name of project without path.

    RESULTS
        Success = 5, if project is already active.
                = 10, if argument is missing, or there is no such project.

    SEE ALSO

                F_GetProjName
```

1.167 f_usercommand

```
    NAME
        F_UserCommand -- Calls a userdefined command.

    SYNOPSIS
        F_UserCommand <Command_Number>

    FUNCTION
        Calls a command, which has been defined in the "User" menu.

    INPUTS
        Command-Number - Number of command, counted from zero. If this number
                         does not exits, nothing will be done.

    RESULTS
        none

    NOTE
        This command is only for implementing a icon bar or similar
        things. It should not be used in normal scripts, because the command
        may change freely.

    SEE ALSO
```

1.168 f_isvirgin

```
                    NAME
        F_IsVirgin -- Is the project unchanged?

    SYNOPSIS
        F_IsVirgin
        rc = <IsVirgin>
```

```
FUNCTION
    Tests, whether the current project has been changed since the last
    saving. If it has been changed, Quit, Erase, Load and so on, will put
    an requester.

INPUTS
    none

RESULTS
    rc = 0 - Unchanged
    rc = 5 - Changed

SEE ALSO

            F_MakeVirgin
```

1.169 f_makevirgin

```
                NAME
    F_MakeVirgin -- Say fiasco, that the current project is unchanged

SYNOPSIS
    F_MakeVirgin

FUNCTION
    Pretends, that the current project is unchanged. This prevents
    certain procedures (Erase, Load, Quit,...) to put up a requester.

INPUTS
    none

RESULTS
    A project, that thinks, it has not been changed since the last
    saving.

SEE ALSO

            F_IsVirgin
```

1.170 f_

```
NAME

SYNOPSIS

FUNCTION

INPUTS

RESULTS
```

SEE ALSO

1.171 Example Projects

The directory databases contains several Fiasco projects. Some ↩ of them may be also used for own purposes.

Addresses
Addressbook

DatatypesDemo
Demonstration of datatypes Fieldtype (req. OS 3.0)

FamilyTree
stores data about you ancestors.

Videos
Catalog of video tapes

1.172 Addresses

The Address project can be used as a simple addressbook. It contains fields for Name, Address, Phone, etc. The german version also uses relations to translate the abbreviations of country names (like "I" for Italy) to the long versions. I don‘t know, how this is handled in other Countries.

The fields for Phone, Fax or Zipcode or string fields, because they also have to take characters like "/" or must have a leading "0" (which would be swallowed by a integer field).

An additional Idea would be to use relations to search for the name of the city using the zip code.

1.173 Datatypes Demo

This project is a easy Demonstration of the Datatypes fieldtype , which requires the datatypes.library. For this reason, it is only available for users of Amiga OS 3.0 or higher. The mask contains three fields, which can be used to display all Data, which have the correct datatypes installed (Fiasco seems to have problems with AmigaGuide and Anim). Two fields have scrollbars at the bottom and at the right side. You can use these scrollers to move the contents of the field. The stringgadget below the display contains the name of the file.

The browsing between records may get a bit slower, because the data are stored in an external file and must be loaded first.

1.174 FamilyTree

The family tree consists of the projects "persons.fdb" and "families.fdb". "persons.fdb" contains all persons, which are used in the family tree. You may also enter sex, date of birth, etc. here.

These data are used by "families.fdb" with relations, to get names of spouses, children, etc. Additionally, there are fields for marriage and divorce. Caused by the intensive use of relations, this project only contains 10 "real" fields, which are stored on disk. The other 12 fields are loaded from "persons".

1.175 Videos

The video database can be used to manage your homevideo collection. The database consists of two projects: "films.fdb" and "tapes.fdb". "Films" takes the informations for each film (Genre, Director, etc.). The field "Tape" connects each film with one tape, which can be found in "tapes". Here is the play length of each tape defined. An ARexx script calculates the left free space on the tapes.

1.176 Localization an

The locale.library makes it since Amiga OS 2.1 possible, to adapt programs to any language. Because my knowledge of languages is not too big, Fiasco supports only German and English (built in).

If someone wants to create a new language catalog for fiasco, the files fiasco.cd and fiasco.ct are provided. These files are required for creating locale catalogs. I hope, that the translators are familiar with the creation of catalogs. More information on fiasco and localization can be found at the beginning of Fiasco.cd.

Although it is not necessarily required, I would be happy, if the authors of catalogs send them and the source files to me. The catalogs will be included in the next release of Fiasco.

You may also translate the documentation and the examples.

1.177 How to get contact

Send gifts, ideas, bug reports, etc. to:

Nils Bandener
Dekanatsgasse 4
D-34369 Hofgeismar
Germany

Sorry, I don't have email, yet.

1.178 bugs

If you find some bugs in Fiasco, send a detailed description to
me
.
Please include information about your processor, OS version and other configuration.

These bugs are known:

- The frame of the list window flashes sometimes in a weird way under kickstart 37.x

- ARexx seems to have problems with filenames, which contain spaces. The name is only interpreted to the the first space. This affects the full path, because Fiasco expands the name to the full path before calling ARexx scripts.

- Seems to leave sometimes some memory allocated.

1.179 todo

Fiasco is of course not perfect, at all. Here is a list of all things,
which will be perhaps added at a later point (no guarantee!). If you have an Idea, send it to
me
!

- Less memory fragmentation. Currently one allocation is made of each field in each record.

- Better scrolling in the mask window. I currently use GadTools gadgets, which have to be recreated if you want to change their positions. I plan to emulate the used gadgets. This would not require any more GZZ windows.

- Relations faster

- New searchfunction.

- Sorting should be take several fields into account and should get faster.

- New ARexx commands: ReadRecord and WriteRecord. Should read all fieldcontents and put them in ARexx variables with the field id as names.

- New searchpatterns. Does anyone know, whether dos patternmatching functions support more than 30 characters officially?

- "Virtual" fields, whose content is not save, but calculated in run-time.

- Hiding of field in the mask,

- A field similar to cycle fields, which uses relations.

- Reports

- Stringfields, which support multiple lines

- An option to convert one fieldtype to another.

- Reversed logic while searching/counting/sorting

- AppWindows for Datatypes and Extern Fields

- "Packing" of projects: search for unused fields and make used as small as possible.

- Checking, whether a similar record already exists (automatically)

1.180 index