in

Conversion program

COLLABORATORS			
	TITLE : in		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Conversion program	October 9, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

in

1.1 RIARexx Library

```
                 .----------------------------------------------------------------------------
|                          RI ARexx Library   V2.5                               |
`----------------------------------------------------------------------------'

                        ©1996 Red When Excited Ltd

       Undocumented commands added by Toby Zuijdveld 02/03/1999.
                      mailto: hotcakes@abacus.net.au


                     Overview
                                                  Command Index


                 RXCreatePort

                  RXErrorCode

                 RXDeletePort

                  RXSendCommand

                 RXCreateMsg

                  RXExecuteScript

                 RXSetArg
                                   RXSetMsgType

                 RXReadPort

                  RXGetResult

                 RXWaitPort

                  RXWaitResult
```

1.2 Whats it all about?

Overview

The RI ARexx Library is designed to allow the Blitz application programmer to add an ARexx port to their program. It provides simple commands for creating an ARexx port, sending ARexx commands to other ports, and reading the ARexx port for incoming commands.

1.3 rxcreateport

```
Function: RXCreatePort
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : msgport.l=RXCreatePort(portname$)
```

This function attempts to create a public message port with the name <portname$>. It returns a pointer to the message port or 0 if it fails. Failure is usually due to a public message port existing with the same name. This must be called before any other RIARexxLib commands.

1.4 rxdeleteport

```
Statement: RXDeletePort
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : RXDeletePort
```

This command removes and free’s your message port so that no further messages can arrive. It will wait until all replies (which want a result) have been received and replies to all new messages with the error code HOSTNOTFOUND.

1.5 rxcreatemsg

```
Function: RXCreateMsg
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : rexxmsg.l=RXCreateMsg
```

This function attempts to allocate memory for an ARexx message. It returns a pointer to the message or 0 for failure. It will fail if no message port has been opened, or, if there is insufficient memory.

This must be called before calling RXCreateArg, RXSendMsg, RXFreeArgs, RXFreeArg, RXFreeArgsWait

1.6 rxsetarg

```
Statement/Function: RXSetarg
-----------------------------------------------------------------------------
Modes  : Amiga
Synrax : [success=]RXSetArg([rexxmsg,]argnumber,arg$)
```

This command attempts to store the argument <arg$> into the message argument slot specified by argnumber. Valid argument numbers are 1-16.

This command will return -1 for success, or 0 for failure.

1.7 rxreadport

```
Function: RXReadPort
-----------------------------------------------------------------------------
Modes  ; Amiga
Syntax : rexxmsg=RXReadPort
```

This function reads the message port created with RXCreatePort to see if a message is waiting. If there is, it returns a pointer to the message else 0. The return value from this function should be used with RXGetArg, RXReplyMsg, RXSendersName, RXNumArgs

1.8 rxwaitport

```
Function: RXWaitPort
------------------------------------------------------------------------------
Modes  ; Amiga
Syntax : rexxmsg=RXWaitPort

  This function waits at the message port created with RXCreatePort, for a
message to arrive. It returns a pointer to the message else 0 (no port).
The return value from this function should be used with RXGetArg,
RXReplyMsg, RXSendersName, RXNumArgs
```

1.9 rxgetarg

```
Function: RXGetArg
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : arg$=RXGetArg([rexxmsg,]argnum)

  This function returns the <argnum> argument from a message read using
RXReadMsg/RXWaitMsg. The return result is a string.
```

1.10 rxnumargs

```
Function: RXNumArgs
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : numargs=RXNumArgs([rexxmsg])

  This function will return the number of arguments contained in a message
received through RXReadPort/RXWaitPort.
```

1.11 rxreplymsg

```
Statement: RXReplyMsg
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : RXReplyMsg errorcode[,return$]

  This statement will reply to a message recevied via RXReadPort/RXWaitPort
signalling to the sender that he/she can free the memory for it.

errorcode is the errorcode to return. 0 for no error

The optional return$ parameter is a result string to be returned.

e.g.

   address 'ThatPort'
   RESULT=MYAREXXCOMMAND("MY PARAMETER")
   SAY(RESULT)

The variable RESULT would contain the string <return$>

NOTE: You should extract any information from the recevied rexxmsg BEFORE
      you reply to it.
```

1.12 rxsendmsg

```
Statement/Function: RXSendMsg
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : [success=]RXSendMsg([rexxmsg,]portname$)

  This command will attempt to send the message created with RXCreateMsg
to the public message port specified in <portname$>. It will return -1
if all was ok, else 0. (eg. couldnt find port, no such rexxmsg)
```

1.13 rxsendersname

```
Function: RXSendersName
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : portname$=RXSendersName

  This function will return the mesage port name of the task which sent
you the message read by RXReadPort/RXWaitPort.
```

1.14 rxerrorcode

```
Function: RXErrorCode
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : errorcode=RXErrorCode[(rexxmsg)]

  If rexxmsg is not given, this function returns the error code number for
the most recent message sent with RXRunCommand/RXRunScript/RXSendCommand
RXExecuteScript

If rexxmsg is given, this function returns the error code number for
the given rexxmsg.

e.g.

If RXSendCommand("REXX","SAY('Hello')"="" Then NPrint "Error "+Str$(RXErrorCode)
```

1.15 rxsendcommand

```
Statement/Function: RXSendCommand
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : [result$=]RXSendCommand([port$,]command$)

  This function is a quick way of sending ARexx messages.

If port$ is omitted, this will send the message to the default port
specified with RXDefaultPort (or REXX if no default port is set)

command$ should be a valid arexx command. Multiple commands are available
by separating with a ;

Single quotes (’) can be used instead of Chr$(34) to save typing

e.g.

            RXSendCommand("REQUEST ’Please’ ’Insert Volume’ ’Yes|No’)
 instead of RXSendCommand("REQUEST "+Chr$(34)+"Please"+Chr$(34)+" "+Chr.....

This function returns a result string. NULL string for error or an
empty return string.
```

1.16 rxexecutescript

```
Statement/Function: RXExecuteScript
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : [result$=]RXExecuteScript(scriptname$)

  This command tells ARexx to execute the ARexx script <scriptname$>. The
result string is the result returned by ARexx.
```

1.17 rxruncommand

```
Statement: RXRunCommand
-----------------------------------------------------------------------------
Modes  : Amiga
Syntax : RXRunCommand command$

  This function is a quick way of sending ARexx messages. It is identical
to RXSendCommand except that:

  a) It does not return a result string
  b) It does not wait for a reply
```

1.18 rxrunscript

```
Statement: RXRunScript
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : RXRunScript scriptname$
```

This command tells ARexx to execute the ARexx script <scriptname$>. It is identical to RXExecuteScript except that:

```
  a) It does not return a result string
  b) It does not wait for a reply
```

1.19 rxgetresult

```
Function: RXGetResult
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : result$=RXGetResult
```

This function will look at your message port for a reply to the last message sent and return the result string.

1.20 rxwaitresult

```
Function: RXWaitResult
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : result$=RXGetResult
```

This function will wait for a reply to the last message sent and return the result string.

1.21 rxmsgscript

```
Statement/Function: RXMsgScript
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : [suc=]RXMsgScript rexxmsg
```

This command sends a previously created rexxmsg to REXX telling it to execute the first argument as a script file.

e.g.

```
  rexxmsg.l=RXCreateMsg
        RXCreateArg rexxmsg,1,"RAM:TEST.rexx"
        RXMsgScript rexxmsg
```

1.22 rxresults

```
Statement: RXResults
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : RXResults On|Off
```

This command lets you choose whether the ARexx commmands which return a result will actually look for a result or not.

e.g.

```
     RXResults Off : RXSendCommand("BLAH")
```

It will not wait for a result. When the message is replied to, the message is automatically free’d and no result is taken.

1.23 rxpriority

```
Statement: RXPriority
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : RXPriority pri
```

This command lets you specify a priority for your function host (the higher the value, the earlier your port is passed a function). This must be called BEFORE RXCreatePort

1.24 rxcomresult

```
Function: RXComResult
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : RXComResult
```

This function determines whether or not the last RX______ Command was successfull or not. 0 indiciates success, -1 indicates failure.

1.25 rxdefaultport

```
Statement: RXDefaultPort
------------------------------------------------------------------------------
Modes  : Amiga
Syntax : RXDefaultPort portname$
```

This command lets you specify the name of the default port for commands which are not given a port name. The default default :-) portname is REXX

1.26 examples

```
                                    Example
```

1.27 Example Programs

```
                              Example Programs

EXAMPLE 1 - An example of ARexx :

   Load Example 1
   Compile It!
```