InputStream

COLLABORATORS			
	TITLE : InputStream		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		November 24, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

InputStream

1.1 main

```
                InputStream

            © 1994 TVI Interactive,  by Robert Hardy

            UUCP: Robert_Hardy@tviney.com
                or
            Fido: 1:153/250.0


i) Contents

        1)  Legal Stuff
        2)  System Description
        3)  Program Startup
        4)  Rexx Commands
        5)  Installation
        6)  Input Strings
```

1.2 legal_stuff

```
1) Legal Stuff

1.1) Disclaimer

    InputStream is provided "AS IS", WITHOUT ANY WARRANTY to its
    quality, performance or fitness for a particular purpose.  In
    no event shall the author be liable or responsible to the user
    or any other person, for any kind of damage caused by the use
    of this software.

1.2) Conditions of Use and Distribution

    InputStream is MessageWare, if you like and use it, send me a
    message (Internet or Fido).  Feel free to include complaints or
    suggestions.
```

InputStream may be freely distributed provided:

1. The files are ALL left intact and unmodified.

2. No charge is made for InputStream (other than a reasonable copy charge)

3. InputStream is not packaged as a part of a commercial product without prior written consent.

4. InputStream may not be uploaded to any electronic service that claims a copyright to it's files and programs.

1.3) System Requirements and Limitations

InputStream requires AmigaDOS V2.04 or higher.

1.3 system_desc

2) System Description

InputStream is a rexx server that gives rexx scripts the ability to feed almost anything they want into the input stream. Special input events are surrounded by angle brackets <>, see Input Description Strings. Anything not surrounded by angle brackets is inserted as is.

As an added bonus, windows and screens can be manipulated directly by title.

1.4 startup

3) Program Startup

The program takes no arguments or tooltypes. It detaches on startup and outputs nothing. Simply type the name at the shell prompt or double click it's icon. Running it again will shut it down.

1.5 rexx

4) Rexx Commands

STRING: (syn KEY)

This command will insert any character, string or input event into the input stream. Input events are surrounded by angle brackets <>, see Input Description Strings. Anything not surrounded by angle brackets is inserted as is.

Arguments:

```
        <string> - Insert one or more characters into the input stream.
      Or
        VALUE <number> - Insert a character into the input stream.

    EG:
        address 'input_stream_rx'

        KEY 'VALUE 7'       /* Insert a bell character */

        STRING '<ralt rshift return>this is a test' /* Bring CED to the front
                                                     * and insert the string
                                                     */

MOUSE:

    This command will insert a mouse event into the input stream.
    Absolute and relative mouse moves can be fed or a button up or
    down event.

    Arguments:

         REL <x> <y>  - Move the mouse pointer relative to it's
                        current position.
      Or
        ABS <x> <y>  - Move the mouse pointer relative to the top
                        left of the screen.
      Or
        BUTTON L|R|M U|D - Insert a left, right or mid button event.
                           Also specify whether it is an up or down
                           event.

    EG:

        address 'input_stream_rx'

        MOUSE 'abs 0 0'     /* Move to the top left corner */

        MOUSE 'BUTTON L D'  /* insert a left mouse button down event */

WINDOW:

    This command will maniptulate windows.  Be carefull,  some of the
    options can be dangerous with some applications.  All arguments
    are executed as encountered.  Order is not important.

    Arguments:

        <name> - Title of the window to play with.  The name is not
                 case sensitive.  The full window title is not needed.
                 Only enough to make it unique.

        Options:

            FRONT  - Brings the window to front.

            BACK   - Send the window to the back.
```

```
            SELECT - Activate the window.

            ZOOM   - If the window has a zoom gadget, the window will
                     switch to it's alternate size and position.

            SIZE <dx> <dy> - Adjust the size of a window.

            MOVE <dx> <dy> - Move the window.

            POS_SIZE <left> <top> <width> <height>

                           - Abritrarily set the size and position of
                             the window.


    EG:

        address 'input_stream_rx'

        WINDOW 'Cygnus back zoom'         /* Flip Ced window to the back
                                           * and shink it.
                                           */

        /* Do some stuff */


        WINDOW 'CygnusEd zoom front select' /* Unshrink Ced, bring it to
                                             * the front and activate it
                                             */



SCREEN:

    This command will maniptulate screens.  Be carefull,  some of the
    options can be dangerous with some applications.  All arguments
    are executed as encountered.  Order is not important.

    Arguments:

        <name> - Title of the screen to play with.  The name is not
                 case sensitive.  The full screen title is not needed.
                 Only enough to make it unique.


        Options:

            FRONT - Bring the screen to the front.

            BACK  - Send the screen to the back.

            MOVE <dx> <dy> - Shift the screen's position

    EG:

        address 'input_stream_rx'
```

```
          SCREEN 'VLT front'  /* Bring VLT's screen to the front */


   BYE:

       Signal the server to shut down.

       Arguments:

           None.

       EG:

           options results

           we_ran_it = 0

               /* Find Out if it is already running */
           if ~show('p', 'input_stream_rx') then
           do
              we_ran_it = 1

              address command 'ex:InputStream'

                     /* Wait till it's loaded */
              address command 'waitforport input_stream_rx'
           end

               /**********************
                ** Do Your stuff...
                **********************/

           if we_ran_it = 1 then BYE
```

1.6 installation

```
5) Installation

Copy InputStream to somewhere convenient that's in your Path.  Then copy this
doc whereever you keep your docs.

To add InputStream to your User-Startup,  simply add the line...

InputStream

Or simply drag it's icon to your WBStartup drawer.

To use it 'on demand', see the example under the BYE command.
```

1.7 input_strings

```
Note:

    This description is copied directly from the docs for one of
    Stefan Sticht's commodities.

6) Input Description Strings

With input description strings you can specify almost any input
action, for example the action lshift f1, which means that pressing
the left shift and the f1 key together is the action. In this
commodity you can specify the action to open the commodity's window,
as described above.

Input description strings have the following template:

      [class] (([-]qual)|syn)* [[-]upstroke] [highmap|ANSIcode]

(* means zero or more occurances of the of the expression in brackets)

class   is one of the following strings:
        rawkey, rawmouse, event, pointerpos, timer, newprefs,
        diskremoved, diskinserted.
        If not specified, the class is taken to be "rawkey".

qual    is one of the strings:
        lshift, rshift, capslock, control, lalt, ralt, lcommand,
        rcommand, numericpad, repeat, midbutton, rbutton, leftbutton,
        relativemouse
        A preceding '-' means that the value of the corresponding
        qualifier is to be considered irrelevant.

syn     (synonym) is one of the strings: shift, caps, alt
        shift means "left or right shift"
        caps means "shift or capslock"
        alt means "either alt key"

upstroke (literally "upstroke")
        if this token is absent, only downstrokes are considered
        for rawmouse (mousebuttons) and rawkey events.  If it is
        present alone, only upstrokes count.  If it preceded by
        '-' it means that both up and down strokes are included.

highmap one of the strings:
        comma, space, backspace, tab, enter, return, esc, del, up, down,
        right, left, help, f1, f2, f3, f4, f5, f6, f7, f8, f9, f10,
        0, 1, 2, 3, 4, 5, 6, 7, 8, 9, (, ), /, *, -, +

ansicode a single character token is interpreted as a character code,
        which is looked up in the system default keymap.
```