

M2IARexx

COLLABORATORS

| | | | |
|------------|---------------------------------|---------------|-----------|
| | TITLE : M2IAR _{exx} | | |
| ACTION | NAME | DATE | SIGNATURE |
| WRITTEN BY | | July 26, 2024 | |

REVISION HISTORY

| | | | |
|--------|------|-------------|------|
| NUMBER | DATE | DESCRIPTION | NAME |
| | | | |

Contents

| | | |
|----------|---|----------|
| 1 | M2IARexx | 1 |
| 1.1 | M2I version V / AREXX | 1 |
| 1.2 | AREXX and M2I | 2 |
| 1.3 | Executing an AREXX command from M2I | 2 |
| 1.4 | Getting started with AREXX | 3 |
| 1.5 | Commands index | 4 |
| 1.6 | M2I address | 6 |
| 1.7 | Return codes | 6 |
| 1.8 | Using spaces and quotes | 6 |
| 1.9 | 'Search-icon' arguments | 7 |
| 1.10 | Project instructions | 11 |
| 1.11 | Collecting Informations | 11 |
| 1.12 | Using Drawers / Windows | 12 |
| 1.13 | Messages Boxes / Requesters | 12 |
| 1.14 | Executing Commands / Tools | 13 |
| 1.15 | Editing menu | 13 |
| 1.16 | Using public screens | 13 |
| 1.17 | Miscellaneous | 14 |
| 1.18 | OPENDRW | 14 |
| 1.19 | CLOSEDRW | 14 |
| 1.20 | CLOSESUBS | 15 |
| 1.21 | ICONIFY | 15 |
| 1.22 | REMICON | 16 |
| 1.23 | EXECTOOL | 16 |
| 1.24 | INTERNAL | 17 |
| 1.25 | QUIT | 17 |
| 1.26 | WBTASKS | 18 |
| 1.27 | OPENMENU | 18 |
| 1.28 | SAVEMENUAS | 18 |
| 1.29 | SAVEMENU | 19 |

| | |
|-------------------------------|----|
| 1.30 REQUEST | 19 |
| 1.31 MESSAGE | 20 |
| 1.32 ERROR | 20 |
| 1.33 FLUSH | 20 |
| 1.34 SOUND | 21 |
| 1.35 PLAYFILE | 21 |
| 1.36 POPUP | 21 |
| 1.37 GETPATH | 22 |
| 1.38 GETNAME | 22 |
| 1.39 GET | 23 |
| 1.40 SET | 24 |
| 1.41 COUNT | 26 |
| 1.42 REQDIR | 26 |
| 1.43 REQFILE | 27 |
| 1.44 REQSTRING | 27 |
| 1.45 REQMULTIFILE | 27 |
| 1.46 OPENPUBDRW | 28 |
| 1.47 PUBLICDRW | 29 |
| 1.48 GETPUBLIC | 29 |
| 1.49 SETDEFPUB | 29 |
| 1.50 PUBTOFRONT | 29 |
| 1.51 VERSION | 30 |
| 1.52 EDMODE | 30 |
| 1.53 LOCKUSER | 30 |
| 1.54 UNLOCKUSER | 30 |
| 1.55 REDRAW | 31 |
| 1.56 GETCHG | 31 |
| 1.57 ADDCHG | 31 |
| 1.58 ADDDRW | 31 |
| 1.59 ADDTOOL | 32 |
| 1.60 ADDICON | 32 |
| 1.61 DRWTOFRONT | 33 |
| 1.62 GETPUBSIZE | 33 |
| 1.63 GETMOUSE | 33 |
| 1.64 GETWINSIZE | 34 |
| 1.65 MOVEITEM | 34 |
| 1.66 LASTSELECTPATH | 35 |
| 1.67 LASTSELECTNAME | 35 |
| 1.68 DELETE | 36 |
| 1.69 WBINFO | 36 |
| 1.70 AUTOSCAN | 36 |

Chapter 1

M2IARexx

1.1 M2I version V / AREXX

```
=====
M2I version V / AREXX
```

```
Copyright (C) 1991-1998 Thomas PIMMEL
```

```
Using M2I with AREXX          Release 5.76
=====
```

CONTENTS

Introduction

AREXX and M2I

Executing an AREXX command from M2I

Getting started with AREXX

Commands index

General

M2I address

Return codes

Using spaces and quotes

'Search-Icon' arguments

Functions

Project instructions

Collecting informations

Using Drawers / Windows

Messages Boxes / Requesters

Executing Commands / Tools

Editing menu

Using public screens

Miscellaneous

1.2 AREXX and M2I

AREXX AND M2I

All what you can do with your mouse, AREXX can do it either, and much more. AREXX uses commands that control M2I in a BASIC language.

AREXX gives you some other commands that you can't access with your mouse (for example, concerning public screens).

M2I has 51 AREXX commands.

Usage examples :

- adding Tools after reading a drawer on your disk.
- opening a Drawer on a public screen.
- remembering configurations and selecting one just with a click.
- changing parameters for several Drawers.
- ...

1.3 Executing an AREXX command from M2I

EXECUTING AN AREXX COMMAND FROM M2I

Nothing's easier : edit or create a Tool and select AREXX in the cycle gadget SHELL / WBENCH / AREXX.

In the "Tool" gadget, type your AREXX script name.

If your script is in REXX:, you don't have to type the path. If its name extension is ".rexx" or the extension you define in the MISC preferences, you don't have to type it.

In the "Arguments" gadget, type the arguments if necessary, or nothing.

If you need an output, type it. If your output has to be interactive, remember to include a CON: window.

You can add a hotkey to your command in the "Keyboard" gadget.

Click OK.

Change to Use mode.

Double-Click on your Tool.

If AREXX isn't activated, M2I will try to run REXxMast as you define it in the MISC preferences, "RExxMast" gadget.

Script execution.

If AREXX error occurs, M2I display it (48 error messages).

The most frequent error is "Program not found". AREXX didn't find your script.

1.4 Getting started with AREXX

GETTING STARTED WITH AREXX

AREXX is nothing more than a program, that reads texts including commands and that executes them.

AREXX allows communication between programs.

INDISPENSABLE TOOLS

The REXxMast program (included in your Amiga package). Normally, it is in your "System" drawer.

The REXxSysLib library that is in LIBS:. It is normally included in your package too.

A text editor, Ed or anyone else.

USEFUL TOOLS

The TS and TE programs that respectively turn on and off the step by step execution mode for AREXX. Very useful when you have an error.

WHERE DO AREXX FIND ITS SCRIPTS ?

Normally, in a REXX: assign. If you give a complete path to M2I, AREXX can find them where you want.

WHAT DOES AN AREXX SCRIPT LOOK LIKE ?

An AREXX script must begin with a comment. Here is an example of a script which works (!!!)

```
/* This is an AREXX comment */
```

UPCASE / LOWCASE

Generally, AREXX isn't case dependent, therefore M2I too.

There's one exception : the address of the task you control.
M2I address is : "M2IPort.Rexx".

CONTROLLING M2I

Here is the minimal script to control M2I :

```
/* We don't do anything for the moment...
   it's only a start */

OPTIONS RESULTS    /* Allows M2I to send back results */

ADDRESS "M2IPort.Rexx" /* Following commands will concern M2I */

EXIT              /* "exit" is for ending (optionnal) */
```

A FIRST SCRIPT

```
/* This script counts icons in your menu and displays
   a message in M2I */

OPTIONS RESULTS
ADDRESS "M2IPort.Rexx"

/* Count icons */
Count "*"

/* Display message */
Message "You have" RESULT "icon(s) in M2I"

EXIT      /* good bye */
```

VARIABLES

RC always represent the return code (0 if successful). RESULT is the result of the function. in our example, RESULT has been sent by the "Count" function.

MORE INFORMATIONS

I don't want to write an AREXX doc. You can find some in the public domain. I just give you some basis. I think that you could read M2I scripts.

Now it's time to work. You can send me your productions.

1.5 Commands index

COMMANDS INDEX

Alphabetical order :


```
ADDCHG
ADDRW      "Label" "Search-icon"
ADDICON    "Icon" "Search-icon"
ADDTOL     "Label" "Search-icon"
AUTOSCAN   "Search-icon"
CLOSEDRW   "Search-icon"
CLOSESUBS  "Search-icon"
COUNT     "Search-icon"
DELETE     "Search-icon"
DRWTOFRONT "Search-icon"
EDMODE     [ON/OFF]
ERROR      "Message"
EXECTOOL   "Search-icon"
FLUSH
GET        Keyword "Search-icon"
GETCHG
GETMOUSE   X/Y
GETNAME    "Search-icon"
GETPATH    "Search-icon"
GETPUBLIC  "PubScreen"
GETPUBSIZE W/H "PubScreen"
GETWINSIZE W/H "Search-icon"
ICONIFY    "Search-icon"
INTERNAL   "Command"
LASTSELECTNAME
LASTSELECTPATH
LOCKUSER
MESSAGE    "Message"
MOVEITEM   [Méthode] "Search-icon" "Search-icon"
OPENDRW    "Search-icon"
OPENMENU   ["Menu name"]
OPENPUBDRW "PubScreen" "Search-icon"
PLAYFILE   "File"
POPUP
PUBLICDRW  "Search-icon"
PUBTOFRONT ["PubScreen"]
QUIT
REDRAW     "Search-icon"
REMICON    "Search-icon"
REQDIR     ["Drawer"]
REQFILE    ["File"]
REQMULTIFILE "Drawer" "Pattern" ["Window message"]
REQSTRING  "Message"
REQUEST    [GADGETS "Gad1|Gad..."] "Message"
SAVEMENU   ["Menu name"]
SAVEMENUAS ["Menu name"]
SET        Keyword "Value" "Search-icon"
SETDEFPUB  ["PubScreen"]
SOUND      [ON|OFF]
UNLOCKUSER
VERSION
WBINFO     "Search-icon"
WBTASKS
```

1.6 M2I address

M2I AREXX address :

```
"M2IPort.Rexx"
```

(case dependant).

Advised extension for M2I scripts :

```
".M2I"
```

1.7 Return codes

RETURN CODES

They are generally returned by M2I. RC variable contains it.

30 Memory error

20 Unknown command

11 Bad args

10 Error while executing command

6 Return from "Search-icon", no element found

5 Warning

1 Yes (returned by REQUEST)

0 No error / No (returned by REQUEST)

NOTE : if RC~=0, the RESULT variable isn't initialised.

1.8 Using spaces and quotes

USING SPACES AND QUOTES

M2I manage labels and "Search-icon" containing spaces.

For that, the program must know you use a space in your string.

Here is the way AREXX works with a string it sends to a program :

```
/* Example SET KeyWord "Value" "Search-icon" */
```

```
Variable=":xxx/yyy/zzz.t"
```

```
set label "My Label" Variable
```

```
Exit
```

M2I receive that :

```
SET LABEL My Label :xxx/yyy/zzz.t
```

Words without quotes are "upcased". If they represent an AREXX variable, they are replaced by the current value of this variable and the quotes disappear. M2I will search the icon named Label and won't find it.

Here is the way to do it :

```
set label "'My Label'" Variable
```

M2I will receive :

```
SET LABEL 'Mon Label' :xxx/yyy/zz.t
```

You can exchange the quotes type :

```
set label '"My Label"' Variable
```

USING VARIABLES

If you aren't sure of your variable, you can put it between quotes :

```
REQSTRING "How can I name the first drawer of the root ?"
```

```
IF RC=0 THEN
```

```
DO
```

```
    NAME="'RESULT'"
```

```
    SET LABEL NAME ":1"
```

```
END
```

If the user types "My Label", M2I will receive :

```
SET LABEL 'My Label' :1
```

1.9 'Search-icon' arguments

"Search-icon" ARGUMENTS

"Search-icon" is an argument which allows to designate an icon or a group of icons, that is Drawers or Tools.

A command using a "Search-Icon" argument, will be executed as many times there are icons matching "Search-Icon".

Syntax :

```
Search-icon = "[:][[path][*]/...][label][*][.1|d|t]"
```

M2I "Search-icon" arguments aren't case dependent.

We will try the OPENDRW command (open drawer). Its syntax is OPENDRW "Search-icon".

1. Searching an icon from the root

":" represent the root. To open the root you can type :

```
OPENDRW ":"
```

Then you can type a Drawer label :

```
OPENDRW ":xxx"
```

The drawer called "xxx" in the root is opened.

You can type sub-drawers too :

```
OPENDRW ":xxx/yyy" /* open drawer yyy in xxx */
```

And so on...

```
OPENDRW ":xxx/yyy/zzz"
```

NOTE : If you have two drawers named ":xxx", the two will be opened.

2. Specifying an icon type.

"Search-icon" considers that the "/" char always follows a Drawer. However, the last "Search-icon" element doesn't have a default type.

If you want to force the icon type, just add after the last element :

".d" you search a Drawer.

".t" you search a Tool.

```
OPENDRW ":xxx/yyy.d" /* you will open the yyy drawer */
```

What happens if you give a Tool argument instead of a Drawer argument ?

Generally, commands consider that you want to operate the parent Drawer of this Tool.

```
OPENDRW ":xxx/zzz.t" /* you will open the xxx drawer */
```

3. Forcing a single argument

Normally, a command is executed for each matching icon.

If you add ".1" at the end of "Search-icon", command will be executed once.

```
OPENDRW ":xxx.1" /* will open the first Drawer called xxx */
```

".1" can be added to another specification :

".1d" or ".d1" means first Drawer.

".1t" or ".t1" means first Tool.

4. Icons without label

You can type the number of an icon in its drawer instead of its label.

```
OPENDRW ":3" /* Third icon in the root */
```

WARNING : the number doesn't depend on the type.

OPENDRW ":3.d" will fail if the third icon isn't a drawer.

Valid example : OPENDRW ":xxx/4/yyy/3"

In all cases, RC value will be 6 if "Search-icon" doesn't find what you search.

Warning : the command COUNT will send back 0 if "Search-icon" doesn't find anything.

5. Wild cards

You aren't compelled to type the complete label of an icon. You can add a "*" after the beginning of the name.

```
OPENDRW ":xx*"
```

will open Drawers beginning with "xx".

You can either type only a star :

```
OPENDRW ":*"
```

will open all Drawers in the root and, if there's a Tool, the root itself.

Previous specifications are available too :

```
OPENDRW ":x*/yyy/z*.d"
```

NOTE : if you replace a label by "*", icons without labels will be concerned too.

You can either replace a char by a "?".

OPENDRW ":xxx?" will open xxx1, and xxxf...

6. The path

You aren't compelled to start from the root to give a "Search-icon" argument.

OPENDRW "xxx/*.d" will open all Drawers in a Drawer called xxx.

If there are several Drawers named "xxx", command will be executed for each.

You can omit the path too.

OPENDRW "zzz.t" will open the Drawer containing the Tool "zzz"

NOTE : "*" means all icons,
"*.t" means all Tools,
"*.d" means all Drawers.

7. Conclusion

Commands using a "Search-icon" argument are very powerful.

Generally, a Drawer followed by a Tool is good enough to find a tool. For example, I have 3 icons labelled Devpac (a Drawer, an AREXX script, and a shell Tool) :

"Devpac.d" is unique. It's my language Drawer.

"ARexx/Devpac" is unique too.

"Devpac/Devpac" too...

NOTE : When you use a "*" try to optimize search.

Example :

OPENDRW "/*.d" does the same as OPENDRW "*", but it's not called as many times.

If you use REDRAW it's even more important.

REDRAW "*" will redraw a window for each Tool it contains.

REDRAW "/*.d" is better.

1.10 Project instructions

PROJECT INSTRUCTIONS

These instructions can be used in the Project Menu :

Load / Save

```
OPENMENU ["Menu name"]
SAVEMENU ["Menu name"]
SAVEMENUAS ["Menu name"]
```

Quit

QUIT

REMEMBER :

GETNAME ":" returns the Menu name

1.11 Collecting Informations

COLLECTING INFORMATIONS

These instructions don't process M2I but they only get informations about the menu.

COUNT "Search-icon"

```
GET Keyword "Search-icon"
GETCHG
GETMOUSE X/Y
GETNAME "Search-icon"
GETPATH "Search-icon"
GETWINSIZE W/H "Search-icon"
```

```
LASTSELECTNAME
LASTSELECTPATH
```

```
VERSION
WBTASKS
```

REMEMBER :

GETNAME ":" returns the Menu name
SOUND without argument, returns the
state of the sound task.
EDMODE without argument, returns the

mode M2I is running.

1.12 Using Drawers / Windows

USING DRAWERS

Open / close

```
OPENDRW      "Search-icon"  
DRWTOFRONT   "Search-icon"  
CLOSEDRW     "Search-icon"  
CLOSESUBS    "Search-icon"
```

Iconify

```
ICONIFY       "Search-icon"  
REMICON       "Search-icon"  
POPUP
```

AutoScan

AUTOSCAN

SEE ALSO

```
SET      XPOS|YPOS "Search-icon"  
GETWINSIZE W/H "Search-icon"
```

FLUSH

1.13 Messages Boxes / Requesters

MESSAGES BOXES / REQUESTERS

Messages

```
ERROR      "Message"  
MESSAGE     "Message"
```

Requesters

```
REQUEST     [GADGETS "Gad1|Gad..." ] "Message"  
REQSTRING   "Message"
```

File / Drawer request

```
REQDIR      ["Drawer"]  
REQFILE     ["File"]  
REQMULTIFILE "Drawer" "Pattern" ["Window message"]
```


1.14 Executing Commands / Tools

EXECUTING COMMANDS / TOOLS

Internal Commands

INTERNAL "Command"

Tools

EXECTOOL "Search-icon"

1.15 Editing menu

EDITING MENU

To display changes

REDRAW "Search-icon"

Instructions that need a REDRAW

ADDDRW "Label" "Search-icon"

ADDICON "Icon" "Search-icon"

ADDTOOL "Label" "Search-icon"

DELETE "Search-icon"

MOVEITEM [Method] "Search-icon" "Search-icon"

Instructions that sometimes need a REDRAW

SET Keyword "Value" "Search-icon"

Optional Instructions

ADDCHG

EDMODE [ON/OFF]

LOCKUSER

UNLOCKUSER

1.16 Using public screens

USING PUBLIC SCREENS

GETPUBLIC "PubScreen"

GETPUBSIZE W/H "PubScreen"

OPENPUBDRW "PubScreen" "Search-icon"

PUBLICDRW "Search-icon"

PUBTOFRONT ["PubScreen"]

SETDEFPUB ["PubScreen"]

SEE ALSO

The WinPatch program.

1.17 Miscellaneous

MISCELLANEOUS

Reduce memory usage

FLUSH

Sounds

PLAYFILE "File"

SOUND [ON|OFF]

Misc

WBINFO "Search-icon"

1.18 OPENDRW

OPENDRW "Search-icon"

Open the specified Drawer, or, if your parameter is a Tool, its Drawer.

Return : 0 success
10 error when opening a window
(immediately stops Search-icon)

Result : Number of times the procedure have been called
by "Search-icon"

NOTE : if the Drawer was iconified, its icon is removed from Workbench.

See CLOSEDRW "Search-icon"
CLOSESUBS "Search-icon"

1.19 CLOSEDRW

CLOSEDRW "Search-icon"

Close the specified Drawer, or, if your parameter is a Tool, its

Drawer.

Return : 0

Result : Number of times the procedure have been called
by "Search-icon"

NOTE : - if the Drawer is already closed, the command is canceled.

- the icons from the window are not removed from memory. You
must use FLUSH.

See OPENDRW "Search-icon"
CLOSESUBS "Search-icon"

1.20 CLOSESUBS

CLOSESUBS "Search-icon"

Close all sub-drawers of the specified Drawer.

Return : 0

Result : Number of times the procedure have been called
by "Search-icon"

NOTE : - CLOSESUBS doesn't work like CLOSEDRW. It frees the memory from
the closed Drawers icons.

- CLOSESUBS only closes sub-drawers and not the specified Drawer.

See OPENDRW "Search-icon"
CLOSESUBS "Search-icon"

1.21 ICONIFY

ICONIFY "Search-icon"

Iconify the specified Drawer, or, if your parameter is a Tool, its
Drawer.

Return : 0 success
10 Error while iconify

Result : Number of times the procedure have been called
by "Search-icon"

NOTE : if the Drawer is already iconified, the command is canceled.

See REMICON "Search-icon"

1.22 REMICON

REMICON "Search-icon"

Remove from Workbench the specified Drawer icon, or, if your parameter is a Tool, its Drawer.

Return : 0

Result : Number of times the procedure have been called
by "Search-icon"

See ICONIFY "Search-icon"

1.23 EXECTOOL

EXECTOOL "Search-icon"

Execute the specified Tool. Don't accept all internal commands. Can be runned either in Shell or in WBench mode. Only one Tool will be executed.

Return : 5 the icon is a Drawer or a Tool without command.
0 Ok
10 Launch error (or internal command forbidden)

Result : Program launched

NOTE : - it doesn't allows commands using :
OPEN, SAVE, PARENT, EDIT, WBINFO
Use equivalent ARexx commands.

- for safety, EXECTOOL executes only one Tool, the first
matching "Search-icon"..

See Project instructions

1.24 INTERNAL

INTERNAL "Command"

Execute the specified internal command.

Return : 5 User cancel (Preferences)
0 Command found and allowed
10 Unknown command

Result : #####

Internal commands :

ABOUT
FLUSH
ICONIFY
MAIN
MISCPREFS
NEW
PREFS
QUIT
SPY
SNDBREAK
SNDPREFS

(See M2I.guide for more about them)

NOTE : Refuse the commands :
OPEN, SAVE, PARENT, EDIT, WBINFO
Use equivalent ARexx commands.

See Project instructions

1.25 QUIT

QUIT

Quit program (same as "quit" menu)

Return : 0

Result : Number of WBench tasks waiting
(can be zero)

1.26 WBTASKS

WBTASKS

Return number of WBench tasks running.

Return : 0

Result : Number of WBench tasks waiting
(can be zero)

1.27 OPENMENU

OPENMENU ["Menu name"]

Open the specified menu, or open a file request if no name is given.

Return : 5 User cancel
0 No error
10 Error while loading menu

Result : Menu name if Return = 0

See SAVEMENU ["Menu name"]
SAVEMENUAS ["Menu name"]

1.28 SAVEMENUAS

SAVEMENUAS ["Menu name"]

Open a file request. If you type a name, it will be proposed to the user.

Return : 5 User cancel
0 No error
10 Error while saving menu

Result : Menu name if Return = 0

See OPENMENU ["Menu name"]
SAVEMENU ["Menu name"]

1.29 SAVEMENU

```
SAVEMENU ["Menu name"]
```

Save the menu, using the default name (defined when you load the menu). If you type a name, the menu is saved with its new name.

```
Return :  5 No default name.  
         0 No error  
        10 Error while saving menu
```

Next time you will use "Save" without argument, default name will be those you just type.

```
Result :  Menu name if Return = 0
```

```
See OPENMENU  ["Menu name"]  
    SAVEMENUAS ["Menu name"]
```

1.30 REQUEST

```
REQUEST [GADGETS "Gad1|Gad..."] "Message"
```

Open a requester with the message "Message" and 2 gadgets (Yes/No).

```
Return :  1 Yes  
         0 No  
        11 Error in arguments
```

```
Result :  #####
```

NOTE : Extended usage.

If you add the keyword "GADGETS" as the first argument, M2I waits for the gadgets user texts separated by a "|". Therefore, the Return code will be :

```
First Gadget  : 1
```

```
Second Gadget : 2
```

```
n th Gadget  : n
```

```
Last Gadget   : 0
```

Warning : don't define more than 9 gadgets to avoid problems with an error Return.

Example :

```
REQUEST GADGETS "Eat|Sleep|Drink" "This evening, you want to :"
```

```
1 = "Eat"  
2 = "Sleep"  
0 = "Drink"
```

SPACES

If you have spaces in your gadgets texts, don't forget the double quotes :

Examples :

```
REQUEST GADGETS "'Choice 1|Choice 2'" "What do you want ?"
```

or

```
REQUEST GADGETS '"Choice 1|Choice 2"' "What do you want ?"
```

1.31 MESSAGE

```
MESSAGE "Message"
```

Open a message request with the message "Message" and the OK gadget.

```
Return : 1 Ok  
        (10 Error!) not yet implemented
```

```
Result : #####
```

See ERROR "Message"

1.32 ERROR

```
ERROR "Message"
```

Open an error request with the message "Message" and the OK gadget.

```
Return : 1 Ok  
        (10 Error!) not yet implemented
```

```
Result : #####
```

See MESSAGE "Message"

1.33 FLUSH

FLUSH

Free the memory (no argument).

Return : #####

Result : Size of freed memory

1.34 SOUND

SOUND [ON|OFF]

Install or remove the sound task, or give the task state.

If there's an argument :

Return : 5 No change
0 Ok
10 error, can't do that

If no argument :

1 Sound installed
0 No sound

Result : #####

See PLAYFILE "File"

1.35 PLAYFILE

PLAYFILE "File"

Play the sound named "File".

Return : 5 Sound is off
0 Sound sent to the task

Result : #####

See SOUND [ON|OFF]

1.36 POPUP

POPUP

Open M2I after a command "IconifyAll".

```
Return : 5 M2I wasn't iconified
        0 Ok
```

```
Result : #####
```

See INTERNAL "Command"
with command = "Iconify"

1.37 GETPATH

GETPATH "Search-icon"

Return the complete path to an icon.

```
Return : 5 No path (root Drawer)
        0 Success
        10 Error (memory)
```

```
Result : Complete path
```

NOTE : GETPATH gives you the path only to the first icon
matching "Search-icon".

See GETNAME "Search-icon"
LASTSELECTPATH
LASTSELECTNAME

1.38 GETNAME

GETNAME "Search-icon"

Return an icon name followed by its .T or .D extention.
If "Search-icon" is the root (":"), you'll have the menu name.

```
Return : 0
```

```
Result : Icon name (or number if no name)
```

NOTE : GETNAME only gives you the name of the first icon

matching "Search-icon".

See GETPATH "Search-icon"
LASTSELECTNAME
LASTSELECTPATH

1.39 GET

GET Keyword "Search-icon"

With Keyword

Drawer or Tool :

TYPE type (0/1) (Drawer/Tool)
LABEL icon label
ICON icon name

Drawer :

XPOS window X position
YPOS window Y position
LOCK lock window (0/2/4/6) (See note)
LMOD icons loading mode (0/1/2)
IPLACE icons place (0/1/2) (Up / Down / Center)
WLIST display type (0/1/2/3/4/5)
(0, horiz.icons, 1 vert. , 2 or 3 texts,
4 horiz. buttons, 5 vert. buttons)
LC number of lines or colonms (1-9)
WINDOW window opened (0/1) (No/Yes)
APPICON icon in wbench (0/1) (No/Yes)
ITEM sub-items (0/1) (No/Yes)

DRAWERTYPE type of drawer

Values to add :

0/1 Type (Simple / AutoScan)
0/2 WB (Auto / Force WBench)
0/4 Sorting (No / Alpha)
0/8 Sorting (2) (No / Entry Type)
0/16/32 Files (Icons/All/Ignore)
64 Force
128 Directory already scanned (set by M2I)

Exemple : DRAWERTYPE=21 (1+4+16)

Tool :

TOOL tool
ARG arguments
PATH default path to Tool

IO output
STACK stack size
PRI priority (-128 to +127)
WB wbench task
HOTKEY hotkey
SOUND sound file

Return : 0 Success
 5 Empty string
 10 Bad icon type

Result : depends on the type

NOTE : Search-icon executes the command only for one icon.

NOTE : Locked windows

0 normal

2 without title bar

4 borderless (but normal look)

6 (2+4) without title bar and borderless, 3D look

Examples :

GET STACK "dev*.T"
GET ICON ":3/PIPO"

See SET Keyword "Value" "Search-icon"

1.40 SET

SET Keyword "Value" "Search-icon"

With Keyword

Drawer or Tool :

LABEL icon label
ICON icon name

Drawer :

XPOS window X position
YPOS window Y position
(position in screen changes immediately if the window

is opened. Only YPOS).

LOCK locked window (0/2/4/6) See note 4
LMOD icons loading mode (0/1/2)
IPLACE icons place (0/1/2) (Up / Down / Center)
WLIST display type (0/1/2/3/4/5)
 (0, horiz.icons, 1 vert. , 2 or 3 texts,
 4 horiz. buttons, 5 vert. buttons)
LC number of lines or colonms (1-9)

DRAWERTYPE type of drawer

DRAWERTYPE=0 Simple Drawer, DRAWERTYPE<>0 AutoScan

Values to add :

0/1 Type (Simple / AutoScan)
0/2 WB (Auto / Force WBench)
0/4 Sorting (No / Alpha)
0/8 Sorting (2) (No / Entry Type)
0/16/32 Files (Icons/All/Ignore)
64 Force
128 Directory already scanned (set by M2I)

Exemple : DRAWERTYPE=21 (1+4+16)

DIRECTORY Directory to scan (AutoScan)
FILTER filter (AutoScan)
DEFTOOL Default Tool (AutoScan)

Tool :

TOOL tool
ARG arguments
PATH default path to Tool
IO output
STACK stack size (0 to ...)
PRI priority (-128 to +127)
WB wbench task (0 SHELL / 1 WB / 2 AREXX)
HOTKEY hotkey
FLG Flags (0 / 1) See note 1
SOUND sound file

Return : 0 Success
 10 Bad icon type

Result : Number of icons processed

NOTE 1: There is only one flag for the moment. FLG 1 means that
the hotkey must be resident.

Examples :

```
SET ICON "" "*" All icons set to default.
```

```
SET WB "1" "Dev*.T" Tools matching "Dev*.T" must be runned in  
WBench mode.
```

NOTE 2: LABEL and ICON are ignored for the Root.
(Example : SET LABEL "Yahoo" ":")

NOTE 3: Value with spaces or quotes.

```
SET LABEL ' "My Label"' ":1" (My Label)  
SET LABEL "'My Label'" ":1" (My Label)  
SET LABEL ' "What''s up"' ":1" (What's up)  
SET LABEL "'What''s up'" ":1" (What's up)
```

and so on...

NOTE 4: Locked windows

0 normal

2 without title bar

4 borderless (but normal look)

6 (2+4) without title bar and borderless, 3D look

See GET Keyword "Search-icon"
(for the values)

1.41 COUNT

```
COUNT "Search-icon"
```

Return the number of icons matching "Search-icon"

Return : 0 Success

Result : Number of icons.

1.42 REQDIR

```
REQDIR ["Drawer"]
```

Return a drawer selected by the user.

```
Return : 0 Success  
        5 Cancel
```

```
Result : Drawer (with complete path)
```

```
See REQFILE    ["File"]  
REQMULTIFILE  "Drawer" "Pattern" ["Window message"]  
REQSTRING     "Message"
```

1.43 REQFILE

```
REQFILE ["File"]
```

Return a file selected by the user.

```
Return : 0 Success  
        5 Cancel
```

```
Result : File (with complete path)
```

```
See REQDIR     ["Drawer"]  
REQMULTIFILE  "Drawer" "Pattern" ["Window message"]  
REQSTRING     "Message"
```

1.44 REQSTRING

```
REQSTRING "Message"
```

Return a text typed by the user.

```
Return : 0 Success  
        5 Cancel
```

```
Result : String (max. 200 chars, termination not included)
```

```
See REQDIR     ["Drawer"]  
REQFILE        ["File"]  
REQMULTIFILE  "Drawer" "Pattern" ["Window message"]
```

1.45 REQMULTIFILE

```
REQMULTIFILE "Drawer" "Pattern" ["Window message"]
```

Return a list of files selected by the user.

```
Return : 0 Success
         5 Cancel
         10 Error (bad drawer)
         30 Memory
```

```
Result : String
```

NOTE : String contains the selected drawer, followed by the files names separated by a space. The drawer is always followed by ":" or "/". The returned drawer isn't inevitably the same as the one in the command argument.

Returns examples :

```
"WBench:Yahoo/ FirstFile"
```

```
"Ram: FisrtFile SecondFile ThirdFile"
```

```
See REQDIR    ["Drawer"]
REQFILE      ["File"]
REQSTRING    "Message"
```

1.46 OPENPUBDRW

```
OPENPUBDRW "PubScreen" "Search-icon"
```

Open a Drawer in the public screen indicated.

```
Return : 0 success
         5 Drawer already opened or public screen not found
         10 Error while opening window
           (immediately stops Search-icon)
```

```
Result : Number of times the procedure have been called
         by "Search-icon"
```

```
Example : OPENPUBDRW "WorkBench" ":"
```

NOTE : - Works the same way as OPENDRW

- Opens only the closed Drawers.

```
See OPENDRW    "Search-icon"
CLOSEDRW      "Search-icon"
CLOSESUBS     "Search-icon"
```


1.47 PUBLICDRW

PUBLICDRW "Search-icon"

Return the public screen in which the window is opened.

Return : 0 Success
5 Default screen

Result : Public screen name

1.48 GETPUBLIC

GETPUBLIC "PubScreen"

Test a public screen.

Return : 0 It exists
5 Not found

1.49 SETDEFPUB

SETDEFPUB ["PubScreen"]

Set the default screen (without argument, the M2I default screen becomes the system default screen).

Return : 0

Warning :

No error code.

Use GETPUBLIC if you want to be sure to pick an existing screen.

See GETPUBLIC "PubScreen"

1.50 PUBTOFRONT

PUBTOFRONT ["PubScreen"]

Move the public screen to foreground. Without argument, the M2I default screen is moved to foreground (5.35!).

Return : 0 Success
5 Not found

See GETPUBLIC "PubScreen"

1.51 VERSION

VERSION

Return the M2I version.

Return : 0

Result : Version (5.xx)

1.52 EDMODE

EDMODE [ON/OFF]

Switch to edition mode (ON) or to execution mode (OFF).

Return : 0 Success

5 Menu was already in this mode

Without argument, it returns the mode :

Return : 0

Result : 0 Edition

1 Execution

1.53 LOCKUSER

LOCKUSER

Lock user actions to edit menu with AREXX.

Return : 0

Result : #####

See UNLOCKUSER

1.54 UNLOCKUSER

UNLOCKUSER

Unlock user actions after editing menu with AREXX.

Return : 0 Success

5 Not locked

Result : #####

See LOCKUSER

1.55 REDRAW

REDRAW "Search-icon"

Redraw the specified Drawer, or, if your parameter is a Tool, its Drawer.

It redraws only the opened Drawers (windows).

Return : 0 success

Result : Number of times the procedure have been called
by "Search-icon"

1.56 GETCHG

GETCHG

Returns the number of changes since last save.

Return : 0 success

Result : Number of changes

See ADDCHG

1.57 ADDCHG

ADDCHG

Add 1 to the changes counter.

Return : 0 success

Result : Number of changes

See GETCHG

1.58 ADDDRW

```
ADDDRW "Name" "Search-icon"
```

Insert a Drawer with the name "Name" in the path "Search-icon".

If "Search-icon" is a Drawer, the new is inserted at the end.

If "Search-icon" is a Tool, the new is inserted after this Tool.

```
Return : 0 success
        30 not enough memory
```

```
Result : ##### (for the moment)
```

```
See ADDICON "Icon" "Search-icon"
  ADDTOOL "Label" "Search-icon"
```

1.59 ADDTOOL

```
ADDTOOL "Name" "Search-icon"
```

Insert a Tool with the name "Name" in the path "Search-icon".

If "Search-icon" is a Drawer, the new is inserted at the end.

If "Search-icon" is a Tool, the new is inserted after this Tool.

```
Return : 0 success
        30 not enough memory
```

```
Result : ##### (for the moment)
```

NOTE : Default stack is 4000, no Tool.

```
See ADDICON "Icon" "Search-icon"
  ADDDRW "Label" "Search-icon"
```

1.60 ADDICON

```
ADDICON "Icon" "Search-icon"
```

Insert a Tool or a Drawer same as the icon "Icon" in the path "Search-icon".

If "Search-icon" is a Drawer, the new is inserted at the end.

If "Search-icon" is a Tool, the new is inserted after this Tool.

```
Return : 0 success
        10 Error
```

Result : Icon label

NOTE : this works like "drag and drop" in a window.

```
See ADDDRW      "Label" "Search-icon"  
  ADDTOOL      "Label" "Search-icon"
```

1.61 DRWTOFRONT

DRWTOFRONT "Search-icon"

Move the Drawer to foreground.

If "Search-icon" is a Tool, its parent Drawer is moved to foreground.

Return : 0 success

Result : Number of windows in foreground
(can be 0)

NOTE : process only opened Drawers.

1.62 GETPUBSIZE

GETPUBSIZE W/H "PubScreen"

Return the public screen width (W) or height (H).

Example : GETPUBSIZE H "Workbench"

Return : 0 success
5 Screen not found

Result : Size in pixel

1.63 GETMOUSE

GETMOUSE X/Y

Return the X or Y mouse position in active screen.

Return : 0 success
5 No active screen

Result : Screen position

1.64 GETWINSIZE

GETWINSIZE W/H "Search-icon"

Return the specified window width (W) or height (H).

Return : 0 success
5 Closed window

Result : Size in pixel

Example : GETWINSIZE W ":YAHOO/YAHEE.D"

1.65 MOVEITEM

MOVEITEM [Method] "Search-icon" "Search-icon"

=> MOVEITEM [BEFORE/AFTER/LAST] "SOURCE" "DESTINATION"

Move the specified items from source to destination.

Return : 0

Result : Number of icons moved

With Method :

Without method, the copy is made in the destination Drawer at the first place, or, if the destination is a Tool, in its parent Drawer at the first place.

LAST same as before, except that copy is made at the last place.
Warning : Result isn't right if source and destination are the same. The same Icon can be moved several times).

AFTER the copy is made after the destination icon.

BEFORE the copy is made before the destination icon.

NOTE : A copy from a father to a son is canceled. Source can be multiple, destination is unique.

Example (1) :

```
MOVEITEM AFTER "*.t" ":"
```

Example (2) :

```
/* Rotation of icons in a window */  
MOVEITEM LAST "Yahoo/*" "Yahoo.d"
```

1.66 LASTSELECTPATH

LASTSELECTPATH

Return the path to the last selected icon.

```
Return : 0 success  
        5 No selection  
        30 Memory error
```

Result : Path to the last selected icon.

See LASTSELECTNAME

1.67 LASTSELECTNAME

LASTSELECTNAME

Return the last selected icon name.

```
Return : 0 success  
        5 No selection  
        30 Memory error
```

Result : Name of the last selected icon.

Script example :

```
/* using LASTSELECTPATH and LASTSELECTNAME */
```

```
ADDRESS "M2IPort.Rexx"  
OPTIONS RESULTS
```

LOCKUSER

```
RESULT=""  
NAME=""
```

LASTSELECTPATH

```
IF RESULT~="" THEN NAME=RESULT"/"

LASTSELECTNAME
NAME=NAME""RESULT

IF NAME="" THEN ERROR "No selection"
ELSE MESSAGE "Selection : "NAME

UNLOCKUSER

EXIT
```

See LASTSELECTPATH

1.68 DELETE

```
DELETE "Search-icon"
```

Delete the specified icon.

If "Search-icon" is a Drawer, the complete tree is deleted.

Return : 0 success

Result : Number of deleted icons

NOTE : If a Drawer is deleted, Result doesn't count all deleted icons.

1.69 WBINFO

```
WBINFO "Search-icon"
```

Call the Workbench function "Information" for each Tool icon.

Return : 0 function called
10 Drawer in parameter

Result : Number of times the procedure have been called
by "Search-icon"

NOTE : Same as the internal command "WBINFO" or the item "WB Info" in menu "Icons".

1.70 AUTOSCAN

AUTOSCAN "Search-icon"

Scan the directory specified in the directory gadget of an "AutoScan" drawer, or, if your parameter is a Tool, its Drawer.

Retour : 0 success

Result : Number of tools or drawers created or 0

NOTE : If a drawer already has icons or if the window is opened, this command has no effect.

See OPENDRW "Search-icon"
