KingFisher

COLLABORATORS			
	TITLE : KingFisher		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		September 19, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

KingFisher

1.1 KingFisher 2.0

```
                 KingFisher 2.0                                      July ↩
                    10, 1994
  Copyright © 1992,1993,1994 Udo Schuermann     6000 42nd Avenue, Apt. 405
  All rights reserved                        Hyattsville, MD 20781-1518 (USA)
  Shareware, $20 (US)                               email: walrus@wam.umd.edu
```

1.2 1 INTRODUCTION

KingFisher 2.0 is a special purpose database tool designed to ↩ for storing
and retrieving information about software. It is fully compatible with Fred Fish’s Product-Info Specification v6 (which I helped design) which means that, ideally, information in the database is broken up into a large number of distinct fields, each of which has a specific meaning and can be examined individually. The benefit to you is greater flexibility: you can search for software by a specific author, for programs that have at least reached version 2, or software that is not commercial or been released after a certain date. Furthermore, how information is formatted can also be specified on a field-by-field basis, providing you with the means to produce custom databases for other applications.

An ARexx interface with the same formatting capabilities and same advanced search capabilities provides a perfect interface between the databases and any application wishing to retrieve information from them.

KingFisher 2.0 is a complete revision of the original KingFisher. As such, it is no longer aimed at Fred Fish’s AmigaLibDisks ("Fish Disks") only, but will serve to index the Aminet as well as your club’s software collection. The original KingFisher was, above all, a single user’s tool for a single database. In a multi-user environment it fared less than well, and storing all but "fish" in its database was not an excercise for the non-technically minded.

KingFisher 2.0 seeks to provide you with all the power of the original, and build upon this power to provide you with multiple user interfaces, serve you with multiple databases, and allow an unlimited number of simultaneous users to search and browse the database.

```
This flexibility is achieved through
                 Client-Server Architecture
                , a method
of isolating the database itself from the  presentation  portion, whereby a
so-called  Server  provides  access  services,  and  one  or  more  Clients
establish connections to the Server and obtain data from  it.   It  is  the
responsibility of the Clients  to  present the data, and it is the Server’s
responsibility to read and write the database.   The Server will allow more
than a single Client to attach to  it, and because the Server can arbitrate
conflicts, a true multiuser environment becomes possible.
```

1.3 1.1 Components

```
                KingFisher is  no  longer a single, self-contained program.  It   ←
                   is now the
collective name for several programs:


                 KFServer
                    The KingFisher Database Server.
        This program is absolutely required!

  KingFisher    A GUI Client based on GadTools.
        This is what you may use the most to access
        the database.

  RexxFisher    A Client based on ARexx.
        This is what you’d use to give one or more
        BBS users access to the database.

  CLIent      A demonstration Client based on the CLI.

  ReOrder     A demonstration Client based on the CLI.
```

1.4 1.2 Distribution rights

```
KingFisher  2.0 is a Shareware  product made available in two versions: one
meant to be distributed in archived form for the price of no more  than  $3
(US) per disk or  equivalent in  foreign currency; the  other is  available
only to registered users and  available only  from the author or authorized
distribution sites.

Distribution of  any portion  of KingFisher 2.0 as part  of  a  software or
hardware product, including CD-ROM, where KingFisher is stored in  ready to
use form,  such as a CD-ROM index/search tool, is  not permitted without  a
prior license agreement with the author of KingFisher 2.0 for such use.
```

1.5 1.3 Registration and Updates

```
                There are two sites to receive your registration fee.  If you live ←
                    anywhere
in Europe, you should use the site in Germany.  Anywhere else in the world,
such as Australia, Japan, or places even further away than that ;-) should
register directly with the author, in the United States.

  United States:
  ~~~~~~~~~~~~~~
  Send $20 (US) in the form of a personal check drawn on a U.S. bank
  or money order or similar form of payment to:

    Udo Schuermann
    6000 42nd Ave. Apt. 405
    Hyattsville, MD  20781-1518
    USA


  Europe:
  ~~~~~~
  Send DM30 in the form of a EuroCheque or check drawn on a German
  bank.  Direct bank transfers (Überweisungen) are also available.

    Uwe Schürkamp
    Jöllenbecker Weg 4
    32051 Herford
    GERMANY


Please do not ask for
                 technical support
                 from any site but the author!

Updates to newer versions are available at the same addresses.  You  should
specify how you want the update to  be processed.  Examples are: the latest
version (we  keep on record the latest version you received from one of the
registration  sites), or the  next  available version within a certain time
frame (say four weeks; I might be close  to releasing a new version shortly
and you might be happy to wait a little longer to get that,  instead of the
soon-to-be outdated one.  We're human beings, we read your  request and  do
our best to listen.

Updates are handled at  the same addresses as above.  In Europe, send DM10;
in the U.S. send $5 (US).  Updates may also  be available electronically on
Aminet (and eventually on BBSs.)  These will be patch files applicable only
to the original and unmodified binaries of specific versions of KingFisher.
Keep your original disk!
```

1.6 1.4 Installation

```
KingFisher  2.0  is best installed  from  the  distribution disk  with  the
standard  Commodore Installer  by double-clicking on the Install-KingFisher
icon.

Should  something go awry,  it is possible to install KingFisher  manually,
```

but this requires some effort and attention to detail. The distribution disk stores most of its files in a compressed format because the complete distribution contains nearly 2MB of data. Manual installation, therefore, requires you to uncompress the files.

1.7 1.5 Technical support

The quickest way to get technical support is through electronic mail. This requires that you have an account with reliable Internet access. In the past, I've tried to help some people and my replies "bounced." I feel bad that my answers never made it back to them, but there really is nothing I can do.

Email: walrus@wam.umd.edu

Postal mail is another way to get in touch with me. I've tried to be good in the past about replying to all letters, especially those that asked for some sort of response, but I confess that I have a problem allocating space on my desk and as a result things have gotten lost.

Udo Schuermann
6000 42nd Ave. Apt. 405
Hyattsville, MD 20781-1518

NOTICE: KingFisher's status as a shareware product means that my first priority for technical support is towards registered users. If you are not a registered user, I will still try to help you, but if your problem is complex and requires too much of my time ... well, you can probably imagine how these things go.

1.8 2 CONCEPTS

1.9 Implementation of Client-Server Architecture

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.

The KFServer creates a message port, a rendevouz, to which clients deposit requests for processing. The KFServer processes these requests one after another (first come, first served) and returns the results to the sender.

And that, my friends, is all there is to it! Quite simple, really. :)

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 a nearly free package^1 to help you get started. This package includes source code, extensive documentation on the KFServer API (Application Programming Interface) and source code to isolate you from the grueling details of setting up messages with parameters for every single request sent to the server.

All code supplied with the developer package is written for and compiles with SAS®/C 6.51.

References: AMIGA ROM Kernel Reference Manual: Libraries. Exec Chapter.
COMMODORE-AMIGA "C" include files: exec/ports.h

^{1} The KingFisher Developer Pack costs the same as an upgrade but you also get the most recent version of KingFisher "thrown in" as a bonus. If you ask for only an upgrade, you get the upgrade. If you ask, instead, for the KingFisher Developer Pack, you'll get both, regardless if the upgrade will actually upgrade your current version. Of course, you could always explain that you're willing to wait some amount of time for the latest release.

The KingFisher Developer Pack is available via email directly from the author at the address walrus@wam.umd.edu

The KingFisher Developer Pack is available by September 1, 1994.

1.10 2.1 Origin and History

There are many terms used by KingFisher and by this documentation which revolve around the concept of those strange dwellers in the water, fish. The reason for this is that some years ago, I believe it was in 1985 to be exact, a fellow named Fred Fish began to collect freely distributable software, put it all on disks and distributed these in a coherent and reliable manner. The concept of "Fish Disks" was born.

Then came a man named B. Lennart Olsson, who created the first widely distributed tool for storing and searching the contents of Fred Fish's disk library. Compared to KingFisher, Aquarium was somewhat primitive, yet it served the Amiga community admirably for a number of years.

Having become the self-appointed keeper of the Aquarium Database, and spending quite some time every month updating the database, the flags that Aquarium needed to find information, and distributing updates to the database so that not everyone had to do the same thing over again, made me come up with the tool that became KingFisher. Its first release, on December 1, 1992 was quickly followed by several more versions until it arrived after seven public updates at version 1.40.

But KingFisher 1.40 still fell short of what I wanted to accomplish. A lot of people wanted features that I couldn't comfortably patch into the code, and it was a hopelessly single-minded system, unable to work with anything but a single database and a single user. Those who are using KingFisher as a BBS support tool will be especially aware of its shortcomings in this area. All this, the feedback from hundreds of users, and Fred Fish's need for an effective database tool to support his move to a CD-ROM distribution led me to design what you have in your hands now: KingFisher 2.0

It serves multiple databases to multiple users in any configuration, fully supports Fred Fish's Product-Info Specification v6 (which I helped design), and offers practically every feature of KingFisher's first release that can be properly supported. The databases can be accessed through an ARexx interface and a resizable and proportional font aware GadTools window that opens on the default public screen.

Example of a crazy session:

Eight users of your BBS are searching for programs with RexxFisher, five are scanning the GoldFish CD-ROM database, two are scanning the latest FreshFish CD-ROM, and another is looking for software on the newest Aminet CD-ROM.

Meanwhile, you decide that you need something to test your .guide documentation file to make sure everything is in order. You start KingFisher 2.0 (the GadTools interface) and select the GoldFish

```
  CD-ROM database and join those five BBS users, all searching ...

  Being a savvy connoisseur of the  Amiga's multitasking, you are not
  content to just sit and stare at  the computer while  it's chugging
  along,  so you start up  another copy  of KingFisher 2.0 and select
  your Amiga Club's database, then add the file of new descriptions.

  And just to get an idea how things are  going, you ask the KFServer
  for a status listing of activities.  At the CLI you type
        kfserver status
  and find out that several of the BBS users have finished and there
  are only two left checking the database ...

  And  just then KingFisher finds you  something  to check unresolved
  links  in an AmigaGuide document  and  you decide  to copy that off
  your GoldFish CD-ROM which  is still mounted with the database that
  one of the BBS users is scanning ...
```

1.11 2.2 Client-Server Architecture

Client-Server architecture is a database concept whereby one ↩ unique portion
of the software, called the server, is responsible for controlling access
to data, while one or more clients{@UB} talk to the 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.

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

KingFisher implements this same powerful 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.

KingFisher's server software is the
KFServer
.

KingFisher itself is "merely" a client that talks to this server.

RexxFisher, also, is a client that talks to the server.

And so is CLIent and ReOrder, two little tools available in source form,
meant more as examples than useful programs.

All of these together form the product named KingFisher.

If you wish to learn a little bit about how the client-server architecture
is implemented in KingFisher, then click
here
.

1.12 2.3 Search Expressions

```
                If you have
                 Simple Substrings
                  selected,  please  note  that  this option
causes an alteration  to the syntax that  the  expression  parser  accepts.
Described  below is the full expression syntax  accepted by the parser when
the Simple Substrings option is not selected.

The smallest expression has the following pattern:

  field  comparison  value

field   is  the name of  any
                 database field
                .  An example  of this
    would be name, description, version, or author.

comparison  is a 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

value   is 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.  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  If either or both of the expressions on the
```

```
    Inclusive OR  left and on  the right side of the operator
        evaluate  to  TRUE,   then   the   combined
        expression also evaluates to TRUE.

  ^ Logical XOR Either one, but NOT both expressions on the
    Exclusive OR  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 '2', then we have found a record that might be interesting.

What does the following mean?

```
    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.

Let's 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, 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, you'll 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.

1.13 2.4 Search Sets

Search sets represent one of KingFisher’s newest, and perhaps ↩
most useful
features. Before searching, you must select whether or not you wish to
make this an interactive or a non-stop search by checking or unchecking the
PREFERENCES/Searching submenu item
Stop on each
. Only when this item is
not checked, will KingFisher produce search sets that you can examine at
your leisure. The value of both choices is discussed there.

Search sets need not be saved to be useful, although you can save yourself
much time if you save the results of oft-repeated searches. Search sets do
not require much diskspace: approximately 5 bytes per record. A thousand
records, which is nearly ¼ of Fred’s 1000 Fish Disks would require about 5K
on disk.

When a search set is loaded, regardless if it is shown in the listview, the
gadget with representations of
Fred’s Fish Logo
will become active. You
can click on this to open and close the Search Set window. Clicking on the
Search Set window’s close gadget will also close the window. Neither of
these actions will destroy the current Search Set!

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

Quitting KingFisher

Beginning another search

Loading a new Search Set

Loading a Search Set will, if necessary, switch to the database to which
the Search Set applies, and will position you at the first record listed in
the Search Set. The current Search Expression is also remembered to remind
you what the Search Set represents.

1.14 2.5 Custom Formats

The format that KingFisher uses to display, print, and export ↩
fish is
programmable! This means that you can customize the display just about any
way you like to! This may require some persistent trial-and-error, but the
results may be worth the effort if your needs are not served by the default
format. Here is what the default format looks like:

```
@{name}@{version| }@{date| (|)}\n
@{short|\t|\n}
@{author|\tBy |\n}
\n
@{description}\n
\n
@{requirements|Requirements:\n\>|\n\n}
@{restrictions|Restrictions:\n\>|\n\n}
@{address|Author’s Address & Email:\n|\n}
```

```
  @{phone|Phone:\>|\n}
  @{fax|Fax:\>|\n}
  @{email|email:\>|\n}
  \n
  @{distribution|Distrib.:\>|\n}
  @{price|Price   :\>|\n\n}
  @{installsize|Installs:\>|\n}
  @{source|Source  :\>|\n\n}
  @{exectype|ExecType:\>|\n}
  @{construction|Constr. :\>|\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}
```

Looks gruesome, doesn't it? I agree, but computers are just so good at making sense of gruesome things, and they're terrible at working with things that we humans have no trouble understanding. This is why the formatting is described by a gruesome mess: KingFisher understands this stuff a lot easier. The bottom line is that it can display things much quicker this way, and in the end that's probably more important than a pretty behind-the-scenes format file.

So, how can we make sense of this gruesome mess?

It's actually less gruesome (nice word, eh?) than you might think. First, you may have already noticed (you're pretty quick, aren't you?) that almost everything begins with the symbols @{ followed by something gruesome 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 gruesome mess between @{ and } symbols can contain more than only a field name. The complete format (without the blank spaces!) is:

```
    @{ field | prefix | suffix }
```

This means that the | symbol is a separator, and the blank space is the value of the prefix portion. But 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 the database 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 do absolutely nothing if the field doesn't.

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. 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 the file is actually considered merely a blank space by KingFisher so that you can break things up into a more readable form.

Here is a listing of the special formatting symbols. They may be used both inside a @{} construct and outside:

\\. 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.

\n A newline, an end-of-paragraph.

\t A tab, which is equivalent to approximately 5 spaces.

\\> A paragraph indent, which allows you to create hanging indentations of text. The indentation will remain in effect throughout the contents of a field, so it is, in effect, a temporary change of margins. Newlines embedded in a field's data will only reset back to approximately the same column as the field's first character.

At the moment, \t and \\> do not quite act the way described. This is not your fault but mine. Things get pretty tricky. Expect this to get fixed soon.

All other text encountered is transferred verbatim to the display.

For descriptions on format and purpose of available fields, please refer to the Product-Info specification from Fred Fish. The following is a list of the fields referenced in the

Product-Info Specification v6

:

```
  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
```

1.15 2.6 KFServer and Databases

Without the KingFisher Database Server, KingFisher is little more than gerupftes Federvieh (a plucked bird, in English, but it sounds much funnier in German :)

KFServer is the all-important portion of the software. Regardless how you access the database, through KingFisher's GadTools GUI, RexxFisher's ARexx interface, through 3rd party client software, or something you wrote yourself, KFServer will always come into play!

Both KingFisher and RexxFisher know how to start the KFServer if it is not already running. KingFisher is, at this time, somewhat better at this bcause it can be told to start KFServer from a directory other than the one in which KingFisher starts itself.

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.

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 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

```
  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 now been altered for more less confusing single-user operation.

```
  window=CON://640/480/KingFisher 2.0 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.

The format of .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. 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 999, 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 2.0 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.
```

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:

```
  index=inram
```

The current implementation of KFServer loads an index file into memory, whereby the index is said to be "INRAM." While an "ONDISK" index has not been tested enough for me to make the claim that this will work, enough of it has been tested that it may actually be usable and even error free.
  Please be advised that an ONDISK index may seem functional, but is not yet officially supported. If you wish to experiment with this option, feel free to do so, but please understand that the results may range anywhere from index-related access errors to a corrupted index file.

```
  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 open 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.

The following entries are, at this time, ignored, but will be used in a future version of KFServer:

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

1.16 2.7 Product-Info Specification

Main text of the Product-Info Specification

Fields defined by the Product-Info Specification

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

1.17 2.7.1 Product-Info Specification: Text

The purpose of Data Transport Markers is to provide explicit delimiters for data that is surrounded by non-database records. The original KingFisher contained a very complex finite state automaton (sic) to extract data out of email and news files. This FSA relied on certain conventions and would fail to work if those conventions were not followed.

In order to enforce a more reliable means to encapsulate and transport data surrounded by irrelevant information, KingFisher 2.0 supports no other format for importing data but that which conforms to the Product-Info specification.

Records must be enclosed by special markers, such as shown in bold in the example below:

```
  .BEGIN-FISH-DESCRIPTION
  .name
  MonkeyCommand
  .author
  KingKong Industries
  .description
  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.
  .path
  FishROM001:games/MonkeyCommand/
  .END-FISH-DESCRIPTION
```

Furthermore, the data enclosed must consist of one or more actual database records, and specification for these requires the first field of every record to be the name field as shown above.

```
KingFisher will read files without Data  Transport Markers, such  as #?.pi,
.Product-Info, or Product-Info files, but no guarantee can and will be made
that it can successfully do so with files that start with data not relevant
to  the  desired information.
  This represents an added flexibility of KingFisher's parser, not an
implied extension to the Product-Info Specification.

According  to this Product-Info  Specification, KingFisher 2.0 will extract
the relevant information from the following sample file (and it does!):

  Hi Tom,

  Remember that monkey game you told my about?

  .BEGIN-FISH-DESCRIPTION
  .name
  MonkeyCommand
  .author
  KingKong Industries
  .description
  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.
  .path
  FishROM001:games/MonkeyCommand/
  .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 breaking through 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 angry beast.
  .restriction
  You need the secret code from the first MonkeyCommand
  which you can only get if you won the game.
  .path
  FishROM002:games/MonkeyCommand2/
  .END-FISH-DESCRIPTION

  (=:Joe:=)
```

1.18 2.7.2 Product-Info Specification: Fields

```
The following are the fields defined by the Product-Info Specification v6
as designed by Fred Fish and Udo Schuermann.

.name
  PURPOSE:  The proram's name
  FORMAT:   1 line only
  EXAMPLE:  KingFisher
  EXAMPLE:  HomeBase VI
  EXAMPLE:  AIBB
  EXAMPLE:  gcc
  ------------------------------------------------------------------
.fullname
  <<<OPTIONAL>>>
  PURPOSE:  The program's full (or complete) name
  FORMAT:   1 line only
  EXAMPLE:  Amiga Intuition Based Benchmarks
  EXAMPLE:  GNU C Compiler
  NOTES:    If the .name is not an abbreviation then omit the
      .fullname.  No sense in giving the name twice!
  ------------------------------------------------------------------
.type
  PURPOSE:  A keyword that describes the nature of the program
  FORMAT:   Preferrably a single word or two.
  EXAMPLE:  Database
  EXAMPLE:  Spreadsheet
  EXAMPLE:  Animation Player
  EXAMPLE:  Animation Tools
  EXAMPLE:  Communications
  EXAMPLE:  Display Commodity
  EXAMPLE:  Mouse Commodity
  NOTES:    Avoid abbreviations.  Refer to the list below for
      suggestions.
  ------------------------------------------------------------------
.short
  <<<OPTIONAL>>>
  PURPOSE:  A one-line description, preferrably not exceeding
      40 characters in length.  This description is to
      give a single-glance insight into the program's
      purpose.
  FORMAT:   1 line only.
  EXAMPLE:  Software catalog/search/maintenance tool, multi-user.
  ------------------------------------------------------------------
.description
  PURPOSE:  A full-text description of your program, containing
      anything that is NOT ALREADY available through the
      other fields (see above and below.)  The reader
      should gain a good understanding what your program
      can and cannot do.  If you mention other programs
      please do not forget to provide a .reference field
      for each such mention.
  FORMAT:   Any number of lines, treated as one line.
      Formatting is permitted, but generally discouraged.
  NOTES:    Do not indent your text if you choose to format
      your text into multiple paragraphs.  Do not use \t
      as a tab.  Leave paragraph formatting to KingFisher.
  ------------------------------------------------------------------
```

```
.version
  PURPOSE:  The program's version number
  FORMAT:   MAJOR.MINOR
      1 line only
  EXAMPLE:  37.100
  NOTES:    Please note that the Commodore guidelines specify
      that the number after the period is NOT a FRACTION
      but rather a WHOLE NUMBER!  Thus, the following is
      a valid progression:
        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
  NOTES:    The format given for this field is really more of a
      SUGGESTION rather than a RULE.  There is no reason
      why you can't store "Today's Version" or "v940205"
      instead of 18.173.  In an ideal world everyone
      would use Commodore guidelines, but there are
      enough exceptions.
  ----------------------------------------------------------------
.date
  PURPOSE:  The program's official release date; not the date
      it made it into the database.
  FORMAT:   year.month.day
      1 line only
  EXAMPLE:  1993.09.27
  NOTES:    The date format is chosen to be easily sortable.
      Note the use of leading zeros in month and day.
      The full year is to be given in anticipation of
      the coming change to a new millenium.
  ----------------------------------------------------------------
.author
  PURPOSE:  Any and all authors who have a part in the program
  FORMAT:   Any number of lines, treated as one line (\n in the
      text will "break up" the line into multiple visual
      lines.)
  EXAMPLE:  Joe R. User, Tea Rexx.
  EXAMPLE:  J. Jones\n
      Random Hacker\n
      B. Clinton
  NOTES:    Addresses should be placed in the .address field.
      There should be only one .address field for each
      .author field.
      If more than 1 .author field is specified, then the
      same number of .address and .email fields must also
      be given in a 1-to-1 relationship (i.e. the 3rd
      .author field must be associated with the 3rd
      .address, and the 3rd .email field.)
      EX: see the example "Joe R. User, Tea Rexx" above;
      Assume that Joe R. User has long vanished and no
      known address, but that Tea Rexx has supported the
      program for a while.  If an .address and/or .email
      field is available for Tea Rexx, then you must
      specify EMPTY .address and/or .email fields for the
      author listed BEFORE the ones for Tea Rexx.
      Likewise, if the two authors names were reversed,
      you would NOT have to specify blank .address and/or
      .email fields for the second author.  I hope that
      makes sense.
  ----------------------------------------------------------------
.restrictions
  PURPOSE:  List restrictions placed upon this program.  These
      should indicate in which way this program has been
      made dysfunctional (for demo purposes), problems
      (bugs) known to exist with this program, or any
      other thing that lets the user know that this
      program, as seen in this distribution, may not
      fully satisfy the user in some form.
  FORMAT:   Free form; see .description for more info.
  EXAMPLE:  Demo version has SAVE and PRINT options disabled.
  EXAMPLE:  The ReadOperatorsMind command fails to work with
      CDTV units.  Incompatible with the Discus Ejector
      utility.
  EXAMPLE:  Crashes if iconified while loading a sample or
      image larger than 64K.
  EXAMPLE:  Requires a PAL display.
  EXAMPLE:  The program is in German but the documentation
      offers translations into English and Swahili on
      a menu-by-menu and gadget-by-gadget basis.
  NOTES:    Do NOT use this field for things like "won't work
      with KS 1.3" or "won't run with less than 2 Megs
      of RAM."
  ----------------------------------------------------------------
.requirements
  PURPOSE:  List requirements for your program.  These should
      give the reader enough information to determine if
      the software will run on his/her system or not.
      Be sure to specify operating system versions,
      (hard)disk space requirements, etc.  If your
      program requires any external libraries that are
      not part of the system software, it would be nice
      to list them here and comment on whether or not
      they are included in the archive.
      If your program is known to run on every existing
      (Amiga) platform, state this in this field!
  FORMAT:   Free form; see .description for more info.
  EXAMPLE:  68020, 68030, or 68040 CPU; 3M free RAM; 18M disk
      space; at least 640x480 display capabilities!
  EXAMPLE:  Requires WB2.1 (V38)
  EXAMPLE:  Requires 1024x768 (or larger) display capability.
  EXAMPLE:  Works only with 4096-channel, 230db BLAZETHUNDER
      Audio board.
  EXAMPLE:  Requires MUI (MagicUserInterface) version 5.
  ----------------------------------------------------------------
.reference
  <<<OPTIONAL>>>
  PURPOSE:  Full path to where this program's files are stored,
      as well as the version that is stored there.
  FORMAT:   2 lines per reference:  the first line specifies
      the full path (with trailing slash) and the second
      line, the version.
  NOTES:    Multiple such fields may be provided to reference
      previous versions of this program, as well as
      other programs that might be of interest.  The
      versions should be listed in reverse chronological
      order and SHOULD include the CURRENT entry.
      Please note that it is VERY VERY VERY important
      that you specify the CORRECT PATH!  Without a
      correct path, this entry will be nearly useless!
      SPECIFY THE PATH WITH A NEW SUBMISSION ONLY IF YOU
      KNOW WHERE IT IS STORED; NEW SUBMISSIONS WILL HAVE
      A PATH ASSIGNED HERE AUTOMATICALLY.  YOU SHOULD
      PROVIDE THE PROPER PATHS TO KNOWN AND EXISTING
      SOFTWARE.
  EXAMPLE:  FishROM-0002:Productivity/Databases/HomeBase VI/
      417.0
      FishROM-0001:Productivity/Databases/HomeBase VI/
      415.12
----------------------------------------------------------------
.distribution           was: .status in rev 1
  <<<OPTIONAL>>>
  PURPOSE:  Describes the distribution and ownership status
      of this software.  Please see below for a list of
      common (and recommended!) terms to use.
  FORMAT:   1 line
  EXAMPLE:  Shareware
  NOTES:    Please see the table below for descriptions of the
      recommended terms.
  ----------------------------------------------------------------
.price
  <<<OPTIONAL>>>
  PURPOSE:  Describes the cost of this program to the user.
  FORMAT:   Any number of lines, treated as one line.
  EXAMPLE:  $50(US), DM75.
  NOTES:    In order to make this field more useful, it is
      STRONGLY recommended that the FIRST currency
      listed is United States Dollars as shown in the
      EXAMPLE above.  This allows a search to be limited
      to a common price base.  If you charge no money
      for this program, omit this field!
  ----------------------------------------------------------------
.address
  <<<OPTIONAL>>>
  PURPOSE:  Describe a full postal address of the author, to
      be used if it becomes necessary or desirable to
      contact the author.  Do not specify the author’s
                        name, as this is already in the .author field.
  FORMAT:   Multiple lines; formatting symbols \n are not
      required, as physical line breaks are equivalent.
  NOTES:    SEE THE .author FIELD FOR IMPORTANT INFORMATION
  ----------------------------------------------------------------
.email
  <<<OPTIONAL>>>
  PURPOSE:        Describe a full electronic mail address.  Make
      sure that this address is complete and reachable
      even from less well-connected sites.  The author
      of KingFisher, for example, can be reached as
      walrus@wam.umd.edu
      It would be an error to specify only "walrus" or
      "walrus@wam" even though these will work within
      the particular organization where this address
      is valid.
  FORMAT:   Multiple lines; formatting symbols \n are not
      required, as physical line breaks are equivalent.
      Do not specify more than one email address per
      line.  The more you abide by RFC-822 specifications
      the better.
  EXAMPLES: walrus@wam.umd.edu (Udo Schuermann)
      Udo Schuermann <walrus@wam.umd.edu>
      "Udo Schuermann" <walrus@wam.umd.edu>
      <walrus@wam.umd.edu> Udo Schuermann
  NOTES:    You may specify multiple electronic mail addresses
      in order of decreasing reliability and permanence,
      each on its own line.
      SEE THE .author FIELD FOR IMPORTANT INFORMATION
  ------------------------------------------------------------------
.exectype
  <<<OPTIONAL>>>
  PURPOSE:  Describe the type of executable(s) that make up
      your program.  Examples:  68xxx, AMOS, Script,
      ARexx, Compiled basic, Amigabasic, etc.
  FORMAT:   Free form; see .description for more information.
  EXAMPLE:  AMOS
  EXAMPLE:  68000, 68020, and 68040.
  EXAMPLE:  Compiled BASIC
  EXAMPLE:  Compiled ARexx
  NOTES:    AMOS-based software has been said to not work on
      some systems at all; this entry allows a user to
      determine if the software is worth obtaining in the
      first place.
  ------------------------------------------------------------------
.installsize
  <<<OPTIONAL>>>
  PURPOSE:  Indicate the minimum and maximum sizes of the
      executable as it is installed.  The minimum size
      should give an indication of how much diskspace
      is required for a minimal installation (perhaps
      lacking help files and miscellaneous tools) while
      the maximum size should indicate the absolutely
      highest amount of diskspace required by the
      program.
  FORMAT:   1 or more lines; Only the first line has a fixed
      format, the rest are free-form.  See examples.
      Always indicate the number scales with a capital
      K (for kilobyte) or M (for megabyte)
  EXAMPLE:  220K - 2M
      Most of the database files can be kept on floppy
      disks, so valuable harddisk space is not wasted.
  EXAMPLE:  18K
  EXAMPLE:  38K - 500K
      Lots of documentation and example scripts make up
      the bulk of the installation.
  ------------------------------------------------------------------
.source
  <<<OPTIONAL>>>
  PURPOSE:  Describe what source code is available with this
      program.  If source code is not available then
      omit this field.  The .construction field often
      helps further identify the type of source if you
      omit details here.  How large is the source?
  FORMAT:   Free form; see .description for more information.
  EXAMPLE:  SAS/C,Manx,DICE source (750K) available for $15
  EXAMPLE:  Oberon source included.  85K
  EXAMPLE:  Limited C source (15K) included.
  EXAMPLE:  All source plus custom libraries, included: 12MB
  ----------------------------------------------------------------
.construction
  <<<OPTIONAL>>>
  PURPOSE:  Describe the type of language(s) used to create
      this program and the methods used to build the
      final executable.  If possible, include the
      compiler version(s) and possibly important
      options, such as optimization.
  FORMAT:   Free form; see .description for more information.
  EXAMPLE:  SAS/C++ 6.5 with full optimization.
  EXAMPLE:  AdaEd.
  EXAMPLE:  Fortran with self-made compiler.
  EXAMPLE:  AMOS
  NOTES:    This is usually closely related to the .exectype
      field but differs from it in that the .exectype
      might be "Compiled C" but the compiler used was
      "RottenC 0.97"
  ----------------------------------------------------------------
.tested
  <<<OPTIONAL>>>
  PURPOSE:  Give an indication of which configurations have
      served as test environments.
  FORMAT:   Free form; see .description for more information.
  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.
  ----------------------------------------------------------------
.run
  <<<OPTIONAL>>>
  PURPOSE:  Specifies how to start the program.
  FORMAT:   visible=type,command
      Where 'type' is either WB or CLI to indicate the
      required startup environment.
  EXAMPLE:  HomeBase VI=WB,HomeBase VI
      HomeBase VI=CLI,ExecuteMe.HB6
      HomeBase VI Fixer=CLI,ExecuteMe.HB6Fixer
  EXAMPLE:  FishTub=WB,ExecuteMe
  NOTES:    KingFisher requires that this entry strictly
      follows the above format.
      The user is shown all text up to the first equal
      sign (the 'visible' portion.)  The 'type' portion
      must be terminated with a comma (,) and following
      it will be the command to be executed.
      Selecting it will either invoke the program from
      the Workbench (invoking it as if double clicked on
      its icon (if the .info file exists), or execute the
      indicated shell command line as if it has been
      typed at an open console window.
  ----------------------------------------------------------------
.docs
  <<<OPTIONAL>>>
        PURPOSE:        List all documentation files, possibly for viewing
                        from within KingFisher for more detailed info.
        FORMAT:         1 line per file
        EXAMPLE:        HomeBase.guide
                        HomeBase.dvi
                        HomeBase.doc
        NOTES:          KingFisher examines the EXTENSION and invokes the
                        appropriate viewing tool: MultiView/AmigaGuide for
                        .guide files, ShowDVI for .dvi files, more for
                        anything else.  These files can also be sent to the
      printer via KingFisher (i.e. print .ps or .doc
      files.)  KingFisher will honor the PAGER
      environment variable (defaults to 'more') to
      display standard text.
  NOTES:    Omit any path to these files, unless it is a
      relative path from within the program's CD-ROM or
      disk directory.  Do not specify these files if
      they are located within archive files; remember:
      the files must exist as they are given here!
  ----------------------------------------------------------------
.described-by
  <<<OPTIONAL>>>
  PURPOSE:  Specifies who created the description (Product-Info
      file) for the program.
  FORMAT:   Free form; should include an electronic mail
      address, too, if available.
  EXAMPLE:  Fred Fish (fnf@fishpond.cygnus.com)
  EXAMPLE:  Udo Schuermann <walrus@wam.umd.edu>
  ----------------------------------------------------------------
.submittal
  <<<OPTIONAL>>>
  PURPOSE:  Identifies who submitted the program to Fred or
      else how this program 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 application
      is stored.  This field should specify EITHER the
      name of a directory (ending with a : or a /) OR the
      name of a file (one that does NOT end with : or /)
  FORMAT:   1 or more lines.
  EXAMPLE:  FF1000:Disks701-1000/Disks941-960/Disk950/Enforcer/
      FF1000:BBS/Disks501-1000/Disks941-960/Disk950/Enforcer.lha
  NOTES:    It is up to the particular application to decide
      how to handle this information.  If the extension
      on the file is .lha, .lzh, .Z, .zoo, .pak, .zip,
      etc. then you could, for example, call upon the
      archiver of choice to unpack the application into a
      temporary directory and let the user run the
      program or list the files, or whatever.
```

1.19 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
.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
.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.
```

1.20 2.8 KingFisher Tooltypes

```
                KingFisher first processes the contents  of the KingFisher2.prefs ↩
                   file  for
which it looks in  the current directory first, then in ENV:KingFisher/ and
last in S:

Once this file has  been  processed, KingFisher  will process command  line
arguments  (if invoked from the CLI) or Icon Tooltypes (if invoked from the
Workbench.)  The format of both  tooltypes and  CLI arguments are the same,
and  can be anything  you  find in  the  KingFisher2.prefs  file  (which is
written each time you quite KingFisher) as well as the following:

SERVERNAME=volume:path/KFServer

  If the
               KFServer
               is not currently running, KingFisher will attempt
  to start it  by  running "KFServer"  in the current  directory.  If
  this is  not how you  have configured your system (the installation
  script set up things this way, so you ordinarily should not have to
  worry about this) then you must specify  the full path and filename
  of the KFServer executable.

  Notice that the supplied script (available from the Workbench, too)
  named KFDown  queries  the standard c:STATUS command for a  process
  with  the name  'KFServer.'  If you  start KFServer with a specific
  path, then KFDown will no longer  be able to shutdown  the KFServer
  as  you  might  expect.   There  are several solutions to  this, if
  KFDown is a tool you expect to make use of (you don't have to):

      1.  Modify  KFDown to  invoke  KFServer  with  a QUIT parameter
```

```
    instead of sending the CLI process a break signal,

      2.  Modify KFDown to query  the  c:STATUS command with the full
    path of KFServer.
```

NOOUTPUT

A flag that tells KingFisher not to print the initial copyright and welcome banner. When invoked from the Workbench this banner causes a console window to open, which may well be undesirable. This option is set by default in the icon supplied with KingFisher.

1.21 3 MENUS

Project

Edit

Search

Preferences

Help

1.22 The Project Menu

Help (index)

About KingFisher

Server status

Open database

Define database

Print

Release printer

Export

Close export file

Quit

1.23 The Edit Menu

```
Append fish from file

Append fish from tree

Delete fish

Edit expression
   Edit search masks
```

1.24 The Search Menu

```
Select expression

Search backward

Search forward

Load search set

Save search set
```

1.25 The Preferences Menu

```
  Global

Auto-save on exit

Confirm quit
  Display

Load custom display format

Drop custom display format
  Printing

Load custom print format

Drop custom print format

One fish per page

Avoid page breaks

Add index info
  Exporting

Load custom export format

Drop custom 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

                 Save Preferences
```

1.26 The Help Menu

```
                 Help (index)

                 Using KingFisher

                 Searching

                 Printing

                 Exporting

                 Databases
```

1.27 PROJECT/About KingFisher

```
                Presents an image of the KingFisher logo as well as  copyright   ↩
                    information
for  the software.  Also given will be the
                 registration site
                 most likely
to apply to you.

The same  window will  always appear when  you  first start KingFisher.  It
will go away by itself only if it is not deactivated.

Language translations:

   Dansk:       Finn Kettner

   Deutsch:     Uwe Schürkamp
   Suomi:       Janne J Kalliola
```

Note: KingFisher 2.0 is currently being translated to additional languages:
      Español, Nederlands, & Svenska.

1.28 PROJECT/Status

Requests from the server some information, which includes an estimate of what percentage of the server's total time your client has taken. If you sit idle, that percentage will decrease. It also specifies which database you have open.

1.29 PROJECT/Open Database

Requests from the server a listof all available databases. This is a list of the descriptions in all files with the extension .kfdb that the server knows about. The server can see these files only in its default directory.

You get to select one of these databases based on the description for the database as stored in the .kfdb files. KingFisher will save the position in your current database and activate the newly selected database, moving to the most recently visited record in that database.

The window that lists you all the available databases becomes far more useful when you have more than one or two databases available to you.

You can cancel the selection by closing the window.

1.30 PROJECT/Define Database

NOTE: This command is not yet available. The following is what you need to
know to setup your own database, manually:

KFServer can only serve databases that are defined by the contents of files with a .kfdb extension. The exact name of this file is immaterial but it is always a good idea to use a sensible name. Let us setup a database for your Amiga Club, using a single file to store all the information, named ClubDisk:Club.data, and an index file for it named ClubDisk:Club.index:

The name of the KingFisher Database file shall be AmigaClub.kfdb

Let us create this file with the following contents. You can use any standard text editor for this task:

```
  database-name=Our Ourstanding Amiga Club's Own Software Collection
  section=00000,99999,ClubDisk:Club.data
  index=inram
  index-increment=100
  index-name=Club.index
```

For more information on these individual items, please see the KFServer section.

1.31 PROJECT/Print

Using the currently active print format (default or custom), KingFisher will print data to the printer. If you print from the main window's menu, KingFisher will print only the current record. If you print from the "Caught Fish" window that displays all matching records in the Search Set, then KingFisher will print all records in the search set.

Notice that printing is configurable with the options of the Printing Preferences menu.

1.32 PROJECT/Release printer

This entry is active only when KingFisher has printed something, after which it will retain "ownership" of the printer device awaiting more print commands. Using the Release printer command returns the printer to the system and tells KingFisher that you are done with printing for the moment.

1.33 PROJECT/Export

Using the currently active export format (default or custom), KingFisher will write data to the export file. If you export from the main window's menu, KingFisher will write only the current record. If you export from the "Caught Fish" window that displays all matching records in the Search Set, then KingFisher will write all records in the search set.

Notice that exporting is configurable with the options of the Exporting Preferences menu.

If exporting is set to Use importable raw format, then neither the default, nor the custom format will be used, and instead KingFisher will write a file that can be re-imported through the Append fish from file command.

1.34 PROJECT/Close export file

This entry is active only after KingFisher has exported something and is keeping the file open and ready for further additions through the Export command. Using the Close export file command closes the file and allows you to access it through other software.

1.35 PROJECT/Quit

The Global Preferences submenu item Confirm quit allows you to specify whether or not you wish KingFisher to ask you 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 you also have the Auto-save on exit option disabled, you must make this change permanent by selecting Save Preferences .

1.36 EDIT/Append fish from file

The file you specify may contain one or more records. The records must conform to the Product-Info Specification v6 All valid records from the given file will be appended to the database. The index is automatically updated (and saved to disk when the database is closed.)

1.37 EDIT/Append fish from tree

Scans a directory tree for #?.pi, .Product-Info, and Product-Info files and adds their contents to the database. 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.)

1.38 EDIT/Delete fish

```
Truncates the database by deleting the current fish  (record) and  all that
follow.  You must confirm the action before it will take place.

That database files themselves are not (at this time) physically altered.
Only the index is altered (and this change made permanent when the database
is closed.)
```

1.39 EDIT/Reconstruct database index

```
This command doesn’t exist yet.
```

1.40 EDIT/Pack database

```
This command doesn’t exist yet.
```

1.41 EDIT/Edit custom format file

```
This command doesn’t exist yet.
```

1.42 EDIT/Edit search expression

```
Edit the current search expression.
```

1.43 EDIT/Edit Masks

```
Oops!  I haven’t typed any text for this node!  :(
```

1.44 EDIT/Copy to clipboard

```
This command doesn’t exist yet.
```

1.45 EDIT/Append from clipboard

```
This command doesn’t exist yet.
```

1.46 EDIT/Choose clipboard

This command doesn’t exist yet.

1.47 EDIT/Clear clipboard

This command doesn’t exist yet.

1.48 SEARCH/Select Expression

If one or more Search Expressions have been used before, you can select one of them to be placed into the Search Expression gadget and used for the next search you begin.

1.49 SEARCH/Search backward

Begins a search in reverse direction. The Stop on each match option determines if the search will stop as soon as a match is found, or if it should continue to build up a Search Set consisting of all fish (records) that match the expression.

You can interrupt a search by closing the Search Status window.

Notice that you can press the "<" key as a short cut for this command.

1.50 SEARCH/Search forward

Begins a search in forward direction. The Stop on each match option determines if the search will stop as soon as a match is found, or if it should continue to build up a Search Set consisting of all fish (records) that match the expression.

You can interrupt a search by closing the Search Status window.

Notice that you can press the ">" key as a short cut for this command.

1.51 SEARCH/Load search set

Loads a new Search Set. Any Search Set that you have currently loaded will be cleared and is lost if it has not been saved.

When a Search Set is loaded, KingFisher may switch to the database to which the search set applies, and will also store the Search Expression to the Expression gadget to give you an idea what the Search Set means.

While loading the Search Set, KingFisher will retrieve some information from the appropriate database to be shown to you in the Search Set Window. This process requires KingFisher to read from the database. Larger Search Sets may not, therefore, seem to load instantly.

If your Search Sets are not given the extension .search on disk, then you must alter the ALS File Requester's Pattern field from the default pattern #?.search to something closer to your needs.

1.52 SEARCH/Save search set

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

If you give you search sets the extension .search, then KingFisher will automatically show you existing Search Sets when you load a Search Set!

1.53 PREFERENCES/Global

Oops! I haven't typed any text for this node! :(

1.54 PREFERENCES/GLOBAL/Auto-save on exit

If you enable this option, then KingFisher will automatically store all settings to the KingFisher2.prefs file in the default directory, or the file named by the SETTINGS tooltype, or the first file it finds while looking in the default directory, then ENV:KingFisher/, and then S:

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

1.55 PREFERENCES/GLOBAL/Confirm quit

Do you hate software that just always asks you if you really want to quit, and you hear yourself mumbling "Of course, I’m sure!"

Do you tend to click on the close gadget and then find yourself saying "oops!" but it’s too late?

Whichever of these questions describes you, with the "Confirm quit" option you can get KingFisher to behave the way you want it to!

1.56 PREFERENCES/Display

Oops! I haven’t typed any text for this node! :(

1.57 PREFERENCES/DISPLAY/Load custom display format

Oops! I haven’t typed any text for this node! :(

1.58 PREFERENCES/DISPLAY/Drop custom display format

Oops! I haven’t typed any text for this node! :(

1.59 PREFERENCES/DISPLAY/Font

This command doesn’t exist yet. :(

1.60 PREFERENCES/DISPLAY/Screen

This command doesn’t exist yet. :(

1.61 PREFERENCES/Printing

Aw, shucks! Nothing here, either! :(

1.62 PREFERENCES/PRINTING/Load custom print format

Oops! I haven’t typed any text for this node! :(

1.63 PREFERENCES/PRINTING/Drop custom print format

```
Oops!  I haven’t typed any text for this node!  :(
```

1.64 PREFERENCES/PRINTING/One fish per page

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

1.65 Nothing

```
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      |
                                       |:::::::::::::::::::::|

NOTE: At this time  this  command  is only  partially  functional  and  not
      guaranteed to always work as predicted.  This will be fixed shortly!
```

1.66 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.

1.67 PREFERENCES/Export

Oops! I haven't typed any text for this node! :(

1.68 PREFERENCES/EXPORTING/Load custom export format

Loads a custom format to be used when writing fish records to an export file. Use the Drop custom export format command to revert to the internal default format.

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

1.69 PREFERENCES/EXPORTING/Drop custom export format

This entry will only be available if you have a custom export format loaded. It will drop the custom format and revert back to the default. Note that the use of the Use importable raw format option overrides the use of custom or default formats entirely.

1.70 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 between sessions. An implicit Close export file will be issued for you.

1.71 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 2.0 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.

1.72 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.

1.73 PREFERENCES/Searching

Oops! I haven't typed any text for this node! :(

1.74 PREFERENCES/SEARCHING/Stop on each

When you begin a search, KingFisher examines this option to see if you wish it to stop immediately whenever it finds a match. If this option is not enabled, KingFisher will build a Search Set instead, presenting you with the final list of all matches, which you can save permanently, and from which you can choose randomly.

1.75 PREFERENCES/SEARCHING/Case sensitive

When this option is enabled, upper and lower case letters are treated as distinct symbols, so that "a" is not the same as "A". 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.

1.76 PREFERENCES/SEARCHING/Trim blanks

When blank spaces are typed into a string gadget, at the end of a string, they are usually quite invisible and difficult to detect. Their presence, however, can produce rather puzzling results because they may end up being considered part of a string constant in your expression!

Enabling this command will guard against such troubles by removing all blank spaces from the end of your expressions. This option will cause you problems if you are looking for a string such as "fred " (i.e. when you really do want a blank space at the end of a string) but the average case may be satisfied better by turning this option on.

1.77 PREFERENCES/SEARCHING/Simple Substrings

If the Simple Substrings option in the Searching Preferences is selected, KingFisher will automatically supply a field and operator selection of "*$" to your search strings, so that the substrings you provide in the style of KingFisher release 1 expressions are treated as substrings and scanned for in every available field of the database records that will be examined during a search.

1.78 PREFERENCES/SEARCHING/Use search masks

Oops! I haven't typed any text for this node! :(

1.79 Nothing

Saves all settings to a file of your choosing. Unless given a ↩ specific
filename with the SETTINGS
tooltype
at startup, KingFisher for the
following files from which to read its settings:

```
     KingFisher2.prefs                    (in the current directory)
     ENV:KingFisher/KingFisher2.prefs
     S:KingFisher2.prefs
```

If it finds one of these files, it will attempt to writes its settings back to this file when you exit (provided the
Auto-save on exit
option is
enabled) or to the first file in that list (i.e. in the default directory when none of these files have been found.

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.

1.80 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.81 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.82 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.83 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.84 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.85 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.86 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.87 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.88 Nothing

```
Oops!  I haven’t typed any text for this node!  :(
```

1.89 Nothing

```
Oops!  I haven't typed any text for this node!  :(
```

1.90 4 GADGETS

```
                The  following is  an  approximate  layout of  KingFisher's   ↩
                   display.   The
buttons below are placed to correspond with gadgets in the display.  Choose
any of  them for  explanations  of their functions,  or press the  HELP key
while the mouse pointer is over the gadget in the real window.
  _____________________________________________________________________
 |                                                                     |
 |
                 Fish/Disk

                 Number

                 Flag Gadgets
                 |
 |  _____________________________________________________________  _   |
 | |                                                             || |  |
 | |                                                             || |  |
 | |                                                             || |  |
 | |
                 Description of record contents
                                           || |  |
 | |                                                             ||_|  |
 | |                                                             ||_|  |
 | |_____________________________________________________________||_|  |
 |  ___  ___   ___  ___   ___  ___   _______________________________   |
 | |   ||   | |   ||   | |   ||   | |                               |  |
 | | < || > | | < || > | | < || > | |
                 Search Expression
                           |  |
 | |___||___| |___||___| |___||___| |_______________________________|  |
 |  Disk       Version    Fish       ____  ____   ___   ___            |
 |                                  |    ||    | |   | |   |           |
 |                                  | <? |          |
 |                                  |____||____| |___| |___|           |
 |_____________________________________________________________________|
NOTE: yes, this looks a bit messed up, but I'll try to fix it soon... --Udo
```

1.91 GADGET:

1.92 GADGET:

1.93 GADGET:

1.94 GADGET:

1.95 GADGET:

1.96 GADGET:

1.97 GADGET: Search Set Window

1.98 GADGET:

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

1.100 GADGET: Disk/Fish(record) (Integer)

Depending on the state of the cycle gadget to the left of this integer gadget, you are expected either to enter a fish (record) number in this gadget, or a disk number.

Notice that a CD-ROM consists only of one 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 "Fish Disk" collection, however, consists of 1000 disks and over 4500 fish (records) so you can quickly jump to these positions in the database.

1.101 GADGET: Search Expression (String)

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 an open book.

Need help with constructing search expressions? Click here .

1.102 GADGET: Fish Description (ListView)

1.103 5 TROUBLE SHOOTING

Sorry, nothing here yet.

1.104 6 THANKS

I would like to extend my thanks to the following people whose feedback, help, input, criticism, requests, and support have helped grind the rough edges off KingFisher and have helped make the program a more polished product that it otherwise would have been:

AMIGA
  The computer that’s fun! Ti amo, Amiga!

Dan Barrett
  For his nitpicking ;-) that finally got me to clean up the KFServer error messages, the welcome banner window, and a lot of other bits. BLAZE on, Dan!

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 2.0. Fred’s efforts have set him apart as one of the Amiga Community’s most important people.

  Fred’s Fish Logo is also used in a gadget with his kind permission.

George Gibeau
  For his efforts at beta-testing KingFisher, and for making numerous suggestions.

Dave Haynie
  For his work on my favorite computer and for DiskSalv 2.0 which has pulled my a** out of a sling several times when that disconnected organ in my skull failed to keep my fingers from typing what they shouldn’t have.

Janne J Kalliola
  For his Finnish translation of KingFisher.

Finn Kettner
  For his Danish translation of KingFisher.

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. :)

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 with regard to using SimpleRexx, which is part of RexxFisher now.

And to all of my beta testers, some of whom have really gone out of their way to make sure things are working correctly and smoothly!

(this list is not complete)