Known Problems
and Additional Information

Installation and System Tools

● **Warning** : Using "Make Script!" to produce a user specified partition will permanently alter the Installation Floppy. When booted, the first menu option will partition according to your specification; the remaining options must not be used as they are not appropriate for the Intermezzo release.

Workaround: Duplicate the original Installation Floppy with the Floppy Tool to preserve the original Installation menu. Use the duplicate floppy for "MakeScript!".

● "Make Script!" does not check volume names for duplication; do not specify volumes with the same name.

● When specifying a logical volume in the System Tool, use the Volume Menu, rather than explicitly typing the volume name; otherwise, related parameters, such as VMem Size, will not be automatically updated.

● The "Duplicate!" command in the Floppy Tool will not duplicate a floppy formatted by Interlisp's FLOPPY.FORMAT function. Attempting to do so will result in the error message "Floppy Read Error".

Workaround: Format the floppy using the Floppy Tool's "Format!" command, then write files to the floppy in Lisp as you have always done. The floppy can then be copied with the "Duplicate!" command. (This will not work with (SYSOUT '{FLOPPY}) which automatically formats each floppy in Lisp.) Alternatively, use the Interlisp functions FLOPPY.TO.FILE and FLOPPY.FROM.FILE. Those functions have the added advantage that they do not require free disk pages for scratch space.

● If an NS host name containing embedded blanks, such as *Alpine Server*, is presented to the Profile Tool, it will be incorrectly supplied to the other tools as *Alpine*.

Recovery Action: When the host name is presented to the Profile Tool, embed the fully qualified name in double quotes; e.g., "Alpine Server:XSIS:Xerox".

● In the Online Diagnostics Tool, the RS232 loopback test program will always fail with MP code 0915.

FLOPPY

It is advised that before using either FLOPPY.COMPACT or FLOPPY.SCAVENGE that the floppy be duplicated .

● FLOPPY.COMPACT can cause the floppy to become unreadable.

● FLOPPY.SCAVENGE can cause a bad file name break and the floppy must be reformatted to be usable.

● In CPM mode while attempting to DELFILE a floppy file, DIR will still show the deleted file until the floppy drive door is opened and closed, at which time the directory is reset.

● In CPM mode, FLOPPY.SCAVENGE will return the message "not a Pilot floppy" .

DEDIT

● The Tab key will be disabled after selecting DEdit from the inspector.

Recovery Action: (KEYACTION TAB '((9 9)))

FILE SYSTEM

● If you load a compiled file whose source is no longer available, but a more recent source is available, then after making a change during a cleanup, MAKEFILE will not copy the sources from the latest source file. It will also not put any function definitions onto the file.

● If your Lisp source file has an extension , then LOAD? fails to notice when the file has been loaded. For example, if the source is named MOCKUP.LSP then an initial (LOAD? 'MOCKUP.DCOM) puts a FILEDATES property on the atom MOCKUP.LSP . Any subsequent LOAD? looks on the property list of the different atom MOCKUP, and failing to find a FILEDATES property there, reloads the file. This is fixed in the next release.

● CNDIR and (DIRECTORYNAME T) return the incorrect pathname when connecting to a subdirectory on DSK. For example, (CNDIR '{DSK}<LISPFILES>FOO >) returns {DSK}<LISPFILES>FOO. This confuses packages that use UNPACKFILENAME.

●Since the ACCESS attribute is only meaningful for open files you must pass the full filename including version number when calling GETFILEINFO with attribute ACCESS.

● COPYFILE between disk and floppy, when the time has not been set, will result in the MP code error 9318.

2

● Warning: COPYFILE will break if the file is already open on a non-random-access device.

● VOLUMEDISPLAY has been changed to DSKDISPLAY and DFSCREATEDIRECTORY has been changed to CREATEDSKDIRECTORY. Both names are supported in Intermezzo but only DSKDISPLAY and CREATEDSKDIRECTORY will be supported in the next release.

SYSTEM SOFTWARE

● In earlier releases, the function CHANGEBACKGROUND returned the previous value of the background shade, either a SMALLP or T, so it was possible to write

(RESETFORM (CHANGEBACKGROUND BLACKSHADE) (FIE FOO))

Under Intermezzo CHANGEBACKGROUND always returns the value 34850, which is the value that represents the default background shade.

ARITHMETIC

● ANTILOG of numbers less then -22.2 will be incorrect, due to a premature overflow.

STORAGE FORMATS AND MANAGEMENT

● It may rarely happen that when a SYSOUT does not complete successfully \VMEM.INHIBIT.WRITE will be set to T .

If this happens: (SETQ \VMEM.INHIBIT.WRITE NIL)

RECORD PACKAGE

● New Feature: When UNDOing a LOAD which created a datatype, the allocation of the datatype is undone. The system will prompt with the message asking "OK to deallocate DATATYPE FOO?" .If you have no instances of the datatype reply yes. If in doubt reply no.

COMMUNICATIONS

● The middle-button CLEAR command in CHAT does not work.

● Middle buttoning NEW while chatting to a server will put the input cursor in the top level window but the HOST: prompt in the prompt window.

NS SERVERS

● The Xerox Network Systems file server's communication code sometimes times out connections with Lisp clients, due to the great variability in the responsiveness of the Lisp communications code. This code runs at the same priority as other Lisp tasks and hence can be locked out by competing processes. In Intermezzo, the Lisp Filing code has been modified to reestablish the connection if it is lost while reading an input file, the most common case to fail previously. Connections can still be lost in some cases when writing files, or when performing non-filing operations (e.g. printing), especially if there are compute-intensive processes running in parallel. In future releases, the process that manages the connections will run at interrupt level, and timeouts will be avoided.

● Note: The function SPP.FLUSH is subsumed by the generic FORCEOUTPUT operation on streams.

FONTS

● (FONTSAVAILABLE '* '* '* '* 'INTERPRESS T) breaks with message
ARG NOT LIST
*

Lisp Library Packages

TEDIT

● When 2 menus are displayed, and the bottom one is deleted, upon redisplay that menu will overlap the top menu.

Workaround: Close top menus first.

● When doing both Get and Include from floppy, both files must be on the same floppy.

● Attempting to Undo an Expand breaks.

TEDIT Programmer Interface

● Some TEdit functions unexpectedly move the caret. If you're using the functional interface you may find, for example, that after a TEDIT.LOOKS your caret has moved. Therefore, a subsequent TEDIT.INSERT puts text in the wrong place. Some problems have been found which have dependencies on such things as the size and shape of the TEdit window.

Workaround: Never insert "at the caret" without setting it immediately beforehand.

● POSTPARALEADING does not work.

● It is not recommended that you use Invisible characters because they are very error prone.

FILEBROWSER

● The SEE menu option in Filebrowser now requires that one load TEDIT.DCOM.

BIG

● If Big.dcom is loaded, it redefines various font classes, such as STANDARD, and adds new ones, such as BIG, and HUGE. If you change your current fontset to be any of these, e.g., (NEWFONT 'BIG), Interpress font information becomes unavailable, i.e., you cannot hardcopy to an NS printserver.

Workaround: Either call (FONTSET 'PARC) or DV FONTDEFS and supply Interpress font definitions.

CMLSPECIALFORMS

● PROGV treats NIL values as "unbound" rendering it useless in most situations.

● DEFUN creates functions which confuse DEdit, since they contain shared structure. We recommend that DEFUN not be used. If it must be used (and the results edited), copy the resulting definition using something similar to (MOVD 'FOO 'FOO T).

COREHAX

● PRINTVMEM breaks with NON-NUMERIC ARG.

CMLARRAY

● MAKEARRAY's DISPLACEDINDEXOFFSET argument does not work correctly for arrays of different types. Also, MAKEARRAY currently won't displace to a FLOATP array.

● ADJUSTARRAY should not be used because it can severely damage your virtual memory.

● AREFs compile into ASETs.

Workaround: This can be fixed by disabling the appropriate compiler optimization: (REMPROP 'ASET 'DMACRO).

● Beware of (MOD n) arrays. They are not handled correctly by the file package nor, for some values of n, by ASET and FILLARRAY.

MAKEDATABASE

● This package has been withdrawn. Please remove page 101 from your Lisp Library manual.

LABEL

● This package is supported only for use with DECL.

SINGLEFILEINDEX

● MERGEDFILEINDEX does not work as documented for a list of files. Setting RELATIVEINDEXFLG to T or BOTH breaks.

FLOATARRAY

• MAPELT applied to more than two arrays will corrupt memory.

• If arrays of inappropriate sizes or types are passed to MAPELT, memory will be corrupted.

Lispusers Packages

Please note that Lispusers packages are supplied by Xerox as a service to the user community. Xerox is not responsible for the correctness of the software or documentation.

EDITFONT

- BLANKFONTCREATE breaks everytime with "undefined car of form SMALLPOSP".

- The documentation for EDITFONT has not been updated for Intermezzo to describe the changes in the middle mouse button menu. The changes are:

- closing the font edit window is equivalent to the old command STOP
- EXIT is equivalent to the old command OK
- DELETE is equivalent to the old command DUMMY

EMACS

- The Backspace key function has changed

- ↑ F, ↑ K, ↑ N ↑ P all work on paragraphs, not lines

- # key is ESC

- #V doesn't scroll

- #G - text disappears

- #E breaks and text disappears

EMACSUSER

- EMACSUSER loads EMACS, an editor built atop TEDIT. Therefore, TEDIT control characters are disabled .

- The subsection ARITHMETIC, describes shorter function names which are supplied for arithmetic operations. These shorter names in fact map to Interlisp integer functions, not fixed point, as documented.

LOADFILES

- The variable COM for use in the ext field is (LIST COMPILE.EXT NIL) not (dcom nil) as described. In addition LOADFILES uses LOAD?.

TEDITKEY

- Loading TEDITKEY turns off Interrupts.

SYSTAT

- It is strongly recommended that you do not use this package.

COMMENTS

- COMMENTS redefines the system function DC.

PRINTOUT

- Printout package command .SUP does not work

TMENU

- TMenus cannot be re-shaped.
- CloseFileMenus does not work.
- CPROMPT surrounds its output with a pair of parentheses.

ACTIVEREGIONS

- DOWNFN is not performed after the first left down button in a region. UPFN is activated whenever the button comes up in a region.

- FINDACTIVEREGION arguments should be (Window xcoord ycoord regionlist/optional) instead of (Window Position Regionlist/optional) as documented.

EDITMACROS

- Actual syntax of SWITCH is (SWITCH @1 WITH @2).

SIMPLEFNS

- STRINGSUBST smashes its arguments.

Apparent Problems with non-Xerox Software

● Interlisp does not support VAX/VMS relative version numbers; i.e., DIR {XS}<PUBLIC>X.X;0 returns NIL even though the file X.X exists. Note, however, that DELFILE correctly interprets relative version numbers; for example, (DELFILE '{XS}<PUBLIC>X.X;0) correctly deletes the highest version of X.X.

● VAX/VMS running DEI has a limit of 9 characters for file names and 3 character extensions. Distributed font file names are substantially longer.

Workaround: Store fonts on your LispFiles volume or rename all the fonts to meet the VAX name requirements.

● Warning - In some versions of VAX/UNIX , files without extensions have fullnames such as "FOO;1." (note the period AFTER the version number). If given to OPENSTREAM, Interlisp breaks with "file not found".

● In order to avoid a problem with certain versions of the Unix leaf server, if you want to open a file with access mode APPEND or BOTH you should specify OLD as the recognition mode.

TABLE OF CONTENTS

Input/Output

● TYPE attribute of files extended to include arbitrary file types

The TYPE attribute of files has been extended to include arbitrary file types. Formerly, the value of this attribute was either TEXT or BINARY. Now, other values are permitted for those devices/hosts that understand more specific TYPEs. The new values can be passed to OPENSTREAM and SETFILEINFO, and retrieved via GETFILEINFO. For example, when creating an Interpress file, Lisp calls OPENSTREAM with a PARAMETERS argument that includes (TYPE INTERPRESS).

Devices that do not "understand" arbitrary values for TYPE treat unknown types as BINARY. Thus, GETFILEINFO may return the more general value BINARY instead of the original type that you passed to SETFILEINFO or OPENSTREAM. The following devices recognize general types: CORE, the Dorado DSK, NS File Servers. Currently, no non-Xerox servers support other than the default types.

The variable FILING.TYPES is used to associate symbolic types with numbers for Xerox products that actually store the TYPE attribute as a number (such as NS file servers). For example, suppose there existed an NS file type MAZEFILE with numeric value 5678. You could add the element (MAZEFILE 5678) to FILING.TYPES and then use MAZEFILE as a value for the TYPE attribute to SETFILEINFO or OPENSTREAM. Other devices are, of course, free to store TYPE attributes in whatever manner they wish, be it numeric or symbolic.

● File operations to files protected against owner produces "PROTECTION VIOLATION" error instead of hanging

If Interlisp is prevented from executing a file operation because of file protection, it tries to connect to the appropriate directory. If a password is needed to connect to the directory, the user is prompted to supply a password. However, if the file is even protected against the person connected to the directory, then Interlisp used to just hang forever. Now, Interlisp detects this condition, and generates a "PROTECTION VIOLATION" error instead of hanging.

● Directory enumeration to VMS/DEI works when device name but not host name is given

Previously, the "device" information in a directory specification was not passed to the remote server correctly if the host name was defaulted to the connected host. This meant that (DIRECTORY '{HOST}DEVICE:<DIR>*) worked but (CNDIR '{HOST}) (DIRECTORY 'DEVICE:<DIR>*) wouldn't.

● Directory enumeration to TOPS-20 hosts fixed in sysout

In the Harmony release, there were some bugs with directory enumeration to TOPS-20 file servers. This could be fixed by the patch file TOPS20FTPPATCH. This fix has been incorporated into the Intermezzo release sysout.

● Directory enumeration to UNIX file servers accepts wider range of directory specifications

The following directory commands to the UNIX file server {UN} are equivalent:

DIR {UN}</user/jones/>*

DIR {UN}<user/jones>*

DIR {UN}/user/jones/*

The file names printed will all have the same structure: {UN}<user/jones>xxx.

● New command NDIR; prints multi-column directory listings

NDIR, formerly only an EXEC command, now works at the top level. "NDIR FileGroup" prints the file group in a multi-column format.

● OPENFILE, DELFILE, etc. accept strings as filenames

Most of the file manipulation functions accept strings as file names. However, some functions (READ, PRINT, CLOSEF) still interpret string arguments as the string to read or write characters to.

● The {NULL} device returns a new stream with each OPEN

Previously, all streams opened to the {NULL} device were the same stream, which caused conflicts if one process wanted to use a null stream for input while another wanted to use a null stream for output. Now, the {NULL} device returns a new stream with each OPEN.

● {NODIRCORE} files now contain the same file attributes as other files

● Sending output to a re-opened {NODIRCORE} file no longer causes an infinite loop

● UNPACKFILENAME accepts UNIX file pathnames

UNPACKFILENAME treats a UNIX file pathname between slashes as a directory specification. For example,

←(UNPACKFILENAME '/a/rosie/rosie2.4/ELLIE.MAY;3)

(DIRECTORY a/rosie/rosie2.4 NAME ELLIE EXTENSION MAY VERSION 3).

● Interlisp-D can now access all 19 partitions on an 1132

Previously, Interlisp was only able to access the first five partitions on an 1132, using the devices {DSK1} through {DSK5}. It is now possible to access all of the partitions on 19-partition 1132's using the devices {DSK1} through {DSK19}.

● After disk full error on 1132, can delete files and continue

Previously, after a disk full error on an 1132, deleting some files and trying to resume the interrupted operation would cause a "Hard disk error." Now, when a disk full error is received from the Dorado local disk, deletion of files will return the disk to a useable state.

Printing

● Can specify printer, file name using "Hardcopy" background menu command

The "Hardcopy" command in the background right-button menu prompts the user for a region on the screen, and send the bitmap image to the first printer on DEFAULTPRINTINGHOST. Sometimes, the user wants to save the image in a Press or Interpress-format file, or to send it to a non-default printer. Previously, the only way to do this was to explicitly call the function HARDCOPYW. Now, the "Hardcopy" menu command has a submenu that allows the user to easily specify this information.

If the "Hardcopy" command is selected, it works exactly as before. However, the command now has a submenu, indicated by a gray triangle on the right edge of the "Hardcopy" menu item. If the mouse is moved off of the right of the menu item, another pop-up menu will apear giving the choices "To a file" or "To a printer." If the user selects the item "To a file," he will be prompted to supply a file name, and the format of the file (Press, Interpress, etc.), and the specified region will be stored in the file.

If the user selects "To a printer," he will be prompted to select a printer from the list of known printers, or to type the name of another printer. If the printer selected is not the first printer on DEFAULTPRINTINGHOST, the user will be asked whether to move or add the printer to the beginning of this list, so that future printing will go to the new printer.

● New "Hardcopy" menu command in default window right-button menu

The command "Hardcopy" has been added to the default right-button window menu. This is similar to the background menu "Hardcopy" command, in that the user can print to a file or specify a printer by using a submenu. However, instead of printing a portion of the screen, it prints the contents of the specified window. In addition, if the window has a windowprop HARDCOPYFN, it is called with two arguments, the window and an image stream to print to, and the HARDCOPYFN must do the printing. In this way, special windows can be set up that know how to print their contents in a particular way. If the window does not have a HARDCOPYFN, the bitmap image of the window (including the border and title) are printed on the file or printer.

● Star-compatible Interpress filetype used when creating, detecting Interpress files on NS file servers

Files on NS file servers have an associated numerical file type. Interlisp does not normally use these file types, although the user can read and set them (using the FILETYPE file attribute). However, this causes problems when accessing files from the Star workstation, which does use the filetypes. Specifically, Star didn't recognize Interpress files generated from Interlisp, because they didn't have the right filetype (4361). Therefore, Interlisp now creates Interpress files with the standard Interpress filetype. This filetype is also used in INTERPRESSFILEP to quickly detect whether a file is an Interpress file: if the filetype is correct, Interlisp won't try parsing the file to see if it is a valid Interpress file.

• New function BITMAPIMAGESIZE returns size of bitmap in a given stream's units

(BITMAPIMAGESIZE Bitmap Dimension Stream)

Returns the size that Stream will be when BITBLTed to Stream, in Stream's units. Dimension can be either WIDTH, HEIGHT, or NIL, in which case the dotted pair (width . height) will be returned.

• (OPENIMAGESTREAM xx 'DISPLAY) returns an imagestream instead of a window

• LISTFILES only creates one process to list multiple files

• (EMPRESS <press file>) will not print three copies of file

Previously, under some circumstances EMPRESS of a Press-format file would cause three copies of the file to be printed, when one copy was specified. This has been fixed.

Fonts

• Printing old documents doesn't give "ILLEGAL ARG NNN" error

In the Harmony release, TEdit could produce files containing fonts with a "face" of NNN instead of MRR. The patch file FontNNNPatch was distributed to fix this bug. The Intermezzo release has been fixed so that it will not generate bad font faces. In addition, in order to allow the user to read old files, NNN is accepted as a font face.

• (FONTSAVAILABLE <family> '* <face> <rotation> 'PRESS T) can return fonts with size = 0

For some Press font families/faces, the font widths for different sizes are consistently scaled versions of the smallest font in the family/face. Therefore, instead of storing data about all of the sizes in the FONTS.WIDTHS file, only the widths for the font of size = 1 are stored, and the other widths are calculated by scaling these widths up. This is signified in the press FONTS.WIDTHS file by a font with size = 0. Therefore, if FONTSAVAILABLE is called with CHECKFILESTOO? = T, and it finds such a "relative" font, it returns a font spec list with size of 0. For example,

```
←(FONTSAVAILABLE 'GACHA '* '* 0 'PRESS T)
((GACHA 0 (BOLD ITALIC REGULAR) 0 PRESS)
 (GACHA 0 (BOLD REGULAR REGULAR) 0 PRESS)
 (GACHA 0 (MEDIUM ITALIC REGULAR) 0 PRESS)
 (GACHA 0 (MEDIUM REGULAR REGULAR) 0 PRESS))
```

This indicates that press files can be created with GACHA files of any size with faces BIR, BRR, MIR, and MRR. Of course, this doesn't guarantee that these fonts are available on your printer.

1108 Local File System

● *Incompatible Change:* Local file system format changed; many local file system functions renamed

The 1108 low-level disk format has been changed. To upgrade from Harmony to Intermezzo, do the following: (1) within Harmony, copy any valuable local disk files to floppy or file server; (2) purge the local file directory from the old sysout using DFSPURGEDIRECTORY; (3) erase the local file volume using the *Installation Utility* floppy (see the *1108 Users Guide*); and (4) use CREATEDSKDIRECTORY to recreate any local file directories on local disk logical volumes.

It is not possible to access the files on a local file directory created in Harmony from an Intermezzo sysout, and vice-versa. However, it will not damage the integrity of the local file volume if it is tried accidentally.

● *Incompatible Change:* Many local file system functions renamed, changed

A number of the user functions for the local file system have been renamed, as follows:

DFSCREATEDIRECTORY = = > CREATEDSKDIRECTORY

DFSPURGEDIRECTORY = = > PURGEDSKDIRECTORY

VOLUMEDISPLAY = = > DSKDISPLAY

SCAVENGEVOLUME = = > SCAVENGEDSKDIRECTORY

(Note: The local file system scavenger has been put into the standard system, so the separate library package DlionFSScavenge is no longer necessary.)

The function VOLUMETYPE has been removed, and its functionality is available in somewhat different form using the new function LISPDIRECTORYP.

The new function (VOLUMESIZE <volume name> <recompute>) is analogous to DISKFREEPAGES, except it returns the TOTAL size of the volume.

For more information, see the *1108 Users Guide*.

● Interlisp on 1108 will not write beyond end of virtual memory file

In previous releases, there was no protection against Interlisp trying to read or write beyond the end of the virtual memory file. Therefore, the virtual memory file was required to be at least 16200 pages long (so that Interlisp would reach the end of its virtual memory limits before it reached the end of the file), or to be the last logical volume on the disk.

This has been fixed so that Interlisp will not read or write beyond the end of the logical volume.

● File enumeration fixed when local file volume is not given; DIR {DSK}FOO* works

Previously, there was a bug such that if the logical volume were not given, directory enumeration of "partial filenames" wouldn't work correctly. This caused DIR {DSK}<LISPFILES>FOO* to work correctly, but DIR {DSK}FOO* to always return NIL. This has been fixed.

● Directory enumeration on 1108 local file system sorts files alphabetically and by version number

● Directory can expand; no limit on maximum number of files

Previously, the 1108 local file system had a fixed directory size, enough to accommodate 500-700 files. The directory can now expand, to accommodate as many files as there is room to store on the local file system volume.

● COPYFILE to local file system preserves file type information

NS File Servers

● Using Services 8.0 NS file servers doesn't create strange subdirectories

Under some circumstances, a file operation that created a subdirectory on an NS file server would create a strange subdirectory of the form {NS:}<Drawer>Drawer!3>dir5>file. This has been fixed.

Window System

● New function SHRINKBITMAP for reducing bitmaps

(SHRINKBITMAP   Bitmap   WidthFactor   HeightFactor   DestinationBitmap)

Returns a copy of Bitmap that has been shrunken by WidthFactor and HeightFactor in the width and height, respectively. If WidthFactor is NIL, it defaults to 4; HeightFactor defaults to 1. If DestinationBitmap is not provided, a bitmap that is 1/WidthFactor by 1/HeightFactor the size of Bitmap is created

and returned. WidthFactor and HeightFactor must be positive integers.

● Menu items shaded with SHADEITEM will be reshaded on redisplay

● Reshaping the typescript window will not lose the caret

Previously, if a long typescript window with the caret near the top was shaped smaller, the caret would disappear. This has been fixed in this release.

● Typing control-E while shaping window will not leave garbage on screen

TEdit

● *Incompatible Change:* TEdit changed to use new, incompatible file format

The Intermezzo release of TEdit uses an incompatible file format, which makes TEdit files smaller and faster to load. Therefore, TEdit files written in Intermezzo will *NOT* be readable in Harmony. TEdit will continue to read old-format files at least through the Jazz release of Interlisp.

● *Incompatible Change:* The meaning of the CHLIM field of the SELECTION datatype has changed

The meaning of the CHLIM field of the SELECTION datatype *HAS CHANGED*. It is now *one higher than* the character number of the last character in the selection. The CH # field and DCH field retain their old meanings. Now, it is possible to derive any one of those fields from the other two. People whose code depends on the correct value of CHLIM must change their code.

● *Incompatible Change:* Only "registered" image objects can be read

The mechanism for printing and reading image objects has been changed. In order to read an image object, it must be "registered" in a database in memory. Unknown image objects are "encapsulated" without being interpreted. This makes it possible to GET a TEdit document even if all of the image objects in the document are not defined. For more details, see the documentation for *Image Objects.*

● Can systematically substitute one set of character looks for another with TEDIT.SUBLOOKS

(TEDIT.SUBLOOKS Stream OldLooksList NewLooksList) [Function]

This function can be used to systematically substitute one set of character looks for another throughout a given document. This can be used to change all instances of a given font to another font. Stream is the textstream which is to be changed. OldLooksList and NewLooksList are both property lists of

properties and values, in the form of the NewLooks argument to TEDIT.LOOKS. OldLooksList identifies the characters that are going to be changed, and NewLooksList tells how the selected characters will be changed. For example,

```
(TEDIT.SUBLOOKS <Stream>
  '(FAMILY MODERN WEIGHT BOLD SLOPE ITALIC)
  '(FAMILY CLASSIC WEIGHT MEDIUM))
```

will find all the Modern-Bold-Italic characters in the specified textstream, regardless of their size, expansion, etc. All such characters will have their family changed to CLASSIC, and their weight changed to MEDIUM. All other properties of the looks would be left unchanged.

Known Problem: Currently, TEDIT.SUBLOOKS does not update the display. Therefore, it is necessary to "redisplay" the TEdit window to see the changes.

● Paragraph formatting can include "special positioning"

TEdit paragraph formatting can include "special positioning" information, which specifies that a given paragraph should start at a given X,Y position on the paper. This can be used to exdent paragraphs into the left margin of the page.

The paragraph-looks menu includes the new fields:

Special Locn: X {}picas, Y {}picas

Including a value in the X or Y fields turns special positioning on for the paragraph. Setting them to zero removes any special positioning.

Special positioning can be specified in a call to TEDIT.PARALOOKS by unsing the property list names SPECIALX and SPECIALY as part of the NewLooks argument.

NB: Special positioning is ignored for a paragraph that is part of a page heading.

● Functional interface to page layout: TEDIT.PAGEFORMAT

(TEDIT.PAGEFORMAT Stream Format)   [Function]

This function sets the page format specifications for the given text stream. Format is either a format specification for a single page, which will be used for all pages in the document, or a "compound specification," which gives individual pages specifications for the first page, all other right (recto) pages, and all left (verso) pages. To create a "single page" format specification, use the function TEDIT.SINGLE.PAGEFORMAT (described below). To create a compound specification, generate three single format specifications, and combine them into one with the function TEDIT.COMPOUND.PAGEFORMAT (described below).

(TEDIT.SINGLE.PAGEFORMAT Page # s? Pg # X Pg # Y
Pg # Font Pg # Alignment Top Bottom Left Right # Cols
ColWidth InterColSpace Units)   [Function]

Creates and returns a PAGEREGION object describing the page format specified by the arguments:

Page # s?: T if you want page numbers on this kind of page, else NIL.

Pg # X: The horizontal location of the page number, measured from the left edge of the paper. Negative values are measured from the paper's right edge.

Pg # Y: The vertical location of the base line for the page numbers, measured from the bottom of the paper. Negative values are measured from the top of the paper.

Pg # Font: The font to be used to display the page numbers. This can be any specification that is acceptable to TEDIT.LOOKS.

Pg # Alignment: An atom that tells how the page number is to be aligned on the location specified by Pg # X and Pg # Y. LEFT means the location is the lower, left corner of the page number. RIGHT means the location is the lower, right corner. CENTERED means the page number will be centered around the Pg # X you specified.

Top: The top margin of the page—the distance from the top of the paper to the top of the first line of body text.

Bottom: The bottom margin—the distance from the bottom of the last line of body text to the bottom of the paper.

Left: The left margin—the distance from the left edge of the paper to the left edge of the first text column.

Right: The right margin—the distance from the right edge of the rightmost text column to the right edge of the paper.

Cols: Number of columns (default 1)

ColWidth: The column width (default is to evenly divide the available space among the # Cols columns)

InterColSpace: The space between the right edge of one column and the left edge of the next column. Defaults to evenly divide the space left after the columns are set up. If there is more than one column, on or the other of ColWidth and InterColSpace *must* be specified.

Units: The units used in setting the values you specified. May be one of the atoms PICAS, IN, INCHES, CM, POINTS. Default is PICAS.

(TEDIT.COMPOUND.PAGEFORMAT FirstSpec
        VersoSpec RectoSpec)   [Function]

Creates and returns a "compound specification," which gives individual pages specifications for the first page, all other right (recto) pages, and all left (verso) pages. FirstSpec, VersoSpec, and RectoSpec should be PAGEREGION objects created by TEDIT.SINGLE.PAGEFORMAT.

● New function TEDIT.RAW.INCLUDE for quickly including characters from unformatted files

(TEDIT.RAW.INCLUDE Stream InFile Start End)   [Function]

This is the same as TEDIT.INCLUDE, except that the specified characters from InFile are included without checking to see if the InFile is a TEdit file or a Bravo file. This is much faster, if you can guarantee that the source is plain unformatted text.

● Abbreviation facility extended: hooks for calling arbitrary functions, abbreviations are upper-cased

The TEdit abbreviation expansion facility has been extended: If an abbreviation's expansion is a LITATOM, it is applied as a function to the text stream and the abbreviation string, and must return a string/charcode value that is the expansion. If the abbreviation's expansion is a list, it is evaluated and the result used. Also, if the abbreviation expander doesn't find an abbreviation as typed, it converts it to upper-case and tries again.

● New function TEDIT.GETPOINT; returns character number that selection is inserting before

(TEDIT.GETPOINT Stream Sel)   [Function]

Returns the character number that the next character typed would be inserted in front of. If Sel is non-NIL, this is the selection that the information is taken from. Otherwise, the info is taken from the current selection of Stream.

● Can change highlighting of a selection with TEDIT.SETSEL or new function TEDIT.SET.SEL.LOOKS

A TEdit selection is highlighted differently depending on whether it is shift-selected (dashed underline), pending-delete selected (inverse letters), or selected normally (solid underline). The following new function can be used to change the highlighting of a selection:

(TEDIT.SET.SEL.LOOKS Sel Operation)   [Function]

This function changes the highlighting for the selection Sel, which should be a TEdit selection currently visible in a window. Operation, which can be one of the atoms NORMAL, MOVE, COPY, DELETE (to make the selection look like that kind of selection), or the atom INVERTED (which just inverts the selection while leaving the caret flashing).

Note that TEDIT.SET.SEL.LOOKS doesn't change TEdit's internal state with respect to what type of selection has been made. For example, if a word has been pending-delete selected (inverse letters), and the highlighting has been changed to NORMAL (solid underline), the next character typed to that TEdit will STILL delete the selected word. Also, TEdit does not remember the "selection looks," so redisplaying the edit window will set them back to what they were originally.

An Operation argument has been added to TEDIT.SETSEL, which is interpreted the same as with TEDIT.SET.SEL.LOOKS.

● Repeatable MP 9318 with very large TEdit documents fixed

Previously, when editing a very large formatted TEdit documents (on the order of 70 pages), TEdit would break with a MP 9318. This has been fixed. Now, the maximum size of a TEdit document

is only limited by the amount of memory space in your virtual memory.

● Line-selection will not allow the user to select protected characters

There was a bug in TEdit where protected characters (with PROTECTED set ON in their character looks) could still be selected by line-selecting. This has been fixed.

● "Font not found" error now displays face information

If a font file cannot be found while formatting in TEdit, the prompt window now displays not only the size and family information of the font, but also the face information.

● Several fields in the TEXTOBJ and SELECTION datatypes changed to contain lists

The L1 and LN fields of a SELECTION, and the \WINDOW, LINES, and CARET fields of a TEXTOBJ may contain lists, as well as single objects

● The function TEXTSTREAM now accepts a TEdit process as its argument, and returns the corresponding textstream

● Skipping to next >>xx<< field sets TEdit so typein has same character looks as replaced text

● TEDIT.OBJECT.CHANGED marks the document dirty, so that TEDIT.FILECHANGEDP returns the right result

● Leading and trailing spaces are removed from file names, if typed accidentally

● TEDIT.FORMATTEDFILEP now correctly finds most demanding of CHARLOOKS, PARALOOKS, and IMAGEOBJ within a document

● Page formatting is cleared when GETing an unformatted document in a page-formatted TEdit window

● TEDIT.DELETE will delete the selection passed to it, instead of the current selection

● TEDIT.GETSEL now returns a COPY of the current selection, so the user can change it without affecting TEdit

● TEDIT.INCLUDE will copy paragraph formatting from the "included" file

● Calling TEDIT.SETFUNCTION with a function of NIL resets the character's syntax class to NONE

● Bravo files with W or w in the final trailer are properly converted to TEdit files

● TEDIT.OBJECT.CHANGED will accept either a text stream or a TEXTOBJ as its STREAM argument

Break Package

- The break command ORIGINAL now works correctly

The command ORIGINAL in the break package will now work. Previously, using the ORIGINAL command could cause an "ill formed iterative statement" error from DWIM.

CHAT

- Chat service to NS hosts supported

Chat can now be used to communicate with NS hosts using the normal Chat interface. The only visible difference from communicating with PUP hosts is that the NS Chat protocol differentiates among a number of virtual terminal services. Calling CHAT on an NS host will pop up a menu to allow you to choose the terminal service you want to use:

Any

Remote System Administration

Remote System Executive

Interactive Terminal Service

The "Any" option in the menu will eventually allow using any terminal service is available on the specified host, but no hosts currently support it.

The "Remote System Administration" service lets you log onto print servers and clearinghouse servers, and issue appropriate commands.

The "Remote System Executive" service is currently only supported by Tajo/Mesa workstations with appropriate software loaded.

The "Interactive Terminal Service" is the TTY-based interface to NS mail.

If you select an invalid service type, you'll get an "ERROR ServiceNotFound" message in the promptwindow. In a future release, Interlisp will be able to discover which services a particular host supports.

File Package

● MOVETOFILE, DELFROMFILE, etc. now work with file package commands like BITMAPS

Previously, the functions for manipulating filecoms only accepted file package "type." It was not possible to use these functions with file package "commands" which didn't have a corresponding type, such as BITMAPS. Functions such as MOVETOFILE, DELFROMFILE, etc. now will accept file package commands as legitimate "types."

● ADDTOFILE much faster

ADDTOFILE (which adds the commands for a given object to the file package commands for a specified file) was unreasonably slow, because it was calling UPDATEFILES unnecessarily. ADDTOFILE has been changed not to call UPDATEFILES, so it is an extremely fast operation.

● New copyright option: if COPYRIGHTFLG = DEFAULT, default copyright used without waiting

If the value of the variable COPYRIGHTFLG is the atom DEFAULT, the value of DEFAULTCOPYRIGHTOWNER is used for putting copyright information in files that don't have any other copyright. The prompt "Copyright owner for file xx:" will still be printed, but the default will be filled in immediately.

● Bug fixed: no copyright will be printed if COPYRIGHTFLG = NIL

In the Harmony release, there was a bug such that if COPYRIGHTFLG were NIL, the copyright message "(* Copyright (c) by NIL. All rights reserved.)" would be put in the file. This has been fixed.

● SHOWDEF accepts a stream as its FILE argument; will not close the file when finished

Compiler

● Compiler will discard argument names if DASSEM.SAVELOCALVARS returns NIL

Sometimes it is desirable to have the compiler discard the names of the atoms used as local variables within a function. This prevents free variable lookup from accessing these variables. It also means that the atoms are not read in when the function is read; this can be important for an application which uses a lot of atoms (there is a fixed maximum number of atoms). A LOCALVARS declaration within a function can be used to discard the names of local variables internal to the function, but there has never been a way of discarding the names of the arguments to the function. This can now be done, by redefining the function DASSEM.SAVELOCALVARS.

(DASSEM.SAVELOCALVARS ⟨function-name⟩)

This function is called by the compiler to determine whether local-var argument information for ⟨function-name⟩ should be written on the compiled file for ⟨function-name⟩. If it returns NIL, the local-var argument information is NOT saved, and the function is stored with arguments U, V, W, etc instead of the originals.

Initially, DASSEM.SAVELOCALVARS is defined to return T. (MOVD 'NILL 'DASSEM.SAVELOCALVARS) causes the compiler to retain *no* local variable or argument names. Alternatively, DASSEM.SAVELOCALVARS could be redefined as a more complex predicate, to allow finer discrimination.

DWIM & CLISP

● *Incompatible Change:* (CLISPDEC 'MIXED) is default CLISP declaration

In past releases of Interlisp, and in the Harmony release, the default clisp declaration is FIXED, which means that all CLISP constructs are translated using integer arithmetic, unless the user explicitly changes the declaration. Therefore, (A + B) translates into (IPLUS A B), and (for X from A to B do ...) is translated using integer arithmetic to increment X and compare it to B.

In Interlisp-D, mixed (generic) arithmetic is as fast as integer arithmetic, so we are trying to convert the system to use generic arithmetic as much as possible.

Therefore, the default clisp declaration has been changed to MIXED, so generic arithmetic functions will be used when translating clisp constructs. (A + B) translates into (PLUS A B), and (for X from A to B do ...) is translated using PLUS and

GREATERP. Of course, the user can change this declaration using CLISPDEC.

This change shouldn't effect any programs: the only conceivable problems could be in constructs like (for X from A to B do ...) where the programmer *counted* on the floating-point values A and B being converted to fixed point before the loop.

Storage & Data Types

● Virtual memory expanded to 32 megabytes

The maximum virtual memory space in Interlisp has been expanded from 8 megabytes to 32 megabytes. The increase in available space for user applications is even more striking, since a significant percentage of the original 8MB was used by the system code.

The available virtual memory space that can be used on a given machine is determined by the size of the virtual memory "backing file" on the local hard disk. Users who partitioned their 1108 local disks for previous releases, where the maximum size of the virtual memory file was 16200 pages, will have to re-partition their disks to create a virtual memory file greater than 8MB.

When the virtual memory expands to the point where the backing file is almost full, a break will occur with the warning message "Your virtual memory backing file is almost full. Save your work & reload asap." When this happens, it is strongly suggested that you save any important work and reload the sysout. If you continue working past this point, the system will start slowing down considerably, and it will eventually fall into Raid with MP 9308.

● Some early 1108s cannot support >8MB of virtual memory; new function 32MBADDRESSABLE

Some early versions of the 1108 hardware will not support more than 8MB of virtual address space. Intermezzo will still run, but it will crash when the virtual address space grows beyond 8MB. The function (32MBADDRESSABLE) returns T if your hardware supports the full 32MB address space.

● Maximum number of litatoms doubled to 64K

In Interlisp, there is a fixed limit on the number of different litatoms that can be created in a given sysout. Previously, the limit was 32768 atoms. This has now been doubled to 65536 atoms. It is still possible to run out of atoms, so users building applications that generate a lot of atoms might still want to consider using another data representation, such as strings.

● Storage management changed to improve system performance; "hunking" used for small arrays

The storage management code within Interlisp has been extensively revised, to incorporate a number of changes to improve system performance.

First, there are no longer fixed areas in the virtual memory assigned to allocating fixed and variable-length data. Previously, one could run out of variable-length space (array space) even if there was still room in the fixed-length space (main data space, MDS). Now, both spaces expand as necessary until all of the virtual memory is filled.

Second, the scheme called "hunking" has been introduced for allocating small variable-length objects. In this scheme, small arrays are allocated and managed like fixed-length data objects. Specifically, all the arrays of size 5, 6, 7, etc. are managed like fixed-length objects, instead of being allocated from the main array space. This means that the main array space is not fragmented by small arrays. A large percentage of the arrays used by the system are small, so this improves the perfoamcne of the large array allocation.

These changes are invisible to the casual user, except that system performance is improved: more work can be done before storage problems appear, and the system speed doesn't degrade over time as quickly. Of course, there always limits; the most inconvenient ones have just been pushed back.

One visible change: the STORAGE printout has been changed slightly. The "Data Spaces Summary" now looks like:

	Allocated Pages	Remaining Pages
Datatypes (incl. LISPT etc.)	2192	\
ArrayBLocks (variable)	3644	-- 52644
ArrayBlocks (chunked)	2056	/
Litatoms	744	1304
Litatom Pnames (from bootstrap)	131	0

This shows that both variable and fixed-length data types are allocated out of the same virtual memory space, and that small array blocks are treated separately (hunked) from large array blocks.

Arithmetic

- *Incompatible Change:* (ZEROP X) = (EQP X 0)

In the *Interlisp Reference Manual*, (ZEROP X) is defined to be equivalent to (EQ X 0). This has been changed so that (ZEROP X) is equivalent to (EQP X 0). Users who depend on (ZEROP 0.0) returning NIL should change their code to use (EQ X 0).

- Arithmetic functions accept negative zero; fixes obscure bugs

The IEEE standard number format defines the number "negative zero." On rare occasions, Interlisp code can generate this number. Previously, the arithmetic code would cause strange errors when dealing with negative zero. This has been fixed so that the arithemetic code will treat negative zero the same as positive zero.

● 1132 will cause an error on integer overflow if (OVERFLOW T) is set

Previously, even if (OVERFLOW T) was set, integer overflow would not cause an error on the 1132. Instead, it would act as if (OVERFLOW 0) was set (it would return the result modulo 2↑32.

● Accuracy of ANTILOG improved

The accuracy of ANTILOG has been improved. This also improves functions such as EXPT which call ANTILOG.

Processes

● Mouse process is restarted before password window created during system restart

Previously, in certain rare circumstances it was possible for a window to pop up to prompt for a password (in order to re-open files) before the mouse process was restarted. Without the mouse process alive, it wasn't possible to "button" the window to type in the password.

1108 Microcode

● Intermezzo initial microcode is incompatible with Harmony; new MP code 9099

As part of the process of loading and starting an Interlisp sysout, special "initial microcode" stored on the 1108 local disk is used. This microcode is stored on the local disk when the "SystemTools" volume is initialized using the *Installation Utility* floppy.

The initial microcode for the Harmony release and the Intermezzo release are incompatible. This means that you cannot start a Harmony sysout on a machine with Intermezzo initial microcode, and vice versa. In order to help users recognize this situation, Intermezzo sysouts will halt with MP code 9099 if the initial microcode is incompatible.

Note that this doesn't work when loading a Harmony sysout on an 1108 with Intermezzo initial microcode (the new MP code is only defined in Intermezzo). A common symptom of using the wrong

initial microcode is a flashing MP 0201, although other breaks are possible.

● New Maintenance panal codes: 9327, 9328, 9329

9327, 9328: Bad array block. The array allocator found a bad array block in its free list. Generally means some unsafe code trashed one or more locations in array space.

9329: The garbage collector attempted to reclaim an array block, but the block's header was trashed. You can continue from this error with ↑N from TeleRaid, but it is symptomatic of array trashing, and you should save your state as soon as possible and restart in a good sysout.

Library Packages

● FLOATARRAY: MAPELT2 bug with array plus and difference has been fixed

Previously, on the 1108 CPE, the function MAPELT2 returned the wrong results when applying FTIMES and FDIFFERENCE to two arrays. This was due to a microcode bug that has been fixed.

● GRAPHER: Control over line drawing; grapher image objects

Hooks have been added to allow the user to specify properties of the link between any two nodes in a graph. This allows graph links to drawn with different widths, or with dashing.

Grapher image objects are supported. They can be constructed programmatically, or by copy-selecting a graph. Grapher image objects can be inserted into TEdit documents.

Incompatibility: SHOWGRAPH with ALLOWEDIT = T and EDITGRAPH used to move nodes when a shift key was held down. This conflicts with the common system idiom of using shift-selection as a copy indicator. Thus, this edit feature has been changed so that the CTRL key is the move specifier.

For more information on these changes, see the GRAPHER library package documentation.

● SPACEWINDOW: Display reorganized for 32MB Interlisp

The SPACEWINDOW display has been reorganized to display information useful with the new memory management organization. Now, the display contains four lines: 8MBData, Data, Atoms, and Vmem, each of which contains a bar showing the percentage of storage allocated. "Atoms" displays the percentage of atoms that have been allocated. "Vmem" displays the percentage of the virtual memory backup file that has been used. "Data" displays the percentage of virtual memory space that has been allocated to either fixed or variable length data. "8MBData" displays the the amount of virtual memory space that has been allocated, relative to 8MB.

● VTCHAT: Package allows Chat to emulate VT-100 terminal

The Lispusers package VTCHAT provides a VT100-emulating version of Chat. It loads the subfile VT100KP, which contains routines for emulating the VT100's right keypad with a mouse-sensitive window.

Miscellaneous

- PROMPTFORWORD will not timeout when URGENCY.OPTION = TTY

PROMPTFORWORD is called with URGENCY.OPTION = TTY when a process wants to grab the tty immediately for a prompt. TEdit uses this option to prompt for a file name when retieving or storing a file. In the Harmony release, PROMPTFORWORD with URGENCY.OPTION = TTY would timeout after about 15 seconds, which is clearly wrong. This has been changed so that PROMPTFORWORD will wait forever if URGENCY.OPTION = TTY.

- LET, LET*, PROG*, LIST* moved from CMLSPECIALFORMS into standard Interlisp sysout

The functions/macros LET, LET*, PROG*, and LIST* have been removed from the library package CMLSPECIALFORMS, and included in the standard Interlisp system.

LET, LET*, and PROG* have only macro definitions. LET is essentially a PROG that can't contain GO's or RETURN's, and whose last form is the returned value. LET* and PROG* differ from LET and PROG only in that the binding of the bound variables is done "sequentially." Thus

```
(LET* ((A (LIST 5))
       (B (LIST A A)))
      (EQ A (CADR B)))
```

would evaluate to true; whereas the same form with LET might even find A an unbound variable when evaluating (LIST A A).

LIST*, which has both a macro definition and a function definition, is like an iterated CONS;

```
(LIST* A B C) = > (CONS A (CONS B C))
```

Note that LIST could be defined in terms of LIST*:

```
(LIST A B ... Y Z) = = (LIST* A B ... Y Z NIL)
```

Implementing Device-Independent Graphics Through Image Streams

The Interlisp-D system does all image creation through a set of functions and data structures for device-independent graphics, known popularly as DIG. DIG is achieved through the use of a special flavor of stream, known as an *image stream*.

An image stream, by convention, is any stream that has its IMAGEOPS field (described in detail below) set to a vector of meaningful graphical operations. Using image streams, you can write programs that draw and print on an output stream without regard to the underlying device, be it a window, a disk, or a Dover, 8044, or Diablo printer. For example, the following devices are supported by image streams: windows, Press streams, Interpress streams, and Iris streams.

Image Stream Structure

As indicated above, image streams use a field that no other stream uses: IMAGEOPS. IMAGEOPS is an instance of the IMAGEOPS data type and contains a vector of the stream's graphical methods. The methods contained in the IMAGEOPS can make arbitrary use of the stream's IMAGEDATA field, which is provided for their use, and may contain any data needed.

The IMAGEOPS data type has the following fields:

IMAGETYPE is the name of an image type. Monochrome display streams have an IMAGETYPE of DISPLAY; color display streams are identified as (COLOR DISPLAY). The IMAGETYPE is informational and can be set to anything you choose.

IMFONTCREATE is the device name to pass to FONTCREATE when fonts are created for the stream.

The following fields are all stream methods, and are presented with their arguments in the manner of a function definition. With the exception of IMCLOSEFN, each method has a corresponding function that consists of the method's name with the "IM" prefix removed. All coordinates that refer to points in a display device's space are measured in the device's units. (The IMSCALE method provides access to a device's scale.)

(IMCLOSEFN *Stream*)   [Stream method]

is used before a stream is CLOSEFed, e.g., to flush buffers, write header or trailer information, etc.

(IMDRAWLINE *Stream X1 Y1 X2 Y2*
          Width Operation Color)   [Stream method]

draws a line of width *Width* from (*X1, Y1*) to (*X2, Y2*). (Dashing is currently handled at a higher level, and thus is not an argument.)

(IMDRAWCURVE *Stream Knots Closed Brush Dashing*) [Stream method]

draws a curve through *Knots*.

(IMDRAWCIRCLE *Stream CenterX CenterY Radius Brush Dashing*) [Stream method]

draws a circle of radius *Radius* around (*CenterX*, *CenterY*).

(IMDRAWELLIPSE *Stream CenterX CenterY SemiMinorRadius SemiMajorRadius Orientation Brush Dashing*) [Stream method]

draws an ellipse around (*CenterX*, *CenterY*).

(IMFILLCIRCLE *Stream CenterX CenterY Radius Texture*) [Stream method]

draws a circle filled with texture *Texture* around (*CenterX, CenterY*).

(IMBLTSHADE *Texture Stream DestinationLeft DestinationBottom Width Height Operation ClippingRegion*) [Stream method]

is the texture-source case of BITBLT. *DestinationLeft, DestinationBottom, Width, Height,* and *ClippingRegion* are measured in *Stream*'s units. This method is invoked by the functions BITBLT and BLTSHADE.

(IMBITBLT *SourceBitMap SourceLeft SourceBottom Stream DestinationLeft DestinationBottom Width Height SourceType Operation Texture ClippingRegion ClippedSource Left ClippedSourceBottom Scale*) [Stream method]

contains the bit-map-source cases of BITBLT. *SourceLeft, SourceBottom, ClippedSourceLeft, ClippedSourceBottom, Width,* and *Height* are measured in pixels; *DestinationLeft, DestinationBottom,* and *ClippingRegion* are in the units of the destination stream.

(IMSCALEDBITBLT *SourceBitMap SourceLeft SourceBottom Stream DestinationLeft DestinationBottom Width Height SourceType Operation Texture ClippingRegion ClippedSourceLeft ClippedSourceBottom Scale*) [Stream method]

is a scaled version of IMBITBLT. Each pixel in *SourceBitMap* is replicated *Scale* times in the *X* and *Y* directions; currently, *Scale* must be an integer.

(IMMOVETO *Stream X Y*) [Stream method]

moves to *(X,Y)*. This method is invoked by the functions MOVETO and RELMOVETO. If IMMOVETO is not supplied, a default method composed of calls to the IMXPOSITION and IMYPOSITION methods is used.

(IMSTRINGWIDTH *Stream Str RdtBl*) [Stream method]

returns the width of string *Str* in *Stream*'s units, using *Stream*'s current font. If IMSTRINGWIDTH is not supplied, it defaults to calling \STRINGWIDTH.GENERIC.

(IMCHARWIDTH *Stream CharCode*) [Stream method]

returns the width of character *CharCode* in *Stream*'s units. If IMCHARWIDTH is not supplied, it defaults to calling \STRINGWIDTH.GENERIC.

The following methods all have corresponding DSPxx functions (e.g., IMYPOSITION corresponds to DSPYPOSITION) that invoke them. They also have the property of returning their previous value; when called with NIL they return the old value without changing it.

(IMXPOSITION *Stream XPosition*) [Stream method]

sets a new *X*-position on *Stream*.

(IMYPOSITION *Stream YPosition*) [Stream method]

sets a new *Y*-position on *Stream*.

(IMFONT *Stream Font*) [Stream method]

sets *Stream*'s font to be *Font*.

(IMLEFTMARGIN *Stream LeftMargin*) [Stream method]

sets *Stream*'s left margin to be *LeftMargin*. The left margin is defined as the *X*-position set after the new line.

(IMRIGHTMARGIN *Stream RightMargin*) [Stream method]

sets *Stream*'s right margin to be *RightMargin*. The right margin is defined as the maximum *X*-position at which characters are printed; printing beyond it causes a new line.

(IMLINEFEED *Stream Delta*) [Stream method]

sets *Stream*'s line feed distance (distance to move vertically after a new line) to be *Delta*.

(IMNEWPAGE *Stream*) [Stream method]

causes a new page to be started; the position is set to (*DspLeftMargin*, *DspTopMargin*). If not supplied, defaults to (\OUTCHAR *Stream* (CHARCODE ↑L)).

(IMSCALE *Stream Scale*) [Stream method]

returns the number of device points per screen point (a screen point being ~1/72 inch). *Scale* is ignored.

(IMTOPMARGIN *Stream YPosition*) [Stream method]

Sets *Stream*'s top margin (the *Y*-position of the tops of characters that is set after a new page) to be *YPosition*.

(IMBOTTOMMARGIN *Stream YPosition*) [Stream method]

sets *Stream*'s bottom margin (the *Y*-position beyond which any printing causes a new page) to be *YPosition*.

(IMSPACEFACTOR *Stream Factor*)   [Stream method]

sets the amount by which to multiply the natural width of all following space characters on *Stream;* is used for justification of text. The default value is one. For example, if the natural width of a space in *Stream*'s current font is 12 units, and the space factor is set to two, spaces appear 24 units wide. The values returned by STRINGWIDTH and CHARWIDTH are also affected.

(IMOPERATION *Stream Operation*)   [Stream method]

sets the default BITBLT *Operation* argument. See the DSPOPERATION and BITBLT documentation for more information.

(IMBACKCOLOR *Stream Color*)   [Stream method]

sets the background color of *Stream*.

(IMCOLOR *Stream Color*)   [Stream method]

sets the default color of *Stream*.

In addition to the IMAGEOPS-borne methods described above, there are two other important methods, which are contained in the stream itself.

STRMBOUTFN   [Stream method]

is a function called by BOUT. You can install a STRMBOUTFN in a stream *Stream* using the form (replace (STREAM STRMBOUTFN) of STREAM with (FUNCTION *MYBOUTFN*)).

OUTCHARFN   [Stream method]

is the function that is called to output a single byte. This is like STRMBOUTFN, except for being one level higher: it is intended for text output. Hence, this function should convert (CHARCODE EOL) into the stream's actual end-of-line sequence and should adjust the stream's CHARPOSITION appropriately before invoking the stream's STRMBOUTFN (by calling BOUT) to actually put the character. Defaults to \FILEOUTCHARFN, which is definitely *not* what you want. OUTCHARFNs are installed using a form like (replace (STREAM OUTCHARFN) of STREAM with (FUNCTION *MYOUTCHARFN*)).

IMAGEDATA   [Record field]

is used to hold data pertaining to this type of image stream; the content is completely up to you. For Interpress devices, this is an instance of the data type INTERPRESSDATA; for Press, PRESSDATA; for the display, \DISPLAYDATA.

Creating Image Streams

(OPENIMAGESTREAM *File ImageType Options*)   [Function]

opens and returns an output stream of type *ImageType* (PRESS, INTERPRESS, DISPLAY, or other types) on a destination specified by *File*. *File* can name a file either on a normal storage device or on a printer device. In the latter case, the file is sent to

the printer when the stream is closed. Because of the way defaulted arguments are interpreted, OPENIMAGESTREAM provides a convenient and standard interface for interpreting user output-destination specifications.

If *ImageType* is NIL, the image type is inferred from the extension field of *File* and the EXTENSIONS properties in the list PRINTFILETYPES. Thus, a PRESS extension denotes a Press-format stream, while IP, IPR, and INTERPRESS indicate Interpress format. If *File* has no extension but is a file on the printer device {LPT}, then *ImageType* is the type that the indicated printer can print. If *File* has no extension but is not on the printer device, then *ImageType* defaults to the type accepted by the first printer on DEFAULTPRINTINGHOST.

File = NIL is equivalent to *FILE* = {LPT}. Names for printer files are of the form {LPT}PRINTERNAME.TYPE, where PRINTERNAME, TYPE, or both may be omitted. PRINTERNAME is the name of the particular printer to which the file is transmitted on closing; it defaults to the first printer on DEFAULTPRINTINGHOST that can print *ImageType* files. As just described, the TYPE extension supplies the *ImageType* when it is defaulted. OPENIMAGESTREAM generates an error if the specified printer does not accept the kind of file specified by *ImageType*.

Examples assuming IP: is an Interpress printer, P is a Press printer, and DEFAULTPRINTINGHOST is (IP: P):

(OPENIMAGESTREAM) returns an Interpress image stream on printer IP:

(OPENIMAGESTREAM NIL 'PRESS) returns a Press stream on P

(OPENIMAGESTREAM '{LPT}.INTERPRESS) returns an Interpress stream on IP:

(OPENIMAGESTREAM '{CORE}FOO.PRESS) returns a Press stream on the file {CORE}FOO.PRESS

For completeness and consistency, if *ImageType* is inferred to be DISPLAY, then you are prompted for a window to open. The file name in this case is used as the title of the window.

Options is a list in property list format that may be used to specify certain attributes of the image stream; not all attributes are meaningful or interpreted by all types of streams. Among the properties are:

REGION, whose value is the region on the page (in stream scale units, 0,0 being the lower-left corner of the page) that text will fill up. It establishes the initial values for DSPLEFTMARGIN, DSPRIGHTMARGIN, DSPBOTTOMMARGIN, and DSPTOPMARGIN.

FONTS, whose value is a list of fonts that are expected to be used in the stream. Some streams (e.g., Interpress) are more efficient if the expected fonts are called out in advance, but this is not necessary. The first font in this list is the initial font of the stream; otherwise the DEFAULTFONT for that image type is used.

HEADING is the heading to be placed automatically on each page; NIL means no heading.

IMAGESTREAMTYPES   [Association list]

describes how to create a stream for a given image type. Contains OPENSTREAM, FONTCREATE, and FONTSAVAILABLE methods. The main a-list is indexed by the image-stream type name (e.g., DISPLAY, PRESS, or INTERPRESS) to get another a-list that associates device-dependent functions with generic operation names.

The format of a single a-list entry is:

```
(IMAGETYPE
    (OPENSTREAM FunctionToOpenTheStream)
    (FONTCREATE FunctionToCreateAFontDescriptor)
    (FONTSAVAILABLE
FunctionToReturnAvailableFonts))
```

For example, for Interpress, the a-list entry is:

```
(INTERPRESS
    (OPENSTREAM OPENIPSTREAM)
    (FONTCREATE \CREATEINTERPRESSFONT)
    (FONTSAVAILABLE \SEARCHINTERPRESSFONTS))
```

The OPENSTREAM function is called with the arguments:

(OPENSTREAMFN *File Options*)

File is the file name as it was passed to OPENIMAGESTREAM, and *Options* is the OPTIONS property list passed to OPENIMAGESTREAM. The result must be a stream of the appropriate image type.

The FONTCREATE function is called with the arguments:

(FONTCREATEFN *Family Size Face Rotation Device*)

Family is the family name for the font, e.g., MODERN. *Size* is the body size of the font, in printer's points. *Face* is a three-element list describing the weight, slope, and expansion of the face desired, e.g., (MEDIUM ITALIC EXPANDED). *Rotation* is how much the font is to be rotated from the normal orientation, in minutes of arc. For example, to print a landscape page, fonts have the rotation 5,400 (90 degrees). The function's result must be a FONTDESCRIPTOR with the fields filled in appropriately.

The FONTSAVAILABLE function is called with the arguments:

(FONTSAVAILABLEFN *Family Size Face Rotation Device*)

This function returns a list of all fonts agreeing with the *Family*, *Size*, *Face*, and *Rotation* arguments; any of them may be wild-carded (i.e., equal to '*, which means "any"). Each element of the list should be a quintuple of the form (*Family Size Face Rotation Device*).

Where the function looks is an implementation decision: the FONTSAVAILABLEFN for the display device looks at DISPLAYFONTDIRECTORIES, the Interpress code looks on INTERPRESSFONTDIRECTORIES, and implementors of new devices should feel free to introduce new search path variables.

Image Stream Predicates

(IMAGESTREAMP *X ImageType*)   [Function]

returns *X* (possibly coerced to a stream) if it is an output image stream of type *ImageType* (or of any type if *ImageType* is NIL), otherwise NIL.

(IMAGESTREAMTYPE *Stream*)   [Function]

returns the image type of *Stream*.

(IMAGESTREAMTYPEP *Stream Type*)   [Function]

returns T if *Stream* is an image stream of type *Type*.

Creating Your Own Flavor of Image Stream

In describing a task as complex as building a new flavor of image stream, no document can contain all of the answers, tricks, and shortcuts. There is no substitute for studying a working implementation in doing your own. Therefore, we recommend you look at the FX80STREAM package as an example of how to create a new imaging device. FX80STREAM is a DIG interface for the Epson FX-80 printer—a device simple enough to drive that its details do not obscure the fundamentals of how its image stream works.