MRQ

Matthias Bethke

COLLABORATORS			
	TITLE : MRQ		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Matthias Bethke	August 22, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

MRQ

1.1 Index

```
-----====*****====-----
      MRQ V1.5
-----====*****====-----


         Introduction
         Features
         Requirements
         Installation
         Tooltypes
         Shell Params
         Configfile
         Examples
         Utilities
         FAQ
         History
         Bugs / ToDo
         Legal babble
         Thanks
         Author
```

1.2 introduction

What it is and why I needed it

MRQ is a MUI-based system patch that tries to do everything^1 the well-known requester improver "ARQ" by Martin J. Laubach does - and a lot more.

ARQ has been around for years now and it always was among my personal "Top 5 Commodities", but I really wanted something a little more configurable. Most of its features are hardcoded, you can neither configure the ARexx interface nor the graphics nor the text scanner that chooses the graphics depending on the requester text.

See Features for how MRQ tries to change all this.

^{1} Well, there's a little drawback concerning asyncronous requesters. See ToDo
  for more info!

1.3 features

Features

– Configurable like every MUI program (fonts, frames, group layout, ...)

– Complete keyboard control just like ARQ – the leftmost button reacts on
  "Return", rightmost on "Esc", and all buttons can be operated via the
  function keys (F1-F10 from left to right). The traditional
  lcommand-v/lcommand-b combinations still work of course.

– Requester texts can be scanned for arbitrary combinations of localized
  strings (identified by their catalog and string number, so a single config
  works independently of the used locale) and custom strings.
  The text comparison can be both case-sensitive and -insensitive and can use
  AmigaDOS patternmatching as well as simple substring searches.

– Can decorate the left- and rightmost button with additional imagery (like
  a checkmark and X for OK/Cancel or something like that)

– Manages an arbitrary number of images that are loaded via datatypes
  and remapped to the current palette. They are loaded
  just-in-time and don't occupy any memory when no requester is open.
  Images can be of any depth, format and size (although it is of course
  not awfully smart to use huge pictures or slow formats like JPEG)
  If no predefined image matches the requester text, MRQ looks for a file
  called "MRQ_DefaultImage" in the IMAGES directory (or in
  PROGDIR:MRQ-Images/ and S:MRQ-Images/ if the image directory is
  unspecified).

– Every image can be combined with an ARexx command, both command and port
  are configurable so you can send messages to any program when a particular
  requester pops up, even start programs or shellscripts through ARexx's
  system interface.

1.4 requirements

MRQ requires at least

– a 68020 CPU
  (it wouldn't have been a problem to compile MRQ for 68000, I just don't
  think it makes sense on such systems though, it's too slow. Time to
  upgrade, boyzngals!)

– AmigaOS 3.0 or higher (datatypes y'know!)

– MUI 3.x

1.5   installation

Installation

Since V1.4 MRQ comes with an installer script. I tested it on a couple of
different directories and it seems to work pretty well. Expect it to
contain bugs anyway – it's the first version! In case you either don't have
Installer V42.12 (minimum version required!) or the script doesn't work as
expected, here's how to install MRQ by hand:

– Put MRQ somewhere on your HD, preferrably the WBStartup drawer. Since V1.2
  MRQ can be started from the Shell too, so the icon is no longer obligatory.
– copy MRQ.config to S: or to the same directory as MRQ.
– copy the "mrq-images" drawer anywhere on your HD (PROGDIR: and S: are
  searched automatically, if you put it anywhere else you have to tell MRQ
  through the IMAGES tooltype

The installer script automagically sets the CONFIGFILE and IMAGES tooltypes
in MRQ's icon. If you already have a previous version installed, the
locations of config and image-dir are taken from the icon so you don't need
to select the drawers on every update :-)
Future versions will be configurable from within installer, i.e. you will be
able to set all tooltype values from the script.

1.6   tooltypes

Tooltypes

CONFIGFILE
  Tells MRQ where to find its configfile. If none is specified, MRQ looks
  for the file MRQ.config first in PROGDIR, then in s:
  Example: CONFIFILE=ENV:MRQ.config

DEBUG
  Makes MRQ open a console window on startup where it prints various debug
  infos. Good for tracking problems.

IMAGES
  The drawer where you keep the image files for MRQ. This drawer is
  used if you specify a relative or no path in the configfile's "IMAGE"
  entries. Specifying no drawer has MRQ search first PROGDIR:, then S:
  for a directory called MRQ-Images.
  Example: IMAGES=SYS:Tools/MRQ-Images

SAMEWIDTH
  Tries to make all buttons in a requester the same width.
  Default is to make them only as wide as the text they contain.
  SAMEWIDTH-buttons will probably look more aesthetic to most people.

MOUSEREQ
  Makes requesters open under the mousepointer.
  Default is to open all requesters centered on their screen.

FRONTSCREEN
  Tries to open requesters on the frontmost intuition screen.
  This is a hack!!!
  It is not OS-legal to open windows on alien non-public screens,
  therefore MUI defaults to opening its window on either the default
  PubScreen or one that was configured for the particular application. But
  a couple of tools have always opened their windows on screens they do not
  own, and for all current Amiga models/OS versions it works fine.

PRECISION
  The precision to use for remapping images via datatypes. Specify one of
  "GUI","ICON", "IMAGE" and "EXACT". Defaults to "IMAGE".

TRANSPARENCY
  Controls whether color #0 of the requester image should be rendered
  transparent, so custom MUI background images can shine through.

SINGLEFRAME
  Use a single frame for image and requester text instead of framing them
  separately.

SIZEABLE
  Make requester windows resizeable

CENTERTEXT
  Center all texts in the requester window. Gives a better look if the
  gadgets are very wide and there's little text in the requester.
  If there's only a single line in the requester, it will always appear
  centered.

SINGLEIMB
  If a requester has only a single button, MRQ won't use an imagebutton by
  default. If you want the "OK" imagebutton to be used in this case (like
  it was up to V1.4), set this tooltype.

OK-IMG / CANCEL-IMG
  If you want the leftmost and rightmost button of every requester to be
  decorated with something like the green checkmark and red X on many Windoze
  requesters, you can specify an image to use for either of them here. Included
  are two brushes called MRQ_OK.brush and MRQ_Cancel.brush (in the same
  directory as the other images) that show said checkmark and X respectively.
  Of course you can use pictures of any format and size here, too. (tested with
  the checkmark and a 250x350 JPEG at the same time :-))
  The IMAGES directory doesn't apply here, specify the full path and filename for
  both images!

1.7 shellargs

Shell Parameters

MRQ's ReadArgs() template when started from the Shell looks like this:

```
Configfile,IMD=ImageDir/K,OB=OKButton/K,CB=CancelButton/K,RP=RemapPrecision/K,
MR=MouseReq/S,SW=SameWidth/S,FS=FrontScreen/S,SF=SingleFrame/S,
TR=Transparency/S,SI=Sizeable/S,CT=CenterText/S,SIB=SingleImB,Debug/S
```

The corresponding tooltype for every parameter should be obvious, see there
for further info!

Example Usage:
  mrq IMD=Work:Graphics/mrq-images OB=s:mrq-images/MRQ_OK.brush
  CB=s:mrq-images/MRQ_Cancel.brush RP=exact sw fs tr mr

1.8   configuration

Configuration

MRQ has a configfile that tells it how to behave. Here's all the keywords:
(all but NEWCLASS may be abbreviated, the two-character abbreviation is
given in ReadArgs()-syntax as <abbrev.>=<keyword>!)

NEWCLASS
   This starts a new entry. For each image and ARexx-command you need to
   define one "event class" - just like the "delete", "printerstuff",
   "software failure" etc. classes know from ARQ, you can just have more of
   them. The following keywords each need an event class they belong to and
   thereby define this class' behavior.

IM=IMAGE
   Specifies filename (and optional path) of an image to display when a
   requester of the current class is detected. The image can be of any
   size and any format you have a datatype for, but remember it will be
   loaded every time a requester pops up, unless you use the PRELOAD switch
   (see below), so don't use too big images/slow formats if you don't have
   a super-fast machine.
   If you specify a full path here, MRQ will use this to look for the image
   file; otherwise the "IMAGES"-tooltype's value is prepended.

PL=PRELOAD
   This is a modifier for IMAGE. It changes the default behavior of loading
   images every time a requester pops up to loading the image while the
   config is being parsed and keeping it in memory. This increases both memory
   usage and speed, the startup time does not change because all images have to
   be loaded once upon startup anyway to read their size.
   The default image is always preloaded.

TR=TRANSPARENT
   A modifier for IMAGE as well, this thells MRQ to render the image with
   transparent background. Every pixel in color #0 is considered background.
   The global TRANSPARENCY option still works to enable transparency
   for all images.

RP=REXXPORT
   The name of an ARexx port which MRQ should send a command to when opening a
   requester of the current class.
   Default (i.e. if you only specify REXXCMD) is "PLAY".

RC=REXXCMD

Command to dispatch via ARexx. For example, if you want to keep using UPD as
configured for ARQ, use something like "ID error_task_held" here.

Only one IMAGE, REXXPORT and REXXCMD should be specified for each class!

Strings a requester should be scanned for can be specified with the following
two keywords, each of which may appear multiple times for each class:
(Note: Switches can be abbreviated, the short form is given in italics!)

ST=STRING
  STRING needs only one argument: a string :-) If this string occurs as a
  requester's body text, it tells ARQ to use image and arexx command of the
  current class.

LO=LOCALE
  LOCALE takes any number of arguments, the first of which must be the name of
  a locale catalog (e.g. "sys/devs.catalog") and the rest numeric arguments
  representing string numbers from that catalog.
  See examples if you have no idea what this means :-)
  If you do, well then, how do you get the locale catalog number of a
  given string? That's what dumpcat is for, see there its doc for more
  info!

To modify the behavior of the text scanner, STRING and LOCALE can be combined
with a couple of switches as follows. Note that not every switch makes sense
with both STRING and LOCALE!

PA=PATTERN / PA
  Use the AmigaOS patternmatching routines to compare the given string and the
  requester text. For a complete description of patterns see your AmigaOS
  manuals; some examples are here.
  Can be used with STRING only.

SU=SUBSTRING / SU
  Simpler and less CPU-consuming than PATTERN, SUBSTRING only searches for the
  specified string at any position inside the requester text. SUBSTRING and
  PATTERN are mutually exclusive of course! (if both are found on one line,
  PATTERN is used and a warning printed if the debug console is open)
  Can be used with both STRING and LOCALE.

NC=CMPNOCASE / NC
  Forces case-insensitive comparison.
  Can be used with both STRING and LOCALE.

FO=FORMATTED / FO
  Don't search the input text to EasyRequestArgs() but the already formatted
  text as it is written into the requester.
  To understand the difference, it's necessary to know that EasyRequestArgs()
  (the function MRQ replaces) can construct requester texts from a format
  string and an argument list. Usually programs feed a localized format string
  to EasyRequestArgs() (it can looks like "This is the %s with %d arguments"
  for example) and have certain placeholders (the percent-some-character
  things) replaced with arguments like "body text" and "2". In this case it's
  just fine to search the format string for some pattern or substring to
  determine the correct requester image. But then, a few programs (among them

the Workbench!) pass only a very general format string to EasyRequestArgs(),
like"%s\n%s\n%s". The actual text is filled in with localized argument
strings – the above format string can result in
"You MUST replace volume
Blah
in any drive!"
Of course you can't tell what the requester will look like from the format
string alone, you have to scan the text as it appears in the requester, and
that's what FORMATTED does. As there are always parameters that change from
one requester to another (the volumename in the above example) you'll almost
certainly want to combine FORMATTED with SUBSTRING to scan f.e. for the string
"You MUST replace volume" in the above requester.
FORMATTED can be used with both STRING and LOCALE.

1.9 examples

Examples

Here's a short sample configuration that should make a few things
clearer...

```
NEWCLASS
LOCALE hello.catalog 2 1 3
LOCALE test.catalog 5
STRING "Hello, (world|Brasil|Erlangen)?" PATTERN
STRING "good morning" NC SU
IMAGE hello.ilbm
REXXPORT "MYSOUNDPLAYER"
REXXCMD "play my_sample"
```


Let's have a look at the individual lines now:
NEWCLASS starts a new event class as described in Configuration.
Following that is a LOCALE parameter defining three strings (numbers 1, 2
and 3 – order doesn't matter) from "hello.catalog" – a (hypothetical)
catalog for a localized "Helloworld" program that might contain strings
like "Hallo, Welt!", "Schweinewelt!" and "Wo soll das alles enden?" (in
its german version :)). Now if any of these strings is found in a
requester's text, this counts as positive identification of the current
event class.
The next line adds another string from test.catalog to the list of strings
that identify this event (you see, the total number of strings that identify
a class and where they come from doesn't matter at all!).
Next is an explicitely specified (non-localized) STRING using
case-sensitive patternmatching. The given pattern matches "Hello, world",
"Hello, Brasil" and "Hello, Erlangen", with an optional character (an
exclamation mark or something) at the end.
The string "good morning" has both the "case-insensitive" and "substring"
switches set, that means anything like "good morning", "gOOd MornInG",
"GOOd MOrnINg", ... matches anywhere inside the requester text, no matter how
much additional text is before of after the string (like, "A very nice GOOD
MORNING to you all!" will match as well).
IMAGE should be easy to understand – it' simply the name of a picture file
that should be displayed in the requester if any of the above strings is
found. You need not specify a path as long as you keep all your pictures in

the directory specified with the IMAGES tooltype, but you can if you want to.
The last two lines aren't difficult either – REXXPORT and REXXCMD tell MRQ
to send the command "play my_sample" to the port "MYSOUNDPLAYER" when a
requester of this class opens.

1.10   utilities

There's currently only one utility that you'll need to configure MRQ:

dumpcat
  Dumpcat is a tool to dump the contents of a locale catalog file to
  the shell (stdout). It takes the following parameters:
  Catalog/A,Neg/S,Max/N
  Catalog is simply the name of the catalog file of which you want to
          see the contents, "sys/devs.catalog" f.e.
  Neg tells dumpcat to scan catalog numbers from zero downwards. Normal
      catalog files contain strings numbered from 0 or 1 upwards, sometimes
      with unused numbers for historical or other reasons. I have only
      come across a single catalog that contains strings with negative
      numbers (sys/dos.catalog) though, but if a catalog seems to be empty,
      trying "Neg" could be the solution.
  Max is the maximum number of strings to scan, defaults to 65536
      Theoretically, catalogs may contain strings under every index possible
      in a longword, but scanning all of them would mean 2^32 calls to
      locale.library, so the range has to be somewhat smaller :)

  Catalog strings are printed one per line, preceded with their number (the
  one you want for MRQ's config!) If it finds newline characters in a string,
  it replaces them with the C-notation for newlines, Backslash-n ('\n').

  Dumpcat will hopefully be obsoleted by the coming Prefs Editor which will
  handle all this cryptic stuff internally so you only need to click on the
  string you want and have its number and catalog name stored in the
  configfile.

1.11   faq

Frequently Asked Questions

Q: A few requesters like "delete" and "copyright" have the correct images
   but most show only the default image. what's wrong?

A: Most likely there are some catalog files missing from your
   LOCALE:Catlogs/sys directory because you are running an english system
   and all texts are english by default so there don't need to be any catalog
   files. I haven't had the time to make an extra config for you yet – it
   requires getting all the messages that are now encoded als LOCALE config
   lines and writing them verbatim into the config. If anybody wants to do
   this – go ahead, I'll include the config in the next
   release!

Q: All images look really strange and especially the imagebuttons are
   totally mangled! Is this an MRQ bug?

A: I'm not sure but I don't think so. It seems there's a problem with MUI's
   image class when making images transparent. On CyberGraphX systems,
   everything is fine (at least I haven't had a bug report from anybody
   running CyberGraphX yet) but on native displays and Picasso96 palette
   remapping and transparency get in each other's way.
   Solution: don't use imagebuttons and switch off transparency for all
   images. I could include an option to render imagebuttons without
   transparency as well but it wouldn't look good at all.

Q: Why don't AssignWedge and similar functions in MCP etc. work any more?

A: These programs also patch EasyRequestArgs(). Some (f.e. the original
   AssignWedge by Olaf Barthel) will work fine if started after MRQ.
   That's because MRQ replaces the entire EasyRequestArgs-function with code
   of its own and never calls the old function back like most patches do, so
   previously installed patches won't get called any more.

Q: Why can I resize but not snapshot an MRQ window?

A: I tried to make this possible but it didn't do no good at all. The
   problem is that I have to assign a MUI object-ID to windows which is the
   same for all requesters – and MUI remembers the size of each
   window that has an ID for the next time it is opened. So when a requester
   with lots of text appears, the window will naturally be large. But
   subsequent requesters won't shrink, they are always as big as the largest
   requester you had open before – they have the same ID after all!
   This sucks, so I didn't use it...

Q: Why is MRQ so slow?

A: Both MUI and the datatypes system are not quite the fastest components
   in AmigaOS. They're programmed for versatility, not speed. You can do
   something to speed them up though:
   – use PRELOAD for frequently used images. They will be
     kept in memory and need to be rendered only once by the datatypes system.
   – don't use too many patterns and backgrounds in MUI (looks ugly anyway :))
   – don't use hard-to-decode formats like GIF for your images

1.12 history

History

V0.1   07–Nov–97   – First working version. Shows formatted body
                           and gadget text, knows the MOUSEREQ, FRONTSCREEN
                           and SAMEWIDTH tooltypes. No graphics & ARexx yet.

```
V0.2     10-Nov-97       - Started text analyzer to determine the correct
                           graphics.
                         - Some optimizations.

V0.3     15-Nov-97       - Text analyzer works, is fully configurable and
                           does patternmatching; no graphics yet.
                         - Added CONFIGFILE tooltype
                         - MRQ honors the Commodities Exchange "Active"
                           setting.
                         - Some bugfixes

V0.4     16-Nov-97       - Images are now selected correctly for any
                           configured keyword.
                         - Started ARexx code to support "upd"ish
                           soundplayers.

V0.5     20-Nov-97       - After trying my drawing skills on a couple of
                           requester images (and finding them virtually
                           nonexistant) I decided to include some of the
                           real nice icons from the "LinuxBrushes" package
                           that appeared on AmiNet lately.
                         - ARexx works, port name and command are separately
                           configurable for every event class.
                         - MRQ would try to add buttons to a nonexistant
                           window if MUI for some reason failed to create
                           the window object. Fixed.
                         - some more Enforcer tests. Although Enforcer has
                           been running continuosly during development, V0.3
                           was the last version to hit now and then.

V0.6a    25-Nov-97       - Implemented optional imagebuttons.
                         - Improved ARexx code. No more waiting for a reply
                           from ARexx befor a requester can be closed.
                         - Improved stack swapping; after allocating my temp
                           strings 'n stuff (~2K) there's now a good safety
                           margin of ~10K left for MUI. Doesn't make the
                           stack smaller now if it was already big enough.

V0.7b    04-Dec-97       - fixed a bug in the ARexx code: no more trouble if
                           MRQ was quit while there were still messages
                           pending from ARexx.
                         - using the standard SAS/C WBmsg-code again as mine
                           was a little...err...screwed.
                         - implemented IMAGES (default image dir) tooltype

V0.8b    11-Dec-97       - a "Show Interface" (CXCMD_APPEAR) from Exchange now
                           opens MRQ's MUI-settings. 't was a little silly to
                           leave the MUI titlebar-gadgets activated on every
                           requester, now you can disable them and still open the
                           prefs when you feel like it.
                         - picture size is now taken from the files themselves
                           via datatypes. No more WIDTH and HEIGHT config params!
                         - now notifies you if started twice (before it
                           would silently ignore the second attempt)
                         - using a datatypes object for the image, remapping
                           is also done by the datatypes system. Unfortunately
                           this doesn't solve the problem with wrong colors if
```

                                    a screen has a weird palette. An own remapping routine
                                    seems inevitable.
                                    - added PRECISION tooltype.

V0.9b   16-Dec-97   - Bugfix: bitmap for the "cancel"-imagebutton
                                    contained an error which caused 16 zeropage reads
                                    (as a side-effect the image looks much better
                                    now 8-))
                                 - Bugfix: I thought I had implemented the
                                    "PRECISION" tooltype but it was always set to
                                    PRECISION_IMAGE due to a missing string :-(
                                 - Bugfix: some structures would not get freed,
                                    muimaster.library not closed and a signal bit not
                                    deallocated if the rightmost button was pressed
                                    (remnant from a very early version...)
                                 - Implemented (really!) DEBUG tooltype
                                 - Implemented SUBSTRING switch for configfile
                                 - Improved the text formatting routine - even
                                    though an 1KB buffer for requester texts should
                                    be large enough, it now makes sure not to write
                                    past the end

V0.10b   05-Jan-98   - Bugfix: the 10th button wasn't correctly bound to
         (unreleased)   the F10 key (forgot the special case of a 3-character
                                    muikey-string for "F10")
                                 - found a potentially severe bug: the IDCMPptr
                                    field in EasyRequestArgs()' parameters was
                                    ignored completely, so f.e. requesters can't be
                                    terminated by IDCMP_DISINSERTED. Not fixed yet due
                                    to problems with MUI_RequestIDCMP()
                                 - Bugfix: default image was always assumed to be 64x64
                                    pixels. Now taken from file as well. Along the way
                                    this fix eliminated some now unnecessary string
                                    operations and half a KB of stack usage.
                                 - Bugfix: SUBSTRING was screwed up due to swapped
                                    strstr() parameters 8-)
                                    This is fixed now and SUBSTRING can be made to search
                                    case-insenstitive as well!
                                 - added some debug output
                                 - added safety check for NT_PROCESS on calling task
                                    before doing anything else in EasyRequestArgs().
                                 - images for the OK/Cancel buttons are now
                                    configurable as well (OK-IMG/CANCEL-IMG
                                    tooltypes)
                                 - implemented SINGLEFRAME tooltype
                                 - Found and removed this utterly obsolete block of gfx
                                    data that once was the default MRQ image but now only
                                    increased the executable size by 4K. Back way under
                                    20K :-)
                                 - improved button layout. Buttons are now always the
                                    same height, even when using imagebuttons with images
                                    of different size.
                                 - found some strings that I always wondered where the
                                    hell AmigaOS gets them from: in sys/dos.catalog
                                    at negative indices! Improved the dumpcat utility
                                    to show them and similarily weird cases.Also added
                                    some error messages and meaningful returncodes to

                                      dumpcat.
                                    – SUBSTRING is now also allowed as a modifier to
                                      LOCALE
                                    – some sourcecode cleanup, more subroutines 'n stuff
                                    – Added shortcuts for configfile switches

V1.1 08-Jan-98   – Bugfix: a pointer to a dynamic string was returned
                                      from a subroutine.
                                    – Bugfix: when neither FRONTSCREEN was set nor a
                                      parent window passed in from the calling program,
                                      the default public screen would be left locked (so
                                      one couldn't f.e. close and re-open the WB screen!)
                                    – fixed a bug in the stristr() (case-insenstitive
                                      string-in-string search) routine
                                    – extended & improved the example configfile
                                    – more debug output
                                    – now returns –1 if called with bad values in
                                      EasyStruct
                                    – implemented FORMATTED switch for configfile
                                    – quite a lot of changes to the documentation
                                    – added the standard requester shortcuts (lcommand v
                                      and lcommand b)

V1.2 21-Jan-98   – Bugfix: the debug console filehandle would always be
                                      passed to Close(), whether the console was open or
                                      not. I wonder why this never caused any Enforcer hits?
                                    – Bugfix: more an annoying feature than a bug, there was
                                      always a two-second delay right before the program
                                      exited. Now the delay is only there if you have a
                                      debug console open, to keep the window from
                                      disappearing before you can read the last lines.
                                    – Bugfix: debug output always showed OK- and
                                      Cancel-buttonimages swapped
                                    – Bugfix: very stupid though harmless bug caused ARexx
                                      support not to work (maybe since V0.8b? Why didn't
                                      anybody tell me? X-))
                                    – New feature: MRQ can now be started from the Shell.
                                      See Shell Params for the parameters!
                                    – New configfile switch: PRELOAD loads
                                      images while the configfile is read (they have to be
                                      loaded anyway to get their size!) and keeps them in
                                      memory. For people who absolutely want JPEGs :-)
                                    – Removed most of the currently unused images from
                                      the distribution. Get the LinuxBrushes archive or
                                      another picture collection if you want to add more
                                      events!
V1.3 23-Jan-98   – Bugfix: beeeg baaad bug! A caller that requested
                                      any IDCMP bits would almost inevitably crash
                                      inside MRQ's patch due to a MUI macro that I
                                      misinterpreted.
                                      This bug also got me on the right track to
                                      implement IDCMP termination – it works!
                                      Thanx to Jaco Schoonen and Dr.Ash for their
                                      bugreports!
                                    – Bugfix: icon object was not freed after reading
                                      tooltypes
                                    – Both config file and images directory are now

```
                              searched:
                              <CONFIGFILE tooltype/parameter> if given
                              PROGDIR:MRQ.config
                              s:MRQ.config
                              <IMAGES tooltype/parameter> if given
                              PROGDIR:MRQ-Images/
                              s:MRQ-Images/
                              If one isn't found anywhere, a requester signals
                              the error and MRQ exits.
                           - Improved configfile: "write protected"-event did
                              not work due to a typo; added "object exists"
                              event; "insert disk" event was missing one locale
                              string
                           - Internal: made some forgotten functions "static"
                              to help the optimizer. Executable size shows it!

V1.4b  08-Feb-98          - Implemented SIZEABLE tooltype
    (not on Aminet)       - Problem with invisible images on non-CyberGraphX
                              systems seems to be solved. The imagebuttons are
                              not perfect yet though, there are still a few
                              strange artifacts in their bitmaps.

V1.4b2 11-Feb-98          - Bugfix: the GadgetFormat parameter was scanned
    (not on Aminet)          for pipe characters ('|' - the gadget separator
                              for intuition's requester functions) before
                              argument substitution took place. This caused a
                              few requesters like the "Read error...-Retry|Abort"
                              DOS requester or the one from AdPro's "Rotate"
                              function to have only a single gadget - they
                              supply the pipe characters in the argument string
                              instead of GadgetFormat :-(
                              Also somewhat reduced stack usage.
                           - fixed some confusion between the program and the
                              documentation: the image directory was documented
                              as PROGDIR:MRQ-images or S:MRQ-images in some
                              places; in fact MRQ looked for a MRQ_Images
                              directory (as documented in the history only).
                              Now it's consistently called "MRQ-images"
                              everywhere.
                           - added CENTERED tooltype.
                           - Bitmaps for preloaded images are now cached after
                              the first layout call.
                           - implemented TRANSPARENT config option
                              for per-image transparency settings.
                           - requesters with lots of text looked ugly because
                              there was a space in window background color
                              above and below the image. Now these spaces are
                              filled with the image's background color (#0).

V1.4   15-Feb-98          - wrote an installer script
                           - added FAQ section to guide
                           - renamed the CENTERED tooltype to CENTERTEXT
                           - changed the default image to 1 bitplane (should
                              never have been in 8 planes!), shrunk to less
                              than 20%!

V1.5b  17-Apr-98          - although it doesn't show very much, V1.5b is a
```

major rewrite! Trying to patch other
intuition.library requester functions and add
some features I found myself adding hacks and
kludges to keep MRQ's main function a drop-in
replacement for EasyRequestArgs(). No good!
Now all the actual patch does is to allocate a
message with some space to hold various data that
needs to be kept while a requester is open. This
it sends to MRQ's message port and just waits for
a reply; the main MRQ process handles everything
else. This makes for much better decoupling of
MRQ and the process (now it may even be a simple
task) calling EasyRequestArgs() than before.
- drastically reduced stack usage (the patch only
  calls Exec now!), so no stack extension code
  needed any more. Nice side-effect: smaller
  executable again :-)
- also got a smaller executable again :-)
- Implemented SINGLEIMB tooltype

V1.5   20-Apr-98   - the ARexx soundplayer message isn't dispatched
                           until all the GUI work is done and the window
                           opened. Seem much better on slow machines and
                           with player demons running at high priority.
                         - Bugfix: race condition fixed in the ARexx code.
                           Not a real problem as soundplayers are not likely
                           to close their ports unexpectedly, but anyway...

1.13 bugs

Stuff that should really be fixed

- the remapping of requester images looks very strange sometimes. Not sure
  if this is an MRQ bug or a datatypes/MUI "feature" though.

Things planned for future releases (roughly in order of priority)

- animation.datatype support
  (Don't let the test-entry in the configfile misguide you - currently MRQ
  just accepts anim files but treats them as stills. No animation.datatype-
  specific code in there yet)

- patch other functions than just EasyRequestArgs() - that's one thing where
  ARQ still rules. I've only come across one or two programs that actually use
  asynchronous requesters, but for completeness...

- real prefs editor with IFF configfile (will take some time...)

- ...anything else? Send suggestions!

1.14   legal

MRQ is freeware

You know what that means, don't you?
If you don't - well, I want to spare myself the usual kilobyte-long disclaimers:

I don't guarantee anything, so if MRQ causes you damage of any kind, it's entirely
your own problem. You get what you pay for.

Of course I tried to make it as bugfree as possible, and if you tell me about any
bugs that show up on your system I will probably consider fixing them. Don't rely
on it though - installing MRQ on a mission-critical system is definitely a Very ←
   Bad
Idea(tm)! But you knew that already :-)

1.15   thanks

Thanks go to:

        - Martin Laubach for ARQ
        - Stefan Stuntz for MUI
        - Jonas 'Zaphod' Petersson for UPD
        - Tony Matthews for the LinuxBrushes archive
        - the SAS/C blokes for continuing support
        - all bugreporters for constructive criticism
        - Unconscious Collective, XIS, Hallucinogen, Acid Junkies and the Green
          Nuns of the Revolution for incredible sounds
        - Cris for being just wonderful

1.16   author

Author

Send comments, suggestions, gifts, flames, files etc. to:

email: Matthias.Bethke@stud.uni-erlangen.de

smail: Matthias Bethke
       Haagstr. 5
       91054 Erlangen
       Germany

1.17   MagicUserInterface

                        This application uses

                    MUI - MagicUserInterface

MUI is a system to generate and maintain graphical user interfaces. With
the aid of a preferences program, the user of an application has the
     ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing
lots of examples and more information about registration please look for
a file called "muiXXusr.lha" (XX means the latest version number) on
          your local bulletin boards or on public domain disks.

          If you want to register directly, feel free to send

                        DM 30.- or US$ 20.-

                                 to

                          Stefan Stuntz
                      Eduard-Spranger-Straße 7
                         80935 München
                            GERMANY