Injector.doc

COLLABORATORS			
	TITLE : Injector.doc		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		February 24, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Injector.doc **1**

1.1 Injector documentation - English . . . 1
1.2 1.1 Distribution and disclaimer . . . 1
1.3 1.2 Credits and special thanks . . . 2
1.4 1.3 System requirements . . . 3
1.5 2.1 Introduction . . . 3
1.6 2.2 Starting Injector . . . 3
1.7 2.3 ToolTypes . . . 4
1.8 The PUBSCREEN tooltype . . . 4
1.9 The CX_PRIORITY tooltype . . . 4
1.10 The PREFSPATH tooltype . . . 4
1.11 The AREXXCONSOLE tooltype . . . 4
1.12 2.4 Removing Injector . . . 5
1.13 3.1 Starting the preferences program . . . 5
1.14 3.2 The Graphics User Interface . . . 6
1.15 3.3 Language reference . . . 7
1.16 The FREQ_SHOW command . . . 8
1.17 The FREQ_INJFILE command . . . 8
1.18 The FREQ_INJDIR command . . . 8
1.19 The FREQ_INJPATH command . . . 9
1.20 The SREQ_SHOW command . . . 9
1.21 The SREQ_INJECT command . . . 9
1.22 The CHARS_INJ command . . . 9
1.23 The CHARS_CLIP command . . . 9
1.24 The CHARS_ENV command . . . 10
1.25 The TIME_INJDATE command . . . 10
1.26 The TIME_INJTIME command . . . 10
1.27 The TIME_INJDAY command . . . 10
1.28 The EXEC_NAMED command . . . 11
1.29 The EXEC_REXX command . . . 11

1.30 The EXEC_PREFS command 11
1.31 The CMD_QUIT command 11
1.32 The CMD_UPDATE command 11
1.33 4.1 Defining hotkeys 12
1.34 4.2 History 13
1.35 4.3 Contacting the author 15

Chapter 1

Injector.doc

1.1 Injector documentation - English

Welcome to Injector 2.00 copyright 1994 Frédéric Delacroix.

This is the document that must be spread with all other files of the package. It is to be viewed by AmigaGuide (copyright Commodore), but can also be read by human eyes, with less comfort.

TABLE OF CONTENTS

1 FISRT OF ALL:
  1.1 Distribution <- Important !
  1.2 Credits and thanks
  1.3 System requirements

2 INSTALLING INJECTOR:
  2.1 Introduction
  2.2 Starting Injector
  2.3 ToolTypes
  2.4 Removing Injector

3 CONFIGURING INJECTOR:
  3.1 Starting the preferences program
  3.2 The Graphics User Interface
  3.3 Language reference

4 APPENDIX
  4.1 Defining Hotkeys
  4.2 History
  4.3 Contacting the author <- Do it !

1.2 1.1 Distribution and disclaimer

Injector is Copyright 1994 by Frédéric DELACROIX. Permission to copy & distribute it is granted to anyone who follows this conditions (generally known as SHAREWARE):
  - All files (or the whole archive) are distributed together (this

includes the executables, documentation files, catalog files, the catalog transcription file and all the icons). There is one exception though: If you're planning a release for a "mono-linguistic" community (say, for example, France only), you are allowed to distribute only the files that are relative to your language, that is to say: the doc file and the catalog file. All other files must be present.
- All distributed files remain UNMODIFIED IN ANY WAY (no silly "spread by..." message). If you have comments to add, do so in a separate file! Archiving is although allowed, but executable crunching is not recommended as the program is self-detaching (and thus needs to cut its SegList).
- You may NOT ask for money for this program. A small fee for copy & sending is allowed, but you may NOT charge more than Fred Fish does for a single AmigaLib disk.

- This is for all users of the Injector: as the program is released as SHAREWARE, you must send me a little contribution of $10 (or equivalent) if you continue to use it after a short evaluation period. If you want the source of the program (written for Devpac 3), just add $10 more and I will send it to you. My address can be found at the end of the document.

Injector was tested a lot, but I cannot guarantee that it will always work as expected. I will not be held responsible for any direct or indirect damage/ loss of data that might result by the use of this program. Remember: you use it AT YOUR OWN RISK.

Special note: the catalogs and doc files are only available in english and french for now (my school spanish is soooo bad :-). So if you can make a translation of these files into your own language, I would appreciate you send it to me so I can include it in a future release. For the doc file, it's easy: just edit a copy of this one (please, don't change the node names, they are not displayed anyway). For the catalogs, just fill in the .ct files with your own strings and send it to me. I will send you corresponding catalogs in return. Please note that some strings (like the Commodities description message and gadget labels) are limited in length. For some, there is a shortcut key (in upper case please) or menu equivalent right before the label. You could also draw icons for Injector and the prefs program that would be less ugly (I'm not a graphist :-).

1.3 1.2 Credits and special thanks

Injector was written on the wonderful HiSoft Devpac 3, running on an old A500 with OS 2.1. It uses the great reqtools.library which is Copyright Nico François and errormsg.library, which is copyright by myself. ARexx is copyright ©1987 by William S. Hawes.

The concept of Injector was originally implemented in the FR_Bypass program that was part of the kd_freq.library (Copyright Khalid Aldoseri) distribution, but I have totally rewritten the program, and now it doesn't need that library any more. You can still use a patch program like RTtoKD to replace reqtools's FileRequester by kd_freq's.

1.4 1.3 System requirements

Injector should work on any amiga that matches these requirements:

- You need Kickstart 2.04 or newer (V37+), or Injector will refuse to run. The only thing to do if you are still under 1.3 is UPGRADE ! Believe me, it's worth it.

- You need the reqtools.library V38+ (release 2.1) installed in your LIBS: directory. It is not provided in this archive but you can find it nearly everywhere in public domain collections.

- You need the errormsg.library V1.0+ installed in your LIBS: directory. It is provided somewhere in this archive.

- You need the iffparse.library V37+ installed in your LIBS: directory. This came along with Workbench 2.0+.

- Injector uses the locale.library V38+ if it is available to support multiple languages. This library is normally shipped with OS 2.1+.

- The preferences program makes use of the amigaguide.library to display this file whenever the Help key is pressed, but it is not mandatory.

- ARexx is needed for the ARexx facilities of Injector (!). You can find ARexx along with the original distribution of Workbench 2.0 and higher.

1.5 2.1 Introduction

Those of you who already know Injector will have to read on, as the program was entirely rewritten since version 1.x, and lots of things have changed.

Injector is a hotkey-invoked commodity whose purpose is to inject things into the InputEvent stream (that is to say as if typed on keyboard). It is wholly configurable with a nice preferences program, has ARexx facilities and lots of features.

However, the icon facility has been removed, this will be the job of a more fuctionnal commodity (yet to be written:-).

To make this work, Injector has its own language, made of keywords, with or without arguments enclosed in parentheses. You will have to read the references sections for further informations.

1.6 2.2 Starting Injector

Injector can be started either from CLI of from Workbench.

If used from the CLI, Injector will automatically detach itself, enabling the CLI window to close or the startup-sequence to continue, no

need for the Run command. Unlike previous versions of Injector, there are no CLI arguments recognized. The config file provides for that. Some more options are descibed in the tooltypes section.

If you use Injector from the Workbench, the best place for it to be is the WBStartup drawer. This way, Injector will be started each time the system boots up. You can of course double-click its icon too.

Even when started from Workbench, Injector cuts its SegList so that Workbench can close while Injector is active. Plus, there is no need for the DONOTWAIT tooltype.

1.7 2.3 ToolTypes

This section describes the tooltypes that are recognized by Injector. Tooltypes are the ONLY way to pass these arguments, even from CLI. They are: PUBSCREEN,CX_PRIORITY,PREFSPATH and AREXXCONSOLE.

1.8 The PUBSCREEN tooltype

This tooltype tells Injector which public screen should be used for the file requester and the string requester. The default is '*', which means that the frontmost screen will be used, provided it is public (just as CON: windows). If the screen is not available, the default public screen is used.

1.9 The CX_PRIORITY tooltype

This is used to specify a number that will be used as the priority of Injector's Broker in commodities' list. This is useful if you want Injector's hotkeys to override another commodity's hotkeys (or the contrary). The higher the priority is, the sooner Injector will receive the InputEvents (relatively to other brokers). The default is 0.

1.10 The PREFSPATH tooltype

This option will tell Injector where to find the preferences program whenever it is invoked by the EXEC_PREFS command. The default is 'InjectorPrefs', which must of course be in your valid path. I do use PREFSPATH=SYS:Prefs/Injector .

1.11 The AREXXCONSOLE tooltype

This tooltype describes the console window Injector will open when executing an ARexx program. This MUST be an interactive console, for example a CON: window (maybe an AUX: stream ? I haven't tried it).

The default is CON:////Injector and ARexx/SCREEN*/AUTO/WAIT/CLOSE, which will provide you with a reasonable console window on the front screen if public, or the default public screen.

1.12 2.4 Removing Injector

Injector can be removed by several ways. You can of course use the Exchange program and select Kill Injector. You can also run Injector (the master program, not the preferences program) again. The last way of removing Injector is to make it execute the CMD_QUIT command.

In all cases, Injector cannot be removed while an ARexx program is still running, as Injector must wait for the reply message from ARexx (don't want to wake up the guru, do you ? :-). Future versions might implement a "delayed quit" feature (is it really useful?).

1.13 3.1 Starting the preferences program

To modify its configuration, Injector makes use of a separate program, called the preferences program. This way, memory is not used to store unused data while Injector is active: the preferences program is loaded only when necessary.

The preferences program can be started either from CLI, Workbench, or Injector. It ALWAYS recognizes these tooltypes (even from CLI):

PUBSCREEN=<public screen name>. This defines the screen that should be used to open the window. The default is *, meaning that the front screen should be used if it is public.

CREATEICONS=<YES or NO>. This tells the preferences program of the original state of the "Create icons ?" menu item.

ACTION=<USE,SAVE or EDIT>. This is valid only for project icons that have their default tool set to the prefs program, or given via multi-selection. Such project icons are configuration files that are to be loaded immediately by the prefs program. This ACTION tooltype tells the prefs program what to do with this congfiguration file.

USE will save the given file as the "current" configuration, that is to say into the file "ENV:Injector.Prefs".

SAVE will save the given file as the "permanent" configuration, that is to say into both "ENV:Injector.Prefs" and "ENVARC:Injector.Prefs".

EDIT is the default, it won't save anything (yet), but rather let you edit the file as if you had started the prefs program then selected the Open...

menu item.

Moreover, there are some options that can be used on the command line, that OVERRIDE the above-mentionned tooltypes. The template is:
FILE,PUBSCREEN/K,USE/S,SAVE/S,EDIT/S,NCI=NOCREATEICONS/S

FILE is of course the the name of the file to load, PUBSCREEN as the same meaning as the tooltype of the same name, USE,SAVE and EDIT are switches that are equivalent to the corresponding values of the ACTION tooltype, and NOCREATEICONS is equivalent to the tooltype CREATEICONS=NO.

1.14 3.2 The Graphics User Interface

When started in EDIT mode, the preferences program opens a window with some gadgets and menues. If you're used to normal preferences programs, you should not have problems using it. Yet here are some explanations.

The main display is a list. It contains all currently defined hotkeys. You can select a name in that list by clicking on it, or using the shortcut key (h for the english version) with or without SHIFT to scroll through the list without using the mouse.

On the right, there are 5 gadgets that are used to manage the appearance of the list. They do not change the way Injector will treat it (except maybe that if two hotkeys conflict, the first one will be used). Top will send the selected entry to the top of the list, Bottom will send it to the bottom, Up will move it one entry upwards, Down one entry downwards, and Sort will sort the list in alphabetical order.

Below are some control gadgets. There are three string gadgets that are used to edit hotkeys. For injector, a hotkey is a combination of three things: a name, a key descriptor, and a command line. The 3 string gadgets let you edit those fields.

The Name field is used for displaying the list and by the EXEC_NAMED command.

The Key string gadget lets you edit the field that will tell commodities.library which key combination should trigger the action of Injector. This must be a valid commodities descriptor string. See Defining hotkeys for further information. After typing in that string gadget, the preferences program asks commodities if the string is OK, and will alert you if it is not.

The Command string gadget is there to hold the command stream Injector should execute when receiving the event corresponding to that hotkey. Injector uses its own "language" for defining such actions. See the Language reference section for details.

On the right of the string gadgets are three boolean gadgets named Create, Copy and Delete. They are used respectively for creating a new hotkey (that appears at the bottom of the list), Copying an existing entry (appearing right after the currently selected one) and Deleting an existing Hotkey.

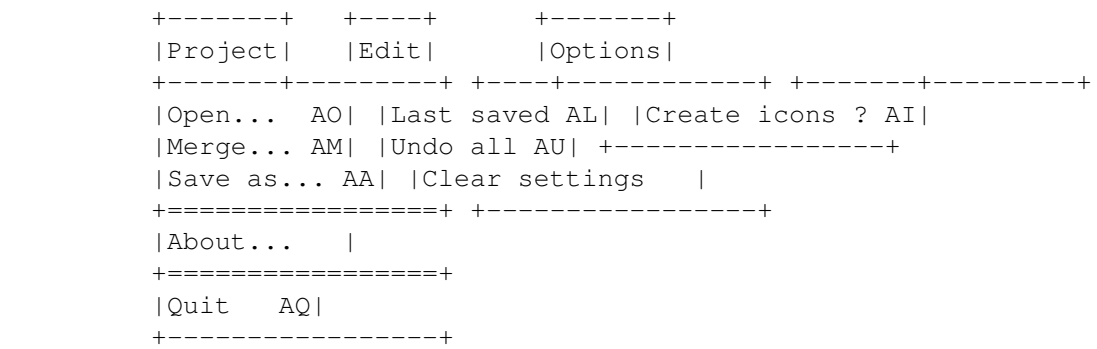
Farther below are 5 gadgets. They control the behaviour of the program. Save will store the edited file as the permanent configuration. Use will save it as the current configuration. Both of them will make the preferences program quit (unless an error occurs while saving).

Test will change the current configuration but won't end the prefs program, so that you can test if the newly created configuration really suit your needs, and change it if it doesn't.

Help will display this file. You need amigaguide.library V34+ for that.

Cancel will revert all you have done. If you had changed the current configuration (via test), then it will be put back as it was before the program was called. Then the prefs program ends.

That's it for the gadgets. Let's throw an eye at the menues:



Open will load a new configuration file. The currently edited one will be lost. Merge will load a new file and merge it with the configuration you are currently editing. Save as will write the currently edited file to a new file. For these three actions, you will get an ASL file requester.

About will inform you about the version of Injector and my address. Quit will end the program (Warning: no "Cancel" nor "Undo" is performed, a tested configuration will remain active).

Last saved will get the last saved version of the permanent configuration file (the configuration will be loaded from ENVARC:Injector.Prefs). Undo All will cancel all the changes you have done. Clear settings will of course clear the file you are currently editing. Use with care !

At last, Create icons is an on/off menuitem that tells the preferences program whether it should create an icon for a file that is saved to disk by the "Save as..." function. The icon will be a project, whose image will be gotten from ENV:sys/def_prefs.info (or the default project icon if inexisting), the default tool is set to the prefs program, and ACTION=USE will be set as a tooltype. This way, double-clicking on that icon will change the current configuration of Injector without really entering the prefs program (does not mean it is not loaded though).

1.15 3.3 Language reference

This section will teach you how to write commands for Injector. These must be entered in the Command string gadget of the preferences program's window when a hotkey is selected, or sent via ARexx to the port named Injector.

A command is made of a keyword and an optionnal argument. Commands that take arguments must have their keyword followed immediately by the argument enclosed in parentheses. Thus it may be necessary for ARexx programs to surround the parentheses by quotes, as the former are treated in a special way by ARexx. But the parentheses must remain. Check the example programs provided. Several commands are separated by spaces.

Now here is the list of the commands Injector understands (those that take an argument have () after them):

```
  FREQ_SHOW   TIME_INJDATE()
  FREQ_INJFILE    TIME_INJTIME()
  FREQ_INJDIR   TIME_INJDAY()
  FREQ_INJPATH
        EXEC_NAMED()
  SREQ_SHOW   EXEC_REXX()
  SREQ_INJECT   EXEC_PREFS

  CHARS_INJ()   CMD_QUIT
  CHARS_CLIP()    CMD_UPDATE
  CHARS_ENV()
```

1.16 The FREQ_SHOW command

This command will show the File requester. The user (you) can then select a file, which can be pasted later with FREQ_INJFILE, FREQ_INJDIR or FREQ_INJPATH.

The screen the file requester appears on can be set by the PUBSCREEN tooltype.

1.17 The FREQ_INJFILE command

This command will inject the file name from a previously called file requester (by FREQ_SHOW) into the InputEvent stream. Only the last component of the name is pasted.

1.18 The FREQ_INJDIR command

This command is used to inject the directory name of a previously called file requester. It works as FREQ_INJFILE.

1.19 The FREQ_INJPATH command

This command is like a medley of FREQ_INJDIR and FREQ_INJFILE, but it also takes care of the handling of colons and slashes between the two. It proves to be useful when using shell, for example.

1.20 The SREQ_SHOW command

This command will make Injector display the string requester. In this requester, you can enter any string you want, limited to 80 characters in length.

1.21 The SREQ_INJECT command

This command will inject the contents of the string requester, assuming it was previously called by SREQ_SHOW. This is useful if you have to type many times the same thing.

1.22 The CHARS_INJ command

This command injects the constant string, which is given in argument, into the InputEvent stream. Some special characters are recognized for this command (they are the same that CatComp uses):

```
  \a  Bell (ASCII 7)
  \b  Backspace (ASCII 8)
  \c  Control sequence introducer (CSI, ASCII 155)
  \e  Escape (ASCII 27)
  \f  Form feed (ASCII 12)
  \n  New Line (ASCII 10)
  \r  Return (ASCII 13)
  \t  Horizontal tabulator (ASCII 9)
  \v  Vertical tabulator (ASCII 11)
  \xNN  Character with NN as hexadecimal ASCII code
  \NNN  Character with NNN as octal ASCII code
  \ Backslash
  \)  Close parenthesis (not end of arguments)
```

All are are case-sensitive.

1.23 The CHARS_CLIP command

This command will inject the contents of the system clipboard. To be precise, all CHRS chunks found will have their contents injected into the InputEvent stream, thus permitting an easy way of sharing data between applications (you can cut some text in the shell and inject it in a word processor that does not support the clipboard).

The argument is the clipboard unit number, from 0 to 255. This is generally 0 (for the shell etc...), but not necessary.

1.24 The CHARS_ENV command

This command will inject the contents of the environment variable whose name is given as an argument. Injector does no sanity checks, so avoid binary files !

1.25 The TIME_INJDATE command

This command will inject the date according to the format requested by the argument.

If locale.library could not be successfully opened, the argument is forced to 2. Else, here are the possible values (other values are reserved for future expansion):

0: short format, according to the current locale
(ex: 20/04/94)

1: long format, according to the current locale
(ex: Mercredi 20 Avril 1994)

2: DOS format. This is how the DOS displays the date.
(ex: 20-Avr-94)

1.26 The TIME_INJTIME command

This will enable you to paste the current time. The argument is a format number similar to that taken by TIME_INJDATE:

ex: 0: 18h46
1: 18h47
2: 18:47:10
(the french locale does not make any difference between long and short formats for the time).

1.27 The TIME_INJDAY command

This is the final TIME_ command: it pastes the name of the day. The argument is still a format number (see TIME_INJDATE).

ex: 0: Mer
1: Mercredi
2: Mercredi

1.28 The EXEC_NAMED command

This command will execute a hotkey as if the corresponding key had been pressed by the user. This way you can do GOSUB-like processings of hotkeys. The argument is of course the name of the hotkey to be processed. There is a security check in Injector: when a hotkey is executing, a special bit in its structure is set, preventing recursive calls to be made (you will just get a message).

1.29 The EXEC_REXX command

This command tells Injector to launch the ARexx program whose name is in argument. The program will have its default host address set to "Injector". The default file extension for such programs is .ijctr .

Thanks to this command (and ARexx!), you can generate complex macros, with tests in them, etc etc etc...

1.30 The EXEC_PREFS command

This command will ask Injector to run the preferences program. This is done via the SystemTagList() call of the dos.library, so the preferences program must be in your path. The default file name for it is "InjectorPrefs", but it can be changed by the PREFSPATH tooltype.

1.31 The CMD_QUIT command

The use of this command is obvious: on receving it, and provided doing so is possible, Injector will suicide. This is not possible for example if some ARexx programs are still running.

1.32 The CMD_UPDATE command

This command is not useful in most cases. On receiving it, Injector will reload its configuration. This is useless in most cases since, like the IPrefs program, Injector uses the notification facilities of AmigaDOS to automatically detect alterations of the preferences file (by the preferences program for example). Generally, ENV: is assigned to the Ram disk, so it is not a problem. But all handlers do not support notification, so, if someone uses an assign over a network filesystem for example, he should use this command for Injector to update the configuration.

When this command is received, Injector does not continue the processing of the current line, even if there were commands remaining.

1.33 4.1 Defining hotkeys

```
The text that follows does depend on the behaviour of the
commodities.library rather than on Injector's. This is how commodities
will understand the key combination you want an action associated with.
A description string is made as follows:

     [<class>] {[-][<qualifiers>]} [-][upstroke] [<key code>]
All keywords are case-insensitive.

class: set to an InputEvent class. Supported classes are rawkey for
keyboard events, rawmouse for mouse events,diskinserted and diskremoved.
The default is rawkey, which should generally be used.

qualifiers: this is a set of keywords representing the state of keyboard
qualifiers (Shift,Alt etc...). This is a list of known keywords. Those
marked with an * are new for commodities.library V38.
lshift,left_shift *:        Left shift key.
rshift,right_shift *:       Right shift key.
shift:            Either shift key.
capslock,caps_lock *:       Caps Lock key.
caps:           Either Caps Lock or shift.
control,ctrl *:         Control key.
lalt,left_alt *:        Left Alt key.
ralt,right_alt *:       Right Alt key.
alt:            Either Alt key.
lcommand,lamiga *,left_amiga *,left_command *:  Left amiga key.
rcommand,ramiga *,right_amiga *,right_command *:Right amiga key.
numericpad,numpad *,num_pad *,numeric_pad *:  For numeric pad keys.
leftbutton,lbutton *,left_button *:   Left mouse button.(1)
midbutton,mbutton *,middlebutton *,middle_button *:
           Middle mouse button.(1)
rbutton:,rightbutton *,right_button *:    Right mouse button.(1)
repeat:           Key-repeat active.(2)

Notes:  (1) commodities.library V37 has a bug which prevents the use of
        leftbutton,midbutton and rbutton as qualifiers. It was fixed
        in V38.
   (2)  for rawkey class only.
   (3)  if a hotkey is to be insensitive to the state of a qualifier,
        place a - before the qualifier name.

upstroke:  Normally an event is generated only when the key is pressed.
You can change this behavior by setting upstroke. This will generate an
event only when the key is released. If both press and release are to
generate an event, use -upstroke.

key code: These are meaningful only to rawkey and rawmouse classes.
For rawkey, here are the key codes, * are new for V38 of commodities.
a to z, 0 to 9:       Normal ASCII characters.
f1 to f10,f11 and f12:      Function keys.
up,cursor_up *:       Up arrow key.
down,cursor_down *:     Down arrow key.
left,cursor_left *:     Left arrow key.
right,cursor_right *:     right arrow key.
esc,escape *:       Esc key.
```

```
backspace       Backspace key.
del          Del key.
help           Help key.
tab          Tab key.
comma          Comma key (,).
return           Return key.
space,spacebar *       Space bar.
enter          Enter key.(4)
insert *         Keypad 0 key.(4)
delete *         Keypad 1 key.(4)
page_up *        Keypad 9 key.(4)
page_down *        Keypad 3 key.(4)
home *           Keypad 7 key.(4)
end *          Keypad 1 key.(4)

Notes:  (4) must be used with the numericpad qualifier.

For rawmouse,valid keycodes are: (only valid for V38 of Commodities):
mouse_leftpress:       Left button pressed.(5)
mouse_middlepress:       Middle button pressed.(5)
mouse_rightpress:      Right button pressed.(5)

Notes:  (5) you must also set the corresponding qualifier.
```

1.34 4.2 History

```
Revision V2.00
--------------
    RUMBLE, RUMBLE... This is finally it (over 6 months after
    the first release!). I have totally rewritten the program,
    with lots of powerful features, a nice prefs program, full
    ARexx support, localization... It has little in common with
    the 1.x versions, so read the doc !

Revision V1.16
--------------
    Commodore have definitely gone wrong with their mis-
    documenting CreateNewProc(): the default value for
    NP_FreeSeglist is FALSE! This caused lots of memory wasted
    in earlier versions of Injector...

Revision V1.15
--------------
    Vicious bug fixed: Failed to unlock a public screens in some
    cases.

Revision V1.14
--------------
    Stupid bug fixed: PASTEFILEKEY and PASTEDIRKEY were inverted

Revision V1.13
--------------
    This is finally it! I've made it to the very first public
    version, released as SHAREWARE. As usual, I've done a bit of
    clean up and packed everything into a nice directory with
```

```
    nice readme files. (10-Oct-93   14:31:10)

Revision V1.12
--------------
    Added a 2.0-version string and written AmigaGuide doc files.
    No ASCII doc for this program,just get AmigaGuide (Fish 870)
    to view them. Optimized placement of icon window (right side
    of requester). Public release is for the next revision.

Revision V1.11
--------------
    The CXPRI tooltype was broken,and is now fixed. I have writ-
    ten a french catalog and I've fixed the locale interface.
    Done some miscellaneous bugfixes and optimizations. Planning
    a public release soon. Promised!

Revision V1.10
--------------
    Well, I have removed the automatic translation mechanism(was
    implemented with Commodities' Translate objects).Advantages:
    it shortens the program, commodities does not cause strange
    gurus when asynchronously accessing data at bad times. Draw-
    back: Cannot use hotkeys while requester is active.

Revision V1.09
--------------
    I've had the PubScreen and IconPath tooltypes working,and I
    have fixed a couple of bugs. Thinking of writing a doc file
    by now...

Revision V1.08
--------------
    I have made a skeleton for option support from CLI and tool-
    types. Supported options for now: all hotkeys,Commodities
    priority. Soon: IconPath and PubScreen.

Revision V1.07
--------------
    Wrote my own InvertString()/FreeIEvents() routines so that
    Injector does not misinterpret filenames with angled brakets
    any more. Also fixed self-detaching from Workbench to enable
    the use of the WBStartup drawer.

Revision V1.06
--------------
    Now the injector self-detaches from the CLI, no longer need
    to use Run. If the icon window is moved, the next time it
    opens at the same place.

Revision V1.05
--------------
    The displayed icon is finally a true selectable gadget,thus
    allowing you to see both inactive and active states. Note
    there could be a little difference with workbench's icons as
    Workbench has a "floodfill" mode which is not supported by
    standard Intuition gadgets. Might be fixed later.
```

```
Revision V1.04
--------------
    Ah, a cool new feature: the icon requester can now display
    the icons themselves in their own little window, using ano-
    ther nice ExtraButton. For now, the icon is not a true gad-
    get (hardly an image), but I shall fix it soon.

Revision V1.03
--------------
    That's it,Injector is now able to copy icons for directories
    ,just select an empty file name.

Revision V1.02
--------------
    Added interesting new feature: Injector can now copy icons
    from one place to another,using kd_freq's ExtraButton fea-
    ture. Proves to be very useful! Still have to take care of
    directories' icons.

Revision V1.00
--------------
    --- Initial release ---
```

1.35 4.3 Contacting the author

I remind you that Injector is SHAREWARE. I know most of you will not pay me anything after their evaluation period, but please consider I've spent a lot of time writting and debugging this program, and I'd just like a small reward that would enable me to buy a good hardware configuration. Consider that paying $10 will provide you with support for future releases. The source is kept warm for those of you who might be interested in programming the wonderful amiga operating system, for $10 more.

For anything like registration, comments, bug reports, improvements requests, postcards, I can be reached at:

Frédéric DELACROIX
5 rue d'Artres
59269 QUERENAING
FRANCE.