HOTLINE!

Bulletin 12
April 7, 1988

HOTLINE! writers would like you to know that we are taking a break (no pun intended) from producing this Bulletin while we attend to some important work. First, we will be involved in Beta Testing Lyric LOOPS, ROOMS, and soon after that, the Medley Release. And as you know from recent announcements, some of us will be moving to the Palo Alto area.

Thank you for letting us serve you!

For more information on the questions or problems addressed in this or other bulletins please call us toll-free in the Continental United States 1-800-228-5325 (or in California 1-800-824-6449). Customer Support can also be reached via the ArpaNet by sending mail to AISUPPORT.PASA@Xerox.com, or by writing to:

Xerox AIS Customer Support
250 North Halstead Street
P.O. Box7018
Pasadena, CA 91109-7018
M/S 5910-432

In this issue

The following topics are covered in this issue:

- Specifying default font for Sketch
- Specifying default font for TEdit
- DATE, GDATE functions have bad SIDE-EFFECTS-DATA decls
- Compiling nlambda expressions
- Saving bitmaps
- TEDIT PAGEFORMAT specifications
- Example use of TableBrowser
- Reporting a problem on the Hotline

Terminology

Terminology used in this *HOTLINE!* bulletin:

UG	-	User's Guide
AR	-	Action Request, a Xerox problem tracking number (e.g. AR 8321)
IRM	-	Interlisp Reference Manual

Specifying default font for Sketch

Release Lyric

Keywords Change Font, Sketch font

Question How do I change the default font in Sketch sessions?

Answer Sketch's font is controlled by the value of SK.DEFAULT.FONT variable. You should reset the value of this variable to the appropriate value. If you want to use Modern 10 Bold regular in all the sketch sessions, type the following in an Interlisp Exec window:

```
(SETQ SK.DEFAULT.FONT '(MODERN 10 BRR))
```

To set the default font in an Init file, user needs to edit the Init file, and add:

```
(VARS (SK.DEFAULT.FONT '(MODERN 10 BRR))
```

at the end of the file.

Reference none

Specifying default font for TEdit

Release Lyric

Keywords Default font, TEdit

Question How do I change the default font in TEdit sessions?

Answer To change the default font in TEdit sessions, you need to change the FONT property value of TEDIT.DEFAULT.PROPS. The FONT property for TEdit can be a FONTDESCRIPTOR, or any form of font description that is acceptable to FONTCREATE.

For example, if you want to use Classic 12 Bold regular in a TEdit session, then type the following in the Interlisp Exec window:

```
(SETQ TEDIT.DEFAULT.PROPS (LIST 'FONT '(CLASSIC 12 BRR)))
```

Reference Xerox Lisp Documentation Tools, Lyric Release pages 8-19.

DATE, GDATE functions have bad SIDE-EFFECTS-DATA decls

Release Lyric

Keywords DATE, GDATE

Problem In Common Lisp the functions DATE and GDATE compile as constants. This problem exits only with the Common Lisp compiler.

Example

```
(DEFINEQ (MYDATEFNS () (DATE)))
```

Calling MYDATEFNS when it's interpreted returns the current date.

Compile MYDATEFNS with (CL:COMPILE 'MYDATEFNS).

Calling MYDATEFNS yields the time at which MYDATEFNS was compiled.

Workaround In the Interlisp Exec window type:

```
(FOR X IN '(DATE GDATE)
    DO (PUTPROP X
        'COMPILER::SIDE-EFFECTS-DATA
        '(:NONE . :ANY))))
```

Reference AR# 8844

Compiling nlambda expressions

Release Lyric

Keywords CL:COMPILE-FILE, IL:TCOMPL

Problem In the Lyric release, when you compile files that contain nlambda functions, a portion of the compiler's stack is held onto by the history list. Thus, if you compile such files frequently enough, you can get a stack overflow without your apparently consuming much stack at all. This problem affects both the Interlisp and Common Lisp compilers.

The problem arises because in Lyric, the compiler variables NLAMA, NLAML and LAMA were changed from globals to specials, so that the effects of a file changing those values is confined to the compilation of the file. Files containing nlambda functions modify these variables with expressions such as (ADDTOVAR NLAMA --), to ensure that calls to the functions are compiled correctly. However, ADDTOVAR, like most special forms emitted by the file manager, is undoable. In order to make the setting of a dynamically bound variable undoable, the system saves a stack pointer to the frame where the variable is bound in the undo information for the event. Hence, every time you run the compiler, the stack from the point where NLAMA et al are bound back to the top is saved, and not released until the event falls off the end of history, or you do a HARDRESET. A HARDRESET is performed automatically when you exit from a Stack Overflow break.

Symptom Stack overflow without your application obviously consuming much stack.

Workaround Don't run the compiler undoable. From the Common Lisp Exec window, type:

```
>SHH(COMPILE-FILE 'filename)
```

or from a program bind LISPXHIST to NIL around the compilation.

References AR# 9486

Note: Thanks to Bill van Melle for help in writing this article.

Saving bitmaps

Release Lyric

Keywords BITMAP,VARIABLES, DEFPARAMETER

Topic Saving bitmaps with the VARIABLES file manager command.

Discussion Bulletin 9.4 stated that bitmap could not be saved using the VARIABLES file manager command and offered a workaround using IL:VARS instead. There is an alternative workaround that uses the VARIABLES file manager command and involves modifying the initial-value form specified in the DEFPARAMETER special form with SEDIT. It is important to note that the modifications are done in SEDIT because the current value of a parameter is not noticed by the file manager ie - (SETQ XXX (IL:BITMAPCREATE 50 50)) will not be noticed. The initial-value form which is specified in the DEFPARAMETER special form is the one that is saved to the file.

If the value you wish to save is a modified value of the parameter, then you should either use DEFPARAMETER again, sufficient for other data types, or in the case of a BITMAP, edit the variable and insert the new value into the declaration. To save the value of a BITMAP, use SEDIT as follows:

Example In the XCL Exec window do the following:

```
(DEFPARAMETER XXX (IL:BITMAPCREATE 50 50))
```

After giving the command (IL:DV XXX), the SEDIT window will show:

```
(DEFPARAMETER XXX (IL:BITMAPCREATE 50 50))
```

Replace the (IL:BITMAPCREATE 50 50) WITH 'XXX to get:

```
(DEFPARAMETER XXX 'XXX)
```

Then select the XXX that is quoted (don't select the quote) and use Meta-Z and the mutation function EVAL to get:

```
(DEFPARAMETER XXX '#<BITMAP @ 377,46256>)
```

After exiting SEDIT, the bitmap can now be saved with the VARIABLES file manager command.

Reference Lyric Release Notes, Section 3, Chapter 13, Interlisp Executive.

Note: Thanks to Roger Beeman for bringing this to our attention.

TEDIT PAGEFORMAT specifications

Release Lyric

Keywords TEDIT, TEDIT.SINGLE.PAGEFORMAT, TEDIT.COMPOUND.PAGEFORMAT

Question Is there a way to provide only one format specification for the PAGEFORMAT property in a TEdit call?

Background I tried to use TEDIT.SINGLE.PAGEFORMAT in order to have the same format specifications for all pages in a document -- something like the following:

```
(TEDIT NIL NIL NIL (LIST 'PAGEFORMAT (LIST
(TEDIT.SINGLE.PAGEFORMAT T 46.5 1.25 NIL 'RIGHT 4.5 4.5
1.25 4.5 1 42.0 NIL NIL 'PICAS))))
```

When I selected "SHOW" for Other Left and Other Right, I got a break with "ARG NOT PAGEREGION NIL."

Answer The PAGEFORMAT prop has to be either the result from TEDIT.COMPOUND.PAGEFORMAT, or a list of three results from TEDIT.SINGLE.PAGEFORMAT (first page, all other right (recto) pages, and all other left (verso) pages). Any other form of the argument isn't guaranteed to do anything reasonable.

To specify the same page format for all pages of a TEdit document, the desired page format will have to be specified as three identical TEDIT.SINGLE.PAGEFORMAT statements within a list or as arguments to a TEDIT.COMPOUND.PAGEFORMAT statement.

```
(TEDIT NIL NIL NIL (LIST 'PAGEFORMAT
(TEDIT.COMPOUND.PAGEFORMAT (TEDIT.SINGLE.PAGEFORMAT T 46.5
1.25 NIL 'RIGHT 4.5 4.5 1.25 4.5 1 42.0 NIL NIL 'PICAS)
(TEDIT.SINGLE.PAGEFORMAT T 46.5 1.25 NIL 'RIGHT 4.5 4.5
1.25 4.5 1 42.0 NIL NIL 'PICAS) (TEDIT.SINGLE.PAGEFORMAT T
46.5 1.25 NIL 'RIGHT 4.5 4.5 1.25 4.5 1 42.0 NIL NIL
'PICAS))))
```

Or:

```
(TEDIT NIL NIL NIL (LIST 'PAGEFORMAT (LIST
(TEDIT.SINGLE.PAGEFORMAT T 46.5 1.25 NIL 'RIGHT 4.5 4.5
1.25 4.5 1 42.0 NIL NIL 'PICAS)
(TEDIT.SINGLE.PAGEFORMAT T 46.5 1.25 NIL 'RIGHT 4.5 4.5
1.25 4.5 1 42.0 NIL NIL 'PICAS)
(TEDIT.SINGLE.PAGEFORMAT T 46.5 1.25 NIL 'RIGHT 4.5 4.5
1.25 4.5 1 42.0 NIL NIL 'PICAS))))
```

References A User's Guide To TEdit, Lyric Release; chapter 8, pages 11-13; Appendix A-Example; AR# 10038

Example use of TableBrowser

Release Koto and Lyric

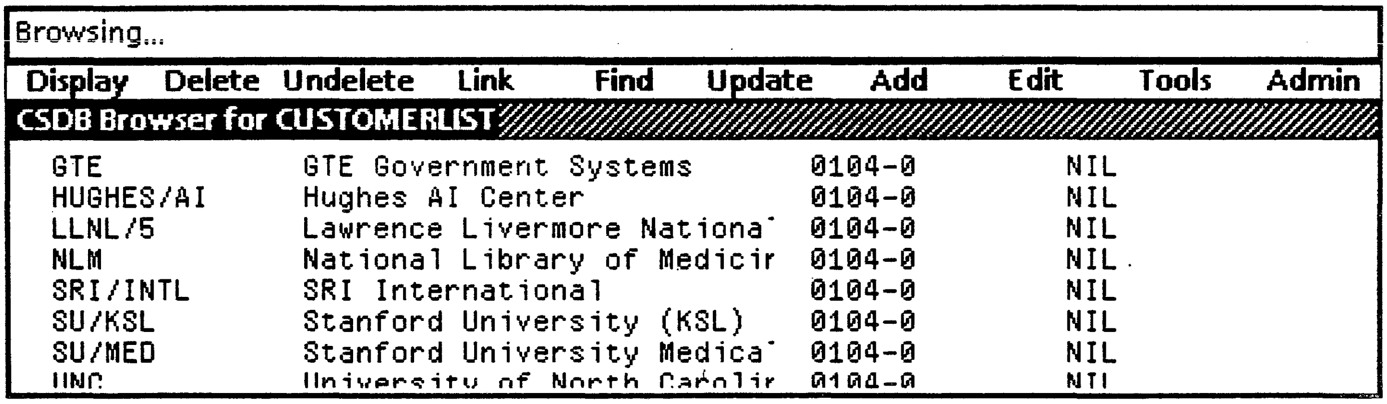
Keyword TableBrowser

Question How can I use the TableBrowser package in my applications?

Discussion I have been reviewing the TableBrowser documentation and it seems to have the functionality I desire but I am having trouble using it. Do you have an example that I can review?

Example TableBrowser implements a simple mechanism for building applications that browser certain kinds of tabular data. It supplies a basic set of functions that matintain the window, allowing scrolling of selected items; the application defines the items, how they print, and any higher-level operations to perform on then. FileBrowser is an example application built upon TableBrowser. If you are familiar with the Lafite mail package, and you consider the mail browser interface, you'll note some striking similarities.

A TableBrowser is a scrollable window which contains table items. Here is an example TableBrowser window with a user defined menu attached to the top:



Browsing...

CSDB Browser for CUSTOMERLIST

```
GTE          GTE Government Systems       0104-0      NIL
HUGHES/AI    Hughes AI Center             0104-0      NIL
LLNL/5       Lawrence Livermore Nationa'  0104-0      NIL
NLM          National Library of Medicir  0104-0      NIL
SRI/INTL     SRI International            0104-0      NIL
SU/KSL       Stanford University (KSL)    0104-0      NIL
SU/MED       Stanford University Medica'  0104-0      NIL
UNC          University of North Carolir  0104-0      NIL
```

What you see is our customer database program written by Greg Wexler. What is displayed is a TableBrowser window displaying about 8 of hundreds of table items, a title bar with a large font, an attached menu window and a prompt window using (GETPROMPTWINDOW) to the main browser window.

Below are a few function definitions and a series of calls that will set up and fill a sample TABLEBROWSER. Enter these into your Interlisp Exec window and you'll be able to experiment with the TableBrowser package functions listed below them.

<u>The Definitions</u>

```
(DEFINEQ (MY.GET.TBPROP (TBITEM PROP)
    (* * Returns a property to the table browser item's
    TIDATA field)
    (LISTGET (fetch (TABLEITEM TIDATA) of TBITEM) PROP)]
```

```
(DEFINEQ (MY.PRINTFN (BROWSER ITEM WINDOW)
    (* * Prints one item in the browser window.)

    (DSPXPOSITION 10 WINDOW)
    (PRINTOUT WINDOW (MY.GET.TBPROP ITEM 'NAME))
    (DSPXPOSITION 150 WINDOW)
    (PRINTOUT WINDOW (MY.GET.TBPROP ITEM 'ADDRESS))
    (DSPXPOSITION 300 WINDOW)
    (PRINTOUT WINDOW (MY.GET.TBPROP ITEM 'PHONE)]

(DEFINEQ (MY.CREATE.TBRECORD (PROPS)

(* * Creates a Tableitem in TBRECORD format)

(PROG (SELECTED DELETED UNDELETABLE UNSELECTABLE
UNCOPYSELECTABLE DATA USERPROPS (NUMBER 1))

    (DECLARE (SPECVARS SELECTED DELETED UNDELETABLE
    UNSELECTABLE UNCOPYSELECTABLE DATA))
    (* * For SET Below)
    (for TAIL on PROPS by (CDDR TAIL) do
        (SELECTQ (CAR TAIL)
            ((SELECTED DELETED UNDELETABLE UNSELECTABLE
            UNCOPYSELECTABLE DATA)
            (SET (CAR TAIL) (CADR TAIL)))
            (push USERPROPS (LIST (CAR TAIL) (CADR TAIL)))))
            (* Bring lists to top level for LISTGET
            operations)

        (SETQ USERPROPS (for X in USERPROPS join X)) (*
        create and return the Tableitem)
        (RETURN (create TABLEITEM TISELECTED ← SELECTED
        TIDELETED ← DELETED TIUNDELETABLE ← UNDELETABLE
        TIUNSELECTABLE ← UNSELECTABLE TIUNCOPYSELECTABLE ←
        UNCOPYSELECTABLE TIDATA ← USERPROPS TI# ←
        NUMBER)))]
```

To set up a sample browser:

```
(SETQ MY.WINDOW (CREATEW '(100 100 400 200) "Phone Book
Database "]

(SETQ MY.BROWSER (TB.MAKE.BROWSER NIL MY.WINDOW '(FONT
(GACHA 10) PRINTFN MY.PRINTFN]
```

To create a few items:

```
(SETQ MY.FIRST.TBITEM (MY.CREATE.TBRECORD '(NAME "John
Smith" ADDRESS "Beverly Hills" PHONE "213-555-1212"]

(SETQ MY.SECOND.TBITEM (MY.CREATE.TBRECORD '(NAME "G.
I. Joe" ADDRESS "Orange County" PHONE "714-555-1212"]

(SETQ MY.THIRD.TBITEM (MY.CREATE.TBRECORD '(NAME "Joe
Surfer" ADDRESS "San Diego" PHONE "619-555-1212"]
```

To put the items into the window:

```
(TB.INSERT.ITEM MY.BROWSER MY.FIRST.TBITEM]

(TB.INSERT.ITEM MY.BROWSER MY.SECOND.TBITEM]

(TB.INSERT.ITEM MY.BROWSER MY.THIRD.TBITEM]
```

Some useful TableBrowser module functions:

```
(TB.MAKE.BROWSER ITEMS WINDOWSPEC PROPS)
```

Makes the tablebrowser using the windowspec and props. Items is a list of TABLEBROWSER Items that is to be put into the browser. There are various properties that can be set. Font is obvious and PRINTFN is the function used to take each TABLEITEM and PRINT the information in the browser as one line.

```
(TB.DELETE.ITEM MY.BROWSER MY.SECOND.TBITEM]
```

```
(TB.UNDELETE.ITEM MY.BROWSER MY.SECOND.TBITEM]
```

```
(TB.REMOVE.ITEM BROWSER ITEM)
```

Removes an item.

```
(TB.NORMALIZE.ITEM BROWSER ITEM)
```

Scrolls the browser until ITEM is vertically centered in the window.

```
(TB.REDISPLAY.ITEMS BROWSER FIRSTITEM LASTITEM)
```

You can call READISPLAYW for that unless you know what items you want to display and wish to maintain speed!

```
(TB.COLLECT.ITEMS BROWSER PREDFN)
```

Returns a list of the items in the browser. PREDFN should be (quoted) DELETED, SELECTED, or NIL. If PREDFN is NIL, TB.COLLECT.ITEMS returns all of the browser's items. Otherwise it returns those items described by PREDFN

Inspecting the CSDB TableBrowser Window

If you are interested in the inner workings of TableBrowser read on, as I give an in-depth look at how information is stored using this package.

Among this scrollable window's properties is a list of table browser item records. First, here is an inspector of the window:

```
SCREEN              NIL
WINDOWENTRYFN       GIVE.TTY.PROCESS
PROCESS             NIL
WBORDER             4
NEWREGIONFN         NIL
WTITLE              "CSDB Browser for CUSTOMERLIST"
MOVEFN              (MOVEATTACHEDWINDOWS)
CLOSEFN             CSDB.CLOSE.BROWSER.WINDOW
HORIZSCROLLWINDOW   NIL
VERTSCROLLWINDOW    NIL
SCROLLFN            TB.SCROLLFN
HORIZSCROLLREG      (5 414 544 24)
VERTSCROLLREG       (0 438 24 221)
USERDATA            (TABLEBROWSER {TABLEBROWSER}#62,161220
                    COPYBUTTONEVENTFN TB.COPYBUTTONEVENTFN
                    SHRINKFN --)
EXTENT              (0 -867 617 1080)
RESHAPEFN           (TB.RESHAPEFN \CSDB.REDISPLAYBROWSER)
REPAINTFN           TB.REPAINTFN
CURSORMOVEDFN       NIL
CURSOROUTFN         NIL
CURSORINFN          NIL
RIGHTBUTTONFN       TB.BUTTONEVENTFN
BUTTONEVENTFN       TB.BUTTONEVENTFN
```

```
REG                   (5 438 544 234)
SAVE                  {BITMAP}#373,47234
NEXTW                 {WINDOW}#55,5000
DSP                   {STREAM}#55,4234
```

Inspecting the USERDATA field

To find the tablebrowser record, and from there the user-defined table items, inspect the USERDATA field from the above example. You will get something like:

```
TABLEBROWSER          {TABLEBROWSER}#62,161220
                      COPYBUTTONEVENTFN
                      TB.COPYBUTTONEVENTFN
SHRINKFN              CSDB.SHRINK.BROWSER.WINDOW
TOTOPFN               (TOPATTACHEDWINDOWS)
OPENFN                (OPENATTACHEDWINDOWS)
EXPANDFN              (EXPANDATTACHEDWINDOWS
                      \CSDB.NEEDSUPDATE?)
CALCULATEREGIONFN     ATTACHEDWINDOWREGION
MINSIZE               MINATTACHEDWINDOWEXTENT
MAXSIZE               MAXATTACHEDWINDOWEXTENT
DOSHAPEFN             RESHAPEALLWINDOWS
ATTACHEDWINDOWS       ({WINDOW#55,5000} {WINDOW#60,63320})
ALL.DISPLAY.KEYS      (SHORTNAME 0 LONGNAME 100 SUPPORTID
                      300 PRINCIPLECONTACT 400)
TOCFILE
{DANTE}<AISUPPORT>CUSTOMERDB>CUSTOMERLIST.TOC
CSDB.GETINW           CSDB.CL.GETINW
INTERNAL-TOCFILE      {CORE}CUSTOMERLIST
DBFILE
{DANTE}<AISUPPORT>CUSTOMERDB>CUSTOMERLIST.DB
PROMPTWINDOW          ({WINDOW#60,63320} . 1)
PWINDOW               {WINDOW#60,63320}
CSDB.WINDOWLOCK       {MONITORLOCK}#71,16270
FOLDERNAME            CUSTOMERLIST
LAST.UPDATE           "20-Aug-86 07:27:32"
ICON                  (\VAG2 54Q 33470Q)
CSDB.CUSTOMERINFOW    (\VAG2 54Q 33320Q)
ICONWINDOW            (\VAG2 54Q 33470Q)
ICONPOSITION          (480 . 727)
CSDB.QUICKFIND.QUERY "044"
```

Inspecting the TABLEBROWSER record

What we are interested in right now is the TABLEBROWSER record from the above inspection of the USERDATA, so lets look at that:

```
TBREADY               T
TBITEMS               ({TABLEITEM}#53,2510
                      {TABLEITEM}#53,2440
                      {TABLEITEM}#53,2454 --)
TB#ITEMS              95
TB#DELETED            0
TB#LINESPERITEM       1
TBFIRSTSELECTEDITEM    47
TBLASTSELECTEDITEM    48
TBMAXXPOS             536
TBFONTHEIGHT          12
TBFONTASCENT          9
```

```
TBFONTDESCENT      3
TBWINDOW           {WINDOW}#57,61640
TBLOCK             {MONITORLOCK}#71,16300
TBUSERDATA         NIL
TBFONT             {FONTDESCRIPTOR}#70,171670
TBEXTENT           (0 -927 536 1140)
TBUPDATEFROMHERE   NIL
TBCOLUMNS          5
TBPRINTFN          CSDB.TB.PRINT
TBCOPYFN           NIL
TBFONTCHANGEFN     NIL
TBCLOSEFN          NIL
TBAFTERCLOSEFN     NIL
TBTITLEEVENTFN     NIL
TBAFTEREXPUNGEFN   NIL
TBORIGIN           213
```

The TABLEBROWSER record has various properites with which the user can work.

TBITEMS shown above contains the list of TABLEITEM records in which is read by the TBPRINTFN which takes the TABLEBROWSER record, a TABLEITEM, and the WINDOW (in which the TABLEBROWSER record is in) and will do fetches of that record and obtain the proper information. Lets look at one such TABLEITEM:

```
TISELECTED         NIL
TIDELETED          NIL
TIUNSELECTABLE     NIL
TIUNSELECTABLE     NIL
TIUNCOPYSELECTABLE NIL
TIDATA             (ALLKEYS (SHORTNAME LONGNAME SUPPORTID
                   PRINCIPLECONTACT) FILEPTR 11244
                   PRINCIPLECONTACT --)
TI#                6
```

Not all equally interesting but let's review the fields:

TISELECTED: tells you if the TABLEITEM has been selected. You may notice that if you had made a selection in your example Tablebrowser window when you began these inspecting exercises, you will now see that data.

TIDELETED: Tells you if the TABLEITEM as been marked for deletion. That is, there would be a straight line though it. (Just like in FILEBROWSER!...as filebrowser uses tablebrowser).

```
TIUNDELETABLE:     cannot be marked for deletion

TIUNSELECTABLE:    cannot be selected

TIUNCOPYSELECTABLE: cannot be selected for copying
```

TIDATA: is just list the USERDATA field of a window. Any additional user defined properties are placed as a PLIST in this field.

TI#: is the number if the TABLEITEM. Note that this field is very important! It is set for you when you add a TABLEITEM to a given browser. Note that if you wish to sort the window property list which contains all the table items, you MUST change the field to reflect the order # where the TABLEITEM is. So if the sorting moved a TABLEITEM from being the 6th item in the list to being the first, you must change this TI# field to be 1 to reflect its location. If you don't, you'll find that if you select this first item, the 6th one will mark the TABLEITEM as selected in the browser and you get the ▶ mark pointing to the wrong TABLEITEM.

Looking at TIDATA:

```
ALLKEYS             (SHORTNAME LONGNAME SUPPORTID
                    PRINCIPLECONTACT)
FILEPTR             11244
PRINCIPLECONTACT    "NIL"
SUPPORTID           "0000-000"
LONGNAME            "Acme Corporation"
SHORTNAME           "ACorp
```

Well...these are some of my own fields but you can put what you like.

Reference Lyric Library documentation under Tablebrowser page 231.

Reporting a problem on the Hotline

Release Koto and Lyric

Keywords Break, Error Messages, Cursor Codes, Diagnostics

Topic Suggestions for information to have available when reporting a problem on the toll-free telephone hotline.

First things first To help us expedite a response to you, if you have certain information readily available when you place your call, we may be able to help you right away.

When you call the Hotline, the first thing we will ask for is your Support ID number. The Principal Contact for every customer who has purchased an SSMA should know the Support ID number for the site. While the workstation is still under its 90 day Warranty period, a valid Support ID number will be given to you.

After that, we need to know which release (Koto, Lyric) or product (PC Emulation, Prolog, LOOPS, or a Beta Test software kit) you are calling about, and which machine type (1108 or 1186) you are using.

What is the nature of your call? Additional information required depends in part on the nature of the report you are making: bug report, feature request, or documentation problem.

Is this a bug report? It is useful if you can describe the exact steps which led to the failure or bug. For example, it is better to say "After pressing F1-0 to boot the system tool, I got to the Base State, clicked the left mouse button, and the screen turned white and the cursor code said 0915," rather than "When I boot the system tool, it crashes with a 0915."

If you have entered a Break and cannot determine its cause, please remain in the Break, rather than pressing the up-arrow and aborting, when you call the Hotline. If this is not possible, please jot down the error message, and the stack as seen by selecting BT! from the menu. And again, the steps immediately preceding the Break. It is also useful to report other software modules loaded into the lisp.sysout. For example, to use RS232 to Chat, one must have many files besides DLRS232.LCOM and CHAT.LCOM loaded. Check the value of IL:LOADEDFILELST for a list of files you have loaded into the sysout. Sometimes just a review of the working environment will help you to solve the problem.

Sometimes it is not appropriate to remain in the Break and call the Hotline, but to take other action. For instance, if you get a Break which says "HARD DISK ERROR-Verify Error" when accessing the local disk, you would first consult your *1186 User's Guide* and see that to fix the problem it is necessary to run Scavenge! from the System Tool. Please be sure to jot down all of the steps taken, and the messages returned by the Scavenge program in the System Tool before calling the Hotline.

Examples of Lisp code, especially brief ones which can be reproduced in our own sysout, are very helpful. Open a file using the function DRIBBLE to collect the code for us. For example, a user gave us the following simple code which we could use to verify his problem:

If SEdit is called on a structure that is circular, it goes away and can't be killed by a Ctrl-D. Before long it gets a stack overflow, then errors within errors, and then bombs the machine with a 9319 code.

A simple case like

```
(SETQ INFINITE '(A))
 (CAR (RPLACD INFINITE INFINITE))
 DV INFINITE
```

can be killed with shift-Stop and killing its process, but then its window cannot be closed. Mousing the window causes an error, ARG NOT EditContext.

And lastly, the misbehavior of the system may not be due to a software bug but a hardware problem. There are many Diagnostics which you can run to determine if this is the case. Here are a few guidelines.

1. In the *1186 Hardware Installation Guide*, Chapter 3, are instructions for using the Configuration Utility. If you have trouble booting the system and have recently swapped displays or rigid disks, or have installed a hardware option yourself, your system may not be correctly configured. Also, if you do swap disks, refer to Appendix B for Installation instructions. If you are upgrading the 1186 from Koto to Lyric, the procedures for enabling 8K Control Store are described in Appendix C of this manual.

2. Problems on the rigid disk could manifest themselves in various ways. To ensure that the disk is not at fault for unexplainable boot or Lisp failures, use the Rigid Disk Confidence Test as described in Chapter 4 of the *1186 Hardware Installation Guide*.

3. The *1186 User's Guide* features a chapter "Diagnostics" with information on the Boot diagnostics which are available from the rigid disk, from a floppy, or from the network. These tests are not available from the rigid disk on the 1108. The User's Guide for this system describes the ALAG test.

4. The System Tool for both the 1186 and 1108 includes Online Diagnostics for Display, Keyboard, Ethernet Echo Test, RS232 Loopback, Floppy Drive, and Ethernet Statistics. Consult the System Tools chapter in the User's Guide.

Are you making a feature request?

If so, we can file a better report if you can help us answer the following questions. What behavior should the feature exhibit, as opposed to the way the system works now? If not obvious, what are the benefits or why do you want this feature? What is the potential impact to yourself and perhaps to other customers if this feature were implemented? Cost savings? Time savings?

Example:

I would like to make a Feature Request. Currently, I cannot use the Sysin! command in the System Tool if my sysout is stored on a device which I can only access by using TCP/IP. It would save all 20 of the workstation users on my network 20 minutes per

Sysin minimum and a lot of aggravation that working with floppies involves if we could have this feature as soon as possible.

Is this call related to the documentation? If you find the documentation lacking in some explanation, please let us know. Have ready the name of the document and its date (from the title page of the manual) and the page number where the unclear explanation is, or perhaps where the missing information should appear.

Summary Here is a checklist of information to provide during the telephone call.

1. Support ID number, Release and Machine Type.
2. Nature of report: bug, feature request, question about documentation.
3. Steps or example of code leading to discovery of bug from a DRIBBLE file.
4. Procedures or steps just prior to system freeze.
5. Cursor Code when system froze.
6. Error Message and Stack shown in Break Window from BT! selection.
7. Diagnostic tests you have run and the results.
8. Other software loaded into the lisp.sysout (e.g. Library and/or User Modules, LOOPS, or some other non-Xerox application.) from IL:LOADEDFILELST.

References 1108 UG Lyric Release, 1186 UG Lyric Release, 1186 Hardware Installation Guide, LOADEDFILELST is documented on IRM II, 17.20, DRIBBLE is documented on IRM III, 30.12 BT! is documented on IRM II, 14.3