KingFisher2

Udo Schuermann

COLLABORATORS			
	TITLE : KingFisher2		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Udo Schuermann	November 11, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

KingFisher2

1.1 KingFisher Release 2

KingFisher Release 2

This document is formatted with AmigaDOS 3.x in mind. Please excuse what display problems you may encounter when running under AmigaDOS 2.x!

Table of Contents

1.2 1 Introduction and Concepts

Welcome to KingFisher!

If you are a registered user of KingFisher, I thank you for your support (now and anytime you read this again ;-) If you have not yet registered, please read section 1.1 to find out why you really should consider registering the software.

1.3 1.1 Why Register?

Why should you register KingFisher?

KingFisher took hundreds, if not thousands of hours to develop, improve, and test. The modest registration fee is the only financial support that software authors, such as I, receive when they release quality software to the public.

You get to evaluate KingFisher in your own home, on your own computer, with your own particular configuration and preferences. You are not "rushed" into a decision and the software's price is not inflated with shipping and handling charges, nor the cost of a fancy box or the store clerk's salary.

Furthermore, KingFisher has not been crippled with annoying requesters or left-out features: you can update databases, print, export, and search without restriction. The only limitation is that you can make no more than two connections to the KingFisher Database Server. I.e. you can run two simultaneous copies of KingFisher or RexxFisher in combination.

In other words, it is your conscience and good sense of morals to which I am appealing!

Shareware can be quite a bargain. Without your support, however, shareware authors may get soon discouraged and stop supporting the software or choose to turn it over to a publisher who exacts from you several times the original price.

If you don't show your support to software authors, who will? So, register today; It's only about the price of a pizza or two and your support can make the difference between a good program falling by the wayside or receiving the support that you and your Amiga deserve!

What YOU get for registering

× A personalized keyfile that allows more than two simultaneous connections to the KFServer, thereby removing the relatively minor restrictions placed on the unregistered version,

× Access to programming documentation and source code (the KingFisher Developer Kit) to allow your own software to connect to the server. Are you the author of BBS software? Would you like to write a "Door" or other interface, rather than going through RexxFisher? Do you want to try your hand at writing a (gasp!) better GUI?

× Priority technical support and notification of new releases via electronic mail. You will also be the first at the chance to receive new versions of KingFisher (even maintenance releases which won't always make it to the general public.)

× My pledge that as long as there is demand for KingFisher, I will continue to support it, regardless of where the Amiga is going (and it's beginning to look as if Amiga Technologies GmbH is quite serious about making the Amiga great again!)

× Your conscience will let you sleep better at night, knowing that you are really supporting your favorite computer. What good is your Amiga if nobody wrote software for it anymore? Support your software authors today!, because tomorrow it may be too late!

Register Now!

1.4 1.2 What is KingFisher?

KingFisher is a special purpose database tool designed to work with variable size records. This storage method allows for a wide variety of descriptions for a program, without wasting diskspace or limiting the size of a description.

Each record in a KingFisher database is indexed so that KingFisher can find it quickly, even if there are tens of thousands of records or more in the database. The index also stores Version Links so that you can quickly follow a program's development to

earlier or later releases. The index also stores a small set of flags that allow you to quickly mark a program as deleted, hidden, or already owned while you are casually browsing through the database. Looking for such programs at some later time becomes a simple and very rapid operation.

A QuickIndex is also available, which allows KingFisher to determine the name of any program in the database without having to load this record from disk. This operation allows you to scan the database for programs whose names (or portions of the name) you know, in a mere second or two. Use of the QuickIndex is completely automatic.

1.5 1.3 What does KingFisher consist of?

KingFisher is the name of the GUI (Graphic User Interface) as well as the entire package. For most people, the distinction is of little value, but technically the package KingFisher consists of:

* KFServer, the server portion of the KingFisher database system,

* KingFisher, the font-sensitive, resizable GadTools GUI, which is a client to the server,

* RexxFisher, the ARexx client interface to the server.

The KFServer is always involved when accessing a database, and only one copy of it is running at any one time. There may be any number of clients operating at one time.

1.6 1.4 How does KingFisher work?

KingFisher is based on Client-Server Technology, whereby you can access one or more databases simultaneously and each of them multiple times. The value of this may be lost on you until, on a bright and productive day, you find yourself wanting to perform a lengthy search on one database, then search another, rebuild a third database from a CD-ROM, and casually browse a fourth database for something you can't quite remember.

Both KingFisher and RexxFisher (the ARexx client) automatically start the KFServer if it is not already running. Once the server has been started by the first client, all future clients attach to it quickly and are served the default database. A client can either use this database without further care or ask the server for a specific database.

Two or more clients can access a single database or access independent databases. The server maintains the required order among databases and clients, avoiding unnecessary duplication of indexing data if many clients access the same database.

The result is a modern and efficient database system, employing the natural strengths of the Amiga's multitasking nature.

1.7 1.5 Why 'KingFisher'?!?

Good question!

If you are an old-time Amiga user, you probably recall how Fred Fish began collecting freely distributable software and made it widely available in a collection that became known as the "Fish Disks." On Fred's 301st disk was a program named Aquarium (Fred FISH ... AQUARIUM, get it?) by B Lennart Olsson, designed to store and search a database of descriptions of Fred Fish's disks. The name KingFisher carries on this "fishy tradition."

More historic information

Aquarium was a fine program for its time: it was stable, quite fast, and had an innovative way of classifying software into as many as 30 or so categories by which it was possible to search for software extremely rapidly. Aquarium, however, had some significant limitations, not the least of which was its inability to utilize multiple floppy disks for the database, a problem that was held at bay for a while by enterprising individuals who found ways to remove unused space from the database. In the end, however, the database completely outgrew the capacity of an 880K floppy disk to hold it.

Not long after Aquarium appeared on the scene I took it upon myself to add the descriptions of Fred's new releases to the Aquarium database and eventually released these as upgrades so that not everyone else had to perform the same laborious process of updating the 30 categories for every program on every "Fish Disk." Being a fairly consciencious human being I usually spent

about 30 minutes for every batch of 10 "Fish Disks" -- you can imagine how quickly I tired of this chore, but there were many who appreciated my efforts and I could not easily afford to stop.

Together with Aquarium's inherent limitations, this set the stage for KingFisher: I needed a program that would make adding Fred's new releases as easy as possible, could maintain its database over an arbitrary number of disk volumes, was at least as quick as Aquarium and hopefully just as stable. The result was released on December 1, 1992 and grew to KingFisher 1.40 by summer 1993. KingFisher could extract descriptions for Fred's releases straight out of an email or Usenet message. Many did not believe it until they saw it, and KingFisher was, where keyword searches were concerned, about 3 times faster than Aquarium.

The feedback I received during that time was enormous: Hundreds of emails, nearly half that many letters, and some money that I had not even asked for. While KingFisher evolved, I became aware of a number of flaws in its design. Rather than evolve the program into a quagmire of patched code, I decided to rewrite it.

The result was KingFisher Release 2, supporting everything that the original KingFisher did, while gaining multi-database support, Style Guide Compliance, a more intuitive, resizable, font-sensitive interface, and support for Fred Fish's Product-Info specification (which I designed, with feedback from him.)

1.8 1.6 Feature Overview

DATABASES

* Client-Server Technology: an advanced design, neatly separating access and presentation portions of the database system for improved performance and reliability,

* Multiple databases: KingFisher supports access to any number of databases, allowing you to switch from one to the other, and remembering your last record of access in each of them,

* Multivolume databases: A database may be spread over any number of (removable) disk volumes, with prompts for any access to a database portion not currently on a mounted disk,

* Simultaneous access: Access to one database is permitted to multiple clients simultaneously, unless one of these clients has obtained exclusive access,

INTERFACE

* Standard, resizable, font-sensitive GadTools GUI: The fully (proportional) font sensitive GUI (Graphic User Interface) can be freely resized. The displayed information will be fitted in the scrolling view (listview) within the available width. Screen mode[1], font, and file requesters are the system standard ASL model, and the Display Database is fully supported,

* Paint your own buttons: Any of KingFisher's 28 most prominent interface buttons may be replaced with your own images,

* Context sensitive help: Pointing at a button or menu item and pressing HELP will summon AmigaGuide® help on the indicated item; similarly, pressing HELP while entering a Search Expression brings up the Search Expression Tutorial,

DATA HANDLING

* User-definable layout: The layout of display, printing and exporting of fish (records) may be defined through custom format files. Databases retain their last-used custom (display, print and export) formats until changed,

* Clipboard support: Fish (records) may be imported or exported through the clipboard,

* Product-Info support: Fred Fish's Product-Info Specification is supported, both for data import and export. Files may be "fed" to KingFisher individually or KingFisher may be instructed to scan an entire directory tree for files named in compliance with the Product-Info Specification,

* Live Update / Notification: Changes to a Product-Info file being displayed, or a custom format used for formatting text for the display, printing, or exporting are now immediately reflected within KingFisher through AmigaOS Notification.

---

[1] Screen mode requesters require Kickstart V38 (v2.1)

1.9 1.7 Redistributing KingFisher

KingFisher is freely distributable shareware. It is, however, also copyrighted software. This means you may distribute KingFisher in archive form as part of a larger collection of software, such as a collection of software on a CD-ROM. You may not -- and I stress that -- you may NOT install KingFisher on a CD-ROM in such a manner that it can be used more or less directly to perform searches on the CD-ROM contents, unless you have obtained my permission first.

In plain language, you may distribute KingFisher so long as it is not in some ready-to-use format.

Please obtain my written permission first!

If you violate these wishes through action, inaction, ignorance, or by device, you will cause great damage to your soul and suffer pain, anguish, and ill health as a consequence. Don't say I didn't warn you!

1.10 1.8 KingFisher History

KingFisher's development history is detailed in a file that may or may not have shipped with your copy of KingFisher, depending if enough diskspace was available on the disk. Typically it is shipped only on CD-ROM, with the KingFisher Developer Kit, or with upgrades posted to Aminet, as the distribution disks are generally too full.

If you have it installed then you should be able to View the History file.

1.11 2 Installing and Uninstalling

Installing

Always install all components of KingFisher with the Installer. Most of the files in the distribution are compressed to save diskspace during transport. The Installer knows how to install these files properly.

KingFisher is installed into a directory structure such as the following:

:KingFisher2/

All files are installed to this directory. This includes especially KFServer, KFServer.prefs, KingFisher, KingFisher2.guide, and all #?.kfdb (Database Descriptor) files used by the KFServer.

:KingFisher2/Catalogs/

Any and all language catalogs are installed into this directory. Each language has its own subdirectory there, and a string catalog named "KingFisher2.catalog" is found in each language subdirectory.

:KingFisher2/Developer/

If you install the KingFisher Developer Kit, it is installed into this subdirectory.

Uninstalling

All of KingFisher is installed into a single directory, which means that uninstalling is as simple as dragging the drawer into the Trashcan!

Installer License

The following notice is required as part of the Installer License Agreement with Commodore:

Installer and Installer project icon

(c) Copyright 1991-93 Commodore-Amiga, Inc. All Rights Reserved.

Reproduced and distributed under license from Commodore.

INSTALLER SOFTWARE IS PROVIDED "AS-IS" AND SUBJECT TO CHANGE;

NO WARRANTIES ARE MADE. ALL USE IS AT YOUR OWN RISK. NO LIABILITY

OR RESPONSIBILITY IS ASSUMED.

1.12 3 Working with KingFisher

This chapter addresses the every-day use and aspects of KingFisher. If your needs are not complex and your desire for overwhelmingly accurate detail not too great, you may find here all that you ever need about operating KingFisher. The next chapter, Advanced use of KingFisher will delve into considerably more depth.

KingFisher's User Interface

The Search Expression Tutorial

Search Sets

Adding databases

Product-Info Specification

1.13 KingFisher's User Interface

KingFisher's user interface is a combination of windows, gadgets, and menu items. You may obtain Context Sensitive Help on any gadget or menu item by pressing HELP when the mouse pointer is over the item in question. If you want the Table of Contents, instead, hold down the SHIFT key or the MMB (Middle Mouse Button), or point somewhere in the window that is not a gadget.

The help system functions asynchronously, meaning that the help window may remain open while you operate KingFisher. Pointing at any gadget or menu item will then merely update the help window with the appropriate information; there is no need to close the help window.

Here are these same items for you to browse:

KingFisher's Buttons and other Gadgets

KingFisher's Menus

Operating the Search Result Window

1.14 KingFisher's Buttons and other Gadgets

KingFisher's gadget interface consists of five groups. These are conveniently labeled for you in this GUI Image :

The upper left has four POSITIONING gadgets:

1. The Disk/Fish Selector cycle gadget, which controls the exact function of the string gadget (#2),

2. The Disk/Fish Number string gadget that stores/displays disk or record (fish) numbers,

3. The Home button that jumps to the first fish (record) in the database,

4. The End button that jumps to the last fish (record) in the database.

The upper right has sixteen (16) flag buttons arranged in two rows of eight (8) each. You can set or clear any combination of flags and later look for such a combination extremely rapidly using the Search Mask setting!

The lower left has three groups of two BROWSING gadgets, each:

1. The Previous/Next Disk buttons,

2. The Previous/Next Version buttons,

3. The Previous/Next Fish (record) buttons.

The lower right has a Search Expression string gadget, and beneath that, four gadgets related to searching:

1. The Reverse Search button,

2. The Forward Search button,

3. The Search Expression History button,

4. The Search Result Set button.

The remaining space, in the middle, between the first two and last two groups, is occupied by the Database Title field and a scrolling view ("listview"), named the Fish Description gadget, which describes (as the name so cleverly suggests) the contents of the current fish (record.)

1.15 GADGET: Disk/Fish(record) Selector (Cycle)

To locate this gadget, click here .

This cycle gadget controls the exact function of the string (numeric) gadget to its right, the Disk/Fish Gadget. If the gadget shows "Disk", the string gadget displays and accepts numbers relating to whole disks. If the gadget shows "Fish", the string gadget displays and accepts fish (record) numbers.

1.16 GADGET: Disk/Fish(record) Number (Integer)

To locate this gadget, click here .

Depending on the state of the Disk/Fish Selector cycle gadget to its left, this gadget displays and accepts numbers relating either to whole disks or fish (records.)

Notice that a CD-ROM consists of only a single disk, which means that the disk gadget will always show a disk number of 1 and you cannot select a disk other than that.

The "1000 Fish Disk" collection, however, consists of 1000 disks and over 4500 fish (records) so you can quickly jump to these positions in the database.

1.17 GADGET: Home/End of Database (Buttons)

To locate these gadgets, click here .

These buttons immediately jump to the first or last record in the database. They are ghosted if you are already located on the first and/or last record.

NOTE: The images of these buttons are user-definable. Click here for more information.

1.18 GADGET: Flag Buttons (Toggle Buttons)

To locate these gadgets, click here .

Each fish (record) in a KingFisher database has 16 special purpose flags associated with it. These are represented by two rows of 8 flags each. Their layout and appearance resemble the following diagram:

D _ _ _ T H O M

8 7 6 5 4 3 2 1

The top row represents pre-defined flags; do not assign a different meaning to these flags than is given below:

D Deleted. This record is marked to be deleted from the database. A command available in a future version of KingFisher will use this flag to physically remove fish (records) from the database, thereby allowing you to eliminate unwanted records and recovering disk storage.

_ (undefined at this time; reserved for future use)

T Twin. This record has at least one duplicate in the database. KingFisher does not [yet?] set this flag for you when it encounters duplicates in the database during a version linking scan.

H Hidden. You do not wish to have this item appear in your searches (provided you set this flag in the Avoid mask and turn on the use of Search Masks.

O Owned. You already own this item.

M Marked. You could use this as a "bookmark" to quickly find records you wish to come back to later.

8-1 These are your own flags to define and use in any way that you may wish. KingFisher will never assign special meaning to these flags. You could use these flags as "bookmarks" in addition to the "M" flag, for example.

NOTE: These flags will be ghosted (i.e. unavailable for changes) if the main index of the current database is located on R/O media, such as a CD-ROM or the media or file is for some other reason not writable.

NOTE: The images of these buttons are user-definable. Click here to learn more.

Any alteration to these flags causes an immediate change in the database index. When the database is closed, KingFisher writes the index back to disk.

1.19 GADGET: Browse across Disks (Buttons)

To locate these gadgets, click here .

These buttons jump over one or more fish (records) if the database index defines them to be located on multiple disks.

The original Fish Disks database that ships with KingFisher defines 1000 such disks, but a CD-ROM usually consists of no more than a single disc, wherefore the distinction of separate disks is less clear, if even meaningful at all. In these cases you will find that all functionality referring to multiple disks becomes unavailable.

NOTE: The images of these buttons are user-definable. Click here for more information.

1.20 GADGET: Browse across Versions (Buttons)

To locate these gadgets, click here .

These buttons jump to previous or later releases of the same program, provided that information is recorded in the database index. The Version Link information is lost when a database is reindexed, and can be rebuilt with the EDIT/Build Version links command.

One or both of these buttons will be ghosted if a link in that direction is not available.

NOTE: The images of these buttons are user-definable. Click here for more information.

1.21 GADGET: Browse from Fish to Fish (Buttons)

To locate these gadgets, click here .

These buttons permit you to browse through the database one fish (record) at a time. They will be ghosted if you are at one end of the database and cannot move in that direction any further.

NOTE: The images of these buttons are user-definable. Click here for more information.

1.22 GADGET: Search Expression (String)

To locate this gadget, click here .

The search expression in this gadget is the expression that is used when you start a search. You may select another expression from the list of previously used expressions by clicking on the gadget beneath the search expression string gadget, that looks like a scroll.

Need help with constructing search expressions? Click here.

1.23 GADGET: Search Gadgets (Buttons)

To locate these gadgets, click here .

These buttons initiate a search either in reverse or forward direction.

NOTE: The images of these buttons are user-definable. Click here for more information.

1.24 GADGET: Search Expression History (Button ==> ListView)

To locate this gadget, click here .

If ghosted, no history of previous Search Expressions has yet been recorded.

Clicking on this button summons a list of Search Expressions that have been used previously and been recorded so that you need not retype them. Retrieve a search expression by clicking on one of the expression in the scrolling view ("listview").

Pressing the Escape key or closing the window instead of selecting an expression will cancel the operation.

NOTE: The image of this button is user-definable. Click here for more information.

1.25 GADGET: Search Result Window (Button ==> ListView)

To locate this gadget, click here .

If ghosted, a Search Set has not been loaded or generated.

This button opens and closes the Search Result window from which you can select individual matches, jumping immediately to the requisite database record to view the item's information. Closing this window will not lose the search set.

My thanks go to Fred Fish for his kind permission to use his Fish Logo!

NOTE: The image of this button is user-definable. Click here for more information.

1.26 FIELD: Current Database Name

To locate this field, click here .

With the PREFERENCES/DISPLAY/Show Database Title command, the display of the current database title can be removed from or restored to the display.

If you routinely work with multiple databases and frequently switch between them, this field will help you keep track of which database is the current one. On the other hand, if your screen is of relatively small size, you may wish to remove this field from the display to make the the central listview larger.

This gadget will show (No database selected) if the current database selection is for some reason not a valid database. This may happen if you remove a database from the disk, then restart KingFisher which will try to access the last viewed record in that database. Failing to do so, KingFisher will display this "bogus" database name.

1.27 GADGET: Fish Description (ListView)

To locate this gadget, click here .

This gadget is sized to fit in the available space in the window and the textual description for each fish (record) is word-wrapped within it. You can scroll this gadget with the up and down arrow gadgets.

The layout of the displayed text is defined by the Display Format option.

1.28 The Main Menu Strip

Behold, the menu strip:

Project Edit View Search Preferences Help

1.29 The Project Menu

About KingFisher

Server status

Open database

Install database

Define database

Print

Release printer

Export

Close export file

Quit

1.30 The Edit Menu

Append fish from file

Append fish from tree

Delete fish

Copy to clipboard

Append from clipboard

Reconstruct database index

Build Version links

Edit format file

Edit expression

Edit search masks

1.31 The View Menu

Database

Product-Info file

Product-Info clipboard

1.32 The Search Menu

Select expression

Search backward

Search forward

Load search set

Save search set

1.33 The Preferences Menu

Global

Auto-save on exit

Confirm quit

Display

Load custom display format

Show all fields in record

Use default display format

Custom screen

Default public screen

Center main window

Sticky result window

Show database title

Use external gadget images

Frame groups

Smart refresh

Printing

Load custom print format

Use default print format

One fish per page

Avoid page breaks

Add index info

Exporting

Load custom export format

Use default export format

Choose export file

Use importable raw format

Add index info

Searching

Stop on each match

Case sensitive

Trim blanks

Simple substrings

Use search masks

External command on failure

External command on success

Flash/beep screen when done

Save Preferences

1.34 The Help Menu

Help (index)

Using KingFisher

Searching

Printing

Exporting

Databases

1.35 PROJECT/About KingFisher

KingFisher Release 2.0

For Distribution Rights, click here

For my translators, click here

For registration, upgrades, and support click here

The KingFisher Logo is available in up to 16 grey shades if you are running Kickstart 3 or later on a display providing 8 or more colors.

1.36 PROJECT/Status

Presents you with some information about your connection to the server, including an estimate of what percentage of the server's total time your client has used (this is always 100% unless more than one client have connected to the server.)

With the SHIFT key or MMB (Middle Mouse Button) held down, you get a very detailed listing (dumped to a console window, from where you can cut & paste it) which is the same as when you invoke KFServer with the STATUS parameter when it is already running. The purpose is to give you information about the databases and client connections in the server. This sort of information is useful mainly for diagnostic purposes but might be fun to examine if you're curious.

If you have KingCON installed you can resize the console window and scroll through the contents forward and backwards!

1.37 PROJECT/Open Database

Select a different database from among those that the server knows about. This function is rarely useful if you have only a single database installed.

The server knows only about databases that have a .kfdb file in the server's directory.

You can cancel the selection by closing the window or by pressing the RMB (Right Mouse Button.)

1.38 PROJECT/Install Database

Copies a .kfdb file from one disk to the server's directory so that the server can find it (immediately.)

The first file requester selects the .kfdb file to install. The second file requester saves the file, possibly under a new name, to the server's directory. The database is immediately available with the PROJECT/Open Database command.

There are some points of which you should be aware:

* All that the server needs to access a database is a properly configured .kfdb file. Such a file occupies a trivial amount of diskspace.

* Certain settings in a .kfdb file specify the location of files (record and index files); if these settings do not specify a path, KingFisher will add the path to the directory where the .kfdb file was copied from. This makes the database immediately available!

* When installing a CD-ROM database, you may wish to manually copy the index files from the CD-ROM to a writable volume and adjust the .kfdb file to reflect this. KingFisher will then permit you to alter the flags, reindex, and rebuild the Version Links of the database (although you will not be able to add records to the database unless those files are also copied to a writable volume.)

1.39 PROJECT/Define Database

NOTE: This command is not yet available. Click here for the necessary steps.

1.40 PROJECT/Print

Send one or more fish (records) to the printer. If you print from KingFisher's main window, only the currently displayed fish is printed. If you print from the "Caught Fish" window, then all fish in the Search Set are printed.

Output is configurable with the options in the PREFERENCES/Printing menu. In particular, output may--if you so desire--use a different layout than the display window.

NOTE: The printer device will remain open until you explicitly close it with the PROJECT/Release printer command. This prevents other tasks from mixing their output with that of KingFisher.

1.41 PROJECT/Release printer

If this option is not ghosted, then it closes the printer device after you have printed all you wish to print. This permits other tasks to use the printer again.

The current page in use will be ejected by this command.

1.42 PROJECT/Export

Save one or more fish (records) to a file. If you export from KingFisher's main window, only the currently displayed fish is saved. If you export from the "Caught Fish" window, then all fish in the Search Set are saved.

Output is configurable with the options in the PREFERENCES/Exporting menu. In particular, output may--if you so desire--use a different layout than the display window.

NOTE: The export file will remain open until you explicitly close it with the PROJECT/Close export file command. This prevents other tasks from using (and possibly blocking) the file before you are truly finished with your export selections.

1.43 PROJECT/Close export file

If this options is not ghosted, it closes the export file after you have saved to it all you wish to save. This permits other tasks to access the file.

1.44 PROJECT/Quit

The PREFERENCES/Global/Confirm quit menu item lets you to specify whether or not you wish KingFisher to ask if you really want to quit.

* If you find yourself frequently quitting KingFisher without meaning to, you should turn that option on.

* If the "Really quit KingFisher" requester goes on your nerves, turn the option off.

If the Auto-save on exit option is enabled, then KingFisher will save all your current settings so that they can be restored when next you start the program.

1.45 EDIT/Append fish from file

The file you specify for imported records may contain one or more records. The file must conform to the Product-Info specifications. All valid records from the given file will be appended to the database and Version Links extablished. The index is automatically updated (and saved to disk when the database is closed.)

1.46 EDIT/Append fish from tree

Performs the same operation as EDIT/Append fish from file on every file in a given directory tree that is named #?.pi, .Product-Info, or Product-Info. A status window keeps you informed of progress. You can interrupt the scan by closing the status window; you must confirm such an action before the scan is actually aborted.

The index is automatically updated (and saved to disk when the database is closed.)

1.47 EDIT/Delete fish

Truncates the database by deleting the current fish (record) and all that follow. You must confirm the action before it will be actually performed. The records are actually removed from the database files and are not recoverable unless you make prior arrangements to copy them elsewhere.

1.48 EDIT/Copy to clipboard

Copies the current record to the clipboard from where it can be reimported via the EDIT/Append fish from file command. The next 'Copy to clipboard' operation will overwrite the previous clipboard contents. To save multiple fish (records) to the clipboard, use the 'Copy to clipboard' command in the "Caught Fish" window.

The record is stored in clipboard unit 0 as an IFF FTXT.

1.49 EDIT/Append from clipboard

The contents of all IFF FTXT records found in the clipboard's unit 0 are processed and--if valid records--appended to the current database.

1.50 EDIT/Reconstruct database index

This command asks the server to throw away all indexing information on the current database, then scan the database file(s) in an effort to reconstruct the information necessary to access individual records.

Both the main index, and the QuickIndex are rebuilt by this command, wherefore both must be writable.

WARNING: The main index stores the Version Links as well as the flags. If you reindex a database you will lose both of these sets of information!

After reconstructing the database index you should use the Build Version links command to rebuild the Version Links, as well.

A database can be reindexed only if you are the only client using it.

1.51 EDIT/Build Version Links

Reconstructs the Version Links for the current database. The main database index must be writable.

The following options are available before the operation begins:

[x] Clear existing links

Removes all previous Version Links before the operation begins. This step assures that no confusion ensues when two or more programs reference a single ancestor or successor through the use of multiple 'Build Version Links' commands that were issued with different options.

[x] Eliminate (Roman) numeric additions[2]

Programs such as "bBaseII" and "bBaseIII" or "DiskSalv" and "DiskSalv 2" are considered the same. Use of this option does not adversely affect processing.

[x] Double-check by authors[1] [3]

Two programs with the same name are considered for linking only if their respective author sets intersect. In other words, "MonkeyCommand" by Marcus Antonius, and "MonkeyCommand" by Ricky Rocket are considered distinctly different, but if the second is written by Ricky Rocket and Marcus Antonius (and perhaps others), then the programs are linked.

[x] Sort links by version[3]

As CD-ROM databases may not necessarily list software in version order, this option will examine the 'version' fields and order the links so that earlier versions (even if stored later in the database) are at the start of the version list.

This option depends heavily on correct version numbers in the database (i.e. the .version field must be formatted according to the Commodore Style Guide, as reflected in the Product-Info specification.)

This option is not available with a older Release 1 databases such as the "1000 Fish Disks" database that ships with KingFisher (the option will be disabled in that case.)

Version Links can be build only if you are the only client using this database. The database must be writable.

NOTE: In versions earlier than 2.10, KingFisher relied on an $O(n^2)$ algorithm to perform this operation. A vastly superior algorithm was then designed, which rebuilds the links in linear, O(n) time!

---

[1] This option will be implemented in a future release.

[2] Use of this option will slow down processing only if no QuickIndex is available.

[3] Use of this option will slow down processing!

1.52 EDIT/Edit custom format file

This permits you to create or edit a custom display, print, or export format for KingFisher.

KingFisher will suspend its operation until you are finished with the editor program that it invokes.

If the environment variable EDITOR is not set (i.e. 'ENV:EDITOR' does not exist) then KingFisher will use 'c:ED' instead to edit the .format file, otherwise it will use your favorite editor!

NOTE: If you are running KingFisher on a custom screen instead of the default public screen, then the editor may appear on the Workbench or default public screen; look for it there. This is because KingFisher's custom screens are not made into a (default) public screen that your editor would recognized and use.

NOTE: If you edit the display format file currently in use by KingFisher, you must manually reload this file. A future release of KingFisher will use file notification to assure that any alteration to the file (even when not modified with KingFisher's knowledge) will automagically update the display.

1.53 EDIT/Edit search expression

Edit the current search expression. You may also press the RETURN or the 'x' key to activate the Search Expression gadget.

1.54 EDIT/Edit Masks

When selected for use the Search Mask operates during a search to select only specific records for searching, usually eliminating the vast majority of records.

Alter the Search Mask usable during searches to locate specially flagged fish (records) in the database. The current record's flags are available in the upper right corner. The Search Mask is compared against this combination of flags during a search, allowing you to explicitly exclude certain records, depending on what flags a record has set.

The Search Mask is used only if you have selected the Use Search Masks option.

There are two components to the Search Mask:

Avoid Mask

The 'Avoid Mask' is used to completely eliminate any record from the search that has set one of the flags in the 'Avoid Mask.' By default this includes all records with the Deleted (D), Owned (O), and Hidden (H) flags.

Match Mask

A record not already eliminated by the 'Avoid Mask' is then considered a match only if it has set one of the flags in the 'Match Mask.' By default this includes all records with the Mark (M) flag.

NOTE: Fish (records) without any flags set cannot be located with a Search Mask in use!!

1.55 SEARCH/Select Expression

Every search expression you use in a search is placed into the Expression History list. From this list you may retrieve a previously used expression by clicking on it. The list is available in a scrolling listview by either a menu operation or by use of the Expression History gadget.

The Expression History List can store between 5 and 1024 expressions. The MaxHistory setting controls how many you permit to be stored before the oldest expression is dropped to make room for a new one.

1.56 SEARCH/Search reverse

Begins a search in reverse direction. You may interrupt the search by closing the "Search Status" window.

If you the Search Options are configured so that you are creating a Search Set then the current fish is included in the search, otherwise it is skipped.

The Search Expression string gadget must contain a valid Search Expression; if it does not, you will receive a diagnostic error message and the search will not be performed.

You may also initiate a reverse search by pressing the ">" key or clicking on the forward Search Button

1.57 SEARCH/Search forward

Begins a search in forward direction. You may interrupt the search by closing the "Search Status" window.

If you the Search Options are configured so that you are creating a Search Set then the current fish is included in the search, otherwise it is skipped.

The Search Expression string gadget must contain a valid Search Expression; if it does not, you will receive a diagnostic error message and the search will not be performed.

You may also initiate a forward search by pressing the ">" key or clicking on the forward Search Button

1.58 SEARCH/Load search set

Reload a Search Set from disk where the result of a previously performed search has been stored.

Any Search Set that you have currently loaded will be cleared and is lost if it has not been saved with the SEARCH/Save search set command.

When a Search Set is loaded, KingFisher will activate the database to which the search set applies. KingFisher will also store the relevant Search Expression to the Expression gadget to indicate what parameters produced the Search Set.

While loading the Search Set, KingFisher will retrieve some information from the records of the appropriate database. Large Search Sets may not seem to load very quickly for this reason, bue the slight delay requires much less time than performing the search again.

1.59 SEARCH/Save search set

Saves the current Search Set to a file on disk so it can be retrieved later. This saves you the time and effort of executing the search and having to wait for the result again. Search Sets require approximately 5 bytes per fish (record) on disk, so that 100 matching records will not require more than approximately 500 bytes on disk!

NOTE: You should make sure that you always add the file extension .search to a saved Search Set, otherwise (when you use the SEARCH/Load search set command) the file requester may not show you this Search Set until you alter the file requester's Pattern that specifies which files to include and exclude from the total volume of available files.

1.60 VIEW/Database

View the records in the currently selected database. This option is always the default when KingFisher starts.

KingFisher automatically re-selects this option when either of the VIEW/Product-Info file or the VIEW/Product-Info clipboard is active and you perform an operation that is only available when a database is viewed (such as selecting the next record.) KingFisher then redisplays the current database record and alerts you to this switch by performing a DisplayBeep() operation, effectively flashing the display and/or playing the system default bell tone (see Workbench's Sound Preferences.)

1.61 VIEW/Product-Info file

Select a Product-Info file to view.

While the Product-Info file is displayed, any operation performed that relates to the current database will flash the display and revert to the VIEW/Database selection.

This viewing mode is meant to proof-read a Product-Info file of your own construction. This saves you the repetitive cycle of having to add it to a database, then removing it, modifying the file, and adding it once again.

1.62 VIEW/Product-Info clipboard

View the current clipboard contents as a Product-Info record.

While the Product-Info file is displayed, any operation performed that relates to the current database will flash the display and revert to the VIEW/Database selection.

This viewing mode is meant to proof-read a Product-Info file of your own construction, which you have copied into the clipboard from an editor. This saves you the repetitive cycle of having to add it to a database, then removing it, modifying the record, and adding it once again.

1.63 PREFERENCES/Global

Auto-save on exit

Confirm quit

1.64 PREFERENCES/GLOBAL/Auto-save on exit

If this option is enabled when you quit, KingFisher will automatically store all current settings. These settings are restored when you start KingFisher.

The settings are ordinarily stored to a file named KingFisher2.prefs in the current directory. The same file is used for storing the settings as was used when KingFisher was started, meaning that if KingFisher is told to load its settings from a different file, then it is that same file that is also updated when you quit. See the SETTINGS tooltype for details on how to specify such an alternate settings file.

When restoring settings and not given a specific file to load them from, KingFisher will look for the file KingFisher2.prefs first in the current directory. If not found there, it will look for it in the ENV:KingFisher/ directory, and finally in the directory named by the logical volume S: (usually SYS:s/)

If you turn off this option and wish this change to become permanent, then you must use the Save Preferences command to update the settings file, otherwise the change will not be saved when KingFisher exits!

1.65 PREFERENCES/GLOBAL/Confirm quit

Do you hate software that always asks if you really want to quit? Do you have a reflexive habit of closing the window (and quitting programs) before you're actually done? KingFisher accomodates you in either case!

If "Confirm Quit" is checked, KingFisher requires affirmation before it will really quit. Likewise, if this menu item is not checked, then KingFisher instantly quits when you click (and release) the close window gadget.

1.66 PREFERENCES/Display

Load custom display format

Show all fields in record

Use default display format

Custom screen

Default public screen

Center main window

Sticky result window

Show database title

Use external gadget images

Frame groups

Smart refresh

1.67 PREFERENCES/DISPLAY/Load custom display format

Load a Custom Format from a file and applies this to the way fish (records) are displayed in the scrolling view (listview). To remove a custom display format and revert to the builtin default, select the Use default display format command.

KingFisher remembers the display format in use for every database.

To create or edit a custom format, use the Edit custom format file command.

To revert to the built-in default format, use the Use default display format command.

Using AmigaOS Notification, the file displayed is immediately updated if the version on disk is changed. The display is immediately updated to reflect the new file's contents.

1.68 PREFERENCES/DISPLAY/Show all fields in record

Creates a Custom Format based on every displayed record's own fields and uses that to create the display of information.

If you currently have a custom display format selected, it will be replaced by this command.

To return to the default display format, select the Used default display format command; to return to a custom display format, select a new custom display format.

Notice that this command will display non-standard fields as well as fields ordinarily suppressed because they are empty.

1.69 PREFERENCES/DISPLAY/Use default display format

This command reverts back to the builtin default display format.

The command is unavailable if the default display format is already in use.

1.70 PREFERENCES/DISPLAY/Font

KingFisher uses the Screen Font defined by your Font Preferences. This may be a fixed pitch or proportionally spaced font.

Likewise, you may select any font for KingFisher's interface, scalable, bitmapped, fixed width or proportional.

Be advised that KingFisher will not fall back to a smaller font if you select an unreasonably large font.

1.71 PREFERENCES/DISPLAY/Custom screen

If you are running Kickstart V37 (2.04) then please click here for assistance in getting KingFisher Release 2 to open on a custom screen, as KingFisher requires Kickstart V38 or later to use the ASL Screen Mode requester.

Any screen mode showing up in the screen mode requester is acceptable to KingFisher, although a lowres (ex: 320x200) may be a debatably silly choice unless you have also (and previously!) chosen a suitably tiny font for KingFisher.

Be advised that KingFisher will not fall back to a smaller font if your current selection is not reasonable for the size of the selected screen.

1.72 Getting KingFisher to open on a custom screen under V37

First of all, quit all copies of KingFisher so that your changes to the KingFisher2.prefs file are not accidentally overwritten again when you later quit the program.

You need to edit the KingFisher2.prefs file and locate the "Screen" item, which should be one of the very first ones listed. The following example uses an NTSC:800x600 screen (oversized with autoscroll) with 8 colors (3 bitplanes deep):

Screen=CUSTOM:00008004,800w,600h,3d,1

Notice that the first portion of the Screen specification is "CUSTOM:" If anything else shows up in the first 7 characters, then KingFisher assumes that you are actually specifying the name of a specific public screen, rather than a custom screen specification.

The next item, "00008004" is a hexadecimal representation of the screen mode ID from the Display Database. The following are standard values that you can use as a starting point.

Mode ID Resolution (PAL) Description

-------- ----------------- ----------------------

00008000 640 × 200 (256) Hires

00008004 640 × 400 (512) Hires-Interlace

00008020 1280 × 200 (256) SuperHires

00008024 1280 × 400 (512) SuperHires-Interlace

- -

00039020 640 × 480 Productivity

00039024 640 × 960 Productivity-Interlace

- -

00041000 1008 × 800 (1024) A2024-10Hz

00049000 1008 × 800 (1024) A2024-15Hz

-------- ----------------- ----------------------

The next three items, "800w", "600h", and "3d" are probably quite obvious to you: they are the width, height, and depth of the screen. The letters that follow the numbers are stored in the file only to more easily describe their purpose. The trailing "1" indicates that autoscrolling is enabled for the screen ("0" disables it.)

Here is an entry for an A2024 15Hz screen with 4 colors (2 bitplanes) and autoscroll disabled:

Screen=CUSTOM:00049000,1008,800,2,0

Notice that in this example we left off the 'w', 'h', and 'd' letters. KingFisher doesn't care.

1.73 PREFERENCES/DISPLAY/Default public screen

If you select this item when KingFisher is operating on a custom screen, it will close its window and screen, and reopen the window on the system's default public screen.

If you select this item when KingFisher is already operating on a public screen, it will close and reopen its window on the default public screen provided that the current public screen is not already the default.

1.74 PREFERENCES/DISPLAY/Center main window

KingFisher usually remembers the most recently used position of its window. If you rather KingFisher opened its display always in the center of the screen, simply select this item.

Whenever you resize the window, KingFisher will automatically recenter the window.

Whenever you move the window, KingFisher will turn off the "Center main window" option to accomodate your wish to place the window in a specific position.

1.75 PREFERENCES/DISPLAY/Frame groups

When this option is selected, KingFisher will place "recessed" frames around groups of related gadgets to more clearly denote their relationships to each other. This will cause a somewhat more "cluttered" display and may not be pleasing to everybody, which is why it has been made an option.

1.76 PREFERENCES/DISPLAY/Use external gadget images

This option is available only if you are running Kickstart V39 or later.

If this option is enabled, KingFisher will attempt to load image files to replace any of the 28 major buttons of its interface. This feature relies on Datatypes to recognize and load the image files, so you may store the pictures in any format that your system can support. This may include any and all of the following: IFF, PNG, PCX, TIFF, TARGA, JPEG, BMP, SunRaster, GIF etc.

The size and colors of the supplied images define the size and appearance of the buttons they replace. It is suggested that a uniform, fairly modest set of colors be used to avoid running out of color pens too quickly, especially if you are operating KingFisher on a public screen, such as the Workbench.

KingFisher will cheerfully load huge images and create a completely dysfunctional interface. In the interest of power vs. reason, it was decided to give you the power to hang yourself, hoping that you can explore reasonable boundaries on your own and not feel fettered by artificial restrictions. To guide you in your quest: KingFisher's internal buttons do not exceed 31×31 pixels.

How to specify a replacement button image

Each button image is stored in its own file. Both the unselected and the selected state of the button must be provided, as KingFisher expects both. For a cheesy (but effective) ASCII representation, click here. If you have the example images installed (these ship only with upgrade copies of KingFisher due to diskspace limitations), then you may also get a real image . The height of the image file you specify should have an even number of lines, and the division between unselected and selected image should occur exactly halfway.

The name of the a button's image file is a composite of two variables:

1. The Display Aspect Ratio

The approximate resolution of KingFisher's display screen determines the aspect ratio. The ASPECT-ID is used as part of the filename. For example, the following table describes how KingFisher sees the display aspects:

RESOLUTION ASPECT-ID

640 × 200 "flat"

640 × 256 "flat"

320 × 200 "flat"

640 × 480 "square"

800 × 600 "square"

1024 × 768 "square"

320 × 400 "tall"

356 × 592 "tall"

N.B. If you always want a certain type of image to be loaded, regardless of the current display aspect ratio, then you must add an item DISPLAY-ASPECT to the .prefs file with a value 0 (flat), 1 (square), or 2 (tall).

2. The Button Name

Each of the 28 buttons that KingFisher uses as part of its interface has a GADGET-ID which becomes part of the filename:

Upper left corner:

JumpToFirst, JumpToLast

Flag gadgets:

FlagD, FlagX4, FlagX3, FlagX2, FlagX1, FlagH, FlagO, FlagM, Flag8, Flag7, Flag6, Flag5, Flag4, Flag3, Flag2, Flag1

Browse gadgets:

DiskPrev, DiskNext, VersPrev, VersNext, FishPrev, FishNext

Searching gadgets:

SearchReverse, SearchForward, ExprHistory, SearchResult

Constructing Filenames from these Variables

Filenames are constructed from ASPECT-ID and GADGET-ID as "gad.ASPECT-ID.GADGET-ID.image"

Example: "gad.square.ExprHistory.image"

Example: "gad.flat.FishNext.image"

Notice that the image files do not have filename extensions that specify their filetype (such as .IFF, .PNG, .TIFF, .JPEG, or .GIF) because you are using an Amiga and not one of those other computers.

If, despite all your valiant efforts, KingFisher seems to ignore an image file you've created for a button, check the following:

1. Are you running Kickstart V39 (3.0) or later?

2. Can MultiView display the image?

3. Is the image small enough not to use up available Chip RAM?

4. Have you named the image file(s) correctly?

5. Have you been a good friend to everyone recently? ;-)

1.77 gadimgexample

UUUUU

UUUUU Unselected image portion

UUUUU

UUUUU _____

SSSSS

SSSSS

SSSSS Selected image portion

SSSSS

Image width = Button Width

Image height = 2 × Button Height

1.78 PREFERENCES/DISPLAY/Sticky result window

When this option is selected, and a Search Result Window is open (see Stop on each match) then moving or resizing the main window causes the Search Result window to be repositioned as well. In effect, the Search Result Window "sticks" to the right edge of the main window as much as possible.

1.79 PREFERENCES/DISPLAY/Show database title

When this option is selected, the title of the current database is displayed above the main text display area of the main window. If you rarely work with multiple databases and you are running on a relatively small display screen, you may wish to leave this option disabled to preserve valuable screen space.

1.80 PREFERENCES/DISPLAY/Smart refresh

When this option is selected (Smart Refresh), KingFisher's windows will (whenever needed) be refresh by Intuition's own buffers. This can be significantly faster on screens with little depth, but on deep screens (such as a large 256 color Workbench) this not only slows down the refreshing but also requires Intuition to set aside enough memory for the refreshing operation to work, which can be quite a lot.

In Simple Refresh mode the display may not always be updated at once when portions of the window are "damaged" by obscuring windows, but Simple Refresh (i.e. Smart Refresh OFF) can significantly improve the display's performance and reduce the program's memory requirements. Experiment!

1.81 PREFERENCES/Printing

This submenu allows you to define printing-related things:

Load custom print format

Use default print format

One fish per page

Avoid page breaks

Add index info

1.82 PREFERENCES/PRINTING/Load custom print format

Load a Custom Format from a file and applies this to every fish (record) that is printed.

KingFisher remembers the print format in use for every database.

To create or edit a custom format, use the Edit custom format file command.

To revert to the built-in default format, use the Use default display format command.

Using AmigaOS Notification, the file specifying the print format is immediately updated if the version on disk is changed. The display will not reflect this change; printed output will. The notification change will not take effect during printing.

1.83 PREFERENCES/PRINTING/Use default print format

This command reverts back to the builtin default print format.

The command is unavailable if the default print format is already in use.

1.84 PREFERENCES/PRINTING/One fish per page

By enabling this command, each record (fish) is printed beginning at the top of a new page.

This command is unavailable while the printer is in use by KingFisher. You need to first Release printer before you can change this setting.

1.85 PREFERENCES/PRINTING/Avoid page breaks

The effect of this command is, perhaps, most easily described at hand of a little diagram to compare the effect visually. The idea is to prevent descriptions from being broken up by page breaks, forcing a record which will not fit on the current page to begin at the top of the next page:

NO Avoid page breaks YES Avoid page breaks

———————————— ————————————

|Name: KingFisher | |Name: KingFisher |

|Vers: 2.0 | |Vers: 2.0 |

|Text: blah blah blah | |Text: blah blah blah |

| blah blah blah | | blah blah blah |

| blah blah. | | blah blah. |

| | | |

|Name: MonkeyCommand | |Name: MonkeyCommand |

|Vers: 1.0 | |Vers: 1.0 |

|Text: blah blah blah | |Text: blah blah blah |

| blah. | | blah. |

| | | |

|Name: MonkeyCommand | | |

|Vers: 2.0 | | |

|Text: blah blah blah | | |

|----------| |----------|

| blah blah blah | |Name: MonkeyCommand |

| blah blah blah.| |Vers: 2.0 |

| | |Text: blah blah blah |

|Name: B5-Images | | blah blah blah |

|::::::::::::::::::::| | blah blah blah.|

| |

|Name: B5-Images |

|::::::::::::::::::::|

This command is unavailable while the printer is in use by KingFisher. You need to first Release printer before you can change this setting.

1.86 PREFERENCES/PRINTING/Add index info

Index information is added to the printout for each record in the following format:

.INDEXINFO=|DISK=1|FISH=17|FLAGS=8001|

This format will become a standard for a future Product-Info Specification and will be recognized by KingFisher's "Add Fish..." command.

This command is unavailable while the printer is in use by KingFisher. You need to first Release printer before you can change this setting.

1.87 PREFERENCES/Export

This submenu allows you to define exporting-related things:

Load custom export format

Use default export format

Export filename

Use importable raw format

Add index info

1.88 PREFERENCES/EXPORTING/Load custom export format

The internal default export format is merely one way of formatting the information in the database for export files. Different types of databases, in fact, may require different export formats.

Custom export formats are kept in files on disk, and KingFisher remembers the export format in use for every database.

To remove a custom export format and revert to the builtin default, select the Use default export format command.

To create or edit a custom format, use Edit custom format file.

Note that the Use importable raw format option, overrides the custom export format completely.

Using AmigaOS Notification, the file specifying the export format is immediately updated if the version on disk is changed. The display will not reflect this change; exported data will. The notification change will not take effect during an export operation.

1.89 PREFERENCES/EXPORTING/Use default export format

This command reverts back to the builtin default print format.

The command is unavailable if the default print format is already in use.

Note that the use of the Use importable raw format option overrides the use of custom or default formats entirely.

1.90 PREFERENCES/EXPORTING/Export filename

By default, the export filename, if you never specify a different name, is t:KF2.output. If you prefer a different filename, this command will let you do so, and KingFisher will remember the name from one session to the next until you change it again. An implicit Close export file will be issued for you.

1.91 PREFERENCES/EXPORTING/Use importable raw format

Forces the output to be in a special, re-importable format. The file can be transmitted via electronic mail (although national characters may not be preserved by the email transmission!) and can be added to any KingFisher Release 2 database through the Edit menu's Add fish from file command.

Notice that while this option is selected any custom export format is effectively disabled.

1.92 PREFERENCES/EXPORTING/Add index info

Index information is added to the export file for each record in the following format:

.INDEXINFO=|DISK=1|FISH=17|FLAGS=8001|

This format will become a standard for a future Product-Info Specification and will be recognized by KingFisher's "Add Fish..." command.

1.93 PREFERENCES/Searching

This submenu allows you to define searching-related things:

Stop on each match

Case sensitive

Trim blanks

Simple substrings

Use Search Masks

External command on failure

External command on success

Flash/beep screen when done

From here, you may also learn more about Search Expressions.

1.94 PREFERENCES/SEARCHING/Stop on each match

If checked, KingFisher performs an interactive search, stopping on each fish (record) where the search succeeded. Continue the search the way you started it.

If not checked, KingFisher performs a collective search, creating a Search Set in the process. When the search is complete, you are presented with the Search Result Window which lists all fish (records) where the search succeeded. If no such window opens, then no fish (records) were found at all.

1.95 PREFERENCES/SEARCHING/Case sensitive

When checked, then upper and lower case letters are treated during a search as distinct symbols, so that "a" is not the same as "A".

When not checked, then upper and lower case variations are immaterial during a search.

If, for example, you are looking for references to Kickstart and your search string consists of "KS" (abbreviation for Kickstart) you might be looking explicitly for only the all upper case version, and have no desire to locate words like these, too: ticks or packs.

1.96 PREFERENCES/SEARCHING/Trim blanks

When checked, causes trailing blank spaces in a search expression to be removed from the end of strings, unless these strings are explicitly quoted.

Blank spaces in a string gadget are usually quite invisible but may have a profound effect on the success of your search. Consider:

name$kingfisher

The spaces after "kingfisher" would tend to go unnoticed. If you want the spaces to be considered, it is suggested that you quote the string with spaces, such as: name$"kingfisher ", rather than uncheck this option.

1.97 PREFERENCES/SEARCHING/Simple Substrings

When checked, KingFisher will recognize most of the old KingFisher 1.x expression syntax, which is powerful enough for most purposes (and somewhat simpler) but does not support the specification of field names and the use of DOS Regular Expressions[1].

This option is named "simple substrings" because that is all that KingFisher 1 supported: a search for substrings in the entire record.

The QuickIndex will not be used when this option is in effect, and standard KingFisher Release 2 Search Expressions cannot be used, either.

---

[1] KingFisher 1.x supported DOS Regular Expressions, too, but this feature is not supported as part of the Simple Substrings mode.

1.98 PREFERENCES/SEARCHING/Use search masks

When checked, this option causes searches to rapidly scan for records matching the Search Mask as defined by the Edit Search Masks command. This feature represents an additional layer of selection / elimination before the regular expression evaluation is performed.

1.99 External command on failure

When a search completes and has located no records, and this option is checked, then an external command is executed through the shell (CLI) as specified by the SEARCH-DONE0-COMMAND option in the KingFisher2.prefs file. The "0" in the option key represents "none" or "failure".

Example: SEARCH-DONE0-COMMAND=say "bummer! kingfisher found nothing."

See also Flash/beep screen when done for a less complex alert option.

Note: The human speech software used by the SAY command was discontinued by Commodore after Workbench 2.1 (V38) was released. If you have this software on older disks you may want to consider installing DEVS:narrator.device and LIBS:translator.library. Both work perfectly with later releases of the operating system, and the SAY command can be a lot of fun!

1.100 External command on success

When a search completes and has located at least one record, and this option is checked, then an external command is executed through the shell (CLI) as specified by the SEARCH-DONE1-COMMAND option in the KingFisher2.prefs file. The "1" in the option key represents "one or more" or "success".

Example: SEARCH-DONE1-COMMAND=say "woopee, kingfisher found something!"

See also Flash/beep screen when done for a less complex alert option.

Note: The human speech software used by the SAY command was discontinued by Commodore after Workbench 2.1 (V38) was released. If you have this software on older disks you may want to consider installing DEVS:narrator.device and LIBS:translator.library. Both work perfectly with later releases of the operating system, and the SAY command can be a lot of fun!

1.101 Flash/beep screen when done

When a search completes (regardless of success or failure), and this option is checked, then KingFisher will "flash" all screens so that if you are busy working on something else, the "flash" will alert you that something wants your attention. To customize this feature further, use Workbench's Sound Preferences; you may set this to "flash" all screens and/or play a sound on your speakers.

NOTE: This command will not be executed if the screen upon which KingFisher's window is open is the front-most screen!

This option is available in addition to the more flexible External command on success and External command on failure options which require a little work on your part to configure.

1.102 PREFERENCES/Save Settings

Saves all settings to a file of your choosing, which is the same file from where KingFisher loaded its settings when it started. Specifying a file is done with KingFisher's SETTINGS tooltype at startup.

If no file is provided, KingFisher will save to the file KingFisher2.prefs in the current directory.

If no specific file was provided during startup, then KingFisher attempts to load its settings form the following files, one after other until it succeeds or finally fails:

KingFisher2.prefs (in the current directory)

ENV:KingFisher/KingFisher2.prefs

S:KingFisher2.prefs

If it finds one of these files during startup, then it will save to this file when you Save Settings.

You can save settings with this command to any file of your choosing but KingFisher will not be able to find and actually use the file unless you are saving it according to the above specifications.

1.103 HELP/Using KingFisher

KingFisher helps you...

a) Search for records matching your specification, and

b) Store new records.

Searching for records requires understanding what it is you are searching for. This requires an understanding of the Search Expressions through which you explain to KingFisher what you are (or are not) looking for.

Search Expressions are entered into the Search Expression gadget (press 'x' -- where the cursor appears, that's the place to type a Search Expression, and a search is initiated by pressing either the reverse or forward search buttons (the arrows with question mark super-imposed.) A search may either collect all possible matches and present you with a list from which you may select at will, or it interactively stops as soon as a match is found, requiring you to press the search button again when you are ready continue. The choice is yours.

Each fish (record) may have a combination of flags set to indicate it as deleted, owned, hidden, or marked for later retrieval. Eight (8) flags are available for your own use. Searching with the additional Use Search Mask in use can drastically reduce the number of searchable records and concentrate the search on the more likely candidates.

VersionLinks are a useful feature of KingFisher that link related records so that you can easily jump forward or backward to related records in the database to follow a program's development. This allows you to quickly locate the earliest or latest versions of a program. The information used by the Version Browse Buttons is constructed when you use the Build Version Links command.

The center piece of the GUI is a scrolling panel of text containing the description of the current record. The layout for this description is controlled by a format that you may redefine for your own purposes.

Things that may not be entirely obvious

* You can run multiple copies of KingFisher. Each copy can access the same or a different database. Each may perform an independent search. Some may be used for browsing, some for printing, some to export data...

* KingFisher's window is fully resizable and can live on public and custom screens.

* You can paint your own buttons

* An ARexx interface to the server is provided through RexxFisher

1.104 HELP/Searching

In order to make the most of KingFisher's powerful Search Expressions you must know their syntax: if you do not know your tools, how can you make effective use of them? Work through the Search Expression Tutorial to learn how!

Overview of Searching

1. Enter a Search Expression in to the Search Expression Gadget, or select a previously used expression from a "history" list by pressing the Expression History Button

2. Press either of the two Search Buttons to initiate a search in reverse or forward direction.

Advanced Searching

1. You may restrict your search to fish (records) in the database that match a certain pattern of flags. No significant time is spent examining records that do not match a certain pattern.

2. You may elect to perform an interactive or collective search, depending on the state of the Stop on each match option in the PREFERENCES/Searching menu.

3. Other options are available to alter the behavior of a search.

1.105 HELP/Printing

When you print with KingFisher, the printer device is kept open so that you may add further records to the printout without having another program intersperse its output.

During that time Other programs are unable to open the printer device, of course, meaning that they cannot print until you quit KingFisher or use the Release printer command from the Project menu.

You can also specify a different print format through the PREFERENCES/Printing menu, as well as dictate how records are broken up over page boundaries.

1.106 HELP/Exporting

Exporting records with KingFisher can mean two similar, but distinct operations: you may write records to a file the same way you might wish to print them on paper, or you export them so that they may be read back into KingFisher. Which way the Export command is to be used depends on how you configure the {" Use importable raw format " link PREFSEXPORTMARKERS} command in the PREFERENCES/Exporting menu.

The file to which you export can also be selected from that menu. Selecting a new file will automatically close the previously used file (if it was open.) The file to which KingFisher exports records is kept open much like the printer device is kept open between print operations. Use the Close export file command from the Project menu when you are finished with the file.

1.107 HELP/Databases

If you wish to make an existing database available to KingFisher, you have two options:

1. Install database... through KingFisher's menu command (see PROJECT menu) which handles the installation of a database fully automatically, even adjusting filename references, if necessary, to included the proper disk volume labels!

2. Perform the step manually -- this section also offers information on what changes may be needed to a .kfdb file to make it work when KingFisher itself fails to handle step (1) properly on its own, or when you need to create a new database.

1.108 CAUGHT FISH

This page unintentionally left blank. Please try the Index or Table of Contents.

1.109 CAUGHT FISH/Close window

This will merely close the Search Set window. It will not lose the current Search Set. To open the Search Set window again, click on the icon which contains an image of Fred Fish's "Fish" logo emulating the shark from "Jaws."

1.110 CAUGHT FISH/Apply Mask

This command applies a new mask to all fish in the Search Result window. More flexibility will be offered to this function in the future.

1.111 4 Advanced use of KingFisher

This chapter addresses the variously complex aspects of KingFisher. You should be familiar with the topics of the previous chapter, Working with KingFisher before delving into the more complex topics of this one.

Custom Formats for the display, printer, and export files

Reconfiguring the KFServer through its preferences file

Customizing .kfdb files for optimum performance

The KingFisher2.prefs file for advanced customizations

RexxFisher as an alternate or additional interface to the server

The KingFisher Developer Kit helps you create additional client software for KingFisher!

Paint the Buttons to alter KingFisher's appearance

Limitations part of KingFisher's design

1.112 5 RexxFisher

RexxFisher

RexxFisher is a client application that attaches to and provides an ARexx interface to the KFServer.

If run with a specific portname on the command line, another application can communicate with a particular RexxFisher client. If the portname already exists, then RexxFisher will fail. If not given a specific portname, RexxFisher will create one: REXXFISHER1 followed by REXXFISHER2 and REXXFISHER3 etc.

Results of calls are returned in the standard result variable (use "say result" to display a value returned by RexxFisher) and errors are returned in the portname.LASTERROR variable, where portname does not include the increment number (1, 2, 3, etc.) added when RexxFisher is invoked without a specific portname. I.e. If RexxFisher's port happens to be 'REXXFISHER4', then the error variable is REXXFISHER.LASTERROR, without the '4.'

All commands are understood also with an RF_ prefix (ex. RF_QUIT instead of QUIT), which may be used to avoid confusion with ARexx or other commands. These commands may be used in any mix of upper and lower case characters.

Commands ALWAYS usable

(These require no previous HELLO, nor a connection to the server)

DISABLE

HELP

QUIT

VERSION

LOCK

UNLOCK

Commands to establish a connection to the server

(Will start the server if it is not already running)

HELLO

Commands available only after a successful use of HELLO:

ADDFISH

BYE

FIND

GETFISH

LIST

OBTAIN

SELECT

SET

STATUS

USE

Demonstration Script

For your enlightenment, the RexxDemo.kfrx script is supplied. Please note that this script assumes that both the KFServer and RexxFisher are running in the background. Enter "rx rexxdemo.kfrx" to run the demo script from the CLI.

1.113 ARexx: VERSION

Usage:

VERSION

Returns RexxFisher's version tag, without the $VER: portion, of course. This will ALWAYS use the standard Style Guide compliant format such as:

RexxFisher 1.5 (8.5.94)

Example:

VERSION

say "Welcome to" result

1.114 ARexx: HELP

Usage:

HELP

Returns a list of all acceptable commands as well as some sort of command template to help you figure out what sort of parameters you might be able to get away with.

Example:

say "These commands are available to you:"

HELP

say result

1.115 ARexx: QUIT

Usage:

QUIT

Tells RexxFisher to shutdown. In a real environment, you might want to issue a command such as "DISABLE QUIT" to prevent the QUIT command from being recognized. This will also suppress the command from being listed by HELP.

Despite a disabled QUIT, RexxFisher will respond to a CTRL_C signal such as those sent by the c:BREAK command.

Example:

QUIT

1.116 ARexx: DISABLE

Usage:

DISABLE command

Disables a command so that RexxFisher will no longer be able to execute it. This prevents accidental shutdown of RexxFisher, for example by a "rogue script." Disabled commands will not be part of the HELP listing.

Example:

DISABLE QUIT

1.117 ARexx: HELLO

Usage:

HELLO "arbitrary identification"

Needs no previous login and will establish a connection to the KFServer. If the KFServer is not running, RexxFisher will attempt to start it in the exact same way that KingFisher (the GadTools client) tries to start KFServer. The only problem is that RexxFisher cannot (yet) be told to look in a place OTHER than the default directory for the KFServer. Start RexxFisher in the same directory where KFServer is located and all will be fine.

You should give a nice and descriptive name along with the HELLO, such as:

HELLO "BLAZEFISHER ARexx Script"

(my apologies, Dan! :^)

If you issue HELLO when already connected, then RexxFisher will issue an implicit BYE command to the server to disconnect you. RexxFisher will also do this when it is made to shutdown (either with QUIT or through a c:BREAK signal.)

Example:

HELLO "test script"

1.118 ARexx: BYE

Usage:

BYE

Sign off from the KFServer. This terminates your access to the server. If you forget this, then RexxFisher keeps the connection active for the next script, which may be confusing. RexxFisher has no idea, of course, if your script has terminated or is just idling around for no particular reason.

Example:

BYE

1.119 ARexx: LIST

Usage:

LIST

This obtains a list of all available databases from the server. The format of this list is as follows:

"Description\1database.kfdb\n

Description\1database.kfdb\n"

Which means that there are one or more lines of text each of which begins with a nice descriptive text for the database followed by a \1 character (which is an ASCII 1, ^A symbol) and followed then by the .kfdb name which you would need to give to the server through the USE function to make a selection.

Example:

LIST

1.120 ARexx: USE

Usage:

USE database

This selects a database by giving it the name of a .kfdb file. Please see LIST above for more information.

Example:

USE "Miniature.kfdb"

1.121 ARexx: FIND

Usage:

FIND "expression"

FIND AGAIN

FIND OPTION x

This command initiates, continues, or configures a search operation:

FIND "expression"

Allows the use of the same expression syntax as KingFisher. The expression is compiled and the function begins a search at once. If you receive an error, the result string is in the format "Error X in column Y" where the error value X is one of these:

X Meaning

1 Comparison Operator Expected ($ = != < > >= <=)

2 Logical Operator Expected (AND, OR, XOR)

3 Invalid Comparison Operator (ex: <! >< ... are bogus)

4 Mismatched Parentheses

5 Field Identifier Expected (must use "field op value")

6 Unsupported Feature -- obsolete

7 Internal Error

8 Incomplete Expression

Example:

FIND "name$kingfisher|name$aquarium"

FIND AGAIN

Search onward with the previously used expression. You must have an expression compiled with a previous FIND "expression" command, otherwise this will not work.

Example:

FIND AGAIN

FIND OPTION x

Alter the behavior of the next FIND command according to the option x:

FORWARD Search forward

BACKWARD Search backward

CASEIGNORE Upper/lower case ignored

CASEEXACT Upper/lower case important

TRIMBLANKS Trim trailing blanks off search-strings

NOTRIMBLANKS Do not trim blanks

SIMPLEEXPRESSION Uses original KF1.40 expressions

COMPLEXEXPRESSION Uses new KF2.0 expressions

SHOW List current options

Example:

FIND OPTION FORWARD

FIND OPTION CASEIGNORE

FIND OPTION TRIMBLANKS

FIND OPTION COMPLEXEXPRESSION

FIND OPTION SHOW

say result

NOTE: There is no way to interrupt a search in progress. Depending on user-feedback, a future version of RexxFisher may accept a break signal to interrupt a search.

1.122 ARexx: GETFISH

Usage:

GETFISH fishnum

GETFISH fishnum width

GETFISH fishnum width displayformat

Retrieve a specific fish by record number. The command has a second, optional parameter that determines if the resulting string is formatted or retrieved in raw form. A positive number for the 2nd parameter indicates the column width of the display that the text should fit. The resulting text, when formatted, will have an appearance much like that in KingFisher's ListView.

A 3rd parameter specifies a display format other than the default.

Notice that a record number of 0 retrieves the most recently retrieved record. It is best not to rely on this functionality, especially after a search operation but may be useful in some cases:

Example:

GETFISH 3693

Retrieves record 3693 without special formatting.

GETFISH 3693 75

Retrieves record 3693 without special formatting, but wordwraps the buffer to fit within 75 columns of non-proportional text.

GETFISH 0 75 "NAME=@{name}\nAUTHOR=@{author}\nDESCRIPTION=@{description}"

Retrieves the last used record (0 has that special meaning), wordwraps the buffer to fit within 75 columns of non-proportional text, and uses a special custom format (the strange mess in ""'s) to determine what is included and how it is formatted.

1.123 ARexx: ADDFISH

Usage:

ADDFISH "filename"

Adds all fish from the indicated file to the current database.

Example:

ADDFISH 't:new-fish.txt'

1.124 ARexx: OBTAIN

Usage:

OBTAIN what

Obtains a variety of information from the server, according to the parameter given:

DISK The current disk number.

FISH The current fish number, usable as the 1st parameter to the GETFISH command.

FLAGS The flag bits of the current fish; the values currently defined, although not necessarily setup for each fish, are:

0x0100 Marked for retrieval

0x0200 Marked for ownership

0x0400 Marked to stay hidden in searches

0x8000 Marked to be deleted

Bits in the range 0x0001 through 0x0080 are user defined.

PVER The fish number of the PREVIOUS VERSION; the value 0 is returned if no previous version exists.

NVER The fish number of the NEXT VERSION; the value 0 is returned if no next version exists.

DBNAME The descriptive name of the database in use.

DBFILE The filename (ending with .kfdb) of the database in use. Such a filename can be passed to the USE command.

DBSIZE The number of records in the current database, which is also the highest fish number you can pass to the GETFISH command.

Example:

OBTAIN DISK

OBTAIN FISH

OBTAIN FLAGS

OBTAIN PVER

OBTAIN NVER

OBTAIN DBNAME

OBTAIN DBFILE

OBTAIN DBSIZE

1.125 ARexx: SELECT

Usage:

SELECT fishnumber

Selects a specific database record, but does not actually retrieve information. This is faster if all you need is use the OBTAIN command next, perhaps to locate a record with specific flags, or to follow a list of records along the VersionLinks.

1.126 ARexx: SET

Usage:

SET PVER fishnumber

SET NVER fishnumber

SET FLAGS flags

NOTE: In prior versions this command was SETVLINK.

Sets the previous or next version link for the current fish. A value of 0 for the fishnumber indicates no previous or next version. It is up to you to assure that records are properly linked in both directions, as the KFServer will not interfere with your selections. Furthermore, if a record already has a link number set, this value will be replaced with the new selection.

Example:

/* link record 17 to 76 and vice versa */

SELECT 17

SET NVER 76

SELECT 76

SET PVER 17

NOTICE: In older versions of this example, we used GETFISH instead of SELECT for improved speed.

NOTICE: Links need not be sequential. Thus, record 17 may have record 76 as its predecessor. Versions of KingFisher before 2.9 do not properly handle such cases, however.

1.127 ARexx: STATUS

Usage:

STATUS

Retrieves status information from the server. This is effectively the same as what KingFisher displays in the Status command (rightAmiga-I) except that it applies to RexxFisher.

Example:

STATUS

say result

1.128 ARexx: LOCK and UNLOCK

Usage:

LOCK

UNLOCK key

Locking RexxFisher may be necessary to prevent other programs for inadvertently connecting to it and fouling up the current status while your ARexx script is passing information to another program, thereby having relinquished access to RexxFisher (i.e. ADDRESS REXXFISHER1; ... ; ADDRESS VLT ; ...)

LOCK returns (in the Rexx RESULT variable) a unique value which you should store. Pass this value to the UNLOCK command to unlock RexxFisher again. Without this key, RexxFisher will remain locked. ALWAYS call UNLOCK before exiting your script, else the key is (probably) lost and RexxFisher must be shutdown with a CTRL-C signal and restarted!

Example:

OPTIONS RESULTS

ADDRESS REXXFISHER1

VERSION

say RESULT

LOCK

myKey = RESULT

ADDRESS VLT

: : :

ADDRESS REXXFISHER1

UNLOCK myKey

: : :

BYE

1.129 6 Developing for KingFisher

What is the KingFisher Developer Kit?

If you have registered KingFisher you may get a KingFisher Developer Kit to help you create client software for the KingFisher server:

* Instead of relying on ARexx and RexxFisher, for example, you might wish to write a small, special-purpose client to help your BBS users search for programs,

* You could write client applications to perform extensive database manipulations,

* You might even want to write a replacement for the KingFisher GUI, perhaps using the MUI (Magic User Interface) system.

* Or maybe you're just curious what the interface looks like or what a little playing around could produce!

The KingFisher Developer Kit is available free via email or (if you specify) as part of a regular upgrade[1] only from the author. The support API (Application Program Interface) you will receive in source form is written to compile with SAS/C 6.56 but it should also compile with DICE 3. It includes example source code to help you get started with a simple client and build from there.

Features of the Developer Kit

The following links will only work if you have the Developer Kit installed. The source code links work especially well if you have the C/C++ Datatype installed!

kf-api.h

All your client programs #include this file to obtain prototype declarations for functions, as well as various structures that are needed to interface with the KFServer

kf-api.c

You must compile this file (although a SAS/C 6.56 generated .o (object) file is also supplied for your convenience.) All your client programs then link with this .o (object) file. The file supports automatically starting the KFServer. All you need to do is call the functions defined in the kf-api.h file.

kf.h

If you #include kf-api.h then this file is automatically included, providing you with details on all server calls, such as expected parameters, return values, and details on error conditions.

CLIent.c

The very first client program. I wrote this without the use of the "kf-api.(c|h)" files (they did not exist at the time) to effect early testing of the server even before the KingFisher GUI was written. Although this program compiles and works, it is also an example of how not to write a client -- life is much easier with the use of the "kf-api.(c|h)" files!

ReOrder.c

A good demonstration program to show how to use the functions in the "kf-api.(c|h)" files to your advantage. The program transfers one or more records from one KingFisher database to another, demonstrating how to access two databases simultaneously, how to select records, read from one and write to the other database.

demo.c

A small starter project to help you get stared. It merely provides a framework from which to expand.

Status.IFF4

A 4 bitplane (16 color) IFF ILBM (picture) that provides color-coded details on the output generated by a KFServer STATUS command. This may prove helpful to developers to diagnose their software's behavior with respect to the KFServer.

README

An introductory "ReadMe" file that describes how to get started with the Developer Kit.

---

[1] As diskspace is limited on floppy disks, upgrades that include the developer kit will not include the "1000 Fish Disk" database and vice versa.

1.130 7 Upgrades and Support

Upgrades

You may obtain upgrades in any of the following ways:

* Free by downloading new releases from Aminet, your local BBS, or new CD-ROMs,

* Free by visiting the KingFisher World Wide Web home page at http://www.wam.umd.edu/~walrus/KingFisher.html

* Free keyfiles are are available to registered users via email. By regular mail ("snail mail") they require an upgrade payment to cover distribution costs.

* By sending the appropriate payment to one of the registration sites.

Technical Support

If you do not have access to electronic mail, please write to KingFisher's author. Except for two letters (which got lost a few years ago) I have always answered snail mail correspondence.

Technical support through email is available by email. Send a message to walrus@wam.umd.edu detailing your questions, difficulties, suggestions, or requests. If you place the word "KingFisher" in the subject line, chances are that your message is answered more swiftly.

1.131 KingFisher Database Server

KFServer is the server portion of KingFisher's Client-Server access model. The program is started automatically by the KingFisher (GUI) and RexxFisher (ARexx) clients. KFServer requires that the file KFServer.prefs is available and properly configured. KFServer will not run if the file is missing or fails to specify required information.

Information on .kfdb (Database Descriptor) files is available here.

1.132 The Keyfile

"Not the dreaded keyfile!"

Making one set of patches available after another to keep all previous users of KingFisher up to date has not only become a chore for me but also a confusing and frustrating labor for others. In addition, the set of patches have grown to occupy nearly a megabyte of space on every site of the global Aminet and numerous local BBSs. Downloading a new set of patches requires quite a while each time a new version arrives, with perhaps as much as 90% of the downloaded portion useless to your particular version of KingFisher.

The solution to this increasingly painful dilemma is obviously a keyfile. This is a file named .KingFisher.key (notice the leading dot) whose content tells KFServer and KingFisher that you have registered the program.

Starting with KingFisher 2.20, only a single kind of executable will be distributed. The presence or lack of the keyfile will determine registration status. You paid for your keyfile. Keep it safe and don't let it fall into someone else's hands.

The keyfile should be stored in the directory where KFServer and KingFisher are installed. If not found there, both programs will examine the environment variable KEYPATH (if it exists) and look in the directory specified there. If not found there, either, they will search for the file in the following logical volumes:

KEYS:

KEYFILE:

KEYFILES:

S:

If still not found, the software operates under the assumption that it is an unregistered copy.

Previously registered users are entitled to receive their keyfile via email for free. PGP is supported and recommended, but not required.

1.133 Tooltypes & KingFisher2.prefs

KingFisher's preferences file is named KingFisher2.prefs. A different filename can be specified with the use of the icon tooltype SETTINGS (example: "SETTINGS=KF2.prefs") This documentation assumes that the default name is in use.

The preferences file contains lines that can also be used in the icon tooltypes. KingFisher processes the preferences file before it processes the tooltypes, wherefore tooltypes (in icons) will override equivalent settings in the preferences file. This file is always re-created when KingFisher exits (with Auto-save on exit enabled) or when you specifically Save Preferences .

The following text describes only settings that KingFisher will not allow you to alter from within the GUI.

Screen=

This setting describes the display screen on which KingFisher will open its windows. If blank, it will use the default public screen. If you are running Kickstart 38 or later, you need not concern yourself with this entry. Users of Kickstart 37 (2.04), however, should click here for details on altering this entry.

Help-File=KingFisher2.guide

If you plan to store this .guide in a directory other than KingFisher's default directory, you should specify the full name here.

Search-Priority=-1

Search-Priority-Relative=Yes

Alters the priority of KingFisher while it is performing a search operation. If the 'Search-Priority-Relative' setting is set to 'Yes' then the given 'Search-Priority' is added to the current priority instead of representing an absolute value.

Search-Done1-Command=

Search-Done0-Command=

Notice the 1/0 that distinguishes these two entries. The 1 stands for "success"; the 0, for failure. That said, whenever a search completes, KingFisher will invoke one or the other shell command if they are defined. I use the following for myself:

Search-Done1-Command=SYS:Utilities/Say "KingFisher has found some fish!" >nil:

Search-Done0-Command=SYS:Utilities/Say "KingFisher found nothing." >nil:

To execute multiple commands, place them into a script file and invoke that.

MaxHistory=50

The maximum number of expressions kept in the expression history list. This value may not exceed 1024.

MaxHistHeight=10

The maximum height of the expression history window. This prevents the window from getting ridiculously tall when a lot of expressions are in the field. The window will be further limited by the amount of space beneath the expression history window and the bottom edge of the display screen.

MaxDBInfo=10

The maximum number of DBInfo entries that KingFisher will maintain. This value may not exceed 1024. These entries store information on the position in previously visited databases, the display, print, and export formats in use, etc. Do not alter the DBInfo lines by hand.

NoOutput

Suppresses the copyright banner that KingFisher ordinarily sends to the console when it first starts up.

1.134 Search Expression Tutorial

Search expressions provide a powerful mechanism for specifying what information you are looking for and what you are not looking for. To exploit KingFisher's true potential, it is necessary that you familiarize yourself with its expression syntax. This section helps you do that.

NOTE: If you have the Simple substrings option selected, then KingFisher will accept a simplified version of the expression syntax described here. Watch for the symbol × to to point out important sections pertaining to this topic!

To check the status of the "Simple Substrings" option, check the SEARCHING item in the PREFERENCES menu:

| Preferences |

| Global » |

| Display » |

| Printing » |

| Exporting » |

| Searching |_____

| | Stop on each match |

| | Case sensitive |

|__ | Trim blanks |

| Simple substrings |

| Use search masks |

|_____|

For other options that affect searching, please see the PREFERENCES/Searching menu.

Part 1: A simple expression

Part 2: Composite expressions

Part 3: Sequence of evaluation

Part 4: Negation

Part 5: Expression shortcuts

Part 6: AmigaDOS patterns

1.135 Search Expression Tutorial, Part 1: A simple expression

An expression tells KingFisher exactly what you are looking for. An expression takes the form

field comparison value

× When the Simple substrings option is selected, only the value portion is used, and KingFisher automatically substitutes a * (all) for the field, and $ (substring search) for the comparison operator.

Choices for...

field the name of any database field (see also the Product-Info Specification). An example of this would be name, description, version, or author. You may use a * instead of a specific name to indicate that you wish KingFisher to search all existing fields in the records.

comparison one or two character symbol. The following are valid:

= or == The field contents are equal to the value

<> or != The field contents are not the same as the value

<= The field contents are alphabetically less than or equal to the value

>= The field contents are alphabetically greater than or equal to the value

< ... alphabetically less

> ... alphabetically greater

$ The field contents contain the substring given by the value

The field contents are compared to the given DOS Pattern.

value a string of characters. If the string contains any of the special symbols, such as: ( ) & | ^ or blank spaces, it becomes necessary to enclose the string in single or double matching quotes: "" or ”

Examples:

name=kingfisher

version >=2

author $ "matt dillon"

Notice that spaces surrounding the three parts of the expression (field, comparison, and value) are unimportant. In fact, spaces are not relevant in expressions and will be ignored unless they appear in a quoted string.

Next: Composite expressions

1.136 Search Expression Tutorial, Part 2: Composite expressions

Let us now combine two expressions to form a more complex one:

name = kingfisher & version >= 2

Notice the new symbol, &, that we used. This is a boolean operator that you can use to connect two expressions. The following boolean operators are valid:

& Logical AND

Both the expression on the left and on the right side of the operator must evaluate to TRUE, or else the combined expression formed from the two will evaluate to FALSE.

| Logical OR

Inclusive OR

If either or both of the expressions on the left and on the right side of the operator evaluate to TRUE, then the combined expression also evaluates to TRUE.

^ Logical XOR

Exclusive OR

Either one, but NOT both expressions on the left and on the right side of the operator must evaluate to TRUE, otherwise the combined expression is FALSE.

The expression above, therefore, means: if the name equals 'kingfisher' AND the version is (alphabetically) greater than or equal to '2', then we have found a record that might be interesting.

Test your new-found understanding: What does the following mean? Refer to the previous section if you are not sure about the '$' symbol (I assume you are well versed with '=', no?)

name $ 'aquarium' | name = kingfisher

It means if the string 'aquarium' appears as a substring ($) in the name field, OR the name equals 'kingfisher' then this is a match.

Next: Sequence of Evaluation

1.137 Search Expression Tutorial, Part 3: Sequence of Evaluation

Let us examine a more complex expression. Assume we want to find all the records with 'aquarium' part of the name, OR all the ones named kingfisher which have a version of at least 2. Does the following expression work?

name$aquarium | name=kingfisher & version>=2

The answer is no! KingFisher uses left-to-right evaluation[1], meaning that the expression first evaluates

name$aquarium

then it evaluates

name=kingfisher

and then checks if EITHER is true. Only then will it proceed to test the version. If we use parentheses to demonstrate how KingFisher actually evaluates the expression, we notice immediately that we had something else in mind:

(name$aquarium | name=kingfisher) & version>=2

But KingFisher does understand parentheses, so we can easily fix the expression to do what we meant it to do in the first place. We just have to remember to use them:

name$aquarium | (name=kingfisher & version>=2)

You can use many levels of nested parentheses, and it is always safer to "overdose" on parentheses than to assume that the expression really means what you hope to express.

Next: Negation

---

[1] Many programming languages assign priorities to certain operations. Much like multiplication has priority over addition, the logical AND is often given priority over the logical OR. KingFisher, however, is strictly left-to-right in its evaluation.

1.138 Search Expression Tutorial, Part 4: Negation

Any part of the expression can be negated. If you wish to find all software in which Udo Schuermann had no part, you would begin by entering the following:

name$"udo schuermann"

and then negating the expression by placing the negation operator in front of it:

~name$"udo schuermann"

KingFisher recognizes both the ~ and the ! symbols as negation operators. Which one you use depends entirely on your preference.

Next: Expression Shortcuts

1.139 Search Expression Tutorial, Part 5: Expression Shortcuts

To make a variety of comparisons to a single field a little easier, an expression shortcut represents enhanced ease of entry and formulation; it does not affect the speed of searches in any way. Expression shortcuts are not available when the Simple Substring is active.

Let us assume that you are looking for a large number of programs by name. You could enter the following expression:

name$post | name$music | name$fish | name$king | name$aqua

You will quickly come to the point where typing "name$" over and over becomes rather tedious. KingFisher allows you to enter the same expression this way:

name$(post|music|fish|king|aqua)

Observe that the expression in parentheses consists only of values and boolean operators (the | symbol in our example, signifying an OR-expression.) The field identifier ('name' in our example) and how this field is treated (as a substring, see the $ symbol) is a constant and does not change for the values given.

Consider:

name="kingfisher" | name$"post"

This expression cannot be converted to a shortcut because in one portion we test for equality (=) and in the other we test for a substring ($).

In order to impose an evaluation order, the portion in ()'s may contain multiple levels of parentheses to prioritize the evaluation order:

name$(fish&(king|net|sticks))

represents:

name$fish & (name$king | name$net | name$sticks)

Next: AmigaDOS Patterns

1.140 Search Expression Tutorial, Part 6: AmigaDOS Patterns

The special comparison symbol '#' serves to identify the following string as a DOS Pattern. In case you are not familiar with these, look in your AmigaDOS manual (the AmigaDOS 3.1 manual describes these in Chapter 3, pp. 16-18) under the heading Pattern Matching and/or Wildcards.

Consider:

name#"#?king#?"

This pattern is the exact same thing as

name$"king"

except that the former uses the dos.library's pattern matching, whereas the latter uses a simpler, but faster substring comparison. Note especially the difference between the '#' vs. the '$' operator to tell KingFisher how you want the string treated.

The example above is probably as unexciting as a slice of stale bread, but if you are familiar with DOS Patterns, you should have already realized that they can help you build some powerful expressions. You must understand, however, that the dos.library matches the entire expression against the field contents, whereby you will usually want to surround the string with the "0-or-more-characters" wildcard: #?

The expression name#"king" will NOT find "kingfisher", nor will it match any program that is not exactly named "king". Read the DOS documentation for more details. Enough said.

NOTE: The Trim Blanks option will affect the search string, unless it is enclosed in quotes. You are strongly urged to always enclose your DOS Patterns in quotes!

NOTE: If you specify a DOS Pattern that contains no wildcards, in other words it is a simple string constant, then KingFisher will convert the expression to an equality search. Example:

name # "kingfisher"

will not use the dos.library for pattern matching, but rather convert the expression to

name = "kingfisher"

and evaluate the equality directly and without assistance from DOS.

And that concludes our tutorial on expressions.

1.141 Client-Server Technology

What is "Client-Server"?

Client-Server is a database technology whereby one unique portion of the software, called the server, is responsible for controlling access to data, while one or more clients talk to this server and request data. It is the clients that are responsible for presenting the data, perhaps alter it and then handing it back to the server for storage; It is the server's responsibility to make sure that clients don't update the same data simultaneously.

The Client-Server model provides for efficient, successful, and safe multiuser arbitration and is widely used in the computer industry.

KingFisher implements the powerful Client-Server concept to provide you with safe access to one or more databases, and to extend this access, if you wish, to a number of simultaneous users that may have access to your system through BBS software.

How does KingFisher implement Client-Server Technology on the Amiga?

The Amiga's multitasking Exec (the software that handles all aspects of multitasking, including interprocess communication) provides a highspeed method of passing large amounts of information from one task to another. Hundreds, if not thousands of messages can be exchanged by two tasks every second!

When the KFServer is started, it creates a message port, a type of rendezvous point between programs, to which clients deposit requests for processing. The server processes these requests one after anther (first come, first served; FIFO) and returns the results to the sending process.

And that, quite frankly, is all there is to it!

Naturally, the protocol of how exactly to ask the KFServer for information requires some attention to detail. If you are interested in writing client software for KFServer and you are a registered user, you are eligible for the free KingFisher Developer Kit to help you get started. This package takes care of all the gruesome details of exchanging messages between your application and the server.

References:

* AMIGA ROM Kernel Reference Manual: Libraries. Exec Chapter.

* COMMODORE-AMIGA "C" include files: exec/ports.h

1.142 Registration Sites

Please select the registration site most convenient to you:

Main Registration Site

The main registration site is handled by KingFisher's author in the USA. There you may register, upgrade, and find technical support for KingFisher.

European Registration Site

The European site in Germany only handles registrations and upgrades, but not technical support.

1.143 Author's Address

Technical Support

If you are looking for technical support, then the address below is what you should use. It is the author's address and main registration site.

Registration and Upgrades

If you are not looking for technical support but wish to register or upgrade KingFisher and you live in Europe (or closer to Europe than the USA), then you may wish to use the European Registration Site instead.

In July 1995, my address changed. I tend to stay at any place at least two or three years, and have been known to grow roots for as long as six years in one spot. If you read this in 1997 or later you might want to drop me an email first, as my email address has not changed since 1987.

Registration: $20(US)[1]

Upgrades: $5(US)[2]

Send payment in form of cash, Money Order, or a personal check drawn on an American bank to:

Udo Schuermann

7022 Hanover Parkway, Apt. C2

Greenbelt, MD 20770-2049

email: walrus@wam.umd.edu

www: http://www.wam.umd.edu/~walrus/

---

[1] Members of User Groups get a 20% discount on initial registration, provided you indicate the name and address of your user group. The registration price with a 20% discount is $16(US). Pass the word!

[2] If you are a programmer, you might be interested in the Developer Kit

1.144 European Registration Site

Technical Support

If you are looking for technical support, please check with KingFisher's author. Do not use the European address for technical support.

Registration and Upgrades

Registration: DM30[1]

Upgrades: DM10[2]

Send payment in form of cash, a EuroCheque or a check drawn on a German bank. Direct bank transfers (Überweisungen) should go to: Stadtsparkasse Herford (Germany), Konto-Nr: 514 646:

Uwe Schürkamp

Jöllenbecker Weg 4

32051 Herford

GERMANY

---

[1] Members of User Groups get a 20% discount on initial registration, provided you indicate the name and address of your user group. The registration price with a 20% discount is DM24. Pass the word!

[2] If you are a programmer, and you are interested in the Developer Kit, you must upgrade with the Main Registration Site in the USA.

1.145 Sending Cash by Mail

Caution!

Sending cash by mail is discouraged for reasons that it's untracable. In the event that it gets lost in the mail, the loss would be yours to carry. If you live in a land far, far away from either Europe or the USA, then your best bet may be to send cash, as many (international) banks do not cater to the exchange of small sums. If you decide to send cash in the mail, make sure that your letter in no way reveals its contents to someone peering at it against the light, or feeling the bank notes through the envelope.

1.146 Translators

Through the commendable effort of the following people, KingFisher is available to you in a language other than English:

Dansk (Danish)

Finn Kettner <flynn@scala.ping.dk>

Deutsch (German)

Uwe Schürkamp <hoover@mathematik.uni-bielefeld.de>

Nederlands (Dutch)

Marcel Offermans <marcel@dutw30.tudelft.nl>

Español (Spanish)

Eduardo Delgado <tdatos@cpd.uva.es>

Suomi (Finnish)

Janne J Kalliola <plastic@vipunen.hut.fi>

Svenska (Swedish)

Jan Simonson <Jan_Simonson@augs.se>

I am always looking for new translators, especially if the missing language is a major language (such as French, which is still missing.) If you have corrections to the existing translation, or you find that portions of KingFisher are not yet translated, I would like to hear from you, too! KingFisher's shareware status does not permit me to pay you, but if your effort is significant enough you will get 50% off the registration price of KingFisher. In any case, you will definitely get your name into the list above!

1.147 Creating a new Database

It is the KingFisher Database Server which is responsible for actually reading and writing databases on disk. KFServer knows only about databases for which a Database Descriptor file exists whose name ends with .kfdb. This file must exist in KFServer's default directory, i.e. the directory where the KFServer is located. KFServer will not recognize .kfdb files that are stored in any other place.

It is the purpose of these .kfdb files to describe all components of a database, i.e. what the name of the index file is, which records are stored in which data files, and less importantly the complete name of the database to be presented to the you when you select one for access. More detail on the contents of these .kfdb files is given here.

Things to remember

* In most cases it is sufficient to copy an existing .kfdb file, then alter its contents.

* The name of the .kfdb file does not matter, so long as it ends in .kfdb

Mini tutorial

Let us setup a database for your Amiga Club, using a single file named ClubDisk:Club.data to store all the records. Our index file will be named ClubDisk:Club.index, and a QuickIndex file based on the name field will be stored in ClubDisk:Club-name.index. The name of the KingFisher Database file shall be AmigaClub.kfdb

AmigaClub.kfdb should contain something like this:

database-name=Our Outstanding Amiga Club's Own Software Collection

section=00000,99999,ClubDisk:Club.data

index=inram

index-increment=100

index-name=Club.index

field-index-field=name

field-index-name=ClubDisk:Club-name.index

keep-open=no

read-only=no

Once you save this file (even while KingFisher and the KFServer are running!) you can then immediately use this new database and add records to it!

1.148 Version Links

Version Links connect any record to a preceding and a following record (if any), allowing you to quickly examine earlier or later versions of a program by browsing with the Previous and Next Version gadgets.

Version Links are maintained as part of the main index and will be lost when you perform a EDIT/Reconstruct database index command, but can be rebuilt with the EDIT/Build Version links command.

1.149 Search Sets

If the PREFERENCES/Searching/Stop on each match menu item is checked, KingFisher interrupts a search operation as soon as it finds a record that matches the Search Expression; if the menu item is not checked, KingFisher builds a Search Set, instead. The first is called an interactive search; the latter, a collective search.

A Search Set is a list of all records that have matched a given Search Expression. A Search Set can be saved to disk and loaded again later to keep you from having to perform a particular search more than once.

Search Sets need not be saved to become useful, however. After a collective search is completed, the Search Set is available in a scrolling "listview" window. Clicking on any fish (record) named in this window immediately retrieves this record from the database and displays it in the main window.

When a Search Set is available, the Search Result Set button in the lower right corner will open and close the Search Result Window.

The Search Set is cleared from memory by one of the following actions:

Quitting KingFisher

Beginning another search

Loading a new Search Set

Click here for more information on operating the Search Result Window.

1.150 Custom Formats

The format that KingFisher uses to display, print, and export fish is programmable! This means you can customize the display in just about any way you like. This section is designed to show you how to tap this powerful feature.

NOTE: Through a standard Operating System feature called "Notification," KingFisher will learn of any change to a custom format file which is currently in use, and will re-load the file.

The display will be updated to reflect the changes, making it especially easy to create and fine-tune new custom formats. Likewise, files specifying printer and export formats are also updated, although that will become apparent only when output is generated.

1.151 Part 1: Default Format Tutorial

The built-in default format for the display, the printer, and exporting is this:

@{name}@{version| }@{date| (|)}\n

@{fullname|\t|\n}

@{short|\t|\n}

@{author|\tBy\>|\n}

\n

@{description}\n

\N

@{requirements|Requirements:\n\>|\n}

@{restrictions|Restrictions:\n\>|\n}

\N

@{address|Author's Address:\n\>|\n}

@{phone|Phone:\>|\n}

@{fax|Fax:\>|\n}

@{email|Email:\>|\n}

@{url|URL:\>|\n}

\N

@{distribution|Distribution:\>|\n}

@{price|Price:\>|\n\n}

@{installsize|Installs:\>|\n}

@{source|Source:\>|\n\n}

@{exectype|ExecType:\>|\n}

@{construction|Construction:\>|\n}

@{tested|Tested:\>|\n\n}

@{docs|Docs:\n\>|\n\n}

@{references|References:\n\>|\n\n}

@{reference|References:\n\>|\n\n}

@{keywords|Keywords:\>|\n\n}

@{described-by|Described-by:\>|\n}

@{submitted-by|Submitted-by:\>|\n}

@{submittal|Submittal:\>|\n}

@{stored-in|Stored-In:\>|\n}

@{aminet-dir|Aminet: \>|/}@{aminet-file}

One glance at the "mess" above may have convinced you that creating custom formats for KingFisher is not something you ever want to attempt. Let the apparent complexity not deter you, however!

First, you have probably noticed that almost everything begins with the symbols @{ followed by something ugly and is terminated with a } Coincidence? Definitely not!

Let's look at the first line and show all four elements on that line, one at a time, in bold:

@{name}@{version| }@{date| (|)}\n

@{name}@{version| }@{date| (|)}\n

@{name}@{version| }@{date| (|)}\n

@{name}@{version| }@{date| (|)}\n

You will notice that the first item is @{name} which looks simple enough. It displays the contents of the name field!

The second line, @{version| } looks a little stranger: there is a | symbol stuck in there, along with a blank space. Let me quickly point out that the strange stuff between @{ and } symbols can contain more than only a field name. The complete format (without the blank spaces!) is:

@{ field | prefix | suffix }

The | symbol is a separator, and the blank space is the value of the prefix portion. Prefix? Suffix? What, you may ask, is the point of this weird concoction? Why not put the space outside the whole @{} construct?

The reason is that when the specified field is missing from a database record or contains no information, then neither the prefix nor the suffix, if any are given, will be processed. This neat trick is used extensively and permits us to print something additional before and/or after the field contents if the field contains data, and leave out surrounding text if the field is empty.

Let's look at the third line item, @{date| (|)}, which contains both prefix and suffix strings: If a date field does not exist in the database, there won't be a non-sensical " ()" shown. It's a content sensitive display format!

The fourth item is one that will be quite familiar to C programmers: \n is a newline. KingFisher begins the following text on a new line. This allows you to break up things into more readable sections. The end of line in a database record is actually treated as a mere blank space by KingFisher so that you can break things up into a more readable form on your own.

Click here for the next tutorial section.

1.152 Part 2: Special Formatting Symbols

Special formatting symbols may be used both inside a @{} construct and outside. They have a variety of uses:

Fields denoted as "free form" disregard the physical line breaks, performing word-wrapping of the text within the current margins of the display window or the printer's paper. To affect the layout to some degree, the following symbols may be inserted at strategic places in the text:

\. Produces a single . (dot) especially useful if/when such a dot is found (against normal practice) at the very beginning of a line of text and where it would then be misunderstood to represent a field-name.

\@ Produces a @ symbol.

\n Forces a newline, an end-of-paragraph.

\N A conditional newline (end of paragraph), inserted only if the previous line is not already a blank line. Will also suppress extra spacing between paragraphs (esp. in the "description" field.)

\t A tab, which is equivalent to approximately 5 "n" characters.

\> A paragraph indent, which allows you to create hanging indentations of text. When used as part of a custom format file, the left margin is indented; As part of a field specification, the contents of the field are indented; as part of a field in a Product-Info file, the indentation lasts until the next \n (newline) character is encountered.

\# Toggles between verbatim and flow mode. Most fields flow their contents within the window margins, treating newlines as blanks and multiple blanks as no more than one space. Some other fields use line breaks verbatim and will not collapse multiple blanks into a single space. The \# is especially useful in flow fields, such as "description", if entering a table is required. Remember, however, that this should NEVER depend on a fixed-pitch font! For some more information see the next section.

\- Adds a line of dashes (------...) up to the end of the line and begins subsequent text on the next line. Makes for nice dividers!

All other text encountered is transferred verbatim to the display.

Click here for the next tutorial section.

1.153 Part 3: Field Option Sets

There are times when KingFisher's default format does not always satisfy how a particular field's contents are presented. For example: many people still format the description field as if it were to be displayed with a fixed-pitch font in a non-sizable window. KingFisher, of course, ignores end-of-line characters in the file and word-wraps the entire line as if it was a continuous stream of characters. If an entire database depends on such a misunderstanding, it may be necessary and desirable to alter the basic layout behavior for a field using field options[1].

One or more options for a field are specified by placing them within []'s immediately before the field name. At this time the number of available options is quite small, but may grow in the future. Here is an example of an option set in use:

@{[f]author\tBy\>\n}

Notice that there are no spaces added, and that a '[' cannot, therefore, be the first character of a field name, unless you also specify an option set. The following specification is legal:

@{[][funnyfield]}

This specifies an empty option set and a field [funnyfield]. The empty option set keep the parser from thinking that the first character of the fieldname begins an option set. Empty option sets have no other effect.

Here is a list of the available options:

f,v Force FLOW or VERBATIM mode.

In Flow mode, the presence of a newline in the database record is treated as a blank space would, and only a \n sequence is regarded as a real newline (paragraph break.) In Verbatim mode, a newline in the database record is treated the same as a forced paragraph break.

0,1 Paragraph Spacing

Forces either 1 or no extra lines to be inserted between forced paragraphs. By default, '1' is used for the description field, which is also handled in Flow mode.

|,> Paragraph Indentation

Disables or forces paragraph-level indentation on a field. Indentation is never turned on, but may be desirable for the description field, for example, especially if no extra lines are inserted between paragraphs (see 0,1 above.)

You could alter the display of the description field by adding an option set [0>] (or [>0]) to it (the order of the option letters does not matter.) Try it, you might like it!

The following fields are usually handled in Verbatim mode: reference, address, email, docs, run, stored-in, and author.

All other fields are handled in Flow mode.

---

[1] Instead of specifying a custom display format (which is always associated with a database) you may store alterations to the option flags for any or all fields in a database's .kfdb file. This method is very slightly slower than specifying a custom display format with embedded option flags but will save you the trouble of creating a new display format when all you want is alter one or two field's behavior slightly. See the .kfdb format for more information. RexxFisher cannot make use of this feature and ignores these settings in the .kfdb file.

1.154 List of Fields

For descriptions on format and purpose of available fields, please refer to the Fields Description of the Product-Info Specification v6 from Fred Fish. The following fields are defined:

name

fullname

type

short

description

version

date

author

restrictions

requirements

reference

distribution

price

address

email

exectype

installsize

source

construction

tested

run

docs

described-by

submittal

stored-in

1.155 Examples of Custom Formats

Example1:

@{name}\n

\>@{description}@{author| Author: }\n

@{version|\>This is version |\n}

Output:

MonkeyCommand

Lure the lovestruck monster ape back to

his island. Tools include Fay Wray’s torn
nightgown, a Fokker airplane (you get to
pilot it), a compass and a map. Author:
KingKong Industries
This is version 1.0
Example2:
@{name}|@{version}|@{author}|@{price}|@{short}\n
Output:
MonkeyCommand|1.0|KingKong Industries|||
Example3:
@{name}\>@{description}
@{version| This is version |.}
@{author| Author: |.}\n\n
Output:
MonkeyCommand Lure the lovestruck monster ape back
to his island. Tools include Fay
Wray’s torn nightgown, a Fokker
airplane (you get to pilot it), a
compass and a map. This is version
1.0. Author: KingKong Industries.
Example4:
PROGRAM NAME:\t@{name}\n
VERSION:\t@{version}\n
AUTHOR:\t@{author}\n
RELEASE DATE:\t@{date}\n
\n
@{[0>]description}\n
\n
\-\n
Output:
PROGRAM NAME: MonkeyCommand
VERSION: 1.0
AUTHOR: KingKong Industries
RELEASE DATE:
Lure the lovestruck monster ape back to his island.
Tools include Fay Wray’s torn nightgown, a Fokker
airplane (you get to pilot it), a compass and a map.

1.156 SEARCH EXPRESSION ERROR: Logical Operator Expected

You failed to provide a legal operator, meaning that KingFisher could not figure out in which way you want to combine two or more expressions.

Valid logical operators are:

& Logical AND (as in "if this AND that is true, then ...")

| Logical OR (as in "if either this OR that is true, then ...")

ˆ Exclusive OR (as in "if either this OR that is true, but NOT BOTH, then ...")

A unary operator may be used to reverse the value of an expression:

! Logical NOT. You may use the ~ character instead; your choice.

1.157 SEARCH EXPRESSION ERROR: Comparison Operator Expected

You failed to provide a comparison operator, meaning that KingFisher could not figure out in which way you wish to apply a value to a field.

Valid comparison operators are:

= Equality (as in "if the field contains exactly this value, then ...") You can also use two equal signs, which is what the C programming language uses for equality tests. The choice is yours.

ex: name = 'kingfisher'

>= Alphanumerically greater than or equal.

ex: version >= '2.0'

<= Alphanumerically less than or equal.

ex: date <= '1994.08.31'

> Alphanumerically greater than.

< Alphanumerically less than.

<> Not equal. This is the exact opposite of the '=' operator. You may use != instead, which is what the C programming language uses to test for inequality.

$ You should read this symbol as "contains the substring" so that the expression

ex: name$fish

This reads "If the name field contains the substring 'fish' then ..."

1.158 SEARCH EXPRESSION ERROR: Invalid comparison operator

The symbol begins, but does not complete a valid comparison operator. Please note that the following are NOT valid:

>< => =< !< !> >$

Click here for a list of valid comparison symbols.

Click here for detailed information about creating a search expression.

1.159 SEARCH EXPRESSION ERROR: Mismatched Parentheses

Each open parentheses must be matched by exactly one closing parentheses. You have either used too many (symbols or too many) symbols. You may have forgotten to enclose in quotes the (or) symbols meant to be part of a string constant.

1.160 SEARCH EXPRESSION ERROR: Field identifier expected

KingFisher requires you to provide the name of a field in the database before you can give it a comparison or logical operator. If you need help constructing a legal search expression, click here.

1.161 SEARCH EXPRESSION ERROR: Unsupported Feature

You have hit upon an as-yet unsupported feature of the expression parser. This feature may become operational in the future to allow you to build more complex expressions with fewer symbols.

1.162 SEARCH EXPRESSION ERROR: Internal Error

An internal error has occurred in the expression parser. Please write down the exact expression (with spaces and all characters trailing) to write to the author of KingFisher with this bug report!

1.163 SEARCH EXPRESSION ERROR: Incomplete Expression

The expression is incomplete. You must provide additional components before KingFisher can process the request.

1.164 Product-Info Specification

A rough framework of the requirements, which eventually became the Product-Info Specification , was originally started by Fred Fish and his brother Richard. Basing my work upon their original efforts, and then relying on their feedback as well as feedback from a number of other people, the Product-Info Specification was born. It is meant to address the following basic issues:

1. A text-only format makes it possible to easily create and modify these files without special software. This makes them portable between computer systems, easily transmitted, and more resilient to minor damage,

2. Aside from defining characteristics for some specifically named fields, the files are freely extensible by users to suit their individual requirements; in addition, the ordering of fields is (almost) completely irrelevant,

Main text of the Product-Info Specification

Fields defined by the Product-Info Specification

Starter file for a .Product-Info file of your own

Common errors when creating a Product-Info file

The Product-Info Specification was designed by me (Udo Schuermann) with much feedback from Fred Fish. Fred has repeatedly said that I'm too modest to take the credit for this work. Maybe I am, but without his initiative to get me going and his feedback to help polish it, the Product-Info Specification would not have come to be what it is, either. So, I guess it is a bit of a joint effort.

1.165 2.7.1 Product-Info Specification: Text

Product-Info Specification

Version 8

March 23, 1996

PURPOSE

A ’Product-Info’ should accompany every released product: software, images, animation, sound, etc. The purpose of a ’Product-Info’ is to describe a product in a way that is reasonably complete so that a customer can perform searches on a collection of ’Product-Info’ files (records) with a reasonable expectation of locating desired products or excluding undesired ones. The structure of the data is designed so that a search can be defined with some certainty of its "correctness."

By providing a well-written ’Product-Info’ with your product, you assure yourself of maximum potential exposure to customers who are looking to fill a need that your product might fill.

AUDIENCE

This document is aimed both at the developer of a product who needs to write an effective and complete Product-Info, as well as at the customer who stands a better chance of locating satisfactory products without having to wade through an inordinate amount of irrelevant products, if she understands what a Product-Info offers.

If you are not a developer, you may want to skip down to the EXPORTING AND IMPORTING DATA and the STANDARD FIELDS sections.

COPYRIGHT

AUTHORSHIP

The ’Product-Info’ specification was designed by Udo Schuermann <walrus@wam.umd.edu> with initial design criteria and much feedback offered by Fred Fish <fnf@ninemoons.com> and Richard Fish <rjf@ninemoons.com> Support for ’Product-Info’ files was first offered in the form of my product "KingFisher Release 2", which is available on Aminet, nearly all of the Fish CD-ROMs for the Amiga, and from the KingFisher home page on the World Wide Web: http://www.wam.umd.edu/~walrus/KingFisher.html

STORAGE

A ’Product-Info’ is a set of information stored in a file with your project. There are three standard filenames which you can use (the quotes are not part of the name):

1. "Product-Info"

2. ".Product-Info" (note the leading period)

3. "project.pi" ("project" would be replaced by your project’s name; example: "KingFisher.pi" or "DiskSalv.pi" or "VT.pi")

The file must conform to the following rules:

1. The file may begin with comment line, but is not required to do so. A comment is either a blank line (one that has no text, no blank spaces, no tabs) or a line beginning with a ``’#’ (hash) symbol. Any combination of these will be ignored at the BEGINNING of the file.

2. The first field (following the optional comments) in the ’Product-Info’ must be "name"; Fields are identified by a Field Identifier and followed by their contents. Field contents are ended when another field is encountered or by the end of file.

3. Field Identifier begins with a "." (dot, period) as the first character on a line, immediately followed by the name of the field. The "name" field is identified by ".name"; The name of the field is separated from the field contents by a white space. White space is a blank, a tab, or a new- line. It is recommended to use a newline, rather than space or tab. A field name cannot contain white space, obviously.

4. The contents of the field may span one or more physical lines in the file. Some fields ignore anything past the first newline; others will ignore newlines and treat them as if they were a normal blank space; other fields require newlines to separate subcomponents from each other. It is the purpose of the file you are currently reading to tell you the meaning of each of the defined fields.

5. A collection of fields (beginning with the "name" field) represents a Product-Info record. A record is terminated by the end of file, by the presence of a blank line containing a ^N (control-N) symbol, or by the presence of another "name" field. Thus, a ’Product-Info’ may store information for more than a single project, although this practice is in general not encouraged. This feature is used, however, for the construction of variant-record-size databases such as used by KingFisher, but these are not a distribution format that go along with a project. That’s the difference.

6. Field names and field contents should use the ISO Latin 1 character set which is the default of the World Wide Web and also the standard of the Amiga computer. The character ^N (ctrl-n) is reserved as a separator between disks in a database of collected records, should this be a desirable feature.

7. If, in the contents of a field, a physical line of text begins with a period (such as ".guide files are provided...") then the period should have a "\" symbol added before the period, as in "\.guide files are provided..." to prevent the line to be interpreted as a new field ("GUIDE") with the contents starting as "files are provided..." In general, it is permitted to "escape" a period in this manner anywhere in a field's contents.

8. A backslash ("\") is represented by placing \ in the contents of a field. This permits special layout control sequences to be introduced through the \ key, much like the "\." suppresses the interpretation of a line-leading period as the introducer for a field name.

9. Field name identifiers are not case sensitive.

FUTURE EXTENSIONS

The 'Product-Info' format was designed with future extensibility in mind. Being a text-only format, it lends itself to alteration with any editor of your preference. Its method of defining fields permits it to be adjusted and expanded to meet future requirements, and it is not tied to any particular implementation to facilitate searching.

RECENT ADDITIONS

The following fields have been added since the widely spread v6 of the 'Product-Info' specification: "aminet-dir", "aminet-file", "url", "keywords", and "execute"; some other fields have suffered some slight modifications: "author", "reference", and "installsize" with the aim to bring the fields in line with original intent. Also added or changed: \> \< \N \# sequences in field contents.

KINGFISHER'S IMPLEMENTATION

I believe it useful to provide the following information specific to KingFisher's implementation of 'Product-Info' support, as the databases and software are widely available on CD-ROM. The Product-Info Specification is in no other way tied to KingFisher.

1. A KingFisher database is described by a .kfdb file, whose (textual) contents describes the filenames containing the actual database records, the files used for indexing these database files, as well as some other, less vital information.

2. database consists of zero or more records spread over one or more files and is indexed by a single (primary) index file. Although KingFisher supports the format of (and is shipped with) the original KingFisher 1 database, it does not support adding such records to an existing database: The new format has a header to identify and distinguish the new format. The header is 8 characters (KF20DATA) plus a newline character. Index files are similarly identified by a KF20INDX header plus other information.

3. Records collected from 'Product-Info' files and stored into a database are stripped of their comments to preserve space.

4. As index files may become corrupted or lost, a special marker symbol is inserted on a line by itself between disks. This permits KingFisher to keep track of the disk on which a given item is stored and recover this information effortlessly if the index needs to be rebuilt. KingFisher 1 used a blank line between 2-line records; KingFisher 2 uses a line with a single ^N (ctrl-n) symbol on it.

5. A record begins with a "name" field and continues until another "name" field is encountered.

6. The efficient storage of variable-size records brings with it the problem of very costly insert and removal algorithms. KingFisher offers only truncation and append methods for this reason.

7. If a text file being imported (appended) to a database does not begin with a "name" field (nor comments) but the first line is less than 30 characters long, KingFisher will try to parse the file looking for a "name" field anyway, even though DATA TRANSPORT MARKERS (see next section, "Exporting and Importing Data") should have been used. This is a non- standard extension on this specification to facilitate support for the old KingFisher 1 database format.

This ends the section on KingFisher's implementation of the 'Product-Info' standard.

EXPORTING AND IMPORTING DATA

To enable effortless and error-free methods for export and import of 'Product-Info' records, it is required to enclose records with DATA TRANSPORT MARKERS. These markers are NOT REQUIRED INSIDE OF 'PRODUCT-INFO' FILES; they are used only when unrelated text (and comments) surrounds a 'Product-Info' file!

Multiple DATA TRANSPORT MARKERS may be used in a single text, so that each 'Product-Info' record is enclosed; or one pair of these markers may enclose multiple, consecutive 'Product-Info' records:

EXAMPLE: The following examples are to be read as complete files between the lines that look like this: ----------

Example 1: A file containing text in addition to the database record.

Hello, Joe. Here is the description of that weird game

I was telling you about. See if you can figure out how

to win this. Good luck!

.BEGIN-FISH-DESCRIPTION

.name

MonkeyCommand

.author

KingKong Industries

.description

Lure the love struck monster ape back to his island.

Tools include Fay Wray's torn nightgown, a Fokker

airplane (you get to pilot it), a compass and a map.

\.png image support lets you paint your own airplane

decals.

.END-FISH-DESCRIPTION

Example 2: A file containing nothing but two database records. Notice that the DATA TRANSPORT MARKERS are omitted in this case, as they are unnecessary. It would not hurt to place them there, however!

.name

MonkeyCommand

.author

KingKong Industries

.description

Lure the love struck monster ape back to his island.

Tools include Fay Wray's torn nightgown, a Fokker

airplane (you get to pilot it), a compass and a map.

\.png image support lets you paint your own airplane

decals.

.name

MonkeyCommand II

.author

KingKong Industries

.description

Keep the captured ape from assaulting the defenses
of the prison that was erected at the conclusion of
MonkeyCommand I. The game consists of coordinating
the actions of four native tribal leaders and their
vassals in repairing the damage done by the enraged
monster ape as it tries to escape and revenge itself
on whoever won the original MonkeyCommand.

Example 3: A file containing two database records interspersed with extraneous text. The records are protected by DATA TRANSPORT MARKERS

Hi Tom,
Remember that monkey game you told my about?
.BEGIN-FISH-DESCRIPTION
.name
MonkeyCommand
.author
KingKong Industries
.description
Lure the love struck monster ape back to his island.
Tools include Fay Wray's torn nightgown, a Fokker
airplane (you get to pilot it), a compass and a map.
\.png image support lets you paint your own airplane
decals.
.END-FISH-DESCRIPTION
Well, seems that one wasn't enough and they released
another one. We'll have to figure out how to finally
beat the first one, it seems, before they let us play
the next. Maybe we can look through the binary to find
that code phrase. Here's the text:
.BEGIN-FISH-DESCRIPTION
.name
MonkeyCommand II
.author
KingKong Industries
.description
Keep the captured ape from assaulting the defenses
of the prison that was erected at the conclusion of
MonkeyCommand I. The game consists of coordinating

the actions of four native tribal leaders and their

vassals in repairing the damage done by the enraged

monster ape as it tries to escape and revenge itself

on whoever won the original MonkeyCommand.

.restriction

You need the secret code from the first MonkeyCommand

which you can only get if you won that game.

.END-FISH-DESCRIPTION

(=:Joe:=)

NOTE: When copying data to the Clipboard, KingFisher supplies these markers for you!

1.166 2.7.2 Product-Info Specification: Fields

STANDARD FIELDS

The following are standard fields. Their meaning is fixed, although sometimes open to interpretation. Nothing forces you to obey these rules, but the closer you adhere to them, the more successful a search will be that relies on these rules.

1. If a field does not apply, or you have no data to supply, leave it out. You may provide a blank field, but this will have the same effect.

2. Pay attention to the FORMAT, which describes if a field is a single line field (ignores 2nd and further lines) or if it will flow its contents (ignoring YOUR newlines) or if it has any other constraints.

3. The only absolutely required field must also be the first field in any 'Product-Info': "name"

==========

.name (absolutely required; must also be the first field)

Purpose: The program's popular name. This is sometimes an

abbreviated version of the "fullname"

Format: 1 line only

Example: KingFisher

Example: HomeBase VI

Example: BLAZEMONGER

Example: AIBB

Example: gcc

.fullname (optional)

Purpose: The full (or complete) name of the program if it

differs from the "name"; omit this field if you

would merely duplicate "name"

Format: 1 line only

Example: Amiga Intuition Based Benchmarks

Example: GNU C Compiler

.short (optional, but recommended)

Purpose: A one-line description, preferably not exceeding
40 characters in length, à la Aminet. A single-
glance to the program's purpose.
Format: 1 line, best not to exceed 40 characters
Example: Software catalog/search/maintenance tool
Example: Full featured C/C++ package; CLI only
Example: 230db sound, mind-reading, killer game
.type
Purpose: Categorize the package; multiple keywords permitted,
but adhere as closely as possible to the list given
at the end of this document. Wild variations will
reduce the value of this field.
Format: One or more lines, but best not to exceed a single
word or two.
Example: database
Example: animation player
Example: animation tool
Example: spreadsheet
Example: communications
Example: display commodity
Example: mouse commodity
.description
Purpose: A full-text description of your package, containing
anything that is NOT ALREADY available through the
other fields (above and below.) The reader should
gain a good understanding what your program can and
cannot do. If you mention other (similar) software,
please add these also to the "reference" field.
Please note that this field FLOWS text and
is not designed for fixed-pitch ASCII graphics or
other flash. If you need to insert a newline, do so
with the "\n" sequence (backslash followed by a
lowercase "n"); newline characters are treated as
blanks. See the section on FORMATTING below!
Format: Free form: Any number of lines, treated as a single
stream of text and formatted according to embedded
formatting symbols (see FORMATTING below.)
Notes: The text should be readable regardless of the font
that is used in its display, and regardless of the

line width available (resizable window vs. printer)
.version
Purpose: The program's version number; please note that this
should follow the standard guidelines for versions,
as obeyed by most (but sadly not all, not even all
system software):
37.1 < 37.17 < 37.39 < 37.100 < 37.170
The following are all vastly different versions:
37.1 37.10 37.100 37.1000
Format: One line only: MAJOR.MINOR
Example: 37.100
Notes: Nothing requires you to maintain your versions this
way, but you should be aware that some software may
make use of this field by searching for a release
that has reached, for example, at least version 2
(perhaps as a requirement that said software has
been maintained by its author beyond some initial
release 1); if your program's version is "v940205"
then this would simply count as version 0 and could
be missed.
.date
Purpose: The program's official release date. NOT the date
when it made it into the database.
Format: year.month.day
Example: 1993.09.27
Example: 1996.01.01
Notes: The format was chosen to make it easily sortable.
Note the use of leading zeros in month and day, and
the complete year (with century.)
.author
Purpose: Any and all authors who have a part in the program.
Format: Any number of lines, each name should be placed on
its own line.
Example: Joe R. User
Tea Rexx
Example: J. Jones
Random Hacker
Bone Head
Notes: Addresses matching these authors should be placed

into the .address field, one after the other, and
in the correct order to produce a 1-to-1 relation.
.address (optional)
Purpose: Describe a full postal address of the author(s),
to be used when it becomes necessary or desirable
to contact the author by snailmail. Do not specify
the author's name, as this already appears in the
"author" field.
Format: Multiple lines. Formatting symbols are not used, as
physical newlines are respected.
Example: 7022 Hanover Parkway, Apt. C2
Greenbelt, MD 20770-2049
USA
.email (optional)
Purpose: Full electronic mail address
Format: Multiple lines, each having a 1-to-1 relation with
the "author" field; more than one email address may
appear on a line, separated by blank space (the
preferred way) or by commas.
Example: walrus@wam.umd.edu
Example: walrus@wam.umd.edu walrus@uga.umd.edu
.url (optional)
Purpose: Universal Resource Locator, usually for ftp sites
or World Wide Web support/home pages.
Format: One or more lines, each on a 1-to-1 relation with
the "author" field. More than one URL (Universal
Resource Locator) may appear on a single line.
Example: http://www.wam.umd.edu/~walrus/KingFisher.html
Example: ftp://nowhere.com/Bogus.png
.keywords (optional)
Purpose: List as few or as many keywords as necessary to sum
up the major features of this package. It is
recommended to keep this entry to 10 keywords or
less, but no definitive limit will be enforced. The
idea with this field is to help construct a memory-
resident quick index for systems that can afford to
maintain a large amount of data in RAM.
Format: One or more lines; keywords or keyphrases separated
by white space (space, tab, or newline.)

Example: fish disks
library maintenance
search expressions
product-info
.restrictions (optional)
Purpose: If your software has any restrictions placed upon
it, list them here, in detail. You should indicate
if your package has been made dysfunctional (such
as would be the case with a demo package.) If the
program will not run on anything less than a 68020
CPU, you should list this here, to.
Format: Free form (see "description")
Example: Demo version has SAVE and PRINT disabled.
Example: Documentation available only in German.
Example: Disables video DMA while playing samples.
Notes: List actual restrictions here, not minimum system
requirements.
.requirements (optional)
Purpose: List minimum requirements for your program. These
should give the reader enough information to
determine if the software is likely to run on his/
her system. Be sure to specify operating system,
(hard)disk requirements, non-standard libraries
(such as MUI) and whether or not some or all of
these required libraries are included in the
distribution.
Format: Free form (see "description")
Example: 68020, 68030, or 68040 CPU; 3M free RAM; 18M disk
space; at least 640×480 display capabilities and
16 colors or better.
Example: Requires WB2.1 (V38)
Example: Requires 1024×768 (or larger) displays.
Example: Works only with 4096-channel, 230db BLAZETHUNDER
audio board.
Example: Requires MUI (Magic User Interface) version 10.
.reference (optional)
Purpose: List the full path to software in some way related
to this package. This may include previous versions
of your package or similar packages. The path is a

volume:path/ or volume:path/archive
Format: 2 lines per reference: the first is the path with
trailing slash (if you do not give the slash then
the name of an archive is assumed); the second line
is the version number.
Example: FishROM-0002:Productivity/Databases/HomeBase VI/
417.0
FishROM-0001:Productivity/Databases/HomeBase VI/
415.12
.distribution (optional)
Purpose: Describe the distribution and ownership status of
this software. Please see below for a list of the
most common (and recommended) terms.
Format: 1 line
Example: shareware
Example: commercial demo
.price (optional)
Purpose: If your package is available for a fee, describe
this price here. it is strongly recommended that
the first currency is expressed in US dollars to
provide a common base unit for prices. If your
package is freely available and has no price
attached, then omit this field.
Format: Free form but best adhere to examples.
Example: $50(US), DM75
.installsize (optional)
Purpose: The minimum and maximum sizes of the package as it
is installed. The minimum size should give an
indication of how much disk space is required for
an absolute minimal installation. If the reader has
less diskspace than that, the program is likely not
to be able to function at all. The maximum is the
absolute highest amount of diskspace that the
package is likely to consume when all portions are
installed (including optional tutorials, demo files,
etc.)
Format: One or more lines; only the first line has a fixed
format, the rest are free-form.
Example: 220K - 2400K

Most of the database files can be kept on floppy
disks, so valuable hard disk space is not wasted.
Example: 18K
Example: 38K - 500K
Lots of documentation and example scripts make up
the bulk of this installation.
Notes: It is recommended that sizes are scaled to kilo-
bytes to maintain a uniform scale, rather than
expressing them in bytes or megabytes. Always add
a size quantifier (K=kilobyte=1024bytes, M=megabyte,
T=terabyte, etc.)
.exectype (optional)
Purpose: Describe the type of executable(s) that make up
your program. Most packages are probably compiled
C, but some may be shell or ARexx scripts, BASIC,
AMOS, or otherwise interpreted. The reason for this
is that some packages are known to cause problems
on some systems, or the customer is looking for
something "better" than interpreted BASIC and would
like to know this before "wasting" time locating,
downloading, and installing the package.
Format: Free form (see "description")
Example: Compiled C
Example: AmigaBASIC
.source (optional)
Purpose: Describe what source code is available as part of
your package. If source is not available, then omit
this field. How large is the source? What compiler,
translator, or interpreter is required? It might be
good to give an idea of the version of the compiler
that is required.
Format: Free form (see "description")
Example: SAS/C 6.56, Manx, DICE source (750K) available for
$15(US); some old SAS/C 5.10b sources (30K) included
free.
Example: Limited C source examples (15K) included
Example: All source plus custom libraries: 12M
.construction
Purpose: Describe the types of languages used to create this

program and the methods used to build the final
executable. If possible, include the compiler
versions and possibly important options, such as
optimization for various CPUs (it is possible to
optimize somewhat for a 68040 without sacrificing
68000 compatibility.)
Format: Free form (see "description")
Example: SAS/C++ 6.56 with speed optimizations weighed for
the 68040 CPU (68040 not required, however.)
Example: AdaEd
Example: Handcrafted assembly
Example: Fortran with self-made compiler.
Example: AMOS
.tested (optional)
Purpose: Give an indication of which configurations have
served as test environments. If the software
operates without problems with non-standard system
enhancements, this would be a good place to mention
them.
Format: Free form (see "description")
Example: A500(512K Chip, 0K Fast, 1 Floppy), A2000(1M Chip,
2M Fast, 40M HD, 1 Floppy); not tested on 68020+
CPUs.
Example: A1000, A500, A600, A2000, A2000/30, A3000, A1200,
A4000/30, A4000/40 with various amounts of Chip
and Fast RAM, with and without MMU or FPU. Found
to be free of Enforcer hits and able to work with
virtual memory products; compatible with Retina,
EGS/Spectrum, and Picasso software. Also tested
under V33 through V40 system software.
Example: CPU: 68000, 68020, 68030, 68040 (68060 unknown);
Kickstart V37, V38, V39, V40; Video/Graphics: PAL,
NTSC, Picasso II (Village Tronic and CyberGraphX),
GVP EGS Spectrum (others unknown); Tested with
system enhancements such as GigaMem 3.12, VMM 3.2,
Enforcer 37.62, Executive 1.30, IPrefs2Fast 0.9b,
OpaqueMove 1.0, KingCON 1.3.
.run (optional)
Purpose: Specifies how to start the program.

Format: visible=type,command
where 'visible' would be what the user would see
and select; 'type' is either 'CLI' or 'WB' (the
system interface) and 'command' is the program to
execute. The 'CLI' version may include parameters,
including I/O redirection, as this string is passed
directly to the System() call.
Example: HomeBase VI=WB,HomeBase VI
HomeBase VI=CLI,ExecuteMe.HB6
HomeBase VI Fixer=CLI,ExecuteMe.HB6Fixer
KingFisher=CLI,KingFisher
Example: FishTub=WB,ExecuteMe
Notes: As this item may be dangerous to the unsuspecting
user, it has not been implemented by KingFisher and
is not likely to be implemented.
.execute (optional)
Purpose: An 'execute' script. This entry has been added by
Fred Fish; the script views the documentation,
installs the software, or merely runs the program.
Format: A shell script, line by line.
Notes: It is not recommended that this entry be supplied
by you, as Fred Fish may simply replace it with a
version of his own.
.docs (optional)
Purpose: List all documentation files associated with your
package. Do not specify the files if they are not
available as-is; if they are only located within an
archive, omit them.
Format: 1 line per file, do not include the path as this
is provided by the "stored-in" field.
Example: HomeBase.guide
HomeBase.dvi
HomeBase.doc
.described-by (optional)
Purpose: Who created the description ('Product-Info' file)
for this project.
Format: Free form (should include an electronic mail
address too, if available.)
Example: Fred Fish <fnf@amigalib.com>

Example: Udo Schuermann <walrus@wam.umd.edu>
.submittal (optional)
Purpose: Who submitted to package to Fred (or else how the
package came to be on the reference disk.)
Format: Free form (usually one line)
Example: Submitted on disk directly by the author.
Example: Downloaded from wuarchive.wustl.edu in pub/aminet/util/misc
.stored-in
Purpose: Specifies where and especially HOW the package is
stored. This field should specify EITHER the name
of a directory (ending with a ':' or a '/') OR the
name of a file. In the case of an archive, the name
should reflect that with the appropriate extension
(.lha .zip .gz .Z etc.)
Format: One or more lines
Example: FF1000:d501-1000/Disk950/Enforcer/
FF1000:d501-1000/Disk950/Enforcer/Enforcer
FF1000:BBS/d501-1000/Disk950/Enforcer/Enforcer.lha
Notes: This field is usually generated by the disk creator
software, not by the submitter of the package, as
the final location on the disk may not be controlled
by the submitter of the Product-Info.
.path (obsolete)
.aminet-dir (optional)
Purpose: The path (WITHOUT trailing '/') where this package
is available on the global Aminet.
Format: One line; must NOT end with a '/'
Example: biz/dbase
Example: gfx/edit
Notes: Together with aminet-file, this entry can be used
to construct the complete path to this package on
Aminet or Aminet CD-ROMs.
.aminet-file (optional)
Purpose: The name of the package (archive) as stored on
Aminet.
Example: KingFisher220.lha
Example: VanGogh.lha
Notes: Together with aminet-dir, this entry can be used
to construct the complete path to this package on
Aminet.
==========

1.167 2.7.3 Starter .Product-Info

INSTRUCTIONS: Using MultiView/AmigaGuide's SAVE AS command (menu), write this page to a file. Call it .Product-Info. Fill in what you need based on the description of Fields given in the previous section. Not all fields are required, and some may need special formatting.

Ship the resulting file with your product!

Acceptable names for the file (in increasing order of desirability) are:

.Product-Info

Product-Info

myproject.pi

----------(Delete this line and all text above)----------

.name

Program's Name

.fullname

Long/full name, if any

.type

Type of program (see below)

.short

Short (40 character) description, à la Aminet

.description

Long, possibly quite verbose description

.version

Release.Version

.date

Release date (yyyy.mm.dd)

.author

Author's name

.restrictions

Restrictions (perhaps crippleware info)

.requirements

Special requirements (such as MUI)

.reference

Reference to other related programs, two lines each (1: path, 2:version)

.distribution

Distribution type (see below)

.price

Price (if any)

.address

Author's postal address (not including author's name)

.email

Author’s email address

.url

.exectype

ARexx, shell script, binary, interpreted BASIC, ...

.installsize

How big is this thing, approximately?

.source

Type (language) of source code, if any

.construction

How built? AMOS, SAS/C, DICE, Modula-2, Oberon, Assembler, ...

.tested

Tested on what type of systems

.run

See above

.docs

Filenames of documentation

.aminet-dir

/biz/patch

.aminet-file

KingFisher223.lha

.keywords

fish

product-info

aquarium newaqua fishcat

walrus

.described-by

Who wrote this description?

----------(Delete this line and all text below)----------

Suggested keywords for the TYPE field:

Action Game Animation Animation Player

Animation Tool Archiver CLI Tool

Communications Compiler Compression

Database Disk Tool Display Commodity

Drawing Image Conversion Image Processing

Library Mouse Commodity Music Composition

OS Utility Painting Picture

Printing Sound Analysis Sound Editing

Sound Playing Spreadsheet Strategy Game

Text Text Editing Text Viewer
Thinking Game Word Processing Workbench Tool
Keywords for the DISTRIBUTION field:
Commercial Commercial software is owned and distributed
through licenses. It costs money to individual
end-users and is not freely distributable.
SUCH PIECES SHOULD NOT APPEAR ON DISKS THAT ARE
MEANT FOR FREELY DISTRIBUTABLE SOFTWARE!
Commercial Demo Represents a demonstration of a commercial
package. As such, commercial demos are freely
distributable and may have limitations as
described in the .limitations field.
Shareware Such software is owned and copyrights are
held by the author(s). The software may be
distributed freely, but not sold for profit,
of course. Shareware often specifies a limit
of some time after which you are requested or
required to register the software (i.e. pay
for it.) This provides you with the means to
evaluate the software thoroughly before paying
for it.
Freeware Such software is owned and copyrights are
held by the author(s). The software may be
distributed freely, but not sold for profit,
which would mean the software is no longer
FREEware. No payments are required for such
software.
Public Domain Software labeled PD (Public Domain) belongs to
the public, i.e. ANYONE. Some people release
their software into the public domain with the
mistaken idea that they can continue to own
and control the program. Not so. Software
that is labeled Public Domain (or said by the
author to be released into the public domain)
truly belongs to anyone and everyone. It is
quite legal for someone to take such a program
and sell it for profit as is. Likewise, it
it perfectly acceptable to modify public domain
software to build a better product (or whatever)

out of it and then sell it for profit.

GNU Public License The terms and conditions of this license

are long and not easily reproduced here. Suffice

to say that software released under the GNU Public

License cannot be sold for profit and must be

distributed with source code. They are not

public domain, however.

1.168 Common Errors with Product-Info files

By far the most common error made when creating a Product-Info file is formatting the .description field (and some others) as if the text was always displayed...

1. with a fixed-pitch font, instead of a proportional font,

2. with a physical end-of-line characters preserved as newlines

BOTH OF THESE ASSUMPTIONS ARE WRONG AND WILL LEAD TO BADLY FORMATTED TEXT!!

Unless specifically stated in the Field Definitions, a field's contents flow without regard for your newline characters, unless you specify a newline as a \n symbol where you want KingFisher to begin a new line. In fact, newlines, tabs, and spaces are all treated as blank characters, and multiple such blanks are treated as no more than a single blank!

If you require a portion of a field's data to be treated exactly as you specify it, you must force verbatim mode on with the \# symbol. Do not forget to turn back to flow mode with another \#. Verbatim mode overrides the flow mode feature of filling text within the margins, although word wrapping will still be performed. If the display window is too narrw for your verbatim text, the result will be a mess. Likewise, if you format your text for a fixed pitch font, you will likely produce a mess in anyone's window who is using a proportional font! Tabs and indents may help in that case, but overriding flow mode with \# is recommended only as a last resort.

To proof-read the appearance of your Product-Info files, use the VIEW/Product-Info file menu command. Resize KingFisher's window when you view your file to assure that your creation does not rely on a specific font and window width!

1.169 8 TROUBLE SHOOTING

If you experience trouble of any kind with KingFisher, please write me (by postal or electronic mail) and I'll try to help. Your suggestions will make their way into this section or the general documentation as I learn more about what to expect and what doesn't work.

If you find that KingFisher is deficient in some way that prevents you from getting something done, or you find yourself frustrated and wish an easier way existed, or you come across a serious problem with KingFisher, please attempt to resolve the problem with me first before crying out in public. A reputation for quality is difficult to build and easy to lose. I have made every effort to assure that KingFisher Release 2 is as stable and bugfree as can be expected of a product as complex as this on. If you experience a problem, allow me the chance to help!

8.1 Making a new (CD-ROM) database available to KingFisher

8.2 KingFisher is losing memory!

8.3 KingFisher 1.40 was so much easier to use!

8.4 The search window pops up for only a second

8.5 Why don't my search expressions work?

8.6 KingFisher makes a mess of my nifty ASCII graphics!

1.170 8.2 KingFisher is losing memory!

I am aware of the problem but have only been able to determine that it is the KFServer that is losing about 20K of memory, apparently through uncontrolled creation of an additional console task that never gets removed. The solution to the problem is still eluding me, as it is not my code that seems to be creating this console task... :(

If you are in the habit of starting and stopping KingFisher frequently, or you are running a BBS, you may wish to change the value of "keep-running" (in KFServer.prefs) to "yes" to prevent the server from shutting down when the last client detaches. You can always shutdown the server with a CTRL_D or CTRL_C signal (see also the AmigaDOS 'break' command.)

1.171 8.3 KingFisher 1.40 was so much easier to use!

Only a few concepts have actually changed. If you pretend to be color-blind for a moment, then the interface isn't actually all that much different from the old version:

1. Instead of 6 search expression gadgets you now have only one. But the last 20 entries[1] are remembered and can be recalled by clicking on the gadget that is hoped to resemble a scroll. The list is sorted in order of the most recently used.

2. Search expressions now come in two flavors: the original KingFisher 1.40 expressions and the new KingFisher Release 2 expressions, which allow you to limit their search scope to specific fields. You can use the old format by making sure the "PREFERENCES/Searching/Simple substrings" option is checked. I strongly urge you to familiarize yourself with the Search Expressions as they represent a core concept for KingFisher. The more comfortable you make yourself with these, the more effective your searches will become!

3. Instead of activating a search gadget with the yellow checkmarks, you get a dedicated set of search gadgets (they have the big question marks (?) on them.)

---

[1] This value (MaxHistory) is adjustable in the KingFisher2.prefs file.

1.172 8.4 The search window pops up far too briefly

Most likely you have forgotten to turn off the use of Search Masks. When this option is selected, KingFisher ignores all records that do not match the selected Search Mask. If no records match (such is always the case when none of the records in the database have any flags set) then the search will be over almost as quickly as you can blink your eyes!

Use of the Search Mask will do you no good unless at least some of the records in your database have one or more flags set.

1.173 8.5 Why don't my search expressions work?

KingFisher Release 2 uses a new expression syntax, which is an extension of the one used by KingFisher 1.40 and earlier. The extension involves having to specify to which field a value is to be applied. This gives you the ability to search for "all database programs released in 1993 or later (provided a date is available)":

type$database & (date>=1993.01.01 | date="")

You can force KingFisher to drop back to the original expression syntax by selecting the Simple substrings option from the PREFERENCES/Searching menu but that will effectively search all database fields, thereby eliminating KingFisher's ability to make use of the QuickIndex for ultra fast searches.

You may also place *$ before every keyword, as in:

* $ keyword

to look for keyword in all fields. This closely mimics what KingFisher 1.x used to do.

You are strongly urged to carefully work through the Search Expression Tutorial. The concepts presented there will help you search for information more effectively. Search Expressions are a tool. Without knowing what a tool can do, you are not making the most of what you have in front of you.

1.174 The KFServer.prefs File

For the KFServer to successfully start, it must be able to read its .prefs file. This file is named "KFServer.prefs" and must contain at least the following information. All blank lines or line beginning with a hash (#) mark are considered comments:

default-database=1000Fish.kfdb

This line specifies the so-called "Default Database" which is the database KFServer will always open. Any client connecting to KFServer will have this database made the initial database until it selects a different one. In the case of KingFisher, you may not realize this happening, because KingFisher remembers the last used database and automatically switches to that before displaying the first record.

Versions of the KFServer before 2.20 would exit if the Default Database could not be opened. Starting with 2.20, the server will operate but clients have to make a valid selection from the available databases, else most calls to the server will have very little effect.

Notice that the filename, 1000Fish.kfdb, has a .kfdb extension. It stands for "KingFisher Database." The contents of a .kfdb file will be described below. First, let's examine what optional items you can place into the KFServer.prefs file:

maxclients=5

This line specifies that KFServer will not allow more than 5 simultaneous clients to connect at one time. This value must be at least 1, and cannot exceed KFServer's maximum. Unregistered versions of KingFisher have a limit of 2. Registered versions have a limit which you will never be able to exceed unless you have too much time on your hands and you are ridiculously rich and can afford 256MB of RAM for your Amiga to run hundreds of millions of copies of KingFisher.

verbosity=MUTE

The verbosity value specifies how talkative you want the KFServer to be. Ordinarily you will want to set this value to MUTE to make the KFServer shut up as much as possible. Only real problems will be reported, things you should be aware of (like a database being unavailable.) If you find that something is not working, you might want to try a higher verbosity value, until you either can no longer stand the amount of output or you find the problem. The following values are available, and you can specify them in upper, lower, or mixed case:

MUTE

Cries out in only terribly critical situations

TERSE

Hardly sends any messages to the output window

QUIET

Sends occasional messages of interest to the output window

CHATTY

Rather talkative with lots of status information

DEBUG

Produces a nearly continuous stream of information

priority=default

A value between -128 and 127 will set the server's priority. It is strongly suggested that this value not be higher than 4 so as not to affect important system processes. A value of 1..4 will give the server increased amounts of CPU time (compared to other tasks), whereas a value less than 0 will give other processes more CPU time, thereby lowering the amount of work that the server is likely to get done unless no higher priority process is in need of CPU time.

loadindex=DefaultInRAM

Determines how a database main index is handled. In particular, it can override the INDEX entry in .kfdb files:

AlwaysInRAM Regardless of what .kfdb files specify, the main index is always loaded into memory for maximum speed.

AlwaysOnDisk Regardless of what .kfdb files specify, the main index is always kept on disk for minimal use of RAM.

DefaultInRAM If a .kfdb file fails to specify an INDEX entry, the index is then loaded into RAM. This is the default.

DefaultOnDisk If a .kfdb file fails to specify an INDEX entry, the index is then maintained on disk.

sanity-check=InRAM

Controls the server's sanity check operation, which is performed whenever a database is first opened.

InRAM Performs the operation only on an index that is loaded into memory (default)

OnDisk Performs the operation only on an index that is maintained on disk (this can be rather slow!)

Always Performs the operation on every index, no matter if it is an InRAM or OnDisk index.

keep-running=yes

By default, KFServer will automatically exit when the last client program detaches, requiring to be started again if another client then wants to use the KFServer. By setting the keep-running value to "yes" (instead of "no", or omitting it altogether) the KFServer will remain running even after no more clients seem to need its services. This behavior is best suited for situations where clients start and quit frequently, such as with a BBS.

NOTE: Earlier releases of KFServer kept running unless keep-running was set to a value of "no." This behavior has been altered for less confusing single-user operation.

window=CON://640/480/KingFisher Release 2 Server Messages/AUTO

The output file to which KFServer writes all error messages should be set to a console window (such as given above) rather than a file, although reason could certainly be found where a file would be desirable. KFServer does not care where you send output, so long as you specify a valid file.

suppress-requesters=yes

By default, KFServer will not produce requesters asking the user to insert a required disk. Instead, the server returns the database selection request with an explanation added, thereby allowing the client to retry the operation based on possible user-feedback. This prevents the server from blocking when serving multiple users, which could cause serious problems in a multi-user environment. It is not recommended to turn requester suppression off.

1.175 The Database Descriptor (.kfdb) Files

The KingFisher Database file must have an extension of .kfdb, otherwise KingFisher will not be able to list them to a client, should the client wish to know what databases are available. Furthermore, the .kfdb files must be located in the same directory where KFServer is started. The following are absolutely required in all .kfdb files. All blank lines and those beginning with a hash (#) mark are considered comments:

database-name=1000 Fish Disks

Specifies the descriptive name of the database. This is the name presented by KingFisher to the user when using the Open Database command. Keeping this name relatively short is a virtue. The example is about as long as you would ordinarily want to make it.

section=00000,02500,MyFish:Fish01.data

section=02501,05000,MyFish:Fish02.data

One (or more) sections must be specified. Unlike the original KingFisher, which used a format strikingly similar, the two numeric values (0 and 2500, as well as 2501 and 5000 in our two examples) are not disk, but record numbers. The above values work for my own copy of the database used by KingFisher 1.40, but it may not work for you. As KingFisher Release 2 ships with a functional database of all 1000 Fish Disks, I do not expect this to be an issue.

The three portions of each section value are:

beginning record

The first record in the database is 0, not 1. KingFisher always adds 1 to the record numbers because that is how most people view a database.

ending record

The last record in this section of the database.

storage filename

The exact name of the file where you wish to store a portion of the database. If the datafiles are stored in the same directory as the .kfdb file, then a specific path to the files is not required. If, however, the data and index files are located elsewhere (such as on a CD-ROM or floppy disk) then a complete path to the files is essential!

Note, that you can break up the database into as many section as you wish, or keep it all in one contiguous chunk. The organization of a new database is entirely up to you. The CLI tool 'ReOrder' can be used to effectively change these values by copying all records from one .kfdb file to a new .kfdb file with different ordering, then removing the original .kfdb file and all related files.

index-name=MyFish:1000Fish.index

The index-name specifies the name of the main index. This file will be recreated whenever something about the index changes, such as new records are added, or the database is truncated, or you alter any of the flags that are part of each record. KFServer updates this index file on disk whenever the database is closed.

The following are optional values you can place into the .kfdb file to determine how KFServer is to treat this database:

field-index-name=MyFish:1000Fish-Name.index

The name of the database file that will store the QuickIndex. As a convention, you might want to use the value of the index-name entry but add '-name' (for the name field's QuickIndex, see below) before the .index file extension. This permits easy upgrades in future versions when multiple QuickIndexes may become available.

field-index-field=name

Specifies the database field to which the QuickIndex applies. You should always use the 'name' field here because some functions (such as Build VersionLinks...) require a QuickIndex on that field. KingFisher will be able to make use of a QuickIndex on another single-line field for searching, but does not (and will not) support multi-line fields. Use of multi-line fields may cause undefined behavior. Unless you have very good reason to do otherwise, set this entry to name.

index=inram

The current implementation of KFServer loads an index file into memory, whereby the index is said to be "INRAM."

If lack of RAM is a consideration and speed is not as important, you may opt to use an "ONDISK" index. This is especially valuable with large (CD-ROM) databases, whose index may occupy half a megabyte or more.

index-increment=100

This value defines by how many records at a time KFServer should expand an inram index whenever you add records to the database and the index need to grow. It is more efficient to grow the index in large leaps at a time, but can waste memory if, for example you are growing the index in step of 1000 index records, and after 1000 records, you merely add one additional record to the database. KFServer will then have allocated 2000 records but is only using 1001 of them.

Do not be overly concerned about this, however. The initial index size when KFServer opens a database, is always exactly what is needed, no more. Only adding to the database will bring the index-increment value into play.

keep-open=yes

With the exception of the Default Database specified in the KFServer.prefs file, KFServer will always close a database (and all its files) when no client remains that is using it. If, for some reason, you rather have the files, as well a the database index, remain open and loaded, you can set this behavior for each database with this value.

read-only=yes

Marking a database read-only signifies to the client program that certain operations, such as the adding of new records or the alteration of flags, is not permitted. The server may still accept such commands if they are not disabled by the client, but if a database can truly not be modified, because the index is located on read-only or write-protected media, then the server may produce error messages to this effect when a change needs to be written back to disk.

field-format-handling=fieldname=flags

Specifies a different layout for a given field if it is encountered in the database. It is slightly more efficient (faster) to specify a complete Custom Format with embedded options but that may not be as convenient when all you wish is to alter the behavior of one or two fields. Embedded options for a field in a custom format will override that field's specification given in the .kfdb file.

1.176 Operating the Search Result Window

The TAB key toggles between the Search Result Window and KingFisher's main window, providing for easy keyboard use.

The UP and DOWN cursor keys scroll the contents of the Search Result Window in the appropriate direction.

Clicking on any item in the Search Result Window immediately retrieves this item from the database and displays it in KingFisher's main window.

All menu commands of this window apply to the whole set. Printing, Exporting, Applying a mask, and copying to the clipboard will perform these operations on all fish (records) in the Search Result Window, not just on the current fish (record) as is the case when these commands are used with KingFisher's main window active.

1.177 Amiga Technologies GmbH

Amiga Technologies GmbH is owned 100% by ESCOM AG, which bought all intellectual property, including the Amiga, in April 1995, one year after Commodore International had begun voluntary liquidation proceedings. ESCOM's surprise appearance permitted them to purchase the Amiga at the incredible bargain price of only $12 million dollars.

First Accomplishments!

Within six months of ESCOM's purchase of the Amiga, ESCOM AG created Amiga Technologies GmbH, which managed to work out agreements with other manufacturers and production facilities to revive the Amiga. The first new Amigas came off the assembly line in September 1995 and by now (December) some 15'000+ Amigas (1200s and 4000s) have found their way not merely into stores, but actually into eager customer's hands!

The Amiga's Future

By March 1996 Amiga Technologies introduced Project Walker, a 40MHz 68030 Amiga with upgraded I/O chips, and an innovative multi-purpose bus interface that can accommodate video and accellerator cards, bus extenders and bus converters to allow PCI and Zorro-III cards to be used in the Amiga. The highly unusual form and the extensible chassis of this new prototype Amiga have sparked a great deal of controversy. Based on that assessment alone, it looks like the Amiga has a future again. :-)

Other plans for mid to late 1996 include a PowerPC 604 board from Phase 5 for Amiga 3000, 1200, and 4000 models (not for 2000s? <sniff!>), so that most current models can be upgraded!

Early 1997 we should see a native PowerAmiga with PPC 604 CPU (perhaps running at 133MHz?) with AmigaDOS ported to run natively(!) and most applications running, for the time, on a 68K emulator for approximately 68040 performance[1].

Amiga Technologies on the Internet

If you have access to the World Wide Web, try http://www.amiga.de/

AMIGA - Back for the Future!

NEWS FLASH

The parent company of Amiga Technology, ESCOM AG, is (as of mid- to late-April 1996) in the process of selling Amiga Technology GmbH to the Chicago, USA based company VIScorp (http://www.vistv.com/); preliminary reports about VIScorp's intentions are mixed: official statements seem to concentrate on the Amiga's chipset in VIScorp's set-top box; statements from Carl Sassenrath at VIScorp (he wrote the Amiga's multitasking Exec!) claim that Amiga-lovers have nothing to fear, as the Amiga (under ownership of VIScorp and the direction of Amiga Technology GmbH) will continue to be produced, and research and development will continue (i.e. Walker, PowerAmiga, etc.) Reason, of course, says that VIScorp needs more than just the custom chips for its set-top boxes. Alas, with the likes of Irving Gould and Medhi Ali, reasonable expectations have been proven folly, and we've all been burned too badly to ever trust words more than actions... we need action now: We need Walker with PPC 604; we need AmigaOS 4.0; we need the future now, not promises. Action!!

---

[1] I will buy a PowerAmiga (or Walker + PPC card) when it comes out, as my 2000 is probably not upgradable with a PPC 604 Card (but this is the price of progress.) KingFisher will be ported to the PowerAmiga as soon as I get my hands on it. Both the 68K and the PPC versions of KingFisher will continue to be supported.

1.178 9 Thanks

I would like to extend my thanks to the following people whose feedback (suggestions, criticism, requests) and support have helped grind the rough edges of KingFisher, and by that have helped make the program a more polished product than it otherwise would have been:

AMIGA

"Baby, you were born to run!"

Dan Barrett

My favorite Amiga humorist (and discriminating beta-tester, too!) BLAZE on, Dan!

Olaf "Olsen" Barthel

For the code that I had hoped to use for recoloring an image to match a rastport's existing colors.

Fred Fish

For his many years of service to the Amiga community, and especially for the collection of software that has come to be known as the "Fish Disks," and his recent step up to a CD-ROM distribution which has been one of the reasons I have created KingFisher Release 2. Fred's efforts have set him apart as one of the Amiga Community's most important people.

With his kind permission, a representation of Fred's Fish Logo is used in one of the gadgets. Thanks, Fred!

Dave Haynie, formerly Commodore-Amiga

For his work on my favorite computer and for DiskSalv 3 which has pulled my a** out of a sling when that disconnected organ in my skull failed to shut down the system before lightning knocked out the building's power supply and my file system was badly corrupted.

Jay Miner, "The Father of the Amiga"

For his vision, his dedication, and for everything! IN PACE REQUIESCAT.

Dean Ridgway

For his continued help (beta-)testing KingFisher and helping me find and kill some rather elusive bugs.

Bill Sorenson

For that great walrus! :-)

Uwe "Hoover" Schürkamp

For his past and present help testing KingFisher and for acting as my European registration site. Thanks also for making some rather major corrections to my initial German catalog file, and for being a great friend for all these years. :-)

Mike Schwager

For his past maintenance of a major Fish Disk ftp site and more recently for his input as a beta tester.

Michael Sinz, formerly Commodore-Amiga

For Enforcer 37.xx (a truly wonderful debugging tool!) and his help (during development of KingFisher 1.x) with regard to using SimpleRexx, which is now part of RexxFisher.

Michael B. Smith

For posting code on how to use the ASL Screen Mode Requester, part of which found its way into KingFisher.

Jure Vrhovnik

For his link library routines (with source) to recolor images (see dev/c/SupraLib.lha on Aminet.)

Jochen Wiedmann

For his excellent tool FlexCat and his consistent responsiveness to my suggestions and needs to improve FlexCat.

Big thanks go also to all of my active beta testers, some of whom have really gone out of their way (and are still doing it!) to make sure things are working well:

Dan Barrett,

George Gibeau,

Jeff Hanna,

Richard Hartmann,

Dean Ridgway,

Nick Ridley,

Mike Schwager,

Jim Ventola.

I also extend my gratitude to the efforts of my Translators by whose efforts you can (hopefully) enjoy KingFisher in your native language.

And last, but not least, my thanks to you who have registered KingFisher and shown your very real support for my efforts. You know who you are. Cheers!

1.179 Design Limitations

To assure you that KingFisher is not likely to encounter its design limitations, here are these for your scrutiny:

Maximum Record Size: 4 Gigabytes

Maximum Database Size: 4 Gigabytes

Maximum Disk Count: 65535

Maximum Fish Count: 2+ Billion

Maximum Database Segments: Limited only by RAM

1.180 Aminet

Aminet is a global network of ftp sites (generally large Unix systems, but many BBSs also participate) where many gigabytes[1] of Amiga related files are stored. The network's root node is wuarchive.wustl.edu, and all Aminet sites mirror each other's contents, meaning that you can usually connect to the physically nearest (and by that, usually the fastest) site.

Aminet's main administrator is Urban Müller.

In order to access Aminet, you typically need access to the Internet, either through an ISP (Internet Service Provider) or through a company's or university's internet access. As some BBS systems mirror Aminet, you may be able to access Aminet's files even without an internet link.

---

[1] A gigabyte is a huge amount of data!

1.181 Index

This index hopes to have captured the majority of important terms and where they are used.

Amiga Technologies GmbH

Amiga Technologies

Aminet

Aminet, KF History, Support, Keyfile

Components of KingFisher, GUI Image

Images, external

External images

Index

What is KingFisher?

Introduction

Introduction

Keyfile

Why Register?, Keyfile

KEYPATH (environment variable)

Keyfile

.kfdb

See "Database Descriptor Files"

KingCON

PROJECT/Status

KFServer

Components of KingFisher, How does KingFisher work?, Developer Kit, Client-Server

Mask, Search

EDIT/Edit Masks, Button Definition, Buttons, PREFERENCES/SEARCHING/Use search masks, HELP/Using KingFisher

Menus

See "GUI (Graphic User Interface)"

Negation

Search Expression Tutorial

Notification

Features, Custom Formats, PREFERENCES/DISPLAY/Load custom display format, PREFERENCES/PRINTING/Load custom print format, PREFERENCES/EXPORTING/Load custom export format

Painting Buttons

See "Images"

Paying

See "Registering"

Preferences

KingFisher2.prefs, KFServer.prefs

Product-Info

Specification, Proofing / Preview, Why 'KingFisher'?!?, Append fish from file, Append fish from tree, Build Version Links , PREFERENCES/PRINTING/Add index info, PREFERENCES/EXPORTING/Add index info, List of Fields

Programming

Why Register?

QuickIndex

What is KingFisher?