SysInspector

COLLABORATORS

	TITLE : SysInspector		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 20, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

SysInspector

1.1 SysInspector - Documentation

```
                     SysInspector 1.2
          "Your Amiga can't hide it from the Inspector ;)"

                 Copyright ©1997 by Eric Sauvageau.

                           SHAREWARE.
```

1.2 About

There are already quite a few system monitors available on Aminet.
Unfortunately they all suffer in at least one of these areas:

\textdegree{} They are old and outdated, not supporting any OS 3.x feature like memory handlers.

\textdegree{} They are buggy or very unstable.

\textdegree{} They have a lousy GUI, sometimes not even fully font-sensitive, or looking plainly ugly under modern system configurations.

\textdegree{} One lacks a feature only available in the other, and vice-versa.

\textdegree{} They don't even come with english documentation (!!!), so you're pretty much left "on your own" while using it (and considering how easy it is to crash your system when misusing a system monitor, it often leads to disaster).

So there's SysInspector. The main goals while developping it were:

1) Provide something STABLE. Of course, removing a task is always risky. But there are some system monitors out there who would crash just at trying to give you the list of tasks. Not very handy when your intention is to get rid of an already crashed task, not to crash another one. To help ensuring this, SysInspector has its own guru trapper built-in, which should catch most software failures from itself. It won't help you though if you crash another task using SysInspector.

2) While making it quite powerful and handy for the advanced developper, still keep it simple enough for the casual user who would only want to change a task priority, or close the window left open by a crashed program. Some features are best fit into another tool such as HDToolbox or your usual debugger than into a system monitor.

3) Provide it a modern, font-sensitive and easy to use GUI, without necessarily turning it into a snail nor a resource-hogger.

4) Make it actualy USEFUL. Some monitors let you see what's on your system, might even let you do a few funky things... but nothing useful. What's the use of being able to remove a crashed task if it won't also remove that annoying window cluttering your Workbench?

5) Provide a minimum of configurability.

WARNING!

Careless use of a system monitor can lead to crashes, even possible loss of data. PLEASE take the time to read this documentation! The risky operations are labeled as such, so you'll know what can be safely done and what cannot.

1.3 About ClassAct

[Taken from the ClassAct homepage]

ClassAct is a set of BOOPSI classes co-authored by Christopher Aldi, Timothy Aston, Osma Ahvenlampi and Petter Nilsen, published by Amigability.

ClassAct provides object-oriented building blocks for your application in the form of Intuition BOOPSI classes libraries libraries. As they are standard classes, they may be used with any application environment supporting BOOPSI. ClassAct is a complete GUI system in its own right, supporting everything from simple buttons to an advanced list management class, and includes a complete GUI layout gadget class, arexx class and window class.

ClassAct user interfaces are font-adaptive, and dynamically resizable. Like any modern GUI, the ClassAct layout engine does not use fixed coordinates, but instead hierarchical groupings. The layout engine supports not only gadgets within its group, but images and even other layout groups as well.

Programs that use ClassAct can be made freely distributable, shareware, commercial, etc. There is no fee for your users!. When you purchase the ClassAct development kit license, users of your software get to use all the functions of our classes. ClassAct is not only a powerful and time-saving choice for software developers, but an affordable and convenient one as well.

ClassAct is an expanding project, providing you with all the graphical user interface tools you need to write your application. Currently there are over 25 different classes, and this list is growing all the time!

ClassAct requires at least Workbench 2.04, and will take advantage of higher OS versions if available.

ClassAct development kit supports SAS/C, Dice, and AmigaE.

Latest version of the classes can be found on Internet at:

ftp://ftp.warped.com/pub/amiga/classact/

1.4 Usage

To start SysInspector, you can double-click on the icon, or start it from a shell. Command template is:

```
   DISPLAY/K/N, ICONIFY/S, TASKPRI/N/K
```

```
DISPLAY - Select an initial display type (bypassing prefs).  Must be one
          of these values:

           0 - Assigns
           1 - Commodities
           2 - Devices
           3 - Fonts
           4 - Interrupt handlers/servers
           5 - Libraries
           6 - Filesystem locks
           7 - Memory nodes
           8 - Memory Handlers
           9 - Message ports
          10 - Mounted devices
          11 - Resident modules
          12 - Resources
          13 - Screenmodes
          14 - Screens and Windows
          15 - Signal Semaphores
          16 - System Information (default)
          17 - Tasks and Processes


ICONIFY - Start SysInspector in an iconified (or menufied) state.


TASKPRI - Start SysInspector with a different task priority.  Especialy
          useful if you want to start it while some other (buggy) task
          is eating up most of the CPU time.

          NOTE: Tho there's a TASKPRI tooltype, it's strongly recommended
                to only use this argument from the CLI, since the args
                parsing in that case will be done earlier (it will be
                done before opening any gadget/classes), so you will
                benefit from the new task pri during this part of the
                startup code as well.
```

Those same arguments can also be specified as tooltypes, which will be used if you start SysInspector from the Workbench.

1.5 Requirements

To run, SysInspector requires:

\textdegree{} Any Amiga model, with a 68000 or better.

\textdegree{} Kickstart 3.0 or better. You must also have SetPatch installed, so that memory pools properly works.

\textdegree{} reqtools.library V38 and boards.library V2 or better (included in ↩ the archive).

\textdegree{} ClassAct 2.0 or better (the latest version can be found on ↩ Internet at ftp://ftp.warped.com/pub/amiga/classact/).

1.6 Installation

Just double-click on the supplied Installer script. If you prefer to do it manualy, copy the supplied libraries in your own libs: drawer (check first their version!), and copy the main executable wherever you wish.

If you haven't installed ClassAct yet, do so through its own Installer script.

1.7 Main Window

At the top of the window, there's a chooser gadget (or an array of buttons, depending if you enabled "Use Button Array" in the prefs or not) that let you select the information you wish to display in the list (the "list type"). If you have the Button Array enabled, the corresponding list type will be shown in the window's titlebar while pressing on the button. So, if you're not sure what list type corresponds to a given button, press on that button to see the name in the window's titlebar, and move the mouse pointer off the button before releasing the mouse button, so to not select it. If you can't see all buttons in the window, you can scroll the array by holding down shift, and dragging the button array with the left mouse button pressed.

Just below there's a listbrowser that displays the information of the selected list type. In most cases, this list will be separated in columns, depending on the selected list type. You can navigate through it by using the usual arrows/scrollbar located to the right, or by using the keyboard:

```
     Up/Down Cursor keys: Move up/down by one entry
   Shift Up/Down Cursors: Move up/down by one page
    Ctrl Up/Down Cursors: Move to top/bottom of the list

  Left/Right Cursor keys: Change the displayed list type
```

For some list types (screens/windows, Interrupts, Locks), the list will be displayed in an hierarchical format. Nodes that have childs will display a small folder pictogram to their left. Just click on the pictogram to expand the list, and display the childs for that given node, or to close it. An example is the Screens/Windows list: when you select this type, a list of the currently opened screens will open. To view the windows opened on a specific screen (if there's any), just click on the arrow located to the left of the node.

At the bottom, there's two rows of buttons, and a text gadget displaying the current status. Not all of these buttons are used for each types: once you select a node in the list, the appropriate buttons will be enabled. Their use is documented in the list type sections of this manual.

Two buttons are special cases: "More", and "Refresh".

```
   Refresh: Will rescan your system, to update the current list.
            So, if you were displaying tasks, clicking on it will reflect
            any possible changes, like if new tasks have started, etc...

      More: Will open a window that shows more information on the currently
            selected node (note that a few list types won't use it).  Once
            opened, you can leave this window open: it will simply adjust
            itself if you select a new list type.  A few list types will
            add buttons to this window: they will be documented in the
            list type section of this manual.  One button is common to all
            list types using this window: "Save".  This button will allow
            you to save to a file the information listed in this window.
            If you wish to have this window automaticaly opened at startup,
            you can set this in the preferences by enabling
            "Open More Window?".

            The window can also be opened by double-clicking on a given
            node in the main list.  Note that double-clicking on some
            columns already have an effect (like double-clicking on a
            task's pri column will prompt you to enter a new priority),
            so I recommend double-clicking on the name or on the address
            since these are sure to never execute a different action.

            To close it, either click on its close gadget, or click again
            on the "More" button.


Menus

   Project

        About...          Information about SysInspector.
        Iconify           Iconify the window.
        Save to File...   Dumps the current list content to a file.
        Quit              Quit SysInspector.

   Settings

        Prefs...    Configure SysInspector's settings.
        Classact... Configure ClassAct's GUI settings.

   Tools

        Flush memory...    Flush the memory from unused libraries/devices,
                           invoke the low memory handlers, etc...

        Selective Flush... User-configurable memory flush.

        Generate Report... Goes through all the list types, and saves
                           the output to a file.
```

1.8 Preferences

```
The prefs window consists of three pages, and the usual "Save" "Use" "Cancel"
buttons found in most prefs editors.  To flip through pages, click on the
tabs located at the top of the window.

General

   Deallocation: When you remove a task or a window, determines if
                 SysInspector should try to free all resources it can,
                 never do so, or ask you about it.  Currently,
                 SysInspector can:

                   \textdegree{} Close screens and windows opened by a killed task
                   \textdegree{} Free any message port the task had opened along ↩
                      with
```

```
                     any queued messages
                   \textdegree{} Free any Gadtools gadgets attached to the window
                   \textdegree{} Free any Gadtools menus attached to the window


   Iconified Position: Coordinates where you want SysInspector to put its
                       icon while iconified.  A value of -1 means
                       "free position", where Workbench will select one.

   Initial List: Select which list to display at startup.


Options

   Confirm Actions:         Should SysInspector ask you to confirm before
                            doing any action.

   Strip Process Path:      When displaying a command running from a CLI,
                            should SysInspector only display the command's
                            filename, without the whole path.

   Snapshot Windows:        When saving prefs, should the size and
                            position of the "More" and the main window be
                            saved.

   Open More Window:        Should the "More" window be opened at
                            startup?

   Format Hex Values:       Should SysInspector format hex values
                            ($1234 5678) or not ($12345678).

   Use Button array:        If you wish to replace the popup chooser
                            with an array of 18 buttons to select the list
                            type to display.

   Menufy on Iconification: Instead of putting an AppIcon on the
                            Workbench, will add an entry in the
                            "Tools" menu.

   Disable DOS Requesters:  If SysInspector tries to access a non-available
                            volume in the "Locks" and "Assigns" lists,
                            don't display any requester asking for it.

   Scan RAM: for Locks:     CBM's Ram-Handler doesn't link locks as
                            expected, so by default SysInspector won't
                            try to obtain the list of locks pending on
                            the RAM:.  If you're using a RamDisk
                            replacement that properly links them, you can
                            allow SysInspector to scan RAM: while building
                            the locks list (in the "Locks" list type) by
                            enabling this option.

   Start Iconified:         Should SysInspector be started in an iconified
                            (or menufied) state.

   Double Click Opens More: If you want SysInspector to open the More
                            window on a double click in the list.
```

Sort

Configure how the items should be sorted for these list types:

```
   Libs/Devs   Resident   Semaphores/Ports   Assigns   Tasks   Mounts
```

Other list types will be listed in the same order as they are found on the system (some lists are already sorted by the system, such as the Memory nodes list.)

Menus:

```
   Project

      Quit Prefs   Close the preference window.

   Settings

      Reset to Default   Restore the default settings.
      Last Saved         Restore the last saved settings.
      Restore            Restore the last used settings.
```

1.9 List Types

Here are the various lists SysInspector is able to display:

```
      Assigns                  Mounts
      Commodities              Residents
      Devices                  Resources
      Fonts                    Screenmodes
      Libraries                Screens/Windows
      Locks                    Semaphores
      Memory                   System
      Memory Handlers          Tasks
      Message Ports
```

1.10 LIST: Assigns

This will show a list of DOS assigns currently on your system. Multiple assigns will be folded - click on the folder pictogram to the left of the parent assign to reveal all the "child" (added) assigns.

Available Buttons:

```
   Remove: Will remove the assign, freeing its associated lock at the
           same time.  Unlike the AmigaDOS "Assign" command, this will
           let you remove a single assign from a multi-assign.  Note that
           If you remove the parent assign of a multi-assign, all the
           "child" assigns will also be removed.
```

1.11 LIST: Commodities

```
This will show a list of the commodities currently loaded.


Available Buttons:

   Remove: Will ask the commodity to quit, just like if you had
           done "Remove" in Exchange.

   More: The "More" window will display four more buttons:
         "Enable", "Disable", "Show" and "Hide".  These have the same
         functions as those the "Exchange" program, letting you
         enable/disable a commodity, and show/hide its interface (if it
         has one).
```

1.12 LIST: Devices

```
This will show a list of devices currently present in memory.

See "Libraries" for an explanation of the available options, they are the
same.
```

1.13 LIST: Fonts

```
This will show a list of the fonts currently available in memory.  Color
fonts will have their name highlighted.
```

1.14 LIST: Screenmodes

```
This will show the screenmodes currently available in the display database.
```

1.15 LIST: Interrupts

```
This will give you a list of the interrupt handlers and servers currently
installed on your system, sorted by interrupt types.
```

1.16 LIST: Libraries

```
This will show you a list of the libraries currently in memory.  It also
includes BOOPSI gadgets and datatype classes that have been loaded from
disk.


Available Buttons:

   Flush: Will try to expunge a library from memory.  For this to succeed,
          the library’s open count must be ’0’, meaning no application
          is currently keeping it open.  Also, the library must allow
          itself to be expunged - some libraries don’t.


   Remove: Will forcibly remove a library from memory.  What this really
           does is remove the library node from Exec’s linked list of
           libraries, so note that no resource will be freed - the library
           will still be somewhere in memory.

           NOTE: This is a RISKY operation!  Only use this if you
                 know what you’re doing, like if you must remove a library
                 you’re working on that is stuck in memory, so you can
                 load a new version.
```

1.17 LIST: Locks

```
Will display a list of volumes.  Clicking on the folder pictogram to the
left of a volume will expand the list, adding a list of all filelocks
currently active on the volume.


Available Buttons:

   Remove: Will remove the lock, calling UnLock() on it.

           NOTES: This is a RISKY operation!  Only use this if the program
                  who installed the lock is crashed, suspended, or has
                  ended.

                  The Ram Disk doesn’t properly handle the linked list of
                  locks, so I’m not displaying it in the list.  CBM’s own
                  ShowLocks on the Ram Disk will even get stuck in an
                  infinite loop...  However, if you’re using a RAM:
                  replacement and wish to let SysInspector scan the lock
                  list, you can enable it in the preferences (see
                  Scan RAM: for Locks).
```

1.18 LIST: Memory nodes

```
This will display a list of the currently configured memory nodes.
```

```
Available Buttons:

   Priority: This will let you change the priority of a memory node.
             Memory allocations will follow the node priorities, so
             when asking for MEMF_ANY, the memory node with the higher
             priority will be checked first for any free block.  So, it’s
             logical to have a faster node have a higher priority over
             a slower one (like 32-bits RAM over 16-bits RAM).  Valid
             priorities ranges from -128 to 127.  Note that double-clicking
             on the "Pri" column in the list will have the same effect.
```

1.19 LIST: Memory Handlers

```
This will show you the list of currently installed low memory handlers. These
are usually called if a memory allocation failed.  They will usually try to
free some memory.  Examples: a lowmem handler installed by a paint
package might free some memory currently used for its undo buffer.  The
Ramlib lowmem handler installed by AmigaOS will try to expunge unused
libraries from memory.

Available Buttons:

   Priority: This will let you change the priority of a memory
             handler.  Handlers are called in the order determined by
             their priority.  Valid priorities ranges from -128 to 127.
             Note that double-clicking on the "Pri" column in the list will
             have the same effect.

  Remove: This will remove a memory handler from the system.

           NOTE: This is a RISKY operation!  Only use this if you
                 know what you’re doing.  It will merely remove it from
                 the system (through exec.library RemMemHandler()), not
                 freeing any resource.
```

1.20 LIST: Mounts

```
This will show a list of the mounted devices currently on your system.

   More: When clicking on "Save", you will be asked if you want
         to generate a Mountlist.  If you reply "No", then the window’s
         content will simply be dumped to a file, just like for any
         other type.

         NOTE: The FileSystem is NOT dumped in the Mountlist file.  If
               someone knows how I can determine what filesystem is used
               by a mounted device, please tell me so.
```

1.21 LIST: Message Ports

This will show a list of the public message ports currently on your system.

Available Buttons:

```
   Remove: Will close the message port.  If you have "Confirm Actions?"
           set in your preferences, it will ask you if you want to reply
           any message(s) still queued to the port.  Else, it will
           automaticaly do so.

           NOTE: This is a RISKY operation!  Only use it if you know what
                 you're doing, like for removing a message port left behind
                 by a crashed/terminated program.
```

1.22 LIST: Resources

This will show a list of resources currently present in memory.

```
   Remove: Will forcibly remove a resource from memory.

           NOTE: This is a RISKY operation!  Only use this if you
                 know what you're doing.
```

1.23 LIST: Resident

Will display a list of the resident modules, located in the ROM or added in RAM by the system during initialisation. Modules located in RAM will have their name highlighted.

1.24 LIST: Screens/Windows

This gives you a list of the currently opened screens. Each screen with at least one window opened will have a folder pictogram to its left. Clicking on this will expand the list, inserting the list of opened windows just under the screen's node.

Available Buttons:

```
   Remove: Will close the selected screen/window.  If you try to close
           a screen that still has windows opened, it will also close
           these (asking you first to confirm the operation if you
           selected "Confirm Actions?" in the preferences).  It will also
           ask you if you wish to deallocate any attached gadtools gadgets.

           NOTE: This is a RISKY operation!  Only use this if the task
                 that opened the screen/window is gone, crashed, or
                 suspended.
```

1.25 LIST: Semaphores

This will show a list of the public semaphores currently on your system.

Available Buttons:

```
   Remove: Will remove the semaphore.

           NOTE: This is a RISKY operation!  Only use it if you know what
                 you're doing, like for removing a semaphore left behind
                 by a crashed/terminated program.
```

1.26 LIST: System

This will display various information about your system. Information is sorted in four categories. Clicking on the folder pictogram to the left of a category will display its content.

```
    Expansion Boards: Will display a list of expansion cards installed
                      on your system, using boards.library to display
                      their name and manufacturer.

    NOTE: If SysInspector fails to recognize one of your expansion boards,
          Email the information to Torsten Bach, author of boards.library.
          You can contact him at bach@deadline.snafu.de.


    Hardware: CPU, FPU, CPU caches, gfx chipset, etc...
              Note: the result for "gfx chipset" on the Draco is unknown.
              Tell me what happens (Enforcer hits, I suspect...).

    Software: Kickstart, Workbench and Setpatch (if installed), graphic
              software version for CyberGFX/Picasso96/EGS (if installed),
              etc...

    Vectors: Execbase vectors, Kicktags, etc...

    CPU Traps: CPU Trap vectors, interrupt vectors, etc...  Those are
               usualy pointing into ROM - vectors pointing to RAM (meaning
               they were modified) will be highlighted.
```

1.27 LIST: Tasks

This will show you all the tasks and processes being executed on your system. Note that the topmost task is always SysInspector itself. No button will be activated for this one (beside the "Refresh" and "More" buttons), so you won't be able to crash SysInspector itself by playing with it. (Ever been told that playing with yourself could make you deaf? Same thing with SysInspector - suspend it, and it will no longer hear your orders :)

```
Available Buttons:

   Break: Send a break signal to the process/task.  It is the same thing as
          using the "Break" AmigaDOS command.

          You can send one of four different break signals:

             CTRL-C: This will usually terminate a process.
             CTRL-D: This will abort an AmigaDOS script.
             CTRL-E: Usually unused.
             CTRL-F: Sometime supported by some programs as a wakeup call,
                     to (re)open their GUI.


   Remove: Will let you remove a task/process.  This is handy if a program
           just crashed on your system, and you wish to get rid of it.
           SysInspector will also let you free various resources.
           See "Task Dealloc" in the prefs.

           NOTE: This is a RISKY operation!  If you want to avoid the Guru,
                 it is much safer to either just "Suspend" it (so it will
                 no longer slow down your system eating CPU time), lower
                 its task priority, or simply let it stay there.



   Suspend: Remove the task/process from the active list, preventing Exec
            from scheduling it and giving it any CPU time.  This is handy
            if you wish to suspend the execution of a task/process, either
            because it's crashed or because it's slowing down your system.
            (I even used it once to pause a WB Tetris game I was playing :)
            You can achieve the same result by double-clicking on the
            node's "State" column.


   Resume: Reactivate a suspended task/process.


   Priority: Change the task priority.  Valid priorities ranges from -128
             to 127, but note that it's not recommended to use values
             outside of the -20 to +20 range.  You can achieve the same
             result by double-clicking on the node's "Pri" column.


   Signal: Signal a task.  A new window will open, showing you
           the signal mask the task is waiting for, and 32
           checkboxes, each of them representing one of the 32
           possible signal bits.  Signal bits that the task isn't
           waiting for will show "--" as the bit number and will
           be disabled.  Just toggle the bits you want to signal,
           and click "Signal Task" to proceed.  You can also enter a value
           in the top right corner string gadget.  Hex values must start
           with a "$".
```

1.28 Misc Notes

```
Usage tips:

 - If you want to output a list’s content to the current CLI, just save
   to a file named "*" (no path!).  This will send the output to the
   current output window (stdout).

 - When a program crashes or gets in an endless loop, the safest action
   to take is to lower its task priority to something like -20 so it
   won’t slow down your system.  Almost as safe is to Suspend it so it
   no longer use any CPU at all.

   Removing the task can be more risky, depending on a lot of factors.
   SI tries to make it as safe as possible, allowing you to chose what
   resources to free, but totaly remove the task only if you can handle
   a potential crash.  It might be safer to just suspend the task, and
   then close any window it opened through the "Screen/Windows" list,
   unless you also want SI to automaticaly take care of any msg port,
   allocated gadgets, etc...

-  Remember: a system monitor can be a very powerful tool because it
   gives you access to private stuff of the operating system.  When
   something is defined as being "risky" in the documentation, don’t
   take it lightly.

 - Some programs will refuse to run twice because they leave a public
   message port behind them (like an ARexx port).  So if they crash,
   killing the old message port might allow you to run a new copy of it.

 - If you use a large disk cache and want to flush unused libraries
   without also flushing your disk cache, the "Selective Flush..." item
   comes very handy.

 - As I often do with my programs, I’ve put a secret message somewhere in
   it.  Try to find the right hotkey to get it ;)

 - If you are writing a library, and the test program crashes, leaving it
   opened so it can’t be flushed, just remove/suspend the crashed program,
   and "Remove" the library in question.  This will let you load a new
   (hopefuly fixed!) version in memory.

 - Most libraries/gadgets who are asked to flush themselves from memory
   while they are currently in use will set a "delayed expunge" flag.  This
   means once the library/gadget opencount reaches 0, it will expunge
   itself from memory.  This can be good to know if you try to flush a
   ClassAct gadget that’s currently in use by SI - it will flushes itself
   once you quit SI, unless another application has opened it.

 - When reporting bugs about a program, the author will usually like to
   get information on your system.  Why not use SI to save the "System"
   information, and send that to them with your bug report?  In some cases,
   the "Task" list and the "Libraries" list can also be of help to them.
```

Known Problems:

- On 640x200 screens there might not be enough room to display the complete 18 list types in the chooser popup. If the last(s) type(s) aren't shown, use the arrow on the chooser gadget to access those not accessible from the popup, use a taller screenmode, or try using the button array.

- Generating a Mountlist through the More window for a mounted device will lack the "FileSystem" entry. Please tell me if you know how I could retrieve this information.

- The "Locks" list gets confused when two volumes have the same name.

- The "About" window will only animate with ClassAct's button.gadget V42 (still in beta). Not a bug, just a feature not yet available under V41.

  NOTE: ClassAct's button.gadget V42 has nothing to do with the older CBM one, except with being backward compatible with it.

- AT's LoadV43Module is screwing up the kicktagptr table, trashing the name of any resident module previously linked, and not properly setting bit 31 to indicate a new table (SI has a workaround to avoid that problem). This is not a bug in SI - the list is perfectly clean when displaying the KickTagPtr list created by RADs and FastExec.

- When going from a listview display with no titlebar to one having one, the upper left area of the scroller isn't erase. That's a known bug in the actual release of the listbrowser.gadget (V41.295), just resize the window to refresh it if it annoys you.

- Selecting a process written in E will show a bogus task usage. That's because E processes allocate their own stack space, but they don't update the Upper/Lower fields of the task structure. E programmers can use the fakestack module from Aminet (somewhere in dev/e) to avoid this problem - SI does.

- Some people asked me to add a memory/structure displayer. I don't intend to write one, simply because I feel I can't write something better than Explorer, from Jason Hulance :) I recommend grabbing it from Aminet (Explorer16j.lha is the current release, in dev/e), as well as the Amiga E main archives from that same directory (it uses the E modules to properly disassemble and display data structures). Makes a good companion to SI IMHO.

ToDO:

This is far from a complete list, but most notable are:

- SCSI devices information

- Open on its own screen

- Safer "Locks" scan. Maybe only scan a device at the user's request, so if the user knows a specific device to not properly chain locks, he would simply avoid checking that one? Or a user-configurable list of devices to avoid scanning?

1.29 LEGAL

SysInspector is Copyright ©1997 by Eric Sauvageau. You are allowed to freely redistribute this program and its documentation. Inclusion on PD collections and coverdisks is allowed, but if you do so on a magazine coverdisk, I'd appreciate that you send me a free issue of the magazine in exchange.

I decline any responsability for any problem encountered while using SysInspector. If it crashes your system, makes your dinner burn, or makes Barney appear at night telling you he loves you, it's not my fault. I'm doing everything I can to ensure this product is as stable as possible, but bugs wouldn't exist if they were that easy to fix. If you encounter such a critter, please report it to me so I can try to fix it for a future release.

SysInspector is released as Shareware. It ain't crippled in any way because I'm personaly allergic to what I call "SabotagedWare", so I'm relying on YOU, the user. If you find it useful and worth paying for it, I'm asking you to send me 15$ US (20$ CAN). I can also accept a registered version of a program you wrote in exchange. The response from users will determine the future of this product. If no one feels it's worth registering, then I'll take it as a "it's not worth working on it".

Magazines are free to include it on a coverdisk, all I ask is that in exchange they send me a free copy of the issue that has SysInspector on its coverdisk. PD/Shareware collections such as Aminet, Fred Fish, etc... are free to redistribute it, as long the whole archive is included in its original state.

Send money (only US or CAN dollars please) to:

Eric Sauvageau
5338 10th Avenue
Montreal, Qc
Canada
H1Y-2G6

Email (in order of preference):

merlin@thule.no
dream@step.polymtl.ca (this is a friend's account)

Fidonet: Eric Sauvageau @ 1:242/907.0

The latest version (as well as any beta version) will always be available from my homepage:

http://www.thule.no/~merlin/

1.30 Thanks

I want to thank these guys, who had something to do with SysInspector:

```
  Philip Vedovatti               (beta-testing, artwork, Installer script)
  Christopher Aldi               (beta-testing, ClassAct)
  Petter Nilsen                  (beta-testing, ClassAct)
  Christopher Gaul               (beta-testing)
  Andrew Robson                  (beta-testing)
  Nico François, Magnus Holmgren (Reqtools)
  Tim Aston, Osma Ahvenlampi     (ClassAct)
  Torsten Bach                   (boards.library)
```

And the registered users:

```
  Ben Monroe (USA)
  Magnus Holmgren (Sweden)
  Richard Körber (Germany)
```

Please consider registering. This is the best way to ensure the future
development of this product.

1.31 History

```
   1.1 - April 7th, 1997

       \textdegree{} NEW: TASKPRI argument in the commandline.

       \textdegree{} NEW: Mounts sort method can be configured.

       \textdegree{} NEW: "Generate Report..." in the Tools menu.

       \textdegree{} NEW: "Screenmodes" list type.  You will probably need to
              readjust your "Initial List" setting in your preferences and
              resave.

       \textdegree{} Had forgotten to put non-Newicon imageries for most icons.   ↩
          Oops :)
       \textdegree{} The More window won’t unnecessarily refresh itself while  ←
          disabled.

       \textdegree{} Added "Largest chunk" to the Memory’s More window

       \textdegree{} Expansion boards now shown on two lines, and will also show
         if they’re of ZorroII or ZorroIII type.

       \textdegree{} Assign list now supports paths upto 150 chars long (still  ←
          only the
         first 50 chars displayed).  Should fix the "Fault 120" errors
         happening too often to those with long path names.

       \textdegree{} SI will stop playing shy and will display its own info in the
         More window when its task is selected.



   1.0 - March 23th, 1997

       \textdegree{} First Aminet release.



   ß0.050 - December 29th, 1996

       \textdegree{} First public beta release.





 "I don’t think I’m amazing
  In fact, I’m quite insane
  To live inside my bucket
  With all my plastic chains."

      - Ozzy Osbourne.
```

1.32 Other Programs I wrote

```
  Other programs that I wrote:

        \textdegree{} DevsMan 1.4 - Devs: directory manager, allow easy handling
                       for your DOSDrivers, Datatypes, etc...

        \textdegree{} FileScroller 3.40 - File lister for TransAmiga BBS (3.50 and ←
            up
                             for Excelsior! BBS.)

        \textdegree{} MFormat 1.8a - Replacement for CBM’s "Format" command.  Has  ←
           a
                        complete GUI, configurable device filter, can
                        install a bootable bootblock, etc...
```

```
\textdegree{} NewIcons V3.1 - Replaces the icon system with a new one that
                  allows icons to have palette information stored
                  in them, and to be properly remapped no matter
                  what’s your WB palette, and still more.

\textdegree{} TDPrefs 1.0 - Preferences editor for trackdisk.device, can
                adjust the step rate, disable the drive click,
                etc...

\textdegree{} XPKatana 1.2 - GUI/ARexx interface for the XPK packing  ←
   system,
                 let’s you (un)pack files using standard XPK
                 libraries, can also unpack files packed with alien
                 formats through xfdmaster.library.
```