RexxToolKit

COLLABORATORS

	TITLE : RexxToolKit		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 5, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

RexxToolKit

1.1 Quick Reference for REXX Routines in CNet Amiga ToolKit, v3.00

```
      Click a routine to Review the Usage and/or Format template for ↩
         :


  [01] : From `Expanded' date to `Sorted' or `Internal' date format ↩
     .

  [02] : From x5xxxxx GU Value to `Sorted' or `Internal' date  ↩
     format.

  [03] : From `Sorted' or `Internal' date to `Expanded' date format ↩
     .

  [04] : Numeric Range Parser: [ -2 19- 4 7-9 11.13.15,17 ]

  [05] : CNet-like input routine, using MCI.

  [06] : Find and return BBSTEXT/BBSMENU line entry.

  [07] : Check the CNet Amiga Version file is being run under.

  [08] : Check if a user is Suboperator in current subboard.

  [09] : Checks if MCI is enabled in current subboard.

  [10] : Convert from 12/24 hour time format to 12/24/min format.

  [11] : External Library Loader

  [12] : View, Enable, Disable or Toggle `Priviledge' Flags.

  [13] : Get `Arguments' from last command.

  [14] : Read `Cursor Key' / `Return/Enter' Keyboard Input.

  [15] : Convert `UPPERCASE' to `lowercase' text.

  [16] : Pauses output for `x' number of seconds the RIGHT way!
```

[17] : Checks for ‘Loss of Carrier’ in your Pfiles!

[18] : An informative ‘Error Checking’ routine.

[19] : Positions cursor for printing anywhere on the screen.

[20] : Horizontal Text Scroller Number 1.

[21] : Horizontal Text Scroller Number 2.

[22] : Read the joystick(s) and firebutton(s).

[23] : First attempt at MOUSE capability. (95% Complete!)

[24] : Disable or Enable the MORE? prompt, regardless of setting!

[25] : Muffle ALL ports, regardless of setting!

[26] : Extended SelectFile Routine.

[27] : Add line of text to specified LOG file.

[28] : Check Port Menu(s) Checkmark Status.

[29] : Send Text File as CNet MAIL to specified User.

[30] : Send a CNet/4 ‘File Attach’ EMail Message to specified ↩ User.

[31] : Send a System OLM (OnLine Message) to the current user.

[32] : Add keystrokes to other ports from present port.

[33] : A QUICK ‘Who’ for SysOps, listing Access Group Number.

[34] : View ‘port’ log of specified port. (Pre ‘calls’ log).

[35] : Send Line Noise to a port (Ability to kick them off too!)

[36] : UnLock User Accounts (That May NOT Have Been Previously!)

[37] : Replace <input> with <output> within string of <text>.

[38] : Find and Return or Verify BBSMENU section line(s).

[39] : Clears a specific port, by dumping the user.

[40] : EnCode & DeCode text strings, using a Numeric Key.

[41] : Guideline Entry-text for Mail (and File) Subboards.

[42] : NewDoor starter framework for New ARexx Doors/Pfiles.

1.2 sdate

```
/**[01]*********************************************************************
 *
 * Description: From "Expanded" date to "Sorted" or "Internal" date format.
 *
 *   Author(s): PMK              - (Peter no longer runs a bbs...)
 *              Dotoran          - Frontiers BBS             +1 716/823-9892
 *
 *************************************************************************
 *
 *    Expanded: Sun 25-Dec-1993 11:53a
 *      Sorted: 19970330    (In YYYYMMDD Format)
 *    Internal: 7028        (# of Days since 1/1/78)
 *
 *       Usage: <var>=SDATE(<date>,[mode])
 *
 *       Where: <date>  holds an "Expanded" Date.
 *              [mode]  as "i" returns Internal Days Format.
 *                      (Number of days since January 1, 1978)
 *
 *     Returns: <var>   holds the sorted (or internal) date format.
 *
 *      Note 1: Because of the way the internal ARexx DATE() command works,
 *              you should NOT use dates PREVIOUS to January 1, 1978 when
 *              using the "i" (internal) setting. This routine will, however
 *              return the SORTED date for ANY DATE given.
 *
 *      Note 2: We decided to keep the "i" parameter, because it's a FAST
 *              way to perform MATH functions on dates(13 days ago, etc).
 */
```

1.3 gdate

```
/**[02]*********************************************************************
 *
 * Description: From x5xxxxx GU Value to "Sorted" or "Internal" date format.
 *
 *   Author(s): PMK              - (Peter no longer runs a bbs...)
 *              Dotoran          - Frontiers BBS             +1 716/823-9892
 *
 *************************************************************************
 *
 *     x5xxxxx: 1500000, 1500410, 2500990, etc.
 *
 *      Sorted: 19970330    (In YYYYMMDD Format)
 *
 *    Internal: 7028        (# of Days since 1/1/78)
 *
 *       Usage: <var>=GDATE(<value>,[mode])
 *
 *       Where: <value> holds the 7-Digit x5xxxxx GetUser Value.
 *              [mode]  as "i" returns Internal Days Format.
 *                      (Number of days since January 1, 1978)
 *
 *     Returns: <var>   holds the sorted (or internal) date format.
 *
 *      Note 1: Because of the way the internal ARexx DATE() command works,
 *              you should NOT use dates PREVIOUS to January 1, 1978 when
 *              using the "i" (internal) setting. This routine will, however
 *              return the SORTED date for ANY DATE given.
 *
 *      Note 2: We decided to keep the "i" parameter, because it's a FAST
 *              way to perform MATH functions on dates(13 days ago, etc).
 *
 *       Fixed: The usage template above stated SDATE, instead of GDATE.
 *              The GDATE routine used an UPPERCASE month template, which
 *              returned incorrect month numerations, because LOWERCASE
 *              characters are returned when reading DATE GetUsers.
 */
```

1.4 edate

```
/**[03]*********************************************************************
 *
 * Description: From "Sorted" or "Internal" date to "Expanded" date format.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *
 ***************************************************************************
 *
 *      Sorted: 19970330        (In YYYYMMDD Format)
 *    Internal: 7028            (# of Days since 1/1/78)
 *    Expanded: Sun 30-Mar-1997 (No time format!)
 *
 *       Usage: <var>=EDATE(<date>,[mode])
 *
 *       Where: <date>  holds a "Sorted" or "Internal" Date.
 *              [mode]  Specified as 'i' if <date> supplied is in
 *                      Internal Days Format.
 *                      (Number of days since January 1, 1978)
 *
 *     Returns: <var>   holds the expanded date format.
 *
 *      Note 1: Because of the way the internal ARexx DATE() command works,
 *              you should NOT use dates PREVIOUS to January 1, 1978 when
 *              using the "i" (internal) setting. This routine will, however
 *              return the SORTED date for ANY DATE given.
 *
 *      Note 2: We decided to keep the "i" parameter, because it's a FAST
 *              way to perform MATH functions on dates(13 days ago, etc).
 */
```

1.5 parse

```
/**[04]*********************************************************************
 *
 * Description: Numeric Range Parser: [ -2 19- 4 7-9 11.13.15,17 ]
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *              PMK             - (Peter no longer runs a bbs...)
 *
 ***************************************************************************
 *
 *       Usage: <var>=PARSE(<range>,<min>,<max>,[sort])
 *
 *       Where: <var>    is any Legal Variable Name.
 *              <range>  is the Numeric Range to Parse.
 *              <min>    is the Minimum Value to Use.
 *              <max>    is the Maximum Value to Use.
 *              [sort]   as "s" is OPTIONAL. If specified, the items will
 *                       also be Numerically Sorted. Duplicate Item checking
 *                       is ONLY performed on SORTED item lists.
 *
 *     Returns: <var>    total parsed items.
 *              <it.0>   parsed item string, parsed in SPACES.
 *              <it.1>
 *                 |
 *              <it.?>   the individual parsed item array.
 *
 *      Note 1: This routine functions EXACTLY like CNet's own routine.
 *              Open ended ranges( -5 or 12- ) fully supported. Any use
 *              of DUPLICATE item numbers are removed, and the resulting
 *              it.? array contains items in NUMERICAL Order. All Non-
 *              Numeric items are discarded. Use the "it.0" variable
 *              string in conjunction with the INDEX() command for VERY
 *              FAST verification checking!
 *
 *      Note 2: If sorting is NOT essential to your needs in a particular
 *              application, we suggest NOT using it, as it will speed up
 *              the parsing process CONSIDERABLY! (VERY, VERY QUICK!)
 */
```

1.6 input

```
/**[05]*********************************************************************
 *
 * Description: CNet-like input routine, using MCI.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *
 ***************************************************************************
 *
 *       Usage: <var>=INPUT(<text>,<length>,<MCI opt>,[default])
 *
 *       Where: <text>     holds the prompt text.
 *              <length>   holds the max length of the input.
 *              <MCI opt>  MCI input options (1=caps, 2=filename, etc.)
 *                         (Review the MCI {I } Command for more info!)
 *              [default]  holds the default text to appear under the
 *                         cursor in the prompt.(OPTIONAL)
 *
 *     Returns: <var>      holds data that was input.
 */
```

1.7 bbsline

```
/**[06]*********************************************************************
 *
 * Description: Find and return BBSTEXT/BBSMENU line entry.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *
 *************************************************************************
 *
 *       Usage: <var>=BBSLINE(<file>,<line>)
 *
 *       Where: <file>    which file to use. (0=BBSMENU, 1=BBSTEXT)
 *              <line>    holds the line number in BBSTEXT/BBSMENU.
 *
 *     Returns: <var>     holds the returned BBSTEXT/BBSMENU line entry.
 */
```

1.8 checkver

```
/**[07]*********************************************************************
 *
 * Description: Check the CNet Amiga Version file is being run under.
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 *************************************************************************
 *
 *       Usage: <var>=CHECKVER()
 *
 *     Returns: <var> holds "3" if running under CNet Amiga, v3.05c,
 *                    holds "4" if running under CNet Amiga, v4.26b+.
 *
 *      Option: <var>=CHECKVER()-3
 *
 *     Returns: "0" if NOT CNet/4, or "1" if CNet/4 being used.
 */
```

1.9 subop

```
                /**[08]*****************************************************************

 *
 * Description: Check if a user is Suboperator in current subboard.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *              Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Needs: The
                CHECKVER()
                 routine listed above.
 *
 *       Usage: <var>=SUBOP(<id>)
 *
 *       Where: <id>   is the ID number of the user.
 *
 *     Returns: <var>  holds "1" if the user has Subop access, "0" if not.
 *
 *       Notes: Compatible with BOTH 3 and 4 versions of CNet, regardless
 *              of the GU shift between versions. Checks ALL SIX SubOp slots
 *              for the user ID specified.
 */
```

1.10 mciena

```
                /**[09]*******************************************************************

 *
 * Description: Checks if MCI is enabled in current subboard.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *              Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Needs: The
                CHECKVER()
                 routine listed above.
 *
 *       Usage: <var>=MCIENA()
 *
 *     Returns: <var>  holds "1" if MCI is enabled, "0" if not.
 */
```

1.11 tim

```
/**[10]*********************************************************************
 *
 * Description: Convert from 12/24 hour time format to 12/24/min format.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *              Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Usage: <var>=TIM(<value>,<mode>)
 *
 *       Where: <value>  is the getuser time value to convert.
 *              <mode>   is the format to convert to:
 *                       (12=12 Hr., 24=24 Hr., 0=Mins. Since Midnight)
 *
 *     Returns: <var>    holds the converted time value.
 *
 *       Fixed: If an AM time was converted to 24-Hr or Minutes format, the
 *              returned value reflected the PM time instead of the AM time.
 */
```

1.12 loadlib

```
/**[11]*********************************************************************
 *
 * Description: External Library Loader
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Usage: call LOADLIB("<library>")
 *
 *       Where: <library>  is the filename of the external library to load,
 *                         inside of either double or single quotation marks.
 *
 *     Returns: If library exists, it will be loaded, but if an error occurs
 *              during the load, you'll be told this and your file will
 *              immediately be terminated. (This occurs if the stated library
 *              is not located in your LIBS: path.)
 */
```

1.13 priv

```
/**[12]*********************************************************************
 *
 * Description: View, Enable, Disable or Toggle "Priviledge" Flags.
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Usage: call PRIV(<mode>,<block>,<priv>,<name>)
 *
 *       Where: <mode>  is the KEYWORD (or first LETTER of KEYWORD)
 *                      of the action to be performed:
 *
 *                      V or VIEW    - Current  Priviledge Setting.
 *                      T or TOGGLE  - Reverse  Current    Setting.
 *                      E or ENABLE  - Turn the Priviledge ON.
 *                      D or DISABLE - Turn the Priviledge OFF.
 *
 *             <block>  is either a "0" for ABIT0 (the first 32 privs),
 *                      or a "1" for ABIT1 (the second 32 priviledges).
 *
 *              <priv>  is the Priviledge Index Number found on the
 *                      GetUser 4.00 List(The Number from 0 to 31).
 *
 *              <name>  is the ID Number, Handle, or Real Name of the
 *                      user to perform the action on, whether they are
 *                      ONLINE or NOT! (Uses CNet’s Scratch Buffer!)
 *
 *     Returns: <priv>  holds a "Yes" if user HAS this Priviledge, or
 *                              "No"  if user DOESN’T have Priviledge.
 *                      (Updated AFTER Action Has Taken Place!)
 *
 *            <handle>  of the user action was performed on, even if
 *                      you entered an ID Number as the initial argument!
 *
 *            <status>  will be a "1" if data saved successfully, or
 *                                "0" if there was a problem saving.
 *                      (Note this variable NOT used in VIEW Mode!)
 */
```

1.14 args

```
/**[13]*********************************************************************
 *
 * Description: Get "Arguments" from last command.
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *              PMK             - (Peter no longer runs a bbs...)
 *
****************************************************************************
 *
 *       Usage: <var>=ARGS()
 *
 *       Where: <var>    is any legal variable name.
 *
 *     Returns: <var>    holds total number of arguments. (Max of 6)
 *              <arg.0>  holds command text/name.
 *              <arg.1>  holds 1st argument.
 *                  |    thru
 *              <arg.6>  holds 6th argument.
 *
 *      Note 1: Max length of any one argument is 61 characters, and any
 *              unused arguments will contain the null string.
 */
```

1.15 getcursor

```
/**[14]*******************************************************************
 *
 * Description: Read "Cursor Key" / "Return/Enter" Keyboard Input.
 *
 *   Author(s): PMK              - (Peter no longer runs a bbs...)
 *              Dotoran          - Frontiers BBS             +1 716/823-9892
 *
 **************************************************************************
 *
 *       Usage: <var> = GETCURSOR()
 *
 *     Returns: <var> will be "8" if UP    arrow was pressed.
 *                    will be "2" if DOWN  arrow was pressed.
 *                    will be "4" if LEFT  arrow was pressed.
 *                    will be "6" if RIGHT arrow was pressed.
 *                    will be "5" if ENTER or RETURN pressed.
 *
 *      Note 1: Returned values are identical to the numeric keypad layout,
 *              so programs using this routine can be accessed by people who
 *              do not have directional cursor keys (A600, C64, etc.)
 *
 *      Note 2: If key pressed was none of the above, then <var> will hold
 *              the actual character that WAS pressed. Returned keys will be
 *              UPPERCASE to mimic the same action as the GETCHAR command.
 */
```

1.16 lower

```
/**[15]*******************************************************************
 *
 * Description: Convert "UPPERCASE" to "lowercase" text.
 *
 *   Author(s): Dotoran          - Frontiers BBS             +1 716/823-9892
 *              PMK              - (Peter no longer runs a bbs...)
 *
 **************************************************************************
 *
 *       Usage: <var> = LOWER(<text>)
 *
 *       Where: <var>   is any valid variable name.
 *              <text>  holds the text to be converted.
 *
 *     Returns: <var>   contains the converted lowercase text.
 */
```

1.17 pause

```
/**[16]*******************************************************************
 *
 * Description: Pauses output for "x" number of seconds the RIGHT way!
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *              PMK             - (Peter no longer runs a bbs...)
 *
 ****************************************************************************
 *
 *       Usage: call PAUSE(<seconds>)
 *
 *       Where: <seconds>  is the number of seconds to wait.
 *
 *        Note: This routine uses the DELAY() function, located in the
 *              support library "rexxsupport.library". See the included
 *              intro for more info on using this library.
 */
```

1.18 check

```
/**[17]**********************************************************************
 *
 * Description: Checks for "Loss of Carrier" in your Doors!
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *              Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ****************************************************************************
 *
 *     Usage 1: <var> = CHECK(<result>)
 *
 *     Usage 2: call CHECK
 *
 *       Where: <var>  is any valid variable name.
 *
 *      Note 1: Use [Usage 1] after you INPUT data using these commands:
 *              GETCHAR, RECEIVE, PROMPT, the MCI {i } Command, etc.
 *
 *      Note 2: Use [Usage 2] to simply CHECK for CARRIER. It’s a good
 *              idea to use a few of these calls in places where your
 *              program may be doing numerous things WITHOUT the user
 *              having to enter any input.
 */
```

1.19 syntax

```
/**[18]**********************************************************************
 *
 * Description: An informative "Error Checking" routine.
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ****************************************************************************
 *
 *       Usage: To use this routine, place it somewhere near the end of
 *              your program, then at the top of your program, normally
 *              right after your "options results" statement, place this
 *              line of text:
 *
 *              signal on SYNTAX ; signal on ERROR ; signal on IOERR
 *
 *     Returns: When an error is encountered, your program will halt. The
 *              user will be shown the error number and description, as well
 *              as the naame of the file the error occurred in. In addition,
 *              the line number and actual code found ON that line will also
 *              be displayed. Any CTRL-Y or CTRL-Q MCI codes contained ON the
 *              line will be rendered using \ and { for safety's sake. A copy
 *              of this same line will be copied to your "calls" log, or to
 *              an "ARexx_Says" log, if you have one defined.
 *
 *      Note 1: Each line is formatted for 46 characters, the maximum width
 *              stated on line 845 of BBSTEXT for inclusion into the logs.
 *              If you include MCI/ANSI color codes into these lines, then
 *              change the "%-.45s" on line 845 of BBSTEXT to read "%s".
 */
```

1.20 at

```
/**[19]**********************************************************************
 *
 * Description: Positions cursor for printing anywhere on the screen.
 *
 *   Author(s): PMK              - (Peter no longer runs a bbs...)
 *              Dotoran          - Frontiers BBS             +1 716/823-9892
 *
 ****************************************************************************
 *
 *       Usage: <command> AT(<row>,<col>)"<text>"
 *
 *       Where: <command>  could be TRANSMIT, SENDSTRING, QUERY, etc.
 *              <row>      is the row text will print on.
 *              <col>      is the column text will start at.
 *              <text>     is the text to be printed, within quotes.
 *
 *     Returns: will print given text at the given screen position.
 */
```

1.21 scroller

```
/**[20]**********************************************************************
 *
 * Description: Horizontal Text Scroller Number 1.
 *
 *   Author(s): Dotoran          - Frontiers BBS             +1 716/823-9892
 *
 ****************************************************************************
 *
 *       Usage: call SCROLLER(clr,ro1,co1,ro2,co2,dir,"txt")
 *
 *       Where: <clr>  Clear Screen First? (0=No, 1=Yes)
 *              <ro1>  Row to START scrolling at.
 *              <co1>  Column on "ro1" to START scrolling at.
 *              <ro2>  Row to STOP scrolling at.
 *              <co2>  Column on "ro2" to STOP scrolling at.
 *              <dir>  Scroll Direction: 0=Left, 1=Right, 2=Alternate
 *              <txt>  Text to be Scrolled, inside DOUBLE quotes.
 *
 *     Results: The <txt> line will be scrolled between the two columns
 *              on each ROW individually, starting at "ro1" and ending
 *              at "ro2". You can STOP the Scrolling prematurely by
 *              pressing any key.
 */
```

1.22 scroll

```
/**[21]********************************************************************
 *
 * Description: Horizontal Text Scroller Number 2.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *
 **************************************************************************
 *
 *       Usage: call SCROLL(<row>,<"txt">)
 *
 *       Where: <row>   is the Screen Row to be Scrolled.
 *              <"txt"> is the Text  Line to be Scrolled, within quotes.
 *
 *      Note 1: You can use three special characters inside your text
 *              string to affect the SPEED at which the text is shown:
 *
 *              Press ALT-1 ($^1$) for Fastest Speed.
 *              Press ALT-2 ($^2$) for Medium  Speed.
 *              Press ALT-3 ($^3$) for Slowest Speed.
 *
 *              This routine uses the DELAY() command to create the speeds,
 *              which means the "rexxsupport.library" is also needed.
 *
 *      Note 2: The text string will be scrolled from RIGHT to LEFT,
 *              starting at the right edge of the user's default Line
 *              Length(40,80,etc.) You can abort the scrolling at any
 *              time by pressing any key.
 *
 *      Note 3: An interesting alternate use for this routine is to
 *              scroll the EXISTING text on the screen. To do this,
 *              specify the "row" you wish to Scroll, then use "" as
 *              the Text to Scroll. Nothing NEW will appear on the
 *              screen, but any EXISTING characters ON that row will
 *              be scrolled off the left side of the screen!
 */
```

1.23 joy

```
/**[22]*********************************************************************
 *
 * Description: Read the joystick(s) and firebutton(s).
 *
 *   Author(s): Thomas          - Dreamline Amiga BBS       +45 3582-7043
 *              PMK             - (Peter no longer runs a bbs...)
 *
 **************************************************************************
 *
 *       Usage: <var>=JOY(<joynum>)
 *
 *       Where: <joynum>  is the Joy port (0=Port1, 1=Port2)
 *
 *     Returns: <var>     holds keypad values for directions, "0" if none.
 *                        and value+10 if the firebutton was pressed.
 *
 *      Note 1: This routine will allow the joystick(s) to be used from the
 *              LOCAL port only. It will NOT function from remote.
 *
 *      Note 2: Press your ENTER/RETURN key to exit the example given below.
 */
```

1.24 mouse

```
/**[23]*********************************************************************
 *
 * Description: First attempt at MOUSE capability. (95% Complete!)
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *              Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 **************************************************************************
 *
 *       Usage: <var> = MOUSE(<row>,<col>,<length>)
 *
 *       Where: <var>  is any valid variable name.
 *              <row>  is the ROW the button is located on.
 *              <col>  is the COLUMN the button STARTS at.
 *           <length>  is the LENGTH(in Columns) of this button.
 *
 *     Returns: <var>  will be "1" if the Mouse Button WAS pressed.
 *                     will be "0" is the Mouse Button WASN'T pressed.
 *
 *      Note 1: This routine is lacking the ability to catch when you
 *              "double-click" on a button, and will not ALWAYS catch
 *              when you DO click on a button. If anyone can offer us
 *              a better MOUSE() routine, please send us a copy and
 *              we'll include it in the next version(with your name)!
 *
 *      Note 2: The following example program can also be aborted by
 *              pressing any key, instead of using the mouse.
 */
```

1.25 nomore

```
/**[24]********************************************************************
 *
 * Description: Disable or Enable the MORE? prompt, regardless of setting!
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 *************************************************************************
 *
 *     Usage 1: call NOMORE
 *
 *     Usage 2: call MORE
 *
 *  Before Use: Add this line somewhere at the START of your file, so that
 *              it will only be run ONCE:
 *
 *              getuser 1100454;oldmore=result
 *
 *   After Use: When you're ready to LEAVE your file, place this line BEFORE
 *              every occurrance of the command EXIT:
 *
 *              setobject oldmore;putuser 1100454
 *
 *      Note 1: Use [Usage 1] when you wish to DISABLE the More? Prompt,
 *              Use [Usage 2] when you wish to ENABLE  the More? Prompt.
 *
 *      Note 2: Be sure to add the above two lines to insure the user's
 *              chosen More? setting is returned to it's original setting,
 *              especially within an Error Check routine you may be using!
 */
```

1.26 muffle

```
/**[25]********************************************************************
 *
 * Description: Muffle ALL ports, regardless of setting!
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 *************************************************************************
 *
 *       Usage: call MUFFLE
 *
 *  Before Use: Add this line somewhere at the START of your file, so that
 *              it will only be run ONCE:
 *
 *              getuser 1101745 ; muffle=result ; call MUFFLE
 *
 *   After Use: When you're ready to LEAVE your file, place this line BEFORE
 *              every occurrance of the command EXIT:
 *
 *              if muffle=0 then call MUFFLE
 *
 *      Note 1: Use "call MUFFLE" as a command in your file as well to give
 *              the USER the option of toggling the Muffle Setting.
 *
 *      Note 2: Be sure to add the above two lines to insure the user’s
 *              chosen Muffle setting is returned to it’s original setting!
 */
```

1.27 selfile

```
/**[26]*******************************************************************
 *
 * Description: Extended SelectFile Routine.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *
 ***********************************************************************
 *
 *      Usage: var=SELFILE(<file>,<bcost>,<fcost>,<kill>)
 *
 *      Where: <file>   is the file to add to the Select Buffer (incl. Path)
 *             <bcost>  "0" if the byte is FREE, "100" to deduct 1*bytesize
 *                      "150" to deduct 1.5*bytesize etc.
 *             <fcost>  "0" if the file is FREE, "1" for file price of 1
 *                      "2" for file price of 2 etc.
 *             <kill>   "0"=Don’t Kill, "1"=Kill when downloaded,
 *                      "2"=Kill when dl/unselect, "3"=Kill when unselected.
 *
 *    Returns: <var>    "0" If selectbuffer is full.
 *                      "1" If selecting was sucessfull.
 */
```

1.28 log

```
/**[27]*******************************************************************
 *
 * Description: Add line of text to specified LOG file.
 *
 *   Author(s): Dotoran         - Frontiers BBS            +1 716/823-9892
 *
 ***********************************************************************
 *
 *       Usage: call LOG("<name>","<text>","[path]")
 *
 *       Where: <name>  is the NAME of the LOG file to add to.
 *              <text>  is the TEXT line to be added to the log.
 *              [path]  if present, this specifies an alternate path to
 *                      SAVE the LOG to. Defaults to "SysData:Log/"
 */
```

1.29 pmenu

```
                /**[28]*********************************************************************
 *
 * Description: Check Port Menu(s) Checkmark Status.
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *              PMK             - (Peter no longer runs a bbs...)
 *
 ***************************************************************************
 *
 *       Usage: call PMENU(<port>,<item>)
 *
 *       Where: <port>  is the PORT to check. (Use 100 for ALL Ports).
 *              <item>  is the item to check. Although you can type as much
 *                      of the menu item text as you wish, only the first
 *                      letter matters, as shown below:
 *
 *      For CNet, v3.05c's Routine:         For CNet, v4.26b's Routine:
 *      ~~~~~~~~~~~~~~~~~~~~~~~~~~~         ~~~~~~~~~~~~~~~~~~~~~~~~~~~
 *      's' to check "Sysop is in".         'd' to check "Doors closed".
 *      'n' to check "No new users".        'f' to check "Files closed".
 *      'u' to check "UD base closed".      'm' to check "Msgs closed".
 *      'p' to check "Pfiles closed".       'n' to check "No new users".
 *      'b' to check "Base closed".         's' to check "Sysop is in".
 *
 *     Returns:  0  if there is NO checkmark shown.
 *               1  if there IS a  checkmark shown.
 *
 *        Note: There are TWO DIFFERENT "PMENU" routines listed below. Not
 *              only did the GetUser change between versions, but the ORDER
 *              of the Menu Flags changed as well. Make sure to use the right
 *              version, depending on your CNet version. If creating a dual
 *              compatible program, use the
                CHECKVER()
                 routine above, then
 *              include BOTH routines given below, perhaps RENAMING them to
 *              "PMENU3" and "PMENU4", calling the needed routine, based on
 *              your returned result from
                CHECKVER()
                .
 */
```

1.30 mail

```
/**[29]*********************************************************************
 *
 * Description: Send Text File as CNet MAIL to specified User.
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Usage: call MAIL(<id>,"<subj>","<file>")
 *
```

```
 *       Where: <id>    can be the ID Number, Handle or Real Name of the
 *                      user in which you'd like to send the mail item to.
 *                      If a Handle or Real Name is specified, enclose it
 *                      in double quotation marks.
 *
 *              <subj>  is the Subject to name the Mail Message.
 *
 *              <file>  if the path/filename of the text file to send.
 *
 *     Returns:  0  if mail send FAILED. (File Not Found/Box Closed or Full)
 *               1  if mail was sent successfully.
 *
 *       Notes: The example given below utilizes a text file created using
 *              the "BUGS" command, a BBSMenu enhancement found in the TEXT
 *              ToolKit.
 */
```

1.31 fmail

```
/**[30]*********************************************************************
 *
 * Description: Send a CNet/4 "File Attach" EMail Message to specified User.
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ****************************************************************************
 *
 *        Note: This routine ONLY runs under CNet, v4.26b!
 *
 *       Usage: call FMAIL(<id>,"<subj>","<text>","<file>")
 *
 *       Where: <id>    can be the ID Number, Handle or Real Name of the
 *                      user in which you'd like to send the mail item to.
 *                      If a Handle or Real Name is specified, enclose it
 *                      in double quotation marks.
 *
 *              <subj>  is the Subject to name the Mail Message. Enclose in
 *                      double quotes if subject contains SPACES.
 *
 *              <text>  is a text file that will become the BODY of the EMail
 *                      Message. Enclose the text in double quotes if it will
 *                      contain SPACES in the filename.
 *
 *              <file>  is the path/filename of the text file to attach to
 *                      the email message created. Again, if the name will
 *                      contain SPACES, enclose it in double quotes.
 *
 *     Returns:  0  if mail FAILED. (Text or File Not Found)
 *               1  if command string sent successfully.
 *
 *       Notes: The example given below utilizes a text file created using
 *              the "BUGS" command, a BBSMenu enhancement found in the TEXT
 *              ToolKit. Also, no internal checking is done on whether the
 *              mail send itself was successful. This routine creates the
 *              needed command string, which is then sent to CNet's Keyboard
 *              Buffer. If the user has the VisEd set as their default, then
 *              the appropriate extra commands are prepended and suffixed to
 *              the mailsend commands to temporarily switch to the LineEd, so
 *              the proper dot(.) commands can be executed.
 */
```

1.32 sysolm

```
                /**[31]*******************************************************************

 *
 * Description: Send a System OLM (OnLine Message) to the current user.
 *
 *   Author(s): Dotoran             - Frontiers BBS         +1 716/823-9892
 *              PMK                 - (Peter no longer runs a bbs...)
 *
 ***************************************************************************
 *
 *     Inputs: This routine uses the
                CHECKVER()
                 routine above to define the
 *             "cv" variable, which the SYSOLM() routine uses. You also need
 *             to define variables "handle", "port", and "uid" before calling
 *             this routine.
 *
 *             Routine works on BOTH v3.05c and v4.26b of CNet Amiga!
 *
 *      Usage: call SYSOLM(<msg>)
 *
 *      Where: <msg>  is the Message to appear as a "System OLM", enclosed
 *                    in double quotation marks.
 *
 *     Option: If you wish to add MCI-Commands into your OLM-text, then you
 *             need to REMOVE the \@1 from the END of line number 920 in your
 *             BBSTEXT file, so it reads:
 *
 *             920:   \n1\c7**** System Message\n1\a1
 *
 *             Remember, the \'s are really CONTROL-Y's.
 *
 *       Note: The \'s and {'s used in the example given below are actually
 *             Control-Y's and Control-Q's, used to show the COLORIZED OLM
 *             ability of this routine. Make the proper substitutions when
 *             you try this routine on your ends, as well as modifying Line
 *             920 as stated above.
 */
```

1.33 addem

```
/**[32]*********************************************************************
 *
 * Description: Add keystrokes to other ports from present port.
```

```
*
*   Author(s): Aunt Bea        - Blue Moon BBS             +1 716/871-9866
*              Dotoran         - Frontiers BBS             +1 716/823-9892
*
****************************************************************************
*
*       Usage: call ADDEM(<port>,"<keys>",<mode>)
*
*       Where: <port>  is the port number to add keystrokes to.
*              <keys>  are the keystrokes to add, within double quotes.
*              <mode>  as "1" and the user will see them,
*                      as "2" and the user will NOT see them.
*
*     Returns: Keystrokes will be entered into the command stream on stated
*              port. Nothing will be returned on your port.
*
*    Example1: call ADDEM(2,"o!",2) will LogOff user on Port 2 without them
*                                   knowing what just happened!
*
*    Example2: call ADDEM(1,"ep;9;1;;",1) will set the Visual Editor as the
*                                         default editor for user on Port 1
*                                         allowing them to see the command
*                                         being processed on their screens!
*
*       Notes: The serial port is DISABLED when you choose for the user NOT
*              to be able to see what commands you just entered. For this
*              reason, YOU on the LOCAL port WILL see the commands being
*              executed, however the text output is NOT sent over the serial
*              port.
*/
```

1.34 who

```
               /**[33]*******************************************************************

*
* Description: A QUICK "Who" for SysOps, listing Access Group Number.
*
*   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
*              Aunt Bea        - Blue Moon BBS             +1 716/871-9866
****************************************************************************
*
*      Inputs: The
               CHECKVER()
                routine is used to create 3.05c/4.26b CNet
*              compatibility.
*
*       Usage: call WHO
*
*     Returns: Displays all loaded ports, listing port number, handle,
*              access group, speed, from and where info.
*/
```

1.35 logpo

```
/**[34]*********************************************************************
 *
 * Description: View "port" log of specified port. (Pre "calls" log).
 *
 *   Author(s): Aunt Bea        - Blue Moon BBS             +1 716/871-9866
 *              Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Usage: call LOGPO(<port>)
 *
 *       Where: <port>  is the port number to view the log of.
 *
 *     Returns: Displays the "calls" log entry for this user, as it looks
 *              so far. By activating other log processes through CONFIG
 *              without assigning other log names for them, you can see
 *              what the user has done up to that point this call.
 */
```

1.36 lnoise

```
                /**[35]*******************************************************************

 *
 * Description: Send Line Noise to a port (Ability to kick them off too!)
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *              Aunt Bea        - Blue Moon BBS             +1 716/871-9866
 *
 ***************************************************************************
 *
 *      Inputs: Define a "cv" variable by calling the
                CHECKVER()
                 routine to
 *              determine the correct CNet Version!
 *
 *       Usage: call LNOISE(<port>,<drop>)
 *
 *       Where: <port>  is the port number to send Line Noise to.
 *              <drop>  as "1" will also DROP CARRIER on that port,
 *                      as "0" will NOT drop carrier. Just Annoy! hehe
 *
 *     Returns: A check is done to make sure that the user using this routine
 *              is a Conference Controller and that the port number entered
 *              is a valid number. A check is also done to make sure the
 *              user issuing the Line Noise doesn't lose carrier themselves.
 */
```

1.37 unlock

```
/**[36]*********************************************************************
 *
 * Description: UnLock User Accounts (That May NOT Have Been Previously!)
 *
 *   Author(s): Aunt Bea        - Blue Moon BBS             +1 716/871-9866
 *              Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Usage: call UNLOCK(<id>)
 *
 *       Where: <id>  is the ID number of the account to UnLock. This can
 *                    also be specified as the Handle or Real Name of the
 *                    user whose account you wish unlocked. Use "0" to
 *                    UnLock ALL accounts on your system.
 *
 *     Returns: Will tell you when it's done.
 *
 *    WARNING!: You should ONLY run this file when there are NO OTHER ARexx
 *              tasks running simultaneously, as if one of these other tasks
 *              were to LOCK an account, Unlocking it prematurely may cause
 *              THAT task to fail or crash. This routine is meant as a FIX
 *              for any files using LOADSCRATCH where you believe there to
 *              be a problem with it not UNLOCKING the accounts.
 */
```

1.38 replace

```
/**[37]*********************************************************************
 *
 * Description: Replace <input> with <output> within string of <text>.
 *              (A bit like the AREXX's TRANSLATE command, but NOT limited
 *               to replacing text of equal length)
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *
 ***************************************************************************
 *
 *       Usage: <var>=REPLACE(<text>,<input>,<output>)
 *
 *       Where: <text>    holds the text to do replacement on.
 *              <input>   is the text you want to replace.
 *              <output>  is the text you want to appear instead of <input>.
 *
 *     Returns: <var>     holds the replaced <text>.
 */
```

1.39 bmtxt

```
/**[38]*********************************************************************
 *
 * Description: Find and Return or Verify BBSMENU Line(s).
 *
 *   Author(s): Dotoran         - Frontiers BBS             +1 716/823-9892
 *              PMK             - (Peter no longer runs a bbs...)
 *
 ***************************************************************************
 *
 *       Usage: <var>=BMTXT(<menu>,[<line>])
 *
 *       Where: <menu>  BBSMENU Menu Number.
 *              <line>  Line Number in Menu OR command to Verify.
 *
 *     Returns: <var>   when <line> is NUMERIC, holds the returned BBSMENU
 *                      Menu Line entry. If <line> is text, a "1" is returned
 *                      if a command by that name exists in that menu. ("0"
 *                      is returned if no such command exists. If <line> is
 *                      omitted, the number of lines in that BBSMENU Menu are
 *                      returned in <var>, while the "BMT.<x>" array will hold
 *                      the actual BBSMENU listing.
 */
```

1.40 dumpuser

```
/**[39]*********************************************************************
 *
 * Description: Clears a specific port, by dumping the user. Similar to
 *              the DROPCARRIER command, but allows different log entries.
 *
 *   Author(s): PMK             - (Peter no longer runs a bbs...)
 *              Dotoran         - Frontiers BBS             +1 716/823-9892
 *
 ***************************************************************************
 *
 *       Usage: call DUMPUSER(<port>,<logoff>,[<quick>])
 *
 *       Where: <port>    is the port number to clear. (dumps user)
 *              <logoff>  as  "0" shows "TIME LIMIT EXCEEDED" in the log.
 *                        as  "1" shows "AUTO CALL-BACK FAILED" in the log.
 *                        as  "2" shows "IDLE TIME EXCEEDED" in the log.
 *                        as  "3" shows "LOST CARRIER" in the log.
 *                        as  "4" shows "INSTANT LOGOFF" in the log.
 *                        as  "5" shows "NORMAL LOGOFF" in the log.
 *                        as  "6" shows "RE-LOGON" in the log.
 *                        as  "7" shows "$ BALANCE TOO LOW" in the log.
 *                        as  "8" shows "DUMPED BY SYSOP" in the log.
 *                        as  "9" shows "FILE XFER AUTO-LOGOFF" in the log.
 *                        as "10" shows "MCI % COMMAND" in the log.
 *                        as "11" shows "TERM LINK" in the log.
 *              <quick>   if "1", a quick logoff is performed. (similar to
 *                        the normal "O!" - No SYS.END is displayed.)
 *
 *       Notes: A check is done to make sure that the port number entered
 *              is a valid number, and the port is occupied by a user.
 *
 *              If a type "6", RE-LOGON, is specified, the user will LogOff,
 *              and then BACK on, using Re-Logon, EVEN if the user does NOT
 *              have the "RELOGON" priviledge flag set! (kinda cool!)
 *
 *     Returns: "0" if the dumping failed (no user on port, bad port etc.)
 *              "1" if the user was successfully dumped.
 */
```

1.41 encode

```
/**[40]*********************************************************************
 *
 * Description: EnCode & DeCode text strings, using a Numeric Key. Given the
 *              desired text string, along with a numeric key, the text will
 *              be encoded using a specific code string. The text can then
 *              only be decoded using the same numeric key.
 *
 *   Author(s): Dotoran         - Frontiers BBS             +716 823-9892
 *
 *************************************************************************
 *
 *       Usage: To EnCode a text string, use the ENCODE() function:
 *
 *              <var> = ENCODE( <text> , <key> )
 *
 *       Where: <var>  is the variable the encoded text will be placed.
 *              <text> is the text string(or variable holding text string)
 *                     that needs to be encoded.
 *              <key>  is a numeric value between 1 and 94. Values below 1
 *                     or greater than 94 will return INCORRECT results!
 *
 *       Usage: To DeCode a coded text string, use the DECODE() function:
 *
 *              <var> = DECODE( <text> , <key> )
 *
 *       Where: <var>  is the variable the decoded text will be stored in.
 *              <text> is the ALREADY CODED text string you wish to DeCode.
 *                     This can also be a variable containing coded text.
 *              <key>  is the SAME numeric key you used to MAKE the initial
 *                     coded string. If you do NOT use the SAME numeric key,
 *                     then the text will NOT be DeCoded correctly.
 *
 *       Notes: This technique comes in real handy when you wish to encrypt
 *              data before saving it to disk. The data can then be decoded
 *              as it is read in the next time it is needed.
 */
```

1.42 guidelines

```
/**[41]*********************************************************************
 *
 * Description: Guideline Entry-text for Mail (and File) Subboards.
 *              This will display a small bar, with some of the current
 *              subboard settings, like Sigs allowed?, Handles allowed?,
 *              Subboard Inactivity days. - Add it to your entry texts.
 *
 * Screenshots: Guidelines4BEV1.iff and Guidelines4BEV2.iff
 *
 *   Author(s): Dotoran         - Frontiers BBS             +716 823-9892
 *
 ***************************************************************************
 *
 *       Usage: There are three versions of the Guideline file for BOTH CNet
 *              versions, an Ascii, ANSI, and IBM-ANSI Beveled. Choose the
 *              version you wish to use, then use it's filename in place of
 *              the <guidelines> tag used in the lines below.
 *
 *  These stand alone ARexx files can be used in one of two ways:
 *
 *  1. You can launch them from within a base's "sys.entry" file by doing
 *     the following:
 *
 *          - Enter the base in question, then type ENTRY.
 *          - Once in the "sys.entry" editor, type the following line:
 *
 *                  {#0<path><guidelines>}
 *
 *            Where: <path>         is the location you chose to STORE the
 *                                  ARexx Base Guidelines file.
 *
 *                   <guidelines>   is the filename of the Base Guidelines
 *                                  you'd like to use.
 *
 *          - Save the editor and you're done. You can create additional
 *            "sys.entry" files for additional bases.
 *
 *  2. Add the following code to the BEGINNING of Line 316 in BBSText:
 *
 *              {#0<path><guidelines>}
 *
 *     So the completed line looks like this:
 *
 *              {#0<path><guidelines>\n1\c7*Subboard \c6(\c7{v48}\c6     <--
 *              ) \c7{v49}\n1\q1
 *
 *     This method will display the selected Guidelines upon entry into
 *     EVERY base created on your system.
 *
 *     Options: Method 1 above offers another advantage over method 2, as
 *              you also have the ability of specifying alternate Guideline
 *              designs in different bases. (For instance, perhaps you'd
 *              like to use the ASCII version in bases that would be more
 *              frequented by users that can't support IBM-ANSI graphics,
 *              and use the IBM-ANSI in areas where users CAN support them).
 *              Additionally, using method 1 allows you NOT to use Guidelines
 *              in certain OTHER areas as well.
 *
 *              Method 2 offers it's own advantages. The most noticable being
 *              the fact that you only need to modify ONE file(your bbstext),
 *              instead of having to create numerous "sys.entry" files. The
```

```
 *              other major advantage is that the Guidelines will be shown to
 *              EVERY user, regardless of their Help Level setting, whereas
 *              in Method 1, the Guidelines are ONLY shown to those users of
 *              the "Novice" Help Level.
 */
```

1.43 newdoor

```
                /**[42]*******************************************************************

 *
 * Description: NewDoor starter framework for New ARexx Doors/Pfiles.
 *              (This is similar to the "empty.c" file "C" coders have)
 *
 *   Author(s): Dotoran         - Frontiers BBS             +716 823-9892
 *
 ***************************************************************************
 *
 *       Usage: Included in the CNet Amiga ToolKit, v3.00 is a file titled
 *              "NewDoor". This file can be used by ARexx programmers as
 *              a starting framework for NEW projects. Below is a descriptive
 *              breakdown of the lines found IN that file, so you'll know
 *              exactly what each line is there for! Lines starting with [>
 *              are comments by me, and are NOT part of the NewDoor file!
 */

-- NewDoor STARTS Here --

/**************************************************************************\
    $VER: , v. (--97) by SysOp of BBS!
\**************************************************************************/

[> The $VER line allows you to create a VERSION string for your program. You
[> should place the NAME of your file BEFORE the comma above. Put the version
[> number after the "v", and it's creation date in DD-MMM-YY format within
[> the paries (Using DD-MMM-YY format eliminates confusion between different
[> date formats, like DD-MM-YY (USA), MM-DD-YY (Europe), etc.) Replace SysOp
[> with the Handle or Real Name of the author, and replace BBS with the name
[> of the BBS this file was either created on/for, or where the file can be
[> located for download(in the event the author doesn't run a BBS).

options results;signal on SYNTAX;signal on ERROR;signal on IOERR

[> Standard initial ARexx stuff. Nothing major here. ;-)

a=sourceline(2);parse var a . ", "ver" ("vdate")" .;a=random(,,time("s"))

[> This line creates two variables, "ver" and "vdate". The "ver" variable
[> will contain the VERSION number of the program (like v1.00 or v3.42), and
[> "vdate" will contain the file's creation date (like 10-May-97). You can
[> then USE these variables throughout the program whenever you wish this
[> information displayed to the screen. (This line grabs it's info directly
[> from the $VERsion string, so make sure to follow the correct format, as
[> described above). This line also seeds the random number generator, so as
[> to insure a good randomity to any random numbers your file may be using.
[> If your file will NOT be using the RANDOM() function, you can DELETE the
[> code that states:    ;a=random(,,time("s"))

tr=transmit;se=sendstring;gc=getchar;gu=getuser;gs=getscratch;mg=maygetchar

[> I’ve used these before, and will mention them again. This line creates a
[> series of command aliases, or abbreviations, you can choose to use within
[> your program INSTEAD of using the entire command itself. For instance, the
[> following line would be valid:    se "Want to continue? [Yes]: ";gc

[> Again, once your program is complete, you can DELETE any abbreviations you
[> may not have used in your program. Other aliases I’ve used include these:
[> pu=putuser ; ps=putscratch ; qu=query ; pr=prompt

a="rexxsupport.library";if ~show("l",a) then if ~addlib(a,0,-30) then exit

[> This line insures the ARexx Support library is available for use. This
[> line is not necessary if your program does not use any of the Support
[> functions: ALLOCMEM(), CLOSEPORT(), FREEMEM(), GETARG(), GETPKT(),
[> OPENPORT(), REPLY(), SHOWDIR(), SHOWLIST(), STATEF(), or WAITPKT().

parse source . . fp .;df=left(fp,max(lastpos('/',fp),lastpos(':',fp)))

[> This line creates the "fp" and "df" variables, for use with the Smart Home
[> Path technique for locating the programs origination path. In "fp", you’ll
[> be given the ENTIRE "FilePath" to the program in question, while "df" will
[> return ONLY the PATH of the program. For instance, if the "NewDoor" file
[> was stored in "DOORS:", fp="doors:newdoor", while df="doors:". These two
[> variables, especially the "df" one, come in REAL handy when you’d like the
[> program to be able to find an included config or data file that resides in
[> the SAME directory as the parent file. This line allows the SysOp using
[> your program to be able to STORE the file ANYWHERE on their systems, and
[> it will be able to find it’s support files (provided the SysOp remembers
[> to keep ALL the files TOGETHER).




[> The open GAP in the program is where YOU would include YOUR ARexx coding.

exit

[> As a safety measure, the "exit" insures that whatever code comes BEFORE
[> it doesn’t mistakenly run into the Carrier Check routine below.

CHECK:;if ARG() & ARG(1)~="###PANIC" then return ARG(1)
  getcarrier;if result="TRUE" then if ARG() then return ARG(1);else return
  logentry "Lost Carrier!!";bufferflush;exit

[> This is a "Loss of Carrier" check. Check the
                CHECK
                 routine for more info
[> on it’s correct usage.

SYNTAX:;ERROR:;IOERR:;e1="?n1 Error: "rc" ("errortext(rc)")"
  e2="  Line: "left(sigl,4)"File:";c="`"fp", "ver"'";e2=e2" "c;tr e1;tr e2
  logentry e1;logentry e2;e=strip(translate(sourceline(sigl),"\{","??"))
  do while e~="";e3="Source: "left(e,37);tr e3;logentry e3;e=substr(e,38);end
  bufferflush

[> This is a modified version Error Check routine that has been customized
[> for use within the "NewDoor" file. It utilizes the "fp" variable defined
[> at the START of this file to include the NAME of the file. It also uses
[> the "ver" variable, also defined above, to include the VERSION NUMBER of
[> the program in the returned error message. Note that the "tr" command
[> alias has also been used in this routine. This alias was defined above
[> as well. The returned Error Message contains the same information as the
[> original routine, described earlier in this document, under the
                SYNTAX
                [> Routine discussed above.

[> Please REMEMBER that the above routine is CUSTOMIZED for use INSIDE this
[> ARexx program. Do NOT copy and paste this routine into other ARexx files,
[> unless you'll also be using the new header info as well, else the routine
[> will NOT function correctly. In fact, it'd be rather ironic to have an
[> Error CHECKING routine that had an ERROR in it ITSELF! hehe ;-)

/***************************************************************************\
\************************************************ Your BBS (AAA)/PPP-SSSS **/

[> To finish up, this comment line serves two purposes. It tells the reader
[> they've reached the end of the code, as well as giving the PHONE NUMBER
[> of the Support BBS they should contact in the event any Bugs are found!

-- NewDoor ENDS Here --
```