Varexx Readme 5 Feb 1996

COLLABORATORS

	TITLE : Varexx Readme 5 Feb 1996	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 25, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Varexx Readme 5 Feb 1996

1.1 Varexx Readme 5 Feb 1996

Varexx Documentation
~~~~~~~~~~~~~~~~~~~~~
Release 1.7 (c)1995-1996 Andrew Cook

5 Feb 1996

1.2 Introduction - So what is it and what can it do ?

Varexx is a program that allows you to control graphical user interfaces (GUI's) from Arexx. The program loads the .gui files that are saved by Jan van den Baard's Gadtoolsbox program and displays them on the screen. Arexx scripts can then send messages to a port and control the actions of the Gui. The script can read and set the contents of each of the gadgets, resize the window and such like.

1.3 Legal Bit - What won't I do ?

Some or all of the files in this package may be may be transmitted by any means. Providing words to the effect of

Varexx and Arexxport.library are Copyright(c)1995 Andrew Cook.
~~~~~~~~~~~~~~~~~~~~~

and my email address (amc93el@soton.ac.uk) are included in the accompanying documentation. The files are FreeWare, copyright is retained by the author (Andrew Cook).

These files may be included in any distribution commercial, shareware, freeware or other. I would like to be informed of any such packages, (drop me an email) but this is not a requirement.

ALTHOUGH THE FILES HAVE BEEN CAREFULLY TESTED ANY ERRORS OR ACCIDENTS ARISING FROM THEIR USE IS NOT THE RESPONSIBITY OF THE AUTHOR. USE AT YOUR OWN PERIL.

I take no responsibility for anything Varexx may do to your system. But is less likely to crash your Amiga than a certain new operating system 95 will. (Unless you've got a bridge board of course.)

1.4 About the Author - So who are you ?

I am a third year electronics undergraduate at the University of Southampton, England. I like SF, technology and having a laugh. If you like you can't check out my www site as I haven't got one. However do drop me a email to

    amc93el@soton.ac.uk

This will be valid until August 1996 after which I will get a demon account. So download this package again and pick my new address out of that.

1.5 Installation - What do I do now ?

There are two programs in the Varexx distribution just copy both into your path.

Varexx GUIPATH

This launches the Varexx server. This opens a port called VAREXX and waits for scripts to send messages to it. The server detaches itself properly so it can be run in the user-startup with the line "Varexx". Only one copy of Varexx can (and needs to) be running at one time.

The Varexx program requires the Arexxport.library and the reqtools.library, both of these are included in the distribution, copy them into your libs: assignment.

The command can be optionally followed by a path. This will be used as the default path to load .gui files from.

eg Varexx rexx:gui/

Varexx will now look in rexx:gui/ path for any .gui files it needs.

VXC

This closes the Varexx server down. This can only be done if no gui files are currently loaded into the server.

1.6 The Demo Script - What can it do ?

To run the demos, click on the DemoSetup Icon. Then open the demos drawer and click on the demo or the address book icons.

The demo shows off the features of Varexx and source code will reveal most of the information needed to write your own scripts for Varexx.

The Address Book is a simple address database written in Arexx. It demonstrates the use of two windows at once with Varexx.

When address book is started a file requester will appear asking for a data file to load. An example containing a few interesting people I know is included. (Load the example address.data).
The main window is now displayed. You can scroll through the names using the '<' and '>' buttons. Clicking on the 'List' produces a list of names in the book and picking a name jumps to that entry.
To add an entry click on the 'Clear' button. Fill in the entries and click on 'Update' to add the entry to the book.
To change an entry, find it, change it and click on 'Update'.
To delete an entry, click on delete, clicking twice undelete it. None of the changes are saved until the address book is closed. A that time, the entries are saved and the address book is sorted.

The testgui file, brings up a filerequester asking for the a .gui file. It then displays this and echos all messages from Varexx to the output console.

TestGUI2 does the same except, that it hacks about inside the .gui file and extracts the name of all the windows in the file and gives you the choice of them. This script was written by Nick Ring and contains some hairy code so he will be horrified that I've distributed it unchanged. I apologise to him for that - but it can be useful.

Fergus Duniho has a script to use his spell checker - AlphaSpell - with Varexx. It's on the Aminet as text/util/AspellGUI.lha. (And well worth a look.)

1.7 Programming Varexx - How do they do that ?

So you want to create your own Arexx gui's. You need will need to create a gui file using Gadtoolsbox, see the documentation on using that program and the notes below.

Having created the gui file you must then write the script to load and display the window.

Some Notes / Comments and Ideas

Okay, having seen a few peoples scripts and written a few of my own. I feel that some ground rules (ideas anyway) should be written.

1) Starting, stopping the Varexx server from within a script.
I feel you should only close the Varexx server ('address command VXC' does it) if you have started it yourself. See the example script below.

2) The location of the .GUI file.
Your script should not hard code this. I feel the best thing to set a variable near the start of the script containing the .gui file name. Allowing the user to change it easily. Secondly I would assume that all the .gui files will be stored in a single location, and that the user will tell varexx where that is. So don't hard code paths into the gui file name.

But that's just my $0.02 worth.

1.8 Example Script

```
/* Test script for Varexx */
    guifile = 'myguifile.gui'

    options results

    /* Open libs needed */
    if ~show("L","rexxsupport.library") then
        if ~addlib("rexxsupport.library", 0, -30) then exit

    /* Check Varexx is loaded if not load it */

    if show( 'p', 'VAREXX' ) ~= 1 then do
        address command 'run Varexx'
        waitforport VAREXX
        RanVarexx = TRUE
    end; else
        RanVarexx = FALSE

    address VAREXX

    /* Open the port for gui to talk to */
    call openport("WINDOWPORT")

    /* Load the gui file into Varexx */
    'load ' guifile 'WINDOWPORT'

    /* Set host to the port for this gui file */
    host = result
```

```
    address value host

    /* Display the window */
    show

    /* Wait for the user to close the window */
    do forever
        /* Wait for a message from Varexx */
        call waitpkt( "WINDOWPORT" )

        /* Get the message */
        packet = getpkt( "WINDOWPORT" )

        /* This is not a null message */
        if packet ~= '00000000'x then do

            /* Get the information about the message */
            class  = getarg(packet)

            /* If the message says the user clicked on the closewindow
             * gadget then leave */
            if class = closewindow then leave

        end
    end

    /* Hide the window and unload the gui file from memory */
    'hide unload'

    /* Close the port */
    call closeport( "WINDOWPORT" )

    /* I launched varexx so close it */
    if RanVarexx = TRUE then ADDRESS COMMAND VXC

    exit
```

1.9 The GUI files

When using Gadtoolsbox to design .gui file for Varexx the following hints may be useful.

At the moment, Menus are not supported by Varexx so don't bother to set any up. Also Fixed Text and Bevel boxes are not rendered by Varexx. Varexx will ignore any Custom or Public screen settings. This may change in the future so it is best to design the Gui to open on the default public screen. This allows the screen to be specified by the PS argument to the load command and will not clash with any future behavior of Varexx. The plan is to have Varexx able to open custom public / private screens at some time.

Of the window flags :

```
    SIZEGADGET      - Varexx doesn't resize the gadgets in a window.
                      But this flag may be set.
    DEPTHGADGET     - May be set.
```

```
    SIZEBRIGHT      - May be set. See SIZEGADGET.
    SMART_REFRESH   - Ignored by Varexx.
    SUPER_BITMAP    - Ignored by Varexx.
    BACKDROP        - May be set. (Not often useful.)
    GIMMEZEROZERO   - This has no effect on the final appearance.
                      However less memory is used if it is left off.
    ACTIVATE        - May be set.
    DRAGBAR         - May be set.
    CLOSEGADGET     - May be set.
    SIZEBOTTOM      - May be set. See SIZEGADGET.
    SIMPLE_REFRESH  - Ignored by Varexx.
    OTHER_REFRESH   - Ignored By Varexx.
    REPORT_MOUSE    - Ignored By Varexx.
    BORDERLESS      - May be set. ( But not very useful.)
    RMBTRAP         - Ignored by Varexx.

A subset of the IDCMP options are used by Varexx. If these options are
selected then Varexx will send additional messages to the Arexx script.

    IDCMP_ACTIVEWINDOW - A activewindow message is sent when the Varexx
        window becomes active. ie. when the user clicks on it.
    IDCMP_INACTIVEWINDOW - A inactivewindow message is sent when the Varexx
        window ceases to be active. ie. when the user clicks outside it.
    IDCMP_DISKINSERTED - A diskinserted message is sent when when a disk
        is inserted.
    IDCMP_DISKREMOVED - A diskremoved message is sent if a disk is removed.
    IDCMP_NEWSIZE - A newsize message is sent when the size of the window
        is changed. This includes clicking on the ZIP gadget.
    IDCMP_CHANGEWINDOW - A changewindow message is sent when there is any
        change in the window's size or position.
    IDCMP_VANILLAKEY - When a key is pressed which doesn't affect a gadget
        then a KEYBOARD message will be sent to the script.

Of the tags,
    InnerWidth, InnerHeight,
    AutoAdjust, FallBack    - May be set.
    MouseQueue, RptQueue    - Are all ignored by Varexx.
    WindowTitle is used.
    ScreenTitle is used. Enter a blank title to get the default screen
    title.
    Prefs Zoom and Pos Zoom will give a window with no size gadget a
zoom gadget. Both zip the window to it's title bar. Prefs also moves
the window to the top left corner of the screen.

All gadgets are supported. The label for each gadget is the title used
by Varexx to name each gadget.

FILE_KIND gadgets rely on the script to supply the filerequester, they
just act as BUTTON_KIND gadgets. This allows the script rather than
Varexx to determine the information in the requester.

Key Strokes ( _Cancel etc ) are supported, providing the box is ticked
in Gadtoolsbox. Tabbing between gadget also works.

Do not compress your GUI files or you will be in deep trouble.
Default options listed for the LISTVIEW kind do not work.
```

1.10 Arexx Commands

When Varexx is active there are three different ports that you have to deal with. Two of these are opened by Varexx and you send messages to them. The other is opened by you and Varexx sends messages to it when the user clicks on a gadget.

When the Varexx server it opens a port called 'VAREXX' there are two commands that can be sent to this port.

Commands for the VAREXX port.
Commands for window ports

1.11 Commands for the VAREXX port.

When launched Varexx opens a Arexx port called 'VAREXX'. This port accepts two different commands.

Load
Quit
Version

1.12 Load

load FILE/A,PORT/A,PS/K

Loads the given .gui file into memory. If the file is given an absolute path (ie. rexx:gui/myfile.gui) then that will be loaded. If not then the default path (given when Varexx is run) will be used. If not found there then the file on its own will be looked for.
ie. If Varexx is run with the line "Varexx rexx:guifiles/"

The commands

load fish/mygui.gui
    will try to load "rexx:guifiles/fish/mygui.gui" and then try
    "fish/mygui.gui".
load rexx:fish/mygui.gui
    tries to load "rexx:fish/mygui.gui".
load mygui.gui
    tries to load "rexx:guifiles/mygui.gui" and then "mygui.gui".

This command doesn't display the window however. PORT is the name of the port to which the window will send messages when the user operates the window's gadgets. This port should be created with the OpenPort( ) function in the rexxsupport library.
The windows will open on the default public screen unless the name of an alternative screen is given in the PS option.
This command returns the name (in result - so turn 'options results' on first) of a Arexx port to which messages about this window may be sent.
eg

```
        options results
        ...

        address VAREXX

        call openport("SQUID")

        'load dh1:c_src/Varexx2/gui/test.gui SQUID PS=MYSCREEN'
        host = result
```

This loads the gui file 'test.gui' into memory, setting it to send messages to a port called 'SQUID', display it's windows on the MYSCREEN and it will receive commands at the port named in the variable 'host'. (Use 'address var host' to send commands.)

1.13 Quit

```
quit
```

If there are no .gui file currently loaded by the Varexx server this command will close the server down and free the memory used by it. This is what the VXC command sends to the VAREXX port.

1.14 Version

```
version
```
Returns the version number of currently active copy of Varexx. The version number is returned in RESULT. This command first appeared in v1.3 so use code like this.

```
    Options Results

    ...

    ADDRESS VAREXX
    version
    if rc ~= 0 then say 'Varexx 1.2 or earlier'
        else say 'Varexx 'RESULT
    ...
```

This command may also be sent to the window ports as well.

1.15 Commands for window ports

The following command can be sent to the specific port for a gui file, ie. the portname returned by the load command.

```
 Show
 Hide
 Window
 Busy
```

```
 Set
 Settext
 Setnum
 Setcheck
 Setbar
 SetList
 Read
 Spawn
 Setlabel
 Activate
 Readcords
```

1.16 Show

```
show WINDOW
```

This displays the named window. The window name is given to the window in Gadtoolsbox not the window title. If no name is given then the first window in the file is displayed.

1.17 Hide

```
hide UNLOAD/S
```

If the .gui file has a window open this command will close it. If the unload option is given, the .gui file will be removed from memory as well. You cannot read information from the windows gadgets after this command has been given. This command can be given with the unload option when no window is open to free up memory.

1.18 Window

```
window ZIP/S,FRONT/S,BACK/S,ACTIVATE/S,X/N/K,Y/N/K
```

This command changes the state of the window.
ZIP - move the window to it's alternate place and size. As if the user clicked the zip gadget.
FRONT - Brings the window to front of the screen.
BACK - Puts it to the back (surprised ?).
ACTIVATE - Gives the window the focus for user input.
X - Number specifes the left edge of the window.
Y - Number specifes the top edge of the window.

1.19 Busy

```
busy SET/S

The command 'busy set' sets the busy pointer in the Varexx window
and blocks all user input to the window. Just 'busy' on its own clears
the pointer and unblocks the window again.
```

1.20 Set

```
set LABEL/M/A,ENABLE/S,DISABLE/S

This enables / disables the given gadget(s). A list of gadgets can
be given and all will be affected.
```

1.21 Settext

```
settext LABEL/A,STRING/F

Passes the given string to the gadget. Used by STRING_KIND and
TEXT_KIND  gadgets only. This sets the text in both of those types.
```

1.22 Setnum

```
setnum LABEL/A,NUMBER/N

Passes the number to the given gadget. Sets the value of NUMBER_KIND
and  INTEGER_KIND gadgets. Picks the option for MX_KIND and CYCLE_KIND
(these are  numbered from 0 upwards ). Sets the position of the marker
in SCROLLER and  SLIDER kind.
```

1.23 Setcheck

```
setcheck LABEL/A,CHECK/S

Set a gadget of CHECKBOX_KIND to ticked or not.
```

1.24 Setbar

```
setbar LABEL/A,MIN=VISIBLE/N/K,MAX=TOTAL/N/K

For a SLIDER_KIND this sets the max and min values of the run.
For a SCROLLER_KIND this sets the max values and the number of
units the knob represents.
```

1.25 SetList

```
setlist LABEL/A,CLEAR/S,DEL/S,SELECT/K,STEM/K,ITEMS/M,UPDATE/N/K

Controls the listview type of gadgets. Okay, this is were it gets fun.
CLEAR switch this clears the listview. Before any other action is
taken. ITEMS is a list of items to be added to the bottom of the list.
Or if the DEL option is given as well these items will be removed from
the list instead.
The SELECT key word is used to specify which item is currently
selected. Some examples of these :

setlist label CLEAR cod haddock place

Clears the current list and replaces it with the items 'cod',
'haddock' and 'place'.

setlist label DEL cod

Removes item 'cod' from the list.

setlist label select=place

selects the 'place' item.

The STEM option. This gives the name of an Arexx variable from which the
items to be added to the list are taken. The variable 'stem.COUNT' must
contain the number of items, which are then numbered 'stem.1', 'stem.2'
etc. This option only allows you to add to the list you cannot use the DEL
switch in conjunction with this. But the SELECT and CLEAR options do work.
In addition the variable 'stem.SELECT' may contain the number of item in
the list to be selected.
To do the same as

setlist label CLEAR cod haddock place SELECT cod

You can use

fish.COUNT = 3
fish.1 = cod
fish.2 = haddock
fish.3 = place
fish.SELECT = 1
setlist label CLEAR STEM fish

Note :
Some other rexx utils use stem.0 to hold the number of items rather
than stem.COUNT. A line like stem.0 = 'stem.COUNT' is no real bother.
So Varexx will stick to using stem.COUNT

Finally the update option, this will cause the given item number to be
updated. To the value given on the line.

setlist label UPDATE=2 'skate'
Will change item 2 in the earlier example from 'haddock' to 'skate'.
```

```
If the stem option is given the that item will be read from the variable.
fish.2 = 'skate'
setlist label STEM fish UPDATE=2

If the update and the select option are both given. Then the numbered item
will be selected in the list. If no stem or text is given then no updating
will be done. Due the way Varexx parses this command line, a dummy string
must be given after the select item.
setlist label SELECT s UPDATE=3
will select 'place' in the above list.

setlist label SELECT s UPDATE=3 'skate'
This will change 'place' to 'skate' and then select it.
```

1.26 Read

```
read LABEL/A,VAR,NUMBER/S

Allow Arexx to read the current state of the gadget. The state of the
gadget is returned in RESULT. If the gadget is a listview and the VAR
option is given then the complete list is written into the same stem
variables as the setlist command uses.
The NUMBER option will return the ordinal number of the item in the
list view in the RESULT variable.

If the list contains the items 'cod' 'haddock' 'place' and 'cod' is
selected then

read label cake

will set these variables.

cake.COUNT = 3
cake.1 = cod
cake.2 = haddock
cake.3 = place
cake.SELECT = 1

Also the commands

options results
read label NUMBER

will give RESULT = 1.
If no item is selected in the listview then stem.SELECT = 0.
```

1.27 Spawn

```
spawn PORTNAME/A,PS/K
This command allows you to have more than one window open on the screen at
once. It works in the same way as the load command. except that instead of
loading a new .gui file. It merely clones the already loaded one. The
```

```
command returns a new VAREXX.## portname in the RESULT field.
        options results
        ...

        address VAREXX

        call openport("SQUID")

        'load dh1:c_src/Varexx2/gui/test.gui SQUID PS=MYSCREEN'
        host = result
        address value host
        show window_one

        call openport("SQUID.1")

        'spawn SQUID.1 PS=MYSCREEN2'

        host2 = result

        address value host2

        show window_two
```

This opens the 'window_one' on the MYSCREEN public screen and 'window_two' on MYSCREEN2.
Each window must be passed a hide unload command to free up the memory connected with it. However the order in which these are sent doesn't matter.

1.28 Setlabel

```
Setlabel LABEL/A,TEXT/M,CYCLE/S,SCREEN/S,WINDOW/S
```

This command allows you to set the text in the window. It works in a number of different ways. For all gadgets except MX_KIND. You can use 'setlabel GADGET_LABEL TEXT' to set the gadget's title. This only works if the window is not open on the screen. Ie the 'show' command hasn't been given yet.

For MX_KIND the text for the options. Is given as
'setlabel GADGET_LABEL OPTION_1 OPTION_2 etc'
Again the window must not be open for this to work. In all these cases the underscore '_' is used to specify the keyboard shortcut.

For CYCLE_KIND the options in the gadget may be set by including the CYCLE switch. So 'setlabel GADGET_LABEL cycle OPTION_1 OPTION_2 etc' is used. This will work regardless of wether the window is open or not.

If the SCREEN or WINDOW switch is given. Then the LABEL is assumed to be a window name not a gadget name and either the screen title or the window title for that window will be set. This will work if the window is open as well.

Some examples, (these need the window to be shut.)
setlabel OKAY '_Fin'

```
Sets the label in the OKAY gadget to Fin.

setlabel 'label=CYCLE' '_Choose'
Sets the cycle gadget label. The 'label=' allows varexx to distinguish
between the label and the switch option.

setlabel MXCHOICE 'One' 'Two' 'Three'
Sets the options of a MX_KIND gadget.

These will work regardless of weather the window is open or not.
setlabel NEXT '"' || 'Window Title' || '"' window

Sets the window title of the window called NEXT to 'Window Title' the
strange syntax is to allow varexx to correctly parse the spaces.

setlabel NEXT '"' || 'Varexx by Andrew Cook' || '"' screen
Sets the screen title for when the NEXT window is open.

setlabel 'label=cycle' 'cycle' 'Yes' 'No' 'Maybe'
Sets the options in the cycle gadget.
```

1.29 Activate

```
Activate LABEL/A

This command applies only to STRING_KIND and INTERGER_KIND gadgets only.
It activates the given gadget to allow the user to enter data into it.
```

1.30 Readcords

```
Readcords STEM/A

This stores the X, Y locations of the currently open window into the
variables STEM.X and STEM.Y.
```

1.31 Messages from Varexx

```
Varexx sends messages to the port your script when events occur. Most of
these messages will be as a result of user action. Every time a gadget
is clicked a message is sent consisting the gadgets label and some
information. The information is dependant on the gadgets type.

BUTTON_KIND, FILE_KIND - No information. Just the gadget name is sent.

STRING_KIND - The text entered by the user into the gadget.

NUMBER_KIND - The number entered into the gadget

MX_KIND, CYCLE_KIND - The option number selected.
```

```
CHECKBOX_KIND - TRUE if the gadget is ticked otherwise FALSE.

SLIDER_KIND, SCROLLER_KIND - The slider numeric position.

LISTVIEW_KIND - The contents of the selected item.

When the user selects the close window gadget a message of 'CLOSEWINDOW'
is sent.

If the correct flags have been set in the IDCMP section of Gadtoolsbox
then these messages may be received.

    ACTIVEWINDOW
    INACTIVEWINDOW
    DISKINSERTED
    DISKREMOVED
    NEWSIZE
    CHANGEWINDOW

It is left as an exercise for the reader to decide what events can cause
each one ? ( Hint - Look in section 5.1 :-)

Some Arexx commands ( WINDOW ) may cause some of these messages to be
sent so it is possible to get stuck in a loop situation.

When windows that have been spawned the CLOSEWINDOW, ACTIVEWINDOW,
INACTIVEWINDOW, NEWSIZE and CHANGEWINDOW window messages will append the
name of the window they came from (if and only if they come from a
spawned window.) This allows two windows to share the same port in an
Arexx script.

The KEYBOARD message will be sent if the IDCMP_VANILLAKEY flag has been set.
The message is followed by the value of the key pressed.
ie
    KEYBOARD a
    KEYBOARD V
    KEYBOARD 4
    KEYBOARD @

For other keys the following messages will be sent
ESC TAB
BS (backspace)
DELETE HELP RETURN
UP DOWN LEFT RIGHT (Cursor keys.)
F1, F2, F3 etc.

ie
    KEYBOARD UP
```

1.32 Thanks - Could not be done without ?

```
And now for something tacky

Thanks to
    Jan van den Baard   - For Gadtoolsbox.
```

```
    Nico Francois       - For reqtools and many other programs.
    Richard Rauch       - Some early ideas.
    Fergus Duniho       - For AlphaSpell and relighting my interest in
                          this project with his gui's for XES and
                          ideas and help.
    Matt Dillon         - For DICE.
    Peter Miller        - A PC programmer but with a knowledge of
                          68000 assembler and constant source of
                          (dis)couragement.
    Stephan Sürken      - For Text2Guide.
    Douglas Keller      - The ButClass code. (It on the Aminet in dev/gui).
    Escom               - For this new A1200.
    Commodore           - For this A500.

    These people and others have give ideas, bug reports, moans and even
    in some cases actually used Varexx.

    Gregg Green, Boris Foldman, Nick Ring, Michael Pounders, Kevin Phair,
    Andy Styles, John Collet, Marcel Offermans, Mattias Linder,
    Matthew Soar, Jim Hines, Girard Michel, Perry Mowbray.

        (Sounds for people being sick offstage.)

Cheers

Andy Cook

Email 'amc93el@soton.ac.uk'

Varexx                  Copyright (c)1995-1996 Andrew Cook
Arexxport.library       Copyright (c)1995 Andrew Cook
reqtools.library        Copyright (c)1991-1994 Nico Francois
                                      1995 Magnus Holmgren
rexxreqtools.library    Copyright (c)1992-1994 Rafael D'Halleweyn.
AlphaSpell              Copyright (c)1995 Fergus Duniho.
Text2Guide 03.10        Copyright (c)1994 Stephan Sürken

All other copyrights, trademarks etc acknowledged.

The button class and file_kind gadgets where created with BOOPSI objects
based on those by Douglas Keller.
```

1.33 Arexxport Library - What is that ?

```
This is a shared library I wrote to allow the easy, yet flexible
implementation of Arexx ports in programs. If you would like the
(brief) documentation and support files. Drop me a line.
```

1.34 Todo - What's left ?

```
Features To Add ( possible )
```

```
- The Gadtoolsbox program allows you to define beveled boxes, and
  fixed text neither of these are rendered by Varexx at the moment.
- Menus.

If you want / need one of these (or anything else) email me.
```

1.35 History - What was done when ?

```
v0.0 To v0.5
v1.0, v1.1
v1.2, v1.3
v1.4, v1.5
v1.6
v1.7
```

1.36 v0.0 To v0.5

```
v0.2
02-09-95  - First beta test version.

06-09-95  - Added the Window ZIP/S,FRONT/S,BACK/S,ACTIVATE/S command.

v0.3
09-09-95  - Added stem/A option to setlist command that adds items to
            list view from rexx variables. Suggested by Fergus.
          - Added a /M option to the set command allows enabling and
            disabling of many gadgets in one line.

10-09-95  - When the close gadget is clicked now returns CLOSEWINDOW
            rather than closewindow. This is more consistent but
            breaks all scripts so far written. :-(

14-09-95  - Some internal clean up.
          - BUGFIX Text_kind gadgets now work properly.
          - Removed the ALL/S option from read command. It didn't do
            anything !
          - read and setlist now can make use of a stem.SELECT
            variable.
          - Added NUMBER/S option to read command.

v0.4
17-09-95  - Cleaned up error reporting code.
          - Added code to close the window is the port to which it
            should send messages is closed. Ie when the script dies.

30-09-95  - BUGFIX Varexx died if the libs weren't installed. Now it
            doesn't you get a nice error message instead. Unless of
            course it was the intuition.library that could not be
            loaded. In which case your Amiga may be sick. Reported by
            Boris.
          - BUGFIX Fixed string bugs that caused enforcer hits. Again
            thanks to Boris.
```

```
01-10-95  - BUGFIX Located the bug that caused Guru's after exit. Fixed

            but had to removed key short cuts. Boris and enforcer at
            work again.
          - BUGFIX Can now read properly from unclicked gadgets.

v0.5
02-10-95  - Added my own key handling functions.

04-10-95  - Wrote a routine to load the .gui files myself. Thus
            removing the need for GadToolsBox.library.
          - Then replaced the nofrag library with the exec libpooled
            functions.

05-10-95  - BUGFIX In List view selection. When the current item
            was deleted the selection was not cleared. Reported by
            Fergus.
```

1.37 v1.0, v1.1

```
v1.0
05-11-95  - First non-beta release.

07-11-95  - Minor change to window command. Doesn't call
            WindowToFront() or WindowToBack if not needed.

09-11-95  - Removed the GET command.

20-11-95  - BUGFIX In settext parsing. Reported by Fergus.

v1.1
20-11-95  - BUGFIX The layout of gadgets with different fonts works
            correctly.

21-11-95  - Added support for disk, windowsize, windowactive events.

21-11-95  - BUGFIX Check box gadget now sends the correct messages.

22-11-95  - BUGFIX Large number of selections in cycle gadgets caused
            guru's. Reported by Nick Ring.
          - Add default path for .gui support.
```

1.38 v1.2, v1.3

```
v1.2
22-11-95  - Second public release.

27-11-95  - Can now use set to disable MX_KIND and LISTVIEW_KIND. However
            operation only has an effect on v39+ versions of gadtools.

28-11-95  - Added spawn command. Allows more than one window to open from
```

```
            one GUI at a time.

30-11-95  - BUGFIX Read command now returns 0 in stem.SELECT if not item
            is selected in the list view. Reported by Nick Ring.

04-12-95  - Tidied up the spawn command.

v1.3
04-12-95  - Added a version command.
          - Now remembers location of windows when they are closed.

05-12-95  - Now supports GET_FILE gadgets (suggested by Nick Ring) and
            correctly supports shortcuts for buttons.

06-12-95  - Settext with no line, now clears the gadgets.
          - BUGFIX Dealt with enforcer hits reported by Kevin Phair.

07-12-95  - Tided up the distribution

08-12-95  - MX_KIND gadgets now respond to key strokes.

11-12-95  - BUGFIX Some problems with the setlist command.
```

1.39 v1.4, v1.5

```
v1.4
14-12-95  - Third public release.

03-01-96  - BUGFIX Check box return values inverted. Thanks John Collett.

10-01-96  - BUGFIX Resizing of GUI saved with fonts other than Topaz 8 now
            works. Reported by Michel Girard.
          - Removed case sensitivity from gadget labels.
          - Added the setlabel command.
          - Added X, Y options to window command.

v1.5
11-01-96  - Added UPDATE option to setlist command. Suggested by Fergus.

25-01-96  - Added activate command. Suggested by Perry Mowbray.
          - Added READCORDS command.
          - Added support for the keyboard.
```

1.40 v1.6

```
v1.6
30-01-96  - Fourth public release.

05-02-96  - BUGFIX I had linked v1.6 with a old version of the button BOOPSI
            gadget.
12-02-96  - BUGFIX In the readcords command and some fixes to the arexxport
            library. (Arexxport now v36.22).
```

1.41 v1.7

```
v1.7

12-02-96  - Fifth public release
```

1.42 Index