rexxtricks

COLLABORATORS

	TITLE : rexxtricks	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

rexxtricks

1.1 rexxtricks.guide

```
                ##########################################################################


                    'rexxtricks.library' 38.6 (12.4.96)

   Copyright (C) 1994,95 Jürgen Kohrmeyer, <J_Kohrmeyer@wilam.north.de>

##########################################################################

   'rexxtricks.library' is an ARexx function library. The functions cannot
be assigned to a special purpose, it's just a collection of functions I
missed when programming with ARexx. Let me know if you have any ideas for
other useful functions.


                Copyright
                                Copyright notice

                History
                                  History of rexxtricks.library

                Credits
                                  Thanks must go to...

                Author
                                   Where to send comments, bug reports etc.

                Requirements
                              68040, 18MB Ram ;-)

                Installation
                              How to install rexxtricks.library

                Library functions
                         Summary of all functions

                The VIEWLIST() window
```

1.2 rexxtricks.guide/Copyright

Copyright notice
****

'rexxtricks.library' is FREEWARE, but still Copyright by Jürgen Kohrmeyer. The archive may be freely distributed for non-commercial purposes, as long as all files are present and have not been modified. The archive may be uploaded to any bulletin board systems or FTP servers. You can include the archive on Public Domain disks if you don't ask for more than a fee of US $3 or 3 DM. This limit applies especially to German Public-Domain dealers! It's allowed to include the archive on a CD-ROM if the price of the CD is not more than US $30 or 30 DM.

All files in this archive are provided "as is" without warranty of any kind, either expressed or implied. The user assumes all risks and responsibilities related to its use. The author cannot be made liable for any damage resulting from the use of 'rexxtricks.library' and the other files in this archive.

Programs which are included in the archive:

The ARexx program 'FindGUI' uses the program 'Find'.
'FindGUI' is Copyright (C) by Nils Görs.
'Find' is Copyright (C) by Ralph Seichter.

The Arexx program 'RDBBackup' uses the program 'ReadRDB'.
'ReadRDB' is Copyright (C) by Gérard Cornu.

'FindGUI' and 'RDBBackup' use the 'triton.library'.
The 'triton.library' is Copyright (C) by Stefan Zeiger.

Programs which are not included in the archive:

The functions 'WHATIS' and 'WHATISTYPES' use the 'whatis.library'. The 'whatis.library' is Copyright (C) by Sylvain Rougier und Pierre Carrette.

The functions 'FILEID_IDENTIFY', 'FILEID_GETHIGHID', 'FILEID_GETIDSTRING' and 'FILEID_GETTYPES' use the 'FileID.library'. The 'FileID.library' is Copyright (C) by Oliver Lange.

The function 'CRC32' uses the 32 bit CRC-Table, which is also used by the ZModem file transfer protocol. This table is Copyright (C) by Gary S. Brown. The source of the function 'CRC32' is taken from the program 'crc32', which is Copyright (C) by Stephen Satchell, Satchell Evaluations and Chuck Forsberg, Omen Technology.

1.3 rexxtricks.guide/History

```
Changes since version 38.4
**************************

V38.5
        * New function 'READLINES'

        * New function 'WRITELINES'

        * New function 'STEMCOPY'

        * New function 'STEMREMOVE'

        * New function 'RAND'

        * New function 'READCLIPBOARD'

        * New function 'WRITECLIPBOARD'

        * New function 'SCSI_MANUFACTURER'

        * New function 'SCSI_PRODUCT'

        * New function 'SCSI_REVISION'

        * New function 'SCSI_DEVICETYPE'

        * New function 'SCSI_TESTREADY'

V38.6
        * English documentation

        * Fixed a bug in the function 'VIEWLIST', MouseBlankers didn't work
          if the window was active

        * New options for function 'QSORT'

        * From now on the function 'SEARCHPATTERN' stores the found line in
          the ARexx variable 'RESULT'

        * New function 'FILEID_IDENTIFY'

        * New function 'FILEID_GETHIGHID'

        * New function 'FILEID_GETIDSTRING'

        * New function 'FILEID_GETTYPES'

        * New function 'UUDECODE'

        * New function 'UUENCODE'
```

```
        * New function 'STEMINSERT'

        * New function 'GETDIR'
```

1.4 rexxtricks.guide/Credits

```
Credits
*******

Thanks must go to:

"Gérard Cornu" for his excellent program 'ReadRDB'.

"Hermann 'Uso' Doerries" for his great work and 'Wilam', the best mailbox
and newsserver. :-)

"Nils Görs" for 'FindGUI' and all the other scripts, ideas,
bug reports, beta testing ...

"Ralph Seichter" for his excellent program 'Find'.

"Stefan Zeiger" for 'Triton', one of the best GUI-Layout systems.

All other people, who send me suggestions or bug reports.
```

1.5 rexxtricks.guide/Author

```
Author
******

   You can reach the author at the following addresses, please use e-mail
if possible:

Mail:
          Jürgen Kohrmeyer
          Oststraße 2
          49143 Bissendorf

          GERMANY

Voice:
          +49-5402-5195

E-Mail:
          j_kohrmeyer@wilam.north.de

Support-Mailbox:
          WILAM Mailboxsystem Wildeshausen

          Port 1 - +49-4431-92081 : USR DualStandard V.34
```

```
          Port 2 - +49-4431-92082 : ZyXel 19.2k
          Port 3 - +49-4431-92082 : ISDN X.75

          Login with username MD, download from menu
```

1.6 rexxtricks.guide/Requirements

```
Requirements
************
```

The 'rexxtricks.library' requires Kickstart and Workbench 2.04. Before using the library, you must start the ARexx-Interpreter RexxMast.

The functions 'WHATIS()' and 'WHATISTYPES()' require 'whatis.library' V4.0+ in your LIBS: directory.

You can get 'whatis.library' from many BBS systems, version 4.0 of the library can be found on Fred Fish's 'AmigaLibDisk 995'.

The 'FILEID_xx()' functions require 'FileID.library' V7.0+ in your LIBS: directory.

'FileID.library' V7.0 can be found on the 'Meeting Pearls Vol. III' CD-ROM.

1.7 rexxtricks.guide/Installation

```
Installation
************
```

Installation is easy, just copy 'rexxtricks.library' to LIBS:. Deutsch is the builtin language of 'rexxtricks.library', if you want to use an other language copy the file rexxtricks.catalog of the language to LOCALE:catalogs. Only english is supported at this time.

To use 'rexxtricks.library' and ARexx, you have to start the ARexx interpreter first. If this isn't installed in your system, add the following line to S:User-Startup:

```
SYS:System/RexxMast >NIL:
```

1.8 rexxtricks.guide/Library functions

```
                Functions of 'rexxtricks.library' 38.6 (12.4.96)
*********************************************
```

AmigaDOS-functions

GETENV()
Get environment variable

SETENV()
Set environment variable

UNSETENV()
Remove environment variable

PATHPART()
Extract dirname from path

FILEPART()
Extract filename from path

MAKEPATH()
Append a filename to the end of a path

SUFFIXPART()
Get the suffix of a filename

MAKESUFFIX()
Append a suffix to the end of a filename

GETCOMMENT()
Get comment of a file

SETCOMMENT()
Set comment of a file

GETPROTECTION()
Get protection flags of a file

SETPROTECTION()
Set protection flags of a file

MATCHPATTERN()
Check for a pattern match with a string

SEARCHPATTERN()
Search for pattern in a textfile

GETKEY()
Wait for a key at console window

GETDIR()
Read directory into a compound variable

READFILE()
Read a textfile into a compound variable

WRITEFILE()
Write contents of a compound variable to a textfile

READLINES()
Read a part of a textfile into a compound variable

WRITELINES()
Replace or insert lines in a textfile

SCSI-functions

SCSI_DEVICETYPE()
Get the type of a SCSI peripheral, DISK, TAPE etc.

SCSI_MANUFACTURER()
Get the manufacturer of a SCSI peripheral

SCSI_PRODUCT()
Get the product string of a SCSI peripheral

SCSI_REVISION()
Get the revision string of a SCSI peripheral

SCSI_TESTREADY()
Test whether a SCSI peripheral is ready or not

Clipboard-functions

READCLIPBOARD()
Read text from clipboard

WRITECLIPBOARD()
Write text to clipboard

List-functions

QSORT()
Sort list with QuickSort

BSEARCH()
Search string with Binary Search

LSEARCH()
Search string with Linear Search

```
                STEMCOPY()
                             Copy elements of a compound variable

                STEMINSERT()
                           Insert elements in a compound variable

                STEMREMOVE()
                           Remove elements of a compound variable


                VIEWLIST()
                             Display list in a listview window


Publicscreen-functions


                GETDEFAULTPUBSCREEN()
                  Get name of the default public screen

                SETDEFAULTPUBSCREEN()
                  Set new default public screen


                GETPUBSCREENMODES()
                    Get current public screen modes

                SETPUBSCREENMODES()
                    Set new public screen modes


                PUBSCREENTOFRONT()
                     Bring a public screen to the front

                PUBSCREENTOBACK()
                      Send a public screen to the back


                PUBSCREENLIST()
                        Get list of all public screens currently open


                BEEP()
                                Beep all screens


Icon-functions


                GETTOOLTYPEVALUE()
                     Get the value of a tooltype

                SETTOOLTYPEVALUE()
                     Set the value of a tooltype


                GETTOOLTYPES()
```

```
                Get all tooltypes of an icon

        SETTOOLTYPES()
                Set all tooltypes of an icon


        GETDEFAULTTOOL()
               Get default tool of an icon

        SETDEFAULTTOOL()
               Set default tool of an icon


        GETSTACK()
                    Get the stacksize of an icon

        SETSTACK()
                    Set the stacksize of an icon


        CREATEICON()
                   Create a new Icon

        WBINFO()
                      Call the icon information window from workbench ( ↩
                         OS3.0+)
```

Misc functions

```
        WHATIS()
                      Recognize the type of a file using 'whatis.library ↩
                         '

        WHATISTYPES()
                  Get a list of all 'whatis.library' filetypes


        FILEID_IDENTIFY()
              Recognize the type of a file using 'FileID.library'

        FILEID_GETHIGHID()
             Get maximum filetype ID of 'FileID.library'

        FILEID_GETIDSTRING()
           Get the description of a filtype ID

        FILEID_GETTYPES()
              Get a list of all 'FileID.library' filetypes


        UUDECODE()
                    uudecode a file

        UUENCODE()
                    uuencode a file
```

```
                CRC32()
                               Calculate 32-Bit CRC checksum of a file


                COUNTCHARS()
                          Count chars in a string


                RAND()
                                Get a random number


                REXXTRICKSVERSION()
                    Get version of rexxtricks.library
```

All functions in alphabetical order

```
                BEEP()
                                Beep screens

                BSEARCH()
                             Search string with Binary Search

                COUNTCHARS()
                          Count chars in a string

                CRC32()
                                Calculate 32-Bit CRC checksum of a file

                CREATEICON()
                          Create a new Icon

                FILEID_GETHIGHID()
                     Get maximum filetype-ID of 'FileID.library'

                FILEID_GETIDSTRING()
                   Get the description of a filtype-ID

                FILEID_GETTYPES()
                      Get a list of all currently known filetypes

                FILEID_IDENTIFY()
                      Get filetype, uses 'FileID.library'

                FILEPART()
                             Extract filename from path

                GETCOMMENT()
                          Get comment of a file

                GETDEFAULTPUBSCREEN()
                  Get name of the default-pubcreen
```

```
GETDEFAULTTOOL()
       Get default tool of an icon

GETDIR()
               Read directory into a compound variable

GETENV()
               Get environment variable

GETKEY()
               Wait for a key at console window

GETPROTECTION()
        Get protection flags of a file

GETPUBSCREENMODES()
    Get current pubscreen modes

GETSTACK()
             Get the stacksize of an icon

GETTOOLTYPES()
         Get all tooltypes of an icon

GETTOOLTYPEVALUE()
     Get the value of a tooltype

LSEARCH()
              Search string with Linear Search

MAKEPATH()
             Append a filename to the end of a path

MAKESUFFIX()
           Append a suffix to the end of a filename

MATCHPATTERN()
         Check a string for pattern

PATHPART()
             Extract dirname from path

PUBSCREENLIST()
        Get list of all pubscreens currently open

PUBSCREENTOBACK()
      Move a pubscreen to the back

PUBSCREENTOFRONT()
     Move a pubscreen to the front

QSORT()
                Sort list with QuickSort

RAND()
                 Get a random number
```

```
READCLIPBOARD()
        Read text from clipboard

READFILE()
             Read a textfile into a compound variable

READLINES()
            Read a part of a textfile into a compound variable

REXXTRICKSVERSION()
    Get version of rexxtricks.library

SCSI_DEVICETYPE()
      Get type of a SCSI-Device, DISK, TAPE etc.

SCSI_MANUFACTURER()
    Get manufacturer of a SCSI-Device

SCSI_PRODUCT()
         Get product-string of a SCSI-Device

SCSI_REVISION()
        Get revision-string of a SCSI-Device

SCSI_TESTREADY()
       Test whether a SCSI-Device is ready or not

SEARCHPATTERN()
        Search for pattern in a textfile

SETCOMMENT()
           Set comment of a file

SETDEFAULTPUBSCREEN()
  Set new default pubscreen

SETDEFAULTTOOL()
       Set default tool of an icon

SETENV()
               Set environment variable

SETPROTECTION()
        Set protection flags of a file

SETPUBSCREENMODES()
    Set new pubscreen modes

SETSTACK()
             Set the stacksize of an icon

SETTOOLTYPES()
         Set all tooltypes of an icon

SETTOOLTYPEVALUE()
     Set the value of a tooltype
```

```
                STEMCOPY()
                             Copy elements of a compound variable

                STEMINSERT()
                           Insert elements in a compound variable

                STEMREMOVE()
                           Remove elements of a compound variable

                SUFFIXPART()
                           Get suffix of a filename

                UNSETENV()
                             Remove environment variable

                UUDECODE()
                             uudecode a file

                UUENCODE()
                             uuencode a file

                VIEWLIST()
                             Display list in a listview-window

                WBINFO()
                               Call the icon information window from workbench ( ↩
                                  OS3.0+)

                WHATIS()
                               Get filetype, uses 'whatis.library V4.0+'

                WHATISTYPES()
                          Get a list of all currently known filetypes

                WRITECLIPBOARD()
                       Write text to clipboard

                WRITEFILE()
                            Write contents of a compound variable to a textfile

                WRITELINES()
                           Replace or insert lines in a textfile
```

1.9 rexxtricks.guide/GETENV()

```
                Function GETENV()
*****************

NAME
     contents = GETENV(variable)

ALIAS
     contents = RXTR_GETENV(variable)
```

```
DESCRIPTION
     Returns the contents of an environment variable.

ARGUMENTS
     variable - Name of the variable

RESULT
     contents - Contents of the variable

EXAMPLE
     /* Get Kickstart-version */

     version = GETENV('Kickstart')
     SAY 'Kickstart-Version =' version

SEE ALSO


                 SETENV()
                     Set environment variable

                 UNSETENV()
                   Remove environment variable
```

1.10 rexxtricks.guide/SETENV()

```
                Function SETENV()
*****************

NAME
     boolean = SETENV(variable,text)

ALIAS
     boolean = RXTR_SETENV(variable,text)

DESCRIPTION
     Sets the contents of an environment variable. The variable will be
     created if it doesn't exist.

ARGUMENTS
     variable - Name of the variable

     text     - New contents of the variable

RESULT
     boolean  - 1 if successful, 0 for an error

EXAMPLE
     /* Set Editor-variable */

     IF SETENV('Editor','C:ED') THEN
        SAY "Variable 'Editor' is set to 'C:ED'!"
```

```
     ELSE
        SAY 'Error setting new editor!'

SEE ALSO


                 GETENV()
                     Get environment variable

                 UNSETENV()
                   Remove environment variable
```

1.11 rexxtricks.guide/UNSETENV()

```
                Function UNSETENV()
*******************

NAME
     boolean = UNSETENV(variable)

ALIAS
     boolean = RXTR_UNSETENV(variable)

DESCRIPTION
     Removes an enviranment variable. The variable will be deleted in ENV:,
     but NOT in ENVARC:!

ARGUMENTS
     variable - Name of the variable

RESULT
     boolean  - 1 if successful, 0 for an error

EXAMPLE
     /* Remove the variable 'Editor' */

     IF UNSETENV('Editor') THEN
        SAY "Variable 'Editor' is removed!"
     ELSE
        SAY 'Error removing variable!'

SEE ALSO


                 GETENV()
                   Get environment variable

                 SETENV()
                   Set environment variable
```

1.12 rexxtricks.guide/PATHPART()

```
                Function PATHPART()
*******************

NAME
     dir = PATHPART(path)

ALIAS
     dir = RXTR_PATHPART(path)

DESCRIPTION
     Returns the directory comopnent of a path.

ARGUMENTS
     path - An AmigaDOS path

RESULT
     dir  - The directory component of the specified path

EXAMPLE
     /* Get the directory component of a path */

     dir = PATHPART('DH0:Daten/Testfile')
     SAY 'Directory =' dir

SEE ALSO


                 FILEPART()
                   Extract filename from path

                 MAKEPATH()
                   Append a filename to the end of a path
```

1.13 rexxtricks.guide/FILEPART()

```
                Function FILEPART()
*******************

NAME
     filename = FILEPART(path)

ALIAS
     filename = RXTR_FILEPART(path)

DESCRIPTION
     Returns the filename of a path.

ARGUMENTS
     path     - An AmigaDOS path
```

```
RESULT
     filename - The filename of the specified path

EXAMPLE
     /* Get the filename of a path */

     filename = FILEPART('DH0:Daten/Testfile')
     SAY 'Filename =' filename

SEE ALSO


                 PATHPART()
                   Extract dirname from path

                 MAKEPATH()
                   Append a filename to the end of a path
```

1.14 rexxtricks.guide/MAKEPATH()

```
                Function MAKEPATH()
*******************

NAME
     path = MAKEPATH(dir,filename)

ALIAS
     path = RXTR_MAKEPATH(dir,filename)

DESCRIPTION
     Appends a filename to the end of a path.

ARGUMENTS
     dir      - Directory path to add the 'filename' to

     filename - Filename, directory name or subpath to add

RESULT
     path     - An AmigaDOS path that consists of the specified
                'dir' and 'filename'.

EXAMPLE
     /* Add a filename to the end of a path */

     path = MAKEPATH('DH0:Daten','Testfile')
     SAY 'Path =' path

SEE ALSO


                 PATHPART()
                   Extract dirname from path
```

```
                FILEPART()
                  Extract filename from path
```

1.15 rexxtricks.guide/SUFFIXPART()

```
                Function SUFFIXPART()
*********************

NAME
     suffix = SUFFIXPART(path)

ALIAS
     suffix = RXTR_SUFFIXPART(path)

DESCRIPTION
     Returns the suffix of the file component of a path.

ARGUMENTS
     path   - An AmigaDOS path which last component is a filename

RESULT
     suffix - The suffix of the filename without the point, e.g. the
              suffix of 'DH0:Test.lha' is 'lha'!

EXAMPLE
     /* Get the suffix of a filename */

     suffix = SUFFIXPART('DH0:Daten/Testfile.txt')
     SAY 'Suffix is =' suffix

SEE ALSO


                MAKESUFFIX()
                 Append a suffix to the end of a filename
```

1.16 rexxtricks.guide/MAKESUFFIX()

```
                Function MAKESUFFIX()
*********************

NAME
     new_path = MAKESUFFIX(path,suffix,mode)

ALIAS
     new_path = RXTR_MAKESUFFIX(path,suffix,mode)

DESCRIPTION
```

```
     Appends a suffix to the end of the file component of a path, or
     replaces an old suffix with the specified one.

ARGUMENTS
     path     - An AmigaDOS path which last component is a filename

     suffix   - The new suffix for the filename, without point!

     mode     - Specifies whether the suffix should be appended to the
                filename or an existing suffix should be replaced.

                Possible keywords are:

                'APPEND'  or 'A' - Append the suffix to the filename

                'REPLACE' or 'R' - Replace an existing suffix

RESULT
     new_path - Pathname with the new suffix. A filename that would be
                longer than 30 characters is cut before the new suffix
                is appended.

EXAMPLE
     /* Replace a suffix */

     jpegfile = MAKESUFFIX('DH0:gfx/picture.IFF','JPEG','REPLACE')
     SAY 'The new filename is' jpegfile

SEE ALSO


                  SUFFIXPART()
                   Get the suffix of a filename
```

1.17 rexxtricks.guide/GETCOMMENT()

```
                 Function GETCOMMENT()
*********************

NAME
     comment = GETCOMMENT(filename)

ALIAS
     comment = RXTR_GETCOMMENT(filename)

DESCRIPTION
     Returns the comment string of a file or directory.

ARGUMENTS
     filename - File or directory to get the comment of.

RESULT
     comment  - The comment of the specified file or directory.
```

```
EXAMPLE
     /* Get comment of a file */

     comment = GETCOMMENT('DH0:Data/Testfile')
     SAY 'Comment =' comment

SEE ALSO

                  SETCOMMENT()
                    Set comment of a file
```

1.18 rexxtricks.guide/SETCOMMENT()

```
                Function SETCOMMENT()
*********************

NAME
     boolean = SETCOMMENT(filename,comment)

ALIAS
     boolean = RXTR_SETCOMMENT(filename,comment)

DESCRIPTION
     Sets the comment string of a file or directory.

ARGUMENTS
     filename - The file or directory, which comment should be set.

RESULT
     boolean  - 1 if successful, 0 for an error

EXAMPLE
     /* set comment on a file */

     IF SETCOMMENT('DH0:Data/Testfile','This is a testfile') THEN
        SAY 'New comment is set!'
     ELSE
        SAY 'Error setting the new comment!'

SEE ALSO

                  GETCOMMENT()
                    Get comment of a file
```

1.19 rexxtricks.guide/GETPROTECTION()

```
                Function GETPROTECTION()
************************

NAME
     flags = GETPROTECTION(filename)

ALIAS
     flags = RXTR_GETPROTECTION(filename)

DESCRIPTION
     Returns the protection flags of a file.

ARGUMENTS
     filename - Name of the file

RESULT
     flags    - A string that contains the protection flags in the
                format 'HSPARWED'. An uppercase character indicates
                that the flag is set, a '-' indicates that the flag
                is not set.

EXAMPLE
     /* Get protection flags of a file */

     flags = GETPROTECTION('DH0:Testfile')
     SAY 'Protection flags =' flags

SEE ALSO


                 SETPROTECTION()
                   Set protection flags of a file
```

1.20 rexxtricks.guide/SETPROTECTION()

```
                Function SETPROTECTION()
************************

NAME
     boolean = SETPROTECTION(flags)

ALIAS
     boolean = RXTR_SETPROTECTION(flags)

DESCRIPTION
     Sets the protection flags of a file.

ARGUMENTS
     flags    - A string that contains the protection flags in the
                format 'HSPARWED'. An uppercase character indicates
                that the flag should be set, a '-' indicates that
                the flag should not be set.
```

```
RESULT
     boolean  - 1 if successful, 0 for an error

EXAMPLE
     /* Set protection flags of a file */

     IF SETPROTECTION('DH0:Testfile','-S-RWE-') THEN
        SAY 'Protection flags are set!'
     ELSE
        SAY 'Cannot set protection flags!'

SEE ALSO

                 GETPROTECTION()
                   Get protection flags of a file
```

1.21 rexxtricks.guide/MATCHPATTERN()

```
                Function MATCHPATTERN()
************************

NAME
     boolean = MATCHPATTERN(string,pattern,case)

ALIAS
     boolean = RXTR_MATCHPATTERN(string,pattern,case)

DESCRIPTION
     Checks for a pattern match with a string.

ARGUMENTS
     string  - String to match against pattern

     pattern - AmigaDOS pattern

     case    - Specifies whether the string should be matched
               case-sensitive against the pattern or not.

               Possible keywords are:

               'CASE'   or 'C' - case-sensitive

               'NOCASE' or 'N' - case-insensitive

               If this argument is omitted, the string is matched
               case-insensitive.

RESULT
     boolean - 1 if the string matches pattern, 0 if not or for an error

EXAMPLE
```

```
     /* Check for a pattern match with a string */

     IF MATCHPATTERN('Testtext','t#?','CASE') THEN
        SAY 'The string matches pattern'
     ELSE
        SAY 'No match found'

SEE ALSO


                 SEARCHPATTERN()
                   Search for pattern in a textfile
```

1.22 rexxtricks.guide/SEARCHPATTERN()

```
                Function SEARCHPATTERN()
************************

NAME
     result = SEARCHPATTERN(filename,pattern,start,mode,case)

ALIAS
     result = RXTR_SEARCHPATTERN(filename,pattern,start,mode,case)

DESCRIPTION
     Reads a textfile and checks for a pattern match with each line of the
     file.

ARGUMENTS
     filename - Name of the textfile

     pattern  - AmigaDOS pattern

     start    - Specifies the position within the file to begin with
                the pattern match. This can be the number of a line or
                a byte offset, depending on the 'mode' argument.

     mode     - Specifies the type of the 'start' argument and the
                'result'.

                Possible keywords are:

                'BYTE' or 'B' - 'start' and 'result' are byte offsets
                                from the beginning of the file. An offset
                                of 0 specifies the first byte of the file.

                'LINE' or 'L' - 'start' and 'result' are line numbers.
                                A number of 1 specifies the first line of
                                the file.

                If you have specified 'BYTE', the 'result' is the byte
                offset of the line that matches pattern. You can read
                this line using the ARexx function 'SEEK()' to move the
```

```
                file pointer to the beginning of the line and then read
                it with 'READLN()'.

                If this argument is omitted, 'start' and 'result' are
                line numbers.

     case     - Specifies whether the lines of the file should be
                matched case-sensitive against the pattern or not.

                Possible keywords are:

                'CASE'   or 'C' - case-sensitive

                'NOCASE' or 'N' - case-insensitive

                If this argument is omitted, the lines are matched
                case-insensitive.

RESULT
     result   - Line number or byte offset of the found line, depending
                on the 'mode' argument, or -1 if no match is found.

                The found line is stored in the variable 'RESULT'.

EXAMPLE
     /* Search for pattern in a textfile */

     line = SEARCHPATTERN('S:User-Startup','Assign#?MAILFILTER:',1,'LINE','NOCASE ↩
        ')

     IF line ~= -1 THEN
        SAY 'MailFilter assign found at line:' line
     ELSE
        SAY 'Sorry, no MailFilter assign found.'

SEE ALSO


                 MATCHPATTERN()
                   Check for a pattern match with a string
```

1.23 rexxtricks.guide/GETDIR()

```
Function GETDIR()
*****************

NAME
     boolean = GETDIR(dirname,pattern,stemvar,type,result,subdirs)

ALIAS
     boolean = RXTR_GETDIR(dirname,pattern,stemvar,type,result,subdirs)

DESCRIPTION
```

```
     Reads a directory and stores all entries in a compound variable.

ARGUMENTS
     dirname - Name of the directory to read

     pattern - An AmigaDOS Pattern. Only entries matching this pattern
               are stored in the compound variable. This is optional, all
               entries are stored if no pattern is specified.

     stemvar - Compound variable to store the directory entries:

               stemvar.0 - the number of entries
               stemvar.1 - the first entry
               stemvar.2 - the second entry
               stemvar.n - the n'th entry

     type    - Type of the entries, that should be stored in the
               compound variable.

               Possible keywords are:

               'ALL'   or 'A' - All entries are stored

               'FILES' or 'F' - Only files are stored

               'DIRS'  or 'D' - Only directories are stored

               If this is omitted, all entries are stored.

     result  - Specifies whether the complete path or only the name of
               the entries should be stored in the compound variable.

               Possible keywords are:

               'NAME' or 'N' - Only the name is stored

               'PATH' or 'P' - The complete path is stored

               If this is omitted, only the name is stored.

     subdirs - Specifies whether all found subdirectories should be
               entered and read or not.

               Possible keywords are:

               'SUBDIRS' or 'S' - All found subdirectories are entered
                                  and read

               If this is omitted, only the directory specified with
               'dirname' is read.

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /*
     ** Read a directory and all subdirectories and display the
```

```
     ** complete path of all entries, don't display icons
     */

     IF GETDIR('DEVS:','~(#?.info)','stemvar','FILES','PATH','SUBDIRS') THEN DO
        SAY 'Number of entries:' stemvar.0

        DO i = 1 TO stemvar.0
           SAY stemvar.i
        END
     END

SEE ALSO
```

1.24 rexxtricks.guide/GETKEY()

```
Function GETKEY()
*****************

NAME
     key = GETKEY(timeout)

ALIAS
     key = RXTR_GETKEY(timeout)

DESCRIPTION
     Waits the specified time for a key.

ARGUMENTS
     timeout - Time to wait in Seconds

RESULT
     key     - If the user presses a key within the specified time, the
               corresponding character is returned. If no key is pressed,
               the function returns -1.

               Special keys are returned as follows:

                                  |     qualifiers
                                  |
                       key        | none    | SHIFT
                       -----------+---------+-----------
                       F1         | F1      | SHIFT F1
                       F2         | F2      | SHIFT F2
                       F3         | F3      | SHIFT F3
                       F4         | F4      | SHIFT F4
                       F5         | F5      | SHIFT F5
                       F6         | F6      | SHIFT F6
                       F7         | F7      | SHIFT F7
                       F8         | F8      | SHIFT F8
                       F9         | F9      | SHIFT F9
                       F10        | F10     | SHIFT F10
                                  |         |
                       Esc        | ESC     | ESC
                       Help       | HELP    | HELP
                                  |         |
```

```
                    Return          | CR      | CR
                    Enter           | CR      | CR
                                    |         |
                    Backspace       | BS      | BS
                    Delete          | DEL     | DEL
                                    |         |
                    Cursor Oben     | UP      | SHIFT UP
                    Cursor Unten    | DOWN    | SHIFT DOWN
                    Cursor Links    | LEFT    | SHIFT LEFT
                    Cursor Rechts   | RIGHT   | SHIFT RIGHT

               If you have defined a string for the pressed key, e.g. with
               the commodity 'FKey', the defined string is returned.

EXAMPLE
     /* Wait ten seconds for a key */

     key = GETKEY(10)

     IF key ~= (-1) THEN
        SAY 'Key pressed:' key
     ELSE
        SAY 'No key was pressed!'

SEE ALSO
```

1.25 rexxtricks.guide/READFILE()

```
                 Function READFILE()
*******************

NAME
     boolean = READFILE(filename,stemvar)

ALIAS
     boolean = RXTR_READFILE(filename,stemvar)

DESCRIPTION
     Reads the contents of a textfile into a compound variable. Each line is
     stored as an element of the variable, all linefeeds (ASCII code: 10)
     are removed automatically.

ARGUMENTS
     filename  - File to read from

     stemvar   - Compound variable to store the contents of the textfile,
                 each element will contain one line without linefeed:

                 stemvar.0 - the number of lines
                 stemvar.1 - the first line
                 stemvar.2 - the second line
                 stemvar.n - the n'th line

RESULT
     boolean   - 1 if successful, 0 for an error
```

```
EXAMPLE
     /* Read 's:user-startup' into a compound variable and display all lines */

     IF READFILE('s:user-startup','array') THEN DO
        SAY 'Number of lines:' array.0

        DO i = 1 TO array.0
           SAY 'Zeile' i||':' array.i
        END
     END

SEE ALSO


                 WRITEFILE()
                   Write contents of a compound variable to a textfile
```

1.26 rexxtricks.guide/WRITEFILE()

```
                 Function WRITEFILE()
********************

NAME
     boolean = WRITEFILE(filename,stemvar,append)

ALIAS
     boolean = RXTR_WRITEFILE(filename,stemvar,append)

DESCRIPTION
     Writes the contents of a compound variable to a file. A linefeed
     (ASCII code: 10) is appended to each element of the variable.

ARGUMENTS
     filename   - File to use for writing

     stemvar    - Compound variable which should be written to the
                  file, this variable must have the following format:

                  stemvar.0 - must be the number of lines
                  stemvar.1 - must be the first line
                  stemvar.2 - must be the second line
                  stemvar.n - muste be the n'th line

     append     - With this switch you can specify that teh contents of
                  'stemvar' should be appended to the file. In this case,
                  the file must exist!

                  Possible keywords:

                  'APPEND' or 'A' - Append 'stemvar' to an existing file.

                  All other words are ignored.
```

```
RESULT
     boolean    - 1 if successful, 0 for an error

EXAMPLE
     /* Write contents of a compound variable to a file */

     list.0 = 4
     list.1 = 'Hello'
     list.2 = 'Good morning'
     list.3 = 'Bye bye'
     list.4 = 'Nice to see you'

     IF WRITEFILE('RAM:Test','list') THEN
        SAY 'Okay, no errors!'

SEE ALSO


                 READFILE()
                   Read a textfile into a compound variable
```

1.27 rexxtricks.guide/READLINES()

```
                Function READLINES()
********************

NAME
     boolean = READLINES(filename,start,number,format,stemvar)

ALIAS
     boolean = RXTR_READLINES(filename,start,number,format,stemvar)

DESCRIPTION
     Reads a part of a textfile into a compound variable. The lines are
     read and converted using the specified format string. All linefeeds
     are removed before converting.

ARGUMENTS
     filename - File to read from

     start    - Number of the first line to read, the minimal line
                number is 1. If this argument is omitted, the function
                will read from beginning of the file.

     number   - Number of lines to read. If this argument is omitted,
                the function will read until end of the file.

     format   - Formatstring which describes how to convert each line.
                This string can contain as many conversion specifications
                as you like. You must specify a variable for each of the
                conversion specifications. (see argument 'stemvar')
```

The maximum length for each of the converted elements is 1024 characters! The total length of one line is 4096 characters!

When a line is read, the formatstring is searched for the first conversion specification. If found, a part of the line is converted and stored as an element of the first compound variable. After this, the next part of the line is converted using the next conversion specification, and stored as an element of the next compound variable. etc.

The formatstring may contain three different types of characters and instructions:

1. white space (space, tab and linefeed characters)

   Any white space characters cause the conversion to continue with the next character in the input line that is not white space.

2. all other characters without the percent sign '%'

   For each of this characters, a matching character is read from the input line. If there is not an exact match, the conversion stops at this point.

3. conversion specifications

   A conversion specification indicates how the next part of the input line is to be converted. Each conversion specification starts with the percent sign '%' and follows this format, brackets indicate an optional part:

   %[*][width]type

   % - The percent sign introduces a conversion specification. If you want to match a '%' in the input line, use '%%' in the format string.

   * - The asterisk means that the conversion should be performed, but the result should not be stored. Do not specify a variable for a conversion specification that uses this.

   width - A decimal number which specifies the maximum number of characters to be converted. If 'type' is 'c' exact 'width' characters are converted, the conversion will not stop at a white space character,

   type - Specifies the type of the conversion, two different types are supported:

   s - Indicates a character string terminated

```
                                by white space or end of the input line.
                                Also, the conversion stops if maximum
                                width is specified and 'width' characters
                                are converted.

                            c - Indicates conversion of one character.
                                If maximum width is specified, exact
                                'width' characters will be converted.
                                The conversion doesn't stop at white
                                space characters.

     stemvar  - Compound variables to store the converted parts of the
                input line, for each conversion specification you must
                specify a variable. The variable names are specified
                separated by a 'space' or 'tab':

                Example: READLINES(file,10,3,'%s %5c','stemvar1 stemvar2')

                stemvar1.0 - will be the number of lines,
                             at this example it will be 3
                stemvar1.1 - will be the first item of the first line,
                             at this example the first item of line 10
                stemvar1.2 - will be the first item of the second line,
                             at this example the first item of line 11
                stemvar1.3 - will be the first item of the third line,
                             at this example the first item of line 12

                stemvar2.0 - will be the number of lines,
                             at this example it will be 3
                stemvar2.1 - will be the second item of the first line,
                             at this example the second item of line 10
                stemvar2.2 - will be the second item of the second line,
                             at this example the second item of line 11
                stemvar2.3 - will be the second item of the third line,
                             at this example the second item of line 12

RESULT
     boolean  - 1 if successful, 0 for an error

EXAMPLE
     /*
     ** Create a file using the DOS command 'List', read a part of
     ** the file into compound variables and display the results.
     */

     ADDRESS COMMAND 'List >T:tempfile LFORMAT "%-30N %L Byte %A %D %T"'

     IF READLINES('T:tempfile',5,20,'%30c %s Byte %s %s %s','name bytes flags  ↩
        filedate filetime') THEN DO
        SAY 'Number of lines:' name.0
        SAY

        DO i = 1 TO name.0
           IF bytes.i = 'Dir' THEN
              SAY 'No.' i '-> Directory:' STRIP(name.i)
           ELSE
              SAY 'No.' i '-> File:' STRIP(name.i) '-' bytes.i 'Byte'
```

```
           SAY 'Flags:' flags.i
           SAY 'Date :' filedate.i
           SAY 'Time :' filetime.i
           SAY
        END
     END

SEE ALSO


                 WRITELINES()
                   Replace or insert lines in a textfile
```

1.28 rexxtricks.guide/WRITELINES()

```
                Function WRITELINES()
*********************

NAME
     boolean = WRITELINES(filename,start,stemvar,mode)

ALIAS
     boolean = RXTR_WRITELINES(filename,start,stemvar,mode)

DESCRIPTION
     Replaces lines of a textfile with the contents of a compound variable,
     or inserts the contents of a compound variable in a textfile.

ARGUMENTS
     filename - File to write to

     start    - Number of the first line to write, the minimal line
                number is 1. If this argument is omitted, the function
                will write to the first line of the file.

     stemvar  - Compound variable which should be written to the file,
                this variable must have the following format:

                stemvar.0 - must be the number of lines
                stemvar.1 - must be the first line
                stemvar.2 - must be the second line
                stemvar.n - must be the n'th line

     mode     - Specifies how the contents of the compound variable
                should be written to the file:

                'REPLACE' or 'R'- Lines in the file will be replaced
                                  beginning at the specified 'start'
                                  line. Each element of the compound
                                  variable replaces one line of the
                                  file. For example, if 'stemvar' has
                                  10 elements, 10 lines are replaced.
```

```
                                  All following lines are copied from
                                  the original file.

                'INSERT' or 'I' - All elements of the compound variable
                                  are inserted in the file, beginning
                                  at the specified 'start' line.

RESULT
     bool     - 1 if successful, 0 for an error

EXAMPLE
     /*
     ** Read the lines 10, 11 and 12 from 'RAM:testfile',
     ** change it and write back to the file.
     */

     IF READLINES('RAM:testfile',10,3,'%256c','line') THEN DO
        DO i = 1 TO line.0
           line.i = 'This is line' i+9 '-' line.i
        END

        IF ~WRITELINES('RAM:testfile',10,'line','REPLACE') THEN
           SAY 'Error writing to file!'
     END
     ELSE
        SAY 'Error reading from file!'

SEE ALSO


                 READLINES()
                   Read a part of a textfile into a compound variable
```

1.29 rexxtricks.guide/SCSI_DEVICETYPE()

```
                Function SCSI_DEVICETYPE()
**************************

NAME
     type = SCSI_DEVICETYPE(device,unit)

ALIAS
     type = RXTR_SCSI_DEVICETYPE(device,unit)

DESCRIPTION
     Returns the type of the SCSI peripheral connected to the specified
     device and unit.

ARGUMENTS
     device - Name of the SCSI device

     unit   - Unit number
```

```
RESULT
     type   - Type of the peripheral:

              DISK          - disk drive, hard disk
              TAPE          - tape streamer
              PRINTER       - printer
              PROCESSOR     - processor device
              WORM          - WORM drive
              CDROM         - CD-ROM drive
              SCANNER       - scanner
              OPTICAL       - optical disk drive
              MEDIACHANGER  - medium changer device
              COMMUNICATION - communications device
              UNKNOWN       - unknown device type

              If no peripheral is connected to the specified device
              and unit, an empty string '' is returned.

EXAMPLE
     /* Display the type of the peripheral connected to 'scsi.device' unit 0 */

     SAY SCSI_DEVICETYPE('scsi.device',0)

SEE ALSO


                SCSI_MANUFACTURER()
                  Get the manufacturer of a SCSI peripheral

                SCSI_PRODUCT()
                       Get the product string of a SCSI peripheral

                SCSI_REVISION()
                      Get the revision string of a SCSI peripheral

                SCSI_TESTREADY()
                     Test whether a SCSI peripheral is ready or not
```

1.30 rexxtricks.guide/SCSI_MANUFACTURER()

```
                Function SCSI_MANUFACTURER()
****************************

NAME
     manufacturer = SCSI_MANUFACTURER(device,unit)

ALIAS
     manufacturer = RXTR_SCSI_MANUFACTURER(device,unit)

DESCRIPTION
     Returns the manufacturer of the SCSI peripheral connected to the
     specified device and unit.
```

```
ARGUMENTS
     device       - Name of the SCSI device

     unit         - Unit number

RESULT
     manufacturer - The manufacturer information of the peripheral, or
                    an empty string '' if no peripheral is connected
                    to the specified device and unit.

EXAMPLE
     /* Display the manufacturer of a peripheral */

     SAY SCSI_MANUFACTURER('scsi.device',0)

SEE ALSO


                SCSI_DEVICETYPE()
                    Get the type of a SCSI peripheral, DISK, TAPE etc.

                SCSI_PRODUCT()
                       Get the product string of a SCSI peripheral

                SCSI_REVISION()
                      Get the revision string of a SCSI peripheral

                SCSI_TESTREADY()
                     Test whether a SCSI peripheral is ready or not
```

1.31 rexxtricks.guide/SCSI_PRODUCT()

```
                Function SCSI_PRODUCT()
**********************

NAME
     product = SCSI_PRODUCT(device,unit)

ALIAS
     product = RXTR_SCSI_PRODUCT(device,unit)

DESCRIPTION
     Returns the product description string of the SCSI peripheral
     connected to the specified device and unit.

ARGUMENTS
     device  - Name of the SCSI device

     unit    - Unit number

RESULT
     product - The product description string of the peripheral, or
               an empty string '' if no peripheral is connected
```

```
               to the specified device and unit.

EXAMPLE
     /* Display the product string of a peripheral */

     SAY SCSI_PRODUCT('scsi.device',0)

SEE ALSO


                 SCSI_DEVICETYPE()
                     Get the type of a SCSI peripheral, DISK, TAPE etc.

                 SCSI_MANUFACTURER()
                   Get the manufacturer of a SCSI peripheral

                 SCSI_REVISION()
                       Get the revision string of a SCSI peripheral

                 SCSI_TESTREADY()
                      Test whether a SCSI peripheral is ready or not
```

1.32 rexxtricks.guide/SCSI_REVISION()

```
                Function SCSI_REVISION()
************************

NAME
     revision = SCSI_REVISION(device,unit)

ALIAS
     revision = RXTR_SCSI_REVISION(device,unit)

DESCRIPTION
     Returns the revision string of the SCSI peripheral connected to the
     specified device and unit.

ARGUMENTS
     device   - Name of the SCSI device

     unit     - Unit number

RESULT
     revision - The revision string of the peripheral, or an empty
                string '' if no peripheral is connected to the
                specified device and unit.

EXAMPLE
     /* Display the revision string of a peripheral */

     SAY SCSI_REVISION('scsi.device',0)

SEE ALSO
```

```
                SCSI_DEVICETYPE()
                    Get the type of a SCSI peripheral, DISK, TAPE etc.

                SCSI_MANUFACTURER()
                  Get the manufacturer of a SCSI peripheral

                SCSI_PRODUCT()
                       Get the product string of a SCSI peripheral

                SCSI_TESTREADY()
                     Test whether a SCSI peripheral is ready or not
```

1.33 rexxtricks.guide/SCSI_TESTREADY()

```
                Function SCSI_TESTREADY()
*************************

NAME
     boolean = SCSI_TESTREADY(device,unit)

ALIAS
     boolean = RXTR_SCSI_TESTREADY(device,unit)

DESCRIPTION
     Checks whether a SCSI peripheral is ready for use or not. e.g.: With
     removable cartridge drives you can check if a cartridge is inserted in
     the drive.

ARGUMENTS
     device  - Name of the SCSI device

     unit    - Unit number

RESULT
     boolean - 1 if the peripheral is ready for use, 0 if not

EXAMPLE
     /* Check the peripheral at 'scsi.device' unit 0 */

     IF SCSI_TESTREADY('scsi.device',0) THEN
        SAY 'Peripheral is ready for use!'
     ELSE
        SAY 'Peripheral is not ready!'

SEE ALSO


                SCSI_DEVICETYPE()
                    Get the type of a SCSI peripheral, DISK, TAPE etc.

                SCSI_MANUFACTURER()
```

```
                  Get the manufacturer of a SCSI peripheral

                SCSI_PRODUCT()
                       Get the product string of a SCSI peripheral

                SCSI_REVISION()
                      Get the revision string of a SCSI peripheral
```

1.34 rexxtricks.guide/READCLIPBOARD()

```
                Function READCLIPBOARD()
************************

NAME
     result = READCLIPBOARD(unit,destvar)

ALIAS
     result = RXTR_READCLIPBOARD(unit,destvar)

DESCRIPTION
     Reads text from clipboard and stores each line in the specified
     compound variable, or returns the whole clipboard contents as result.

ARGUMENTS
     unit    - Clipboard-Unit to read from

     destvar - Name of a compound variable to store the clipboard
               contents. Each line will be stored in one element
               of the variable:

               destvar.0 - the number of lines
               destvar.1 - the first line
               destvar.2 - the second line
               destvar.n - the n'th line

               If destvar is not specified, the whole contents will
               be returned as result.

RESULT
     result  - If 'destvar' is not specified, result will be the whole
               clipboard contents.

               If 'destvar' is specified, result will be an empty string ".

EXAMPLE
     /* read from clipboard unit 0 and display the contents */

     SAY READCLIPBOARD(0)

     SAY '------------------'

     CALL READCLIPBOARD(0,'array')
```

```
     SAY 'Number of lines:' array.0

     DO i = 1 TO array.0
        SAY 'Line' i || ':' array.i
     END

SEE ALSO


                 WRITECLIPBOARD()
                   Write text to clipboard
```

1.35 rexxtricks.guide/WRITECLIPBOARD()

```
                Function WRITECLIPBOARD()
*************************

NAME
     boolean = WRITECLIPBOARD(unit,string,stemvar)

ALIAS
     boolean = RXTR_WRITECLIPBOARD(unit,string,stemvar)

DESCRIPTION
     Writes a string or the contents of a compound variable to the
     specified clipboard unit.

ARGUMENTS
     unit    - clipboard unit to write to

     string  - String to write to the clipboard

     stemvar - Name of a compound variable, which contents should be
               written to the clipboard. A linefeed is appended to each
               element of the variable. The compound variable must have
               the following format:

               stemvar.0 - must be the number of lines
               stemvar.1 - must be the first line
               stemvar.2 - must be the second line
               stemvar.n - must be the n'th line

               You can specify either 'string' or 'stemvar'. If 'stemvar'
               isn't specified, the 'string' will be written to clipboard.

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* write text to clipboard unit 0 */

     IF WRITECLIPBOARD(0,'This is a test') THEN
        SAY 'Okay!'
```

```
     ELSE
        SAY 'Error writing to clipboard!'

SEE ALSO


                 READCLIPBOARD()
                   Read text from clipboard
```

1.36 rexxtricks.guide/QSORT()

```
                Function QSORT()
****************

NAME
     boolean = QSORT(list,destvar,case,field,separator)

ALIAS
     boolean = RXTR_QSORT(list,destvar,case,field,separator)

DESCRIPTION
     Sort list in ascending order.

ARGUMENTS
     list      - The list to be sorted, this must be a compound variable
                 with the following format:

                 list.0 - must be the number of elements in the list
                 list.1 - must be the first element of the list
                 list.2 - must be the second element of the list
                 list.n - must be the n'th element of the list

     destvar   - Compound variable to store the sorted list. The sorted
                 list is stored similar to the source list:

                 destvar.0 - is the number of elements
                 destvar.1 - is the first element
                 destvar.2 - is the second element
                 destvar.n - is the n'th element

                 This is optional. If no 'destvar' is specified, the list
                 is sorted directly.

     case      - Specifies whether the list should be sorted case sensitive
                 or not. If this argument is omitted, the list is sorted
                 case insensitive.

                 Possible keywords:

                 'CASE'    or 'C'   - sort case sensitive
                 'NOCASE'  or 'N'   - sort case insensitive
                 'NUMERIC' or 'NUM' - sort by number
```

```
     field     - Number of a field, separated by the character which is
                 specified with the 'separator' argument. The List is
                 sorted according to this field. The number of the first
                 field is 1. If no 'field' is specified, the list is
                 sorted by the complete contents of the variable.

     separator - The character which separates the fields in each element
                 of the list. This argument is only used if a field number
                 is specified. The default separator is space ' ', it will
                 be used if a field number is specified but no separator
                 character.

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* sort 'list' and store in 'destvar' */

     list.0 = 4
     list.1 = 'Musterfrau|Sabine|Bergstraße 75|12345 Musterdorf'
     list.2 = 'Poweruser|Heinz|Megabitallee 128|87654 Mipshausen'
     list.3 = 'Mustermann|Peter|Feldweg 3|35487 Teststadt'
     list.4 = 'Kohrmeyer|Jürgen|Oststraße 2|49143 Bissendorf'

     IF QSORT('list','destvar','NUMERIC',4,'|') THEN DO
        SAY 'Number of elements:' destvar.0

        DO i = 1 TO destvar.0
           SAY destvar.i
        END
     END

     /* now sort 'list' directly */

     IF QSORT('list',,,2,'|') THEN DO
        DO i = 1 TO list.0
           SAY list.i
        END
     END

SEE ALSO


                BSEARCH()
                  Search string with Binary Search

                LSEARCH()
                  Search string with Linear Search
```

1.37 rexxtricks.guide/BSEARCH()

```
                Function BSEARCH()
******************
```

```
NAME
     number = BSEARCH(key,list,start,case)

ALIAS
     number = RXTR_BSEARCH(key,list,start,case)

DESCRIPTION
     Scan a sorted list and search for a string with Binary Search.  The
     list must be sorted in ascending order!!!

ARGUMENTS
     key     - The string to search for

     list    - The list to search, must be a compound variable with
               the following format:

               list.0 - must be the number of elements in the list
               list.1 - must be the first element of the list
               list.2 - must be the second element of the list
               list.n - must be the n'th element of the list

     start   - Index of the element to start with searching. If this
               is omitted, the search starts at element 1.

     case    - Specifies whether the search should be case sensitive or not.
               If this is omitted, the search is case insensitive.

               Possible keywords:

               'CASE'   or 'C' - search case sensitive
               'NOCASE' or 'N' - search case insensitive

RESULT
     number  - Index of the found element or -1 if nothing was found.

EXAMPLE
     /* search for a string with Binary Search */

     list.0 = 4
     list.1 = 'Hello'
     list.2 = 'Good morning'
     list.3 = 'Bye bye'
     list.4 = 'Nice to see you'

     number = BSEARCH('Good morning','list',0,'CASE')

     IF number ~= (-1) THEN
        SAY 'Found at number:' number
     ELSE
        SAY 'Not found!'

SEE ALSO


                 QSORT()
                     Sort list with QuickSort
```

```
                LSEARCH()
                  Search string with Linear Search
```

1.38 rexxtricks.guide/LSEARCH()

```
                Function LSEARCH()
******************

NAME
     number = LSEARCH(key,list,start,case,pattern)

ALIAS
     number = RXTR_LSEARCH(key,list,start,case,pattern)

DESCRIPTION
     Scan a sorted list and search for a string with Linear Search.

ARGUMENTS
     key     - The string to search for. It is possible to use any
               AmigaDOS pattern here, in this case you must specify
               the argument 'pattern'.

     list    - The list to search, must be a compound variable with
               the following format:

               list.0 - must be the number of elements in the list
               list.1 - must be the first element of the list
               list.2 - must be the second element of the list
               list.n - must be the n'th element of the list

     start   - Index of the element to start with searching. If this
               is omitted, the search starts at element 1.

     case    - Specifies whether the search should be case sensitive or not.
               If this is omitted, the search is case insensitive.

               Possible keywords:

               'CASE'   or 'C' - search case sensitive
               'NOCASE' or 'N' - search case insensitive

     pattern - Specifies, whether the 'key' argument contains an AmigaDOS
               pattern or not. If this is omitted, a normal string compare
               is done.

               Possible keywords:

               'PATTERN' or 'P' - Use AmigaDOS pattern match to search
               'STRCMP'  or 'S' - Use normal string compare to search

RESULT
     number  - Index of the next found element or -1 if nothing was found.
```

```
EXAMPLE
     /* search for a string with Linear Search */

     list.0 = 4
     list.1 = 'Hello'
     list.2 = 'Good morning'
     list.3 = 'Bye bye'
     list.4 = 'Nice to see you'

     number = LSEARCH('Bye bye','list')

     IF number ~= (-1) THEN
        SAY 'Found at number:' number
     ELSE
        SAY 'Not found!'

SEE ALSO


                 QSORT()
                     Sort list with QuickSort

                 BSEARCH()
                   Search string with Binary Search
```

1.39 rexxtricks.guide/STEMCOPY()

```
                 Function STEMCOPY()
*******************

NAME
     boolean = STEMCOPY(source,start1,dest,start2,number)

ALIAS
     boolean = RXTR_STEMCOPY(source,start1,dest,start2,number)

DESCRIPTION
     Copies the contents of a compound variable into an other compound
     variable.

ARGUMENTS
     source  - Name of the compound variable to copy from, this variable
               must have the following format:

               source.0 - must be the number of elements
               source.1 - must be the first element
               source.2 - must be the second element
               source.n - must be the n'th element

     start1  - Number of the first 'source' element to copy.

     dest    - Name of the destination variable.
```

```
     start2  - Number of the first 'dest' element to copy to.

     number  - Number of elements to copy from 'source' to 'dest'. If this
               argument is omitted, all elements, beginning at 'start1',
               are copied.

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* copy elements of a compound variable */

     list.0 = 4
     list.1 = 'Hello'
     list.2 = 'Good morning'
     list.3 = 'Bye bye'
     list.4 = 'Nice to see you'

     IF STEMCOPY('list',2,'newlist',1,2) THEN DO
        SAY 'Number of elements in newlist:' newlist.0

        DO i = 1 TO newlist.0
           SAY newlist.i
        END
     END
     ELSE
        SAY 'Error copying list!'

SEE ALSO


                  STEMINSERT()
                    Insert elements in a compound variable

                  STEMREMOVE()
                    Remove elements of a compound variable
```

1.40 rexxtricks.guide/STEMINSERT()

```
                 Function STEMINSERT()
**********************

NAME
     boolean = STEMINSERT(stemvar,start,number,default)

ALIAS
     boolean = RXTR_STEMINSERT(stemvar,start,number,default)

DESCRIPTION
     Inserts one or more elements in a compound variable.

ARGUMENTS
```

```
     stemvar - Name of the compound variable, this variable must have
               the following format:

               stemvar.0 - must be the number of elements
               stemvar.1 - must be the first element
               stemvar.2 - must be the second element
               stemvar.n - must be the n'th element

     start   - Number of the element to insert.

     number  - Number of elements to insert. If this is omitted,
               only one element is inserted.

     default - Default contents of the inserted elements. If this is
               omitted, an empty string '' is used.

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* insert 2 elements before third element */

     list.0 = 4
     list.1 = 'Hello'
     list.2 = 'Good morning'
     list.3 = 'Bye bye'
     list.4 = 'Nice to see you'

     IF STEMINSERT('list',3,2,'New') THEN DO
        SAY 'New number of elements:' list.0

        DO i = 1 TO liste.0
           SAY liste.i
        END
     END
     ELSE
        SAY 'Error inserting elements!'

SEE ALSO


                  STEMCOPY()
                      Copy elements of a compound variable

                  STEMREMOVE()
                    Remove elements of a compound variable
```

1.41 rexxtricks.guide/STEMREMOVE()

```
                  Function STEMREMOVE()
**********************

NAME
```

```
     boolean = STEMREMOVE(stemvar,start,number)

ALIAS
     boolean = RXTR_STEMREMOVE(stemvar,start,number)

DESCRIPTION
     Removes one or more elements of a compound variable.

ARGUMENTS
     stemvar - Name of the compound variable, this variable must have
               the following format:

               stemvar.0 - must be the number of elements
               stemvar.1 - must be the first element
               stemvar.2 - must be the second element
               stemvar.n - must be the n'th element

     start   - Number of the first element to remove.

     number  - Number of elements to remove. If this is omitted, only
               the 'start' element is removed.

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* remove the third element of a compound variable */

     list.0 = 4
     list.1 = 'Hello'
     list.2 = 'Good morning'
     list.3 = 'Bye bye'
     list.4 = 'Nice to see you'

     IF STEMREMOVE('list',3) THEN DO
        SAY 'New number of elements:' list.0

        DO i = 1 TO list.0
           SAY list.i
        END
     END
     ELSE
        SAY 'Error removing element!'

SEE ALSO


                 STEMCOPY()
                     Copy elements of a compound variable

                 STEMINSERT()
                   Insert elements in a compound variable
```

1.42 rexxtricks.guide/VIEWLIST()

```
                Function VIEWLIST()
*******************

NAME
     boolean = VIEWLIST(list,window,destvar)

ALIAS
     boolean = RXTR_VIEWLIST(list,window,destvar)

DESCRIPTION
     Displays a list in a listview window and allows the user to
     select one or more entries.

ARGUMENTS
     list    - A compound variable which contains the list of all
               selectable entries. This list is displayed in the
               listview window:

               list.0 - must be the number of entries in the list
               list.1 - must be the first entry of the list
               list.2 - must be the second entry of the list
               list.n - must be the n'th entry of the list

     window  - A compound variable, which describes the window:

               window.title       - The title text of the window. If
                                    this is not set, the title text is
                                    'RexxTricks-Listview'.

               window.postext     - The text for the positive (Okay)
                                    gadget. Don't use this, use
                                    window.gadgettext instead!

               window.negtext     - The text for the negative (Cancel)
                                    gadget. Don't use this, use
                                    window.gadgettext instead!

               window.gadgettext  - String which describes the action
                                    gadgets, the strings for separate
                                    gadgets must be separated by '|'.
                                    You can specify a hotkey charakter
                                    for each Gadget by preceeding the
                                    charakter to be used with '_'.

                                    For example '_Quit|_Save|_Cancel'
                                    will create 3 gadgets with the
                                    hotkeys q, s and c.

                                    If gadgettext is not set, there
                                    will be 2 gadgets: '_Okay' and
                                    '_Cancel'

               window.pubscreen   - Name of the public screen to open
                                    the window on.
```

```
                        Default: default public screen

window.font         - The font for the window, you can
                      specify whether the screen font
                      or the system default font should
                      be used.

                      DEFAULT - use system default font
                      SCREEN  - use screen font

                      If 'window.font' is not set, the
                      system default font is used.

window.left         - The left edge of the window. If
                      this is not set, the window is
                      centered on screen.

window.top          - The top edge of the window. If
                      this is not set, the window is
                      centered on screen.

window.width        - The width of the window. If this
                      is not set, the window is aligned
                      to the minimum width of the gadgets.

window.height       - The height of the window. If this
                      is not set, the height will be
                      3/4 of the screen height.

window.zoomleft     - The left edge of the zoomed window.
                      If this is not set, 'window.left'
                      is used.

window.zoomtop      - The top edge of the zoomed window.
                      If this is not set, 'window.top'
                      is used.

window.sort         - Specifies, whether the list should
                      be displayed in alphabetical order
                      or not.

                      TRUE  - display sorted list
                      FALSE - do not sort the list

                      This is only for displaying, the
                      compound variable 'list' will never
                      change! If this is not set, the list
                      is displayed in alphabetical order.

window.multiselect  - Enables or disables multiselection.

                      TRUE  - multiselection enabled
                      FALSE - multiselection disabled

                      If this is not set, multiselection
                      is enabled.
```

```
     destvar - A compound variable to store the selected entries. The
               format is similar to the compound variable 'list':

               destvar.0 - the number of selected entries
               destvar.1 - the first selected entry
               destvar.2 - the second selected entry
               destvar.n - the n'th selected entry

               When the function exits, you can read the contents with
               a 'DO i = 1 TO destvar.0' loop. If multiselection is off,
               you can find the selected entry in 'destvar.1'.

               There is a another field in this compound variable:
               -------------------------------------------------
               destvar.gadget - The number of the action gadget, on
                                which the user has clicked to close the
                                window. The gadgets are numbered from
                                the left to the right, beginning with 1.
                                The rightmost gadget (Cancel), and also
                                the close gadget of the window, sets
                                destvar.gadget to 0.

RESULT
     boolean - 1 if at least one entry was selected
               0 if no entries are selected or for an error

EXAMPLE
     /* Open listview window and display all selected entries */

     list.0 = 4
     list.1 = 'Hello'
     list.2 = 'Good morning'
     list.3 = 'Bye bye'
     list.4 = 'Nice to see you'

     window.title      = 'Please select'
     window.gadgettext = '_Yes|_Use|_Edit|_Cancel'
     window.pubscreen  = 'MICRODOT'
     window.font       = 'SCREEN'
     window.sort       = 'TRUE'

     IF VIEWLIST('list','window','destvar') THEN DO
        SAY 'Number of selected entries   :' destvar.0
        SAY 'Number of the selected gadget:' destvar.gadget

        DO i = 1 TO destvar.0
           SAY destvar.i
        END
     END

SEE ALSO


                The VIEWLIST() window
                  How to use the 'VIEWLIST' window
```

1.43 rexxtricks.guide/BEEP()

```
Function BEEP()
***************

NAME
     dummy = BEEP()

ALIAS
     dummy = RXTR_BEEP()

DESCRIPTION
     Beep all screens.

ARGUMENTS
     -/-

RESULT
     dummy - 1

EXAMPLE
     /* Beep screens */

     dummy = BEEP()

SEE ALSO
```

1.44 rexxtricks.guide/GETDEFAULTPUBSCREEN()

```
                Function GETDEFAULTPUBSCREEN()
*****************************

NAME
     screen = GETDEFAULTPUBSCREEN()

ALIAS
     screen = RXTR_GETDEFAULTPUBSCREEN()

DESCRIPTION
     Returns the name of the default public screen.

ARGUMENTS
     -/-

RESULT
     screen - Name of the current default public screen.

EXAMPLE
     /* Get default public screen */
```

```
     screen = GETDEFAULTPUBSCREEN()
     SAY 'Default public screen is:' screen

SEE ALSO


                 SETDEFAULTPUBSCREEN()
                   Set new default pubscreen

                 PUBSCREENLIST()
                         Set new default public screen
```

1.45 rexxtricks.guide/SETDEFAULTPUBSCREEN()

```
                Function SETDEFAULTPUBSCREEN()
*****************************

NAME
     oldscreen = SETDEFAULTPUBSCREEN(newscreen)

ALIAS
     oldscreen = RXTR_SETDEFAULTPUBSCREEN(newscreen)

DESCRIPTION
     Sets a new default public screen, the name of the old screen is
     returned.

ARGUMENTS
     newscreen - Name of the new default public screen. If this is omitted,
                 the screen 'Workbench' is used.

RESULT
     oldscreen - Name of the old default public screen.

EXAMPLE
     /* Set the screen of Cygnus ED as default public screen */

     oldscreen = SETDEFAULTPUBSCREEN('CygnusEdScreen1')
     SAY 'Old default public screen was:' oldscreen

SEE ALSO


                 GETDEFAULTPUBSCREEN()
                   Get name of the default public screen

                 PUBSCREENLIST()
                         Get list of all public screens currently open
```

1.46 rexxtricks.guide/GETPUBSCREENMODES()

```
                Function GETPUBSCREENMODES()
****************************

NAME
     publicmodes = GETPUBSCREENMODES()

ALIAS
     publicmodes = RXTR_GETPUBSCREENMODES()

DESCRIPTION
     Returns the global intuition public screen modes.

ARGUMENTS
     -/-

RESULT
     publicmodes - A string of two characters. The first character
                   represents the SHANGHAI mode, the second character
                   represents the POPPUBSCREEN mode.

                   If SHANGHAI mode is on, all workbench windows
                   are opened on the default public screen.

                   If POPPUBSCREEN mode is on, a public screen pops to the
                   front of the display when a window opens on the screen.

                   -- SHANGHAI and POPPUBSCREEN off

                   S- SHANGHAI on

                   -P POPPUBSCREEN on

                   SP SHANGHAI and POPPUBSCREEN on

EXAMPLE
     /* Get current public screen modes */

     publicmodes = GETPUBSCREENMODES()
     SAY 'Pubscreen modes:' publicmodes

SEE ALSO


                 SETPUBSCREENMODES()
                   Set new public screen modes
```

1.47 rexxtricks.guide/SETPUBSCREENMODES()

```
                Function SETPUBSCREENMODES()
****************************
```

```
NAME
     oldmodes = SETPUBSCREENMODES(publicmodes)

ALIAS
     oldmodes = RXTR_SETPUBSCREENMODES(publicmodes)

DESCRIPTION
     Sets the global intuition public screen modes.

ARGUMENTS
     publicmodes - A string of two characters. The first character
                   represents the SHANGHAI mode, the second character
                   represents the POPPUBSCREEN mode.

                   If SHANGHAI mode is on, all workbench windows
                   are opened on the default public screen.

                   If POPPUBSCREEN mode is on, a public screen pops to the
                   front of the display when a window opens on the screen.

                   -- SHANGHAI and POPPUBSCREEN off

                   S- SHANGHAI on

                   -P POPPUBSCREEN on

                   SP SHANGHAI and POPPUBSCREEN on

RESULT
     oldmodes    - The old public screen modes.

EXAMPLE
     /* Turn on SHANGHAI mode, and turn off POPPUBSCREEN mode */

     oldmodes = SETPUBSCREENMODES('S-')
     SAY 'Old public screen modes:' oldmodes

SEE ALSO


                 GETPUBSCREENMODES()
                   Get current public screen modes
```

1.48 rexxtricks.guide/PUBSCREENTOFRONT()

```
                Function PUBSCREENTOFRONT()
**************************

NAME
     boolean = PUBSCREENTOFRONT(screen)

ALIAS
```

```
     boolean = RXTR_PUBSCREENTOFRONT(screen)

DESCRIPTION
     Brings a public screen to the front of the display.

ARGUMENTS
     screen  - Name of the public screen

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* Bring the screen of Cygnus ED to the front */

     IF PUBSCREENTOFRONT('CygnusEdScreen1') THEN
        SAY 'Cygnus ED is in front of the display!'
     ELSE
        SAY 'Error, cannot bring Cygnus ED to the front!'

SEE ALSO


                 PUBSCREENTOBACK()
                   Send a public screen to the back
```

1.49 rexxtricks.guide/PUBSCREENTOBACK()

```
                Function PUBSCREENTOBACK()
**************************

NAME
     boolean = PUBSCREENTOBACK(screen)

ALIAS
     boolean = RXTR_PUBSCREENTOBACK(screen)

DESCRIPTION
     Sends a public screen to the back of the display.

ARGUMENTS
     screen  - Name of the public screen

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* Send workbench screen to the back */

     IF PUBSCREENTOBACK('Workbench') THEN
        SAY 'Workbench is in the back of the display!'
     ELSE
        SAY 'Error, cannot send workbench screen to the back!'
```

```
SEE ALSO

                PUBSCREENTOFRONT()
                  Bring a public screen to the front
```

1.50 rexxtricks.guide/PUBSCREENLIST()

```
                Function PUBSCREENLIST()
************************

NAME
     boolean = PUBSCREENLIST(destvar)

ALIAS
     boolean = RXTR_PUBSCREENLIST(destvar)

DESCRIPTION
     Creates a list of all public screens currently open.

ARGUMENTS
     destvar - Compound variable to store the public screen names,
               the names are stored as follows:

               destvar.0 - the number of public screens
               destvar.1 - the name of the first public screen
               destvar.2 - the name of the second public screen
               destvar.n - the name of the n'th public screen

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* Display a list of all public screens */

     IF PUBSCREENLIST('screenlist') THEN DO
        SAY 'Number of public screens:' screenlist.0
        DO i = 1 TO screenlist.0
           SAY screenlist.i
        END
     END

SEE ALSO

                GETDEFAULTPUBSCREEN()
                  Get name of the default public screen

                SETDEFAULTPUBSCREEN()
                  Set new default public screen
```

1.51 rexxtricks.guide/GETTOOLTYPEVALUE()

```
                Function GETTOOLTYPEVALUE()
**************************

NAME
     value = GETTOOLTYPEVALUE(filename,keyword)

ALIAS
     value = RXTR_GETTOOLTYPEVALUE(filename,keyword)

DESCRIPTION
     Returns the value of a tooltype. Tooltype Format is <keyword>=<value>

ARGUMENTS
     filename - Name of the icon file, without suffix '.info'!!!

     keyword  - Keyword of the tooltype to search for

RESULT
     value    - The value associated with the specified keyword or an
                empty string '' if the keyword has no value. If the
                keyword or the icon doesn't exist, value will be an
                empty string '' too. In this case, the variable RC
                is set to 10.

EXAMPLE
     /* Get hotkey of CrossDOS commodity */

     hotkey = GETTOOLTYPEVALUE('SYS:Wbstartup/CrossDOS','CX_POPKEY')

     IF hotkey ~= '' THEN
        SAY 'Hotkey is:' hotkey
     ELSE DO
        IF RC = 0 THEN
           SAY 'Tooltype 'CX_POPKEY' has no value!'
        ELSE
           SAY 'Icon or Tooltype doesn't exist!'
     END

SEE ALSO


                 SETTOOLTYPEVALUE()
                   Set the value of a tooltype

                 GETTOOLTYPES()
                       Get all tooltypes of an icon

                 SETTOOLTYPES()
                       Set all tooltypes of an icon
```

1.52 rexxtricks.guide/SETTOOLTYPEVALUE()

```
                Function SETTOOLTYPEVALUE()
**************************

NAME
     boolean = SETTOOLTYPEVALUE(filename,keyword,value)

ALIAS
     boolean = RXTR_SETTOOLTYPEVALUE(filename,keyword,value)

DESCRIPTION
     Creates a tooltype entry <keyword>=<value> and stores it in the icon,
     an existing entry will be replaced.

ARGUMENTS
     filename - Name of the icon file, without suffix '.info'!!!

     keyword  - The keyword of the tooltype entry

     value    - The value of the tooltype entry, you can specify an empty
                string to set the keyword as entry without value.

RESULT
     boolean  - 1 if successful, 0 for an error

EXAMPLE
     /* Set the hotkey of CrossDOS commodity */

     IF SETTOOLTYPEVALUE('SYS:Wbstartup/CrossDOS','CX_POPKEY','alt f1') THEN
        SAY 'Hotkey is set to <alt f1>!'
     ELSE
        SAY 'Error setting the hotkey!'

SEE ALSO


                 GETTOOLTYPEVALUE()
                   Get the value of a tooltype

                 GETTOOLTYPES()
                       Get all tooltypes of an icon

                 SETTOOLTYPES()
                       Set all tooltypes of an icon
```

1.53 rexxtricks.guide/GETTOOLTYPES()

```
                Function GETTOOLTYPES()
***********************

NAME
```

```
     boolean = GETTOOLTYPES(filename,stemvar)

ALIAS
     boolean = RXTR_GETTOOLTYPES(filename,stemvar)

DESCRIPTION
     Creates a list of all tooltype entries stored in the icon.

ARGUMENTS
     filename - Name of the icon file, without suffix '.info'!!!

     stemvar  - Compound variable to store the tooltype entries, each
                element will be one entry:

                stemvar.0 - the number of tooltype entries
                stemvar.1 - the first tooltype entry
                stemvar.2 - the second tooltype entry
                stemvar.n - the n'th tooltype entry

RESULT
     boolean  - 1 if successful, 0 for an error

EXAMPLE
     /* Get all tooltypes of the CrossDOS commodity */

     IF GETTOOLTYPES('SYS:Wbstartup/CrossDOS','tooltypes') THEN DO
        SAY 'Number of tooltypes:' tooltypes.0
        SAY

        DO i = 1 TO tooltypes.0
           SAY tooltypes.i
        END
     END
     ELSE
        SAY 'Error reading tooltypes!'

SEE ALSO


                 SETTOOLTYPES()
                       Set all tooltypes of an icon

                 GETTOOLTYPEVALUE()
                   Get the value of a tooltype

                 SETTOOLTYPEVALUE()
                   Set the value of a tooltype
```

1.54 rexxtricks.guide/SETTOOLTYPES()

```
                Function SETTOOLTYPES()
**********************
```

```
NAME
     boolean = SETTOOLTYPES(filename,tooltypes)

ALIAS
     boolean = RXTR_SETTOOLTYPES(filename,tooltypes)

DESCRIPTION
     Sets the tooltype entries of an icon, all existing entries will be
     removed.

ARGUMENTS
     filename  - Name of the icon file, without suffix '.info'!!!

     tooltypes - Compound variable which contents the new tooltype entries,
                 this variable must have the following format:

                 tooltypes.0 - must be the number of tooltype entries
                 tooltypes.1 - must be the first entry
                 tooltypes.2 - must be the second entry
                 tooltypes.n - must be the n'th entry

RESULT
     boolean   - 1 if successful, 0 for an error

EXAMPLE
     /* Set tooltypes of the CrossDOS commodity */

     tooltypes.0 = 6
     tooltypes.1 = '«PC0,FILTER,TRANS,INTL.crossdos»'
     tooltypes.2 = '«PC1,FILTER,TRANS,INTL.crossdos»'
     tooltypes.3 = 'DONOTWAIT'
     tooltypes.4 = 'CX_POPUP=NO'
     tooltypes.5 = 'CX_PRIORITY=0'
     tooltypes.6 = 'CX_POPKEY=ctrl alt c'

     IF SETTOOLTYPES('SYS:Wbstartup/CrossDOS','tooltypes') THEN
        SAY 'New tooltypes are set!'
     ELSE
        SAY 'Error setting tooltypes!'

SEE ALSO


                 GETTOOLTYPES()
                       Get all tooltypes of an icon

                 GETTOOLTYPEVALUE()
                   Get the value of a tooltype

                 SETTOOLTYPEVALUE()
                   Set the value of a tooltype
```

1.55 rexxtricks.guide/GETDEFAULTTOOL()

```
                Function GETDEFAULTTOOL()
************************

NAME
     defaulttool = GETDEFAULTTOOL(filename)

ALIAS
     defaulttool = RXTR_GETDEFAULTTOOL(filename)

DESCRIPTION
     Returns the default tool of a project or disk icon. All other icon
     types have no default tool, then an empty string is returned.

ARGUMENTS
     filename    - Name of the icon file, without suffix '.info'!!!

RESULT
     defaulttool - The default tool of the icon. The function will return
                   an empty string '' if no default tool is set or for
                   an error. If an error occurs, the variable RC
                   is set to 10.

EXAMPLE
     /* Get default tool of the boot disk */

     prg = GETDEFAULTTOOL('SYS:disk')

     IF prg ~= '' THEN
        SAY 'Default tool is:' prg
     ELSE DO
        IF RC = 0 THEN
           SAY 'No default tool found!'
        ELSE
           SAY 'Error, cannot get default tool!'
     END

SEE ALSO


                 SETDEFAULTTOOL()
                   Set default tool of an icon
```

1.56 rexxtricks.guide/SETDEFAULTTOOL()

```
                Function SETDEFAULTTOOL()
************************

NAME
     boolean = SETDEFAULTTOOL(filename,defaulttool)
```

```
ALIAS
     boolean = RXTR_SETDEFAULTTOOL(filename,defaulttool)

DESCRIPTION
     Sets the default tool of a project, or disk icon.

ARGUMENTS
     filename    - Name of the icon file, without suffix '.info'!!!

     defaulttool - The new default tool.

RESULT
     boolean     - 1 if successful, 0 for an error

EXAMPLE
     /* Set default tool of the boot disk */

     IF SETDEFAULTTOOL('SYS:disk','SYS:System/DiskCopy') THEN
        SAY 'Default tool is set!'
     ELSE
        SAY 'Error, cannot set default tool!'

SEE ALSO


                GETDEFAULTTOOL()
                  Set default tool of an icon
```

1.57 rexxtricks.guide/GETSTACK()

```
                Function GETSTACK()
*******************

NAME
     stacksize = GETSTACK(filename)

ALIAS
     stacksize = RXTR_GETSTACK(filename)

DESCRIPTION
     Returns the stacksize of a tool or project icon. All other icon types
     have no stacksize, then an empty string is returned.

ARGUMENTS
     filename  - Name of the icon file, without suffix '.info'!!!

RESULT
     stacksize - The stacksize of the icon. The function will return
                 an empty string '' if the icon has no stacksize entry
                 or for an error. If an error occurs, the variable RC
                 is set to 10.

EXAMPLE
```

```
     /* Get stacksize of CrossDOS commodity */

     stack = GETSTACK('SYS:Wbstartup/CrossDOS')

     IF stack ~= '' THEN
        SAY 'Stacksize is' stack 'bytes!'
     ELSE DO
        IF RC = 0 THEN
           SAY 'No stacksize found!'
        ELSE
           SAY 'Error, cannot get stacksize!'
     END

SEE ALSO

                 SETSTACK()
                   Set the stacksize of an icon
```

1.58 rexxtricks.guide/SETSTACK()

```
                Function SETSTACK()
*******************

NAME
     boolean = SETSTACK(filename,stacksizee)

ALIAS
     boolean = RXTR_SETSTACK(filename,stacksize)

DESCRIPTION
     Sets the stacksize of a tool, or project icon.

ARGUMENTS
     filename  - Name of the icon file, without suffix '.info'!!!

     stacksize - The new stacksize.

RESULT
     boolean   - 1 if successful, 0 for an error

EXAMPLE
     /* Set stacksize of CrossDOS commodity */

     IF SETSTACK('SYS:Wbstartup/CrossDOS',10240) THEN
        SAY 'Stacksize is set to 10240 bytes!'
     ELSE
        SAY 'Error, cannot set stacksize!'

SEE ALSO

                 GETSTACK()
```

```
                  Get the stacksize of an icon
```

1.59 rexxtricks.guide/CREATEICON()

```
Function CREATEICON()
*********************

NAME
     boolean = CREATEICON(filename,sourceicon,tooltypes,defaulttool,stacksize)

ALIAS
     boolean = RXTR_CREATEICON(filename,sourceicon,tooltypes,defaulttool,stacksize ↩
        )

DESCRIPTION
     Creates a new icon, type and images are taken from the source icon.

ARGUMENTS
     filename    - Name of the new icon, without suffix '.info'!!!

     sourceicon  - Name of the source icon, without suffix '.info'!!!
                   Icon type and images of this icon are used to create
                   the new icon.

     tooltypes   - Compound variable which contents the tooltype entries.
                   If this is omitted, all tooltypes of the source icon
                   are copied into the new icon. This variable must have
                   the following format:

                   tooltypes.0 - must be the number of tooltype entries
                   tooltypes.1 - must be the first entry
                   tooltypes.2 - must be the second entry
                   tooltypes.n - must be the n'th entry

     defaulttool - The default tool of the new icon. If this is omitted,
                   the default tool of the source icon is used.

     stacksize   - The stacksize of the new icon. If this is omitted, the
                   stacksize of the source icon is used.

RESULT
     boolean     - 1 if successful, 0 for an error

EXAMPLE
     /* Create a project icon for a textfile */

     tooltypes.0 = 1
     tooltypes.1 = 'FILETYPE=Text'

     IF CREATEICON('DH0:Text.txt','ENV:sys/def_Project','tooltypes','C:Ed',8192)  ↩
        THEN
        SAY 'Icon created successfully!'
     ELSE
```

```
        SAY 'Error creating the icon!'

SEE ALSO
```

1.60 rexxtricks.guide/WBINFO()

```
Function WBINFO()
*****************

NAME
     boolean = WBINFO(filename,pubscreen)

ALIAS
     boolean = RXTR_WBINFO(filename,pubscreen)

DESCRIPTION
     Calls the icon information window from workbench and opens it on a
     public screen.

ARGUMENTS
     filename  - Name of the icon to edit, without suffix '.info'!!!

     pubscreen - Name of a public screen to open the window on. If this
                 is omitted, the default public screen is used.

RESULT
     boolean   - 1 if successful, 0 for an error

EXAMPLE
     /* Edit icon of the CrossDOS commodity on the CygnusEd screen */

     IF WBINFO('SYS:Wbstartup/CrossDOS','CygnusEdScreen1') THEN
        SAY 'Okay, no errors!'
     ELSE
        SAY 'Error, cannot open the window!'

SEE ALSO
```

1.61 rexxtricks.guide/WHATIS()

```
                Function WHATIS()
*****************

NAME
     type = WHATIS(filename)

ALIAS
     type = RXTR_WHATIS(filename)

DESCRIPTION
     Recognizes the type of a file using 'whatis.library V4.0+' and returns
     the description string of the filetype.
```

```
ARGUMENTS
     filename - Name of the file

RESULT
     type     - A string which describes the type of the file. You can
                get a list of all possible filetypes with WHATISTYPES().

EXAMPLE
     /* Recognize the type of 'S:User-Startup' */

     type = WHATIS('S:User-Startup')
     SAY 'Filetype of S:User-Startup is:' type

SEE ALSO

                 WHATISTYPES()
                   Alle Dateitypen der 'whatis.library' ermitteln
```

1.62 rexxtricks.guide/WHATISTYPES()

```
                Function WHATISTYPES()
**********************

NAME
     boolean = WHATISTYPES(destvar)

ALIAS
     boolean = RXTR_WHATISTYPES(destvar)

DESCRIPTION
     Scans 'whatis.library' for all currently known filetypes and stores
     the description strings in a compound variable.

ARGUMENTS
     destvar - Compound variable to store the description strings
               of the filetypes:

               destvar.0 - the number of filetypes
               destvar.1 - description string of the first filetype
               destvar.2 - description string of the second filetype
               destvar.n - description string of the n'th filetype

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* Display all filetypes of 'whatis.library' */

     IF WHATISTYPES('destvar') THEN DO
        SAY 'Number of filetypes:' destvar.0
        DO i = 1 TO destvar.0
```

```
           SAY destvar.i
        END
     END

SEE ALSO

                 WHATIS()
                   Recognize the type of a file using 'whatis.library'
```

1.63 rexxtricks.guide/FILEID_IDENTIFY()

```
                Function FILEID_IDENTIFY()
**************************

NAME
     type = FILEID_IDENTIFY(filename,mode)

ALIAS
     type = RXTR_FILEID_IDENTIFY(filename,mode)

DESCRIPTION
     Recognizes the type of a file using 'FileID.library V7.0+' and returns
     the description string or the ID number of the filetype.

ARGUMENTS
     filename - Name of the file

     mode     - Specifies whether the description string or the ID
                number of the filetype should be returned.

                Possible keywords:

                'DESCRIPTION' or 'D' - Result is the description string
                                       of the filetype

                'ID' or 'I'          - Result is the ID number of the
                                       filetype

                                       See
                FILEID_GETHIGHID()
                .
                                       See
                FILEID_GETIDSTRING()
                .

                If this argument is omitted, the description string is
                returned.

RESULT
     type     - The filetype, depending on the 'mode' argument.

EXAMPLE
```

```
     /* Recognize teh type of 'LIBS:rexxtricks.library' */

     type = FILEID_IDENTIFY('LIBS:rexxtricks.library','ID')
     SAY 'Filetype ID of LIBS:rexxtricks.library is:' type

     type = FILEID_IDENTIFY('LIBS:rexxtricks.library')
     SAY 'Filetype description of LIBS:rexxtricks.library is:' type

SEE ALSO


                FILEID_GETHIGHID()
                    Get maximum filetype ID of 'FileID.library'

                FILEID_GETIDSTRING()
                  Get the description of a filtype ID

                FILEID_GETTYPES()
                     Get a list of all 'FileID.library' filetypes
```

1.64 rexxtricks.guide/FILEID_GETHIGHID()

```
                Function FILEID_GETHIGHID()
**************************

NAME
     id = FILEID_GETHIGHID()

ALIAS
     id = RXTR_FILEID_GETHIGHID()

DESCRIPTION
     Returns the maximum filetype ID number of 'FileID.library', this is
     also the number of known filetypes.

ARGUMENTS
     -/-

RESULT
     id - Maximum ID number and number of known filetypes

EXAMPLE
     /* Display number of known filetypes */

     SAY 'Number of filetypes:' FILEID_GETHIGHID()

SEE ALSO


                FILEID_IDENTIFY()
                     Recognize the type of a file using 'FileID.library'

                FILEID_GETIDSTRING()
```

```
                  Get the description of a filtype ID

                FILEID_GETTYPES()
                     Get a list of all 'FileID.library' filetypes
```

1.65 rexxtricks.guide/FILEID_GETIDSTRING()

```
                Function FILEID_GETIDSTRING()
*****************************

NAME
     string = FILEID_GETIDSTRING(id)

ALIAS
     string = RXTR_FILEID_GETIDSTRING(id)

DESCRIPTION
     Returns the description string of a filetype ID.

ARGUMENTS
     id     - The ID number of the filetype

RESULT
     string - The description string of the specified ID number

EXAMPLE
     /* Display description of ID 25 */

     SAY 'ID number 25 is:' FILEID_GETIDSTRING(25)

SEE ALSO


                FILEID_IDENTIFY()
                     Recognize the type of a file using 'FileID.library'

                FILEID_GETHIGHID()
                    Get maximum filetype ID of 'FileID.library'

                FILEID_GETTYPES()
                     Get a list of all 'FileID.library' filetypes
```

1.66 rexxtricks.guide/FILEID_GETTYPES()

```
                Function FILEID_GETTYPES()
*************************

NAME
```

```
     boolean = FILEID_GETTYPES(destvar)

ALIAS
     boolean = RXTR_FILEID_GETTYPES(destvar)

DESCRIPTION
     Scans 'FileID.library' for all currently known filetypes and stores
     the description strings in a compound variable.

ARGUMENTS
     destvar - Compound variable to store the description strings
               of the filetypes:

               destvar.0 - the number of filetypes
               destvar.1 - description string of the filetype ID 1
               destvar.2 - description string of the filetype ID 2
               destvar.n - description string of the filetype ID n

               The description string of ID number 0 cannot be stored
               because 'destvar.0' contains the number of filetypes!

               The description of filetype 0 is: 'unknown data file'.

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* Display all filetypes of 'FileID.library' */

     IF FILEID_GETTYPES('destvar') THEN DO
        SAY 'Number of filetypes:' destvar.0
        DO i = 1 TO destvar.0
           SAY RIGHT(i,4) destvar.i
        END
     END

SEE ALSO


                 FILEID_IDENTIFY()
                      Recognize the type of a file using 'FileID.library'

                 FILEID_GETHIGHID()
                     Get maximum filetype ID of 'FileID.library'

                 FILEID_GETIDSTRING()
                   Get the description of a filtype ID
```

1.67 rexxtricks.guide/CRC32()

```
Function CRC32()
****************
```

```
NAME
     crc = CRC32(filename,verbose)

ALIAS
     crc = RXTR_CRC32(filename,verbose)

DESCRIPTION
     Calculates the 32-Bit CRC checksum of a file.

ARGUMENTS
     filename - Name of the file

     verbose  - Specifies the format of the result 'crc'.

                Possible keywords are 'VERBOSE' or 'V', if this is
                specified 'crc' is a string of four words:

                   1. word - checksum hexadecimal
                   2. word - checksum decimal
                   3. word - complement of the checksum hexadecimal
                   4. word - complement of the checksum decimal

                If this argument is omitted, 'crc' is the hexadecimal
                complement of the checksum.

RESULT
     crc      - Checksum of the file depending on the 'verbose' argument.

EXAMPLE
     /* Calculate the checksum of 'S:User-Startup' */

     crc = CRC32('S:User-Startup')
     SAY 'Checksum =' crc

SEE ALSO
```

1.68 rexxtricks.guide/COUNTCHARS()

```
Function COUNTCHARS()
*********************

NAME
     count = COUNTCHARS(string,characters)

ALIAS
     count = RXTR_COUNTCHARS(string,characters)

DESCRIPTION
     Counts the number of occurances of the specified characters in the
     string. The search is case sensitive.

ARGUMENTS
     string     - The string in which the characters should be searched

     characters - The characters to count
```

```
RESULT
     count      - The number of occurances of the specified 'characters'
                  in the 'string':

                  COUNTCHARS('aa,bb,cc,dd,ee','c,') returns 6
                  COUNTCHARS('aa,bb,cc,dd,ee',',')  returns 4
                  COUNTCHARS('aa,bb,cc,dd,ee','c')  returns 2

EXAMPLE
     /* Count chars in a string */

     count = COUNTCHARS('Halli, hallo!','hH')
     SAY 'The characters h and H are' count 'times in the string.'

SEE ALSO
```

1.69 rexxtricks.guide/RAND()

```
Function RAND()
***************

NAME
     integer = RAND(min,max)

ALIAS
     integer = RXTR_RAND(min,max)

DESCRIPTION
     Returns a random number.

ARGUMENTS
     min     - The minimal possible number

     max     - The maximal possible number

RESULT
     integer - A random number between 'min' and 'max'

EXAMPLE
     /* Display random numbers between 1 and 100 */

     DO i = 1 TO 20
        SAY RAND(1,100)
     END

SEE ALSO
```

1.70 rexxtricks.guide/REXXTRICKSVERSION()

```
Function REXXTRICKSVERSION()
****************************
```

```
NAME
     version = REXXTRICKSVERSION()

ALIAS
     version = RXTR_REXXTRICKSVERSION()

DESCRIPTION
     Returns the version of 'rexxtricks.library'

ARGUMENTS
     -/-

RESULT
     version - The version in the following format: VERSION.REVISION

EXAMPLE
     /* Display the version of 'rexxtricks.library' */

     version = REXXTRICKSVERSION()
     SAY 'You are using rexxtricks.library version' version

SEE ALSO
```

1.71 rexxtricks.guide/UUDECODE()

```
                Function UUDECODE()
********************

NAME
     boolean = UUDECODE(sourcefile,destdir,destfile)

ALIAS
     boolean = RXTR_UUDECODE(sourcefile,destdir,destfile)

DESCRIPTION
     Decodes an uuencoded file and saves the binary files in the specified
     destination directory.

ARGUMENTS
     sourcefile - Name of the file, that contains one or more uuencoded
                  binary files. All contained files will be decoded.

                  Also it is possible to decode multipart uuencoded files.
                  All parts must be in the same directory and must have the
                  following format (e.g. 3 parts):

                  Part 1 - name:     uufile.uaa

                           contents: begin 644 binary.lha
                                     ...
                                     include uufile.uab

                  Part 2 - name:     uufile.uab
```

```
                    contents: begin part b uufile.uab
                              ...
                              include uufile.uac

          Part 3 - name:      uufile.uac

                    contents: begin part c uufile.uac
                              ...
                              end
```

To decode multipart uuencoded files of this format, simply specify the filename of the first part at the 'filename' argument. The decoding of all other parts is controlled by the 'include' lines. It's very important that all parts have the correct filename!

destdir - All decoded binary files are saved to this directory.

destfile - This argument is optional and should be used in special cases only. If a filename is specified here, this name is used for the first decoded binary file and the filename specified at the 'begin' line in the sourcefile is then ignored.

RESULT

boolean - 1 if successful, 0 for an error

Informations about the error are stored in the following variables if UUDECODE() fails:

uuerror_number

This variable contains an error number:

```
1 - cannot open a file
2 - incorrect or missing include-file
3 - 'begin' line not found
4 - incorrect checksum of a line
5 - error reading from a file
6 - error writing to a file
7 - incorrect size of the decoded binary file
8 - unexpected end of the sourcefile
```

uuerror_text

This variable contains an error message.

uuerror_file

This variable contains the name of the file, in which the error has occured.

uuerror_line

On error 4, this variable contains the number of the line with the incorrect checksum. On all other errors, this variable is set to 0.

```
                  uuerror_size

                     On error 7, this variable contains the expected
                     size of the decoded binary file. On all other
                     errors, this variable is set to 0.

EXAMPLE
     /* uudecode the file 'DH0:Test.uaa' to 'RAM:' */

     IF UUDECODE('DH0:Test.uaa','RAM:') THEN
        SAY 'Decoding successful, all binary files saved!'
     ELSE
        SAY 'Error decoding file:' uuerror_text

SEE ALSO


                 UUENCODE()
                   uuencode a file
```

1.72 rexxtricks.guide/UUENCODE()

```
                Function UUENCODE()
*******************

NAME
     boolean = UUENCODE(file,uufile,limit,suffix)

ALIAS
     boolean = RXTR_UUENCODE(file,uufile,limit,suffix)

DESCRIPTION
     Converts a binary file to pure ASCII so that it can be transmitted on
     a network which doesn't allow binary data.

ARGUMENTS
     file    - Name of the binary file to encode

     uufile  - Name of the uuencoded file. An existing suffix is replaced
               by '.uaa', '.uab', '.uac' etc., depending on the number of
               the created parts.

               The output format is similar to the format described in the
               chapter 'UUDECODE()'. See
                UUDECODE()
                .

     limit   - The maximum size of a part in KB, allowed are 16, 32, 64,
               128, 256 and 512. 'limit' is optional, the binary file is
               encoded in one part if 'limit' is omitted.

     suffix  - With this switch you can specify that an existing suffix
```

```
               should not be replaced, '.uaa' etc. is appended in that
               case. Possible keywords are 'SUFFIX' or 'S'.

               If this argument is omitted, an existing suffix is replaced.

RESULT
     boolean - 1 if successful, 0 for an error

EXAMPLE
     /* Encode file 'DH0:Test.lha' to 'RAM:Test.uXX', */
     /* maximum size of a part is 64KB */

     IF UUENCODE('DH0:Test.lha','RAM:Test',64) THEN
        SAY 'Encoding of the file was successful!'
     ELSE
        SAY 'Error encoding file!'

SEE ALSO


                 UUDECODE()
                   uudecode a file
```

1.73 rexxtricks.guide/The VIEWLIST() window

```
How to use the VIEWLIST() window
********************************

Entries in the window:
     normal entry         - text color of the screen without background

     selected entry       - text color and fill color of the screen

     actual entry         - a frame is drawn around the entry

Gadgets:
     scroller             - scrolls the list up and down

     All                  - selects all entries

     Pattern              - selects all entries which are matching the
                            pattern in the string gadget

     Toggle               - toggles all entries

     None                 - deselects all entries

     Search               - searches the next entry which matches the
                            pattern in the string gadget

     string gadget        - AmigaDOS pattern to select or search for
                            entries
```

```
     Use                  - closes the window and returns all selected
                            entries

     Cancel               - cancels selection and closes the window

     close gadget         - cancels selection and closes the window

Mouse buttons:
     left button          - selects the entry under the pointer and
                            deselects all other entries

     shift left button    - toggles the entry under the pointer

     doubleclick          - closes the window and returns all selected
                            entries

Keyboard:
     cursor up/down       - moves the actual entry up and down

     shift cursor up/down - moves the actual entry one page up and down

     ctrl cursor up/down  - moves the actual entry to start or end of
                            the list

     return, enter, space - selects or deselects the actual entry

     tab                  - activates the string gadget

     esc                  - cancels selection and closes the window, same
                            as close gadget
```

1.74 rexxtricks.guide/Index

```
                Index
*****


                 Addresses
                                          Author

                 Append a filename to the end of a path
                 MAKEPATH()

                 Append a suffix to the end of a filename
                 MAKESUFFIX()

                 Author
                                             Author

                 BEEP()
                                             BEEP()

                 Bildschirm blitzen
                                   BEEP()
```

Bring a public screen to the front
PUBSCREENTOFRONT()

BSEARCH()
BSEARCH()

Calculate 32-Bit CRC checksum of a file
CRC32()

Call the icon information window from workbench (OS3.0+)
WBINFO()

Check for a pattern match with a string
MATCHPATTERN()

Copy elements of a compound variable
STEMCOPY()

Copyright
Copyright

Copyright notice
Copyright

Count chars in a string
COUNTCHARS()

COUNTCHARS()
COUNTCHARS()

CRC32()
CRC32()

CRC32()
CRC32()

Create a new Icon
CREATEICON()

CREATEICON()
CREATEICON()

Credits
Credits

Display list in a listview window
VIEWLIST()

Extract dirname from path
PATHPART()

Extract filename from path
FILEPART()

FILEID_GETHIGHID()
FILEID_GETHIGHID()

```
FILEID_GETIDSTRING()
                FILEID_GETIDSTRING()

FILEID_GETTYPES()
                   FILEID_GETTYPES()

FILEID_IDENTIFY()
                   FILEID_IDENTIFY()

FILEPART()
                          FILEPART()

Function BEEP()
                     BEEP()

Function BSEARCH()
                  BSEARCH()

Function COUNTCHARS()
               COUNTCHARS()

Function CRC32()
                    CRC32()

Function CREATEICON()
               CREATEICON()

Function FILEID_GETHIGHID()
         FILEID_GETHIGHID()

Function FILEID_GETIDSTRING()
       FILEID_GETIDSTRING()

Function FILEID_GETTYPES()
          FILEID_GETTYPES()

Function FILEID_IDENTIFY()
          FILEID_IDENTIFY()

Function FILEPART()
                 FILEPART()

Function GETCOMMENT()
               GETCOMMENT()

Function GETDEFAULTPUBSCREEN()
      GETDEFAULTPUBSCREEN()

Function GETDEFAULTTOOL()
           GETDEFAULTTOOL()

Function GETDIR()
                   GETDIR()

Function GETENV()
                   GETENV()
```

```
Function GETKEY()
                    GETKEY()

Function GETPROTECTION()
             GETPROTECTION()

Function GETPUBSCREENMODES()
         GETPUBSCREENMODES()

Function GETSTACK()
                  GETSTACK()

Function GETTOOLTYPES()
              GETTOOLTYPES()

Function GETTOOLTYPEVALUE()
          GETTOOLTYPEVALUE()

Function LSEARCH()
                   LSEARCH()

Function MAKEPATH()
                  MAKEPATH()

Function MAKESUFFIX()
                MAKESUFFIX()

Function MATCHPATTERN()
              MATCHPATTERN()

Function PATHPART()
                  PATHPART()

Function PUBSCREENLIST()
             PUBSCREENLIST()

Function PUBSCREENTOBACK()
           PUBSCREENTOBACK()

Function PUBSCREENTOFRONT()
          PUBSCREENTOFRONT()

Function QSORT()
                     QSORT()

Function RAND()
                      RAND()

Function READCLIPBOARD()
             READCLIPBOARD()

Function READFILE()
                  READFILE()

Function READLINES()
                 READLINES()
```

```
Function REXXTRICKSVERSION()
        REXXTRICKSVERSION()

Function SCSI_DEVICETYPE()
          SCSI_DEVICETYPE()

Function SCSI_MANUFACTURER()
        SCSI_MANUFACTURER()

Function SCSI_PRODUCT()
             SCSI_PRODUCT()

Function SCSI_REVISION()
            SCSI_REVISION()

Function SCSI_TESTREADY()
           SCSI_TESTREADY()

Function SEARCHPATTERN()
            SEARCHPATTERN()

Function SETCOMMENT()
               SETCOMMENT()

Function SETDEFAULTPUBSCREEN()
      SETDEFAULTPUBSCREEN()

Function SETDEFAULTTOOL()
           SETDEFAULTTOOL()

Function SETENV()
                   SETENV()

Function SETPROTECTION()
            SETPROTECTION()

Function SETPUBSCREENMODES()
        SETPUBSCREENMODES()

Function SETSTACK()
                 SETSTACK()

Function SETTOOLTYPES()
             SETTOOLTYPES()

Function SETTOOLTYPEVALUE()
          SETTOOLTYPEVALUE()

Function STEMCOPY()
                 STEMCOPY()

Function STEMINSERT()
               STEMINSERT()

Function STEMREMOVE()
               STEMREMOVE()
```

```
Function SUFFIXPART()
               SUFFIXPART()

Function UNSETENV()
                UNSETENV()

Function UUDECODE()
                UUDECODE()

Function UUENCODE()
                UUENCODE()

Function VIEWLIST()
                VIEWLIST()

Function WBINFO()
                  WBINFO()

Function WHATIS()
                  WHATIS()

Function WHATISTYPES()
              WHATISTYPES()

Function WRITECLIPBOARD()
           WRITECLIPBOARD()

Function WRITEFILE()
                WRITEFILE()

Function WRITELINES()
               WRITELINES()

Functions
                           Library functions

Get a list of all 'FileID.library' filetypes
FILEID_GETTYPES()

Get a list of all 'whatis.library' filetypes
WHATISTYPES()

Get a random number
                RAND()

Get all tooltypes of an icon
        GETTOOLTYPES()

Get comment of a file
               GETCOMMENT()

Get current public screen modes
     GETPUBSCREENMODES()

Get default tool of an icon
         GETDEFAULTTOOL()
```

```
Get environment variable
            GETENV()

Get list of all public screens currently open
PUBSCREENLIST()

Get maximum filetype ID of 'FileID.library'
FILEID_GETHIGHID()

Get name of the default public screen
GETDEFAULTPUBSCREEN()

Get protection flags of a file
      GETPROTECTION()

Get the description of a filtype ID
 FILEID_GETIDSTRING()

Get the manufacturer of a SCSI peripheral
SCSI_MANUFACTURER()

Get the product string of a SCSI peripheral
SCSI_PRODUCT()

Get the revision string of a SCSI peripheral
SCSI_REVISION()

Get the stacksize of an icon
        GETSTACK()

Get the suffix of a filename
        SUFFIXPART()

Get the type of a SCSI peripheral, DISK, TAPE etc.
SCSI_DEVICETYPE()

Get the value of a tooltype
         GETTOOLTYPEVALUE()

Get version of rexxtricks.library
   REXXTRICKSVERSION()

GETCOMMENT()
                         GETCOMMENT()

GETDEFAULTPUBSCREEN()
               GETDEFAULTPUBSCREEN()

GETDEFAULTPUBSCREEN()
               GETDEFAULTPUBSCREEN()

GETDEFAULTTOOL()
                     GETDEFAULTTOOL()

GETDIR()
                              GETDIR()
```

```
PATHPART()
                            PATHPART()

PUBSCREENLIST()
                       PUBSCREENLIST()

PUBSCREENLIST()
                       PUBSCREENLIST()

PUBSCREENTOBACK()
                     PUBSCREENTOBACK()

PUBSCREENTOBACK()
                     PUBSCREENTOBACK()

PUBSCREENTOFRONT()
                   PUBSCREENTOFRONT()

PUBSCREENTOFRONT()
                   PUBSCREENTOFRONT()

QSORT()
                                QSORT()

RAND()
                                 RAND()

Read a part of a textfile into a compound variable
READLINES()

Read a textfile into a compound variable
READFILE()

Read directory into a compound variable
GETDIR()

Read text from clipboard
            READCLIPBOARD()

READCLIPBOARD()
                       READCLIPBOARD()

READFILE()
                            READFILE()

READLINES()
                          READLINES()

Recognize the type of a file using 'FileID.library'
FILEID_IDENTIFY()

Recognize the type of a file using 'whatis.library'
WHATIS()

Remove elements of a compound variable
STEMREMOVE()
```

```
Remove environment variable
         UNSETENV()

Replace or insert lines in a textfile
WRITELINES()

Requirements
                         Requirements

REXXTRICKSVERSION()
                 REXXTRICKSVERSION()

SCSI_DEVICETYPE()
                   SCSI_DEVICETYPE()

SCSI_MANUFACTURER()
                 SCSI_MANUFACTURER()

SCSI_PRODUCT()
                      SCSI_PRODUCT()

SCSI_REVISION()
                     SCSI_REVISION()

SCSI_TESTREADY()
                    SCSI_TESTREADY()

Search for pattern in a textfile
    SEARCHPATTERN()

Search string with Binary Search
    BSEARCH()

Search string with Linear Search
    LSEARCH()

SEARCHPATTERN()
                     SEARCHPATTERN()

Send a public screen to the back
    PUBSCREENTOBACK()

Set all tooltypes of an icon
        SETTOOLTYPES()

Set comment on a file
               SETCOMMENT()

Set default tool of an icon
         SETDEFAULTTOOL()

Set environment variable
            SETENV()

Set new default public screen
       SETDEFAULTPUBSCREEN()
```

```
Set new public screen modes
         SETPUBSCREENMODES()

Set protection flags of a file
      SETPROTECTION()

Set the stacksize of an icon
        SETSTACK()

Set the value of a tooltype
         SETTOOLTYPEVALUE()

SETCOMMENT()
                          SETCOMMENT()

SETDEFAULTPUBSCREEN()
               SETDEFAULTPUBSCREEN()

SETDEFAULTPUBSCREEN()
               SETDEFAULTPUBSCREEN()

SETDEFAULTTOOL()
                     SETDEFAULTTOOL()

SETENV()
                               SETENV()

SETPROTECTION()
                      SETPROTECTION()

SETPUBSCREENMODES()
                 SETPUBSCREENMODES()

SETPUBSCREENMODES()
                 SETPUBSCREENMODES()

SETSTACK()
                            SETSTACK()

SETTOOLTYPES()
                       SETTOOLTYPES()

SETTOOLTYPEVALUE()
                  SETTOOLTYPEVALUE()

Sort list with QuickSort
            QSORT()

STEMCOPY()
                            STEMCOPY()

STEMINSERT()
                          STEMINSERT()

STEMREMOVE()
                          STEMREMOVE()
```

```
SUFFIXPART()
                          SUFFIXPART()

Support-Mailbox
                      Author

Test whether a SCSI peripheral is ready or not
SCSI_TESTREADY()

The VIEWLIST() window
               The VIEWLIST() window

UNSETENV()
                            UNSETENV()

uudecode a file
                      UUDECODE()

UUDECODE()
                            UUDECODE()

uuencode a file
                      UUENCODE()

UUENCODE()
                            UUENCODE()

VIEWLIST()
                            VIEWLIST()

Wait for a key at console window
    GETKEY()

WBINFO()
                              WBINFO()

WHATIS()
                              WHATIS()

WHATISTYPES()
                        WHATISTYPES()

WHATISTYPES()
                        WHATISTYPES()

Write contents of a compound variable to a textfile
WRITEFILE()

Write text to clipboard
             WRITECLIPBOARD()

WRITECLIPBOARD()
                     WRITECLIPBOARD()

WRITEFILE()
                           WRITEFILE()
```

```
WRITELINES()
                                WRITELINES()
```