Venue

MEDLEY
DOS USER'S GUIDE

Venue   *Medley for DOS® User's Guide*

Release 2.01
March, 1993

Address comments to:
Venue
User Documentation
1549 Industrial Road
San Carlos, CA 94070
415-508-9672

MEDLEY FOR DOS® USER'S GUIDE

Release 2.01

March, 1993

The information in this document is subject to change without notice
and should not be construed as a commitment by Venue. While every
effort has been made to ensure the accuracy of this document, Venue
assumes no responsibility for any errors that may appear.

Text was written and produced with Venue text formatting tools; a
PostScript printer was used to produce masters. The typeface is
Palatino.

[This page intentionally left blank]

This *Guide* describes Medley release 2.01 for DOS: The release contents, instructions for installing the release, and information on using it.

Audience

The *Medley For DOS User's Guide* is intended for users who want to use Medley on IBM PCs and compatibles running under DOS. The *Guide* assumes that you are already familiar with DOS.

Chapter 1 of this manual gives an overview of the product and its internal architecture, and is of interest to all users.

Chapter 2, Software Installation, and Chapter 3, Getting Started, guide you through the process of installing and configuring Medley for your PC. Experienced Lisp users may want to configure the software; this procedure is described in Chapter 4.

Chapter 1 and Chapters 3 through 6 describe the operation of the system after it has been installed as well as those functions and operations that are specific to PCs.

Using This Manual

Chapter 1, Introduction, describes the hardware, input/output devices, and software needed to run Medley on a PC; describes Medley and how it works with TSR applications; lists the system components; introduces pertinent DOS conventions used throughout the *Guide;* explains Medley's compatibility; and lists the contents of the release.

Chapter 2, Software Installation, contains installation and software configuration procedures.

Chapter 3, Getting Started, shows the keyboard configuration and has instructions for getting started in Lisp on a PC.

Chapter 4, Using Medley on a PC, describes how specific Medley features work on a PC.

Chapter 5, Medley File Systems, discusses the file conventions that need to be followed when running Medley on a PC. Lisp file attributes and variables are also discussed.

Chapter 6, Error Recovery, describes the diagnostic error recovery program URAID. This chapter explains how to recover from fatal error conditions and lists specific Lisp errors that may be encountered when running Medley on a PC.

Appendix Font Directories, contains a listing of the font directories.

The **Glossary** provides definitions of terms used in this *Guide.*

Medley is a Venue product which was built on the Xerox Lisp environment. It provides an integrated programming environment consisting of Interlisp-D and Common Lisp, a windowing system, and a set of programs and utilities. Users not already familiar with the Xerox Lisp environment should try to become somewhat familiar with it before attempting serious development work.

Supporting Documentation

The following reference documents are useful to have on hand during the installation process and when working in Medley on a PC.

Venue Documentation

In addition to this *Guide,* the following documents describe the Medley system and are included with the software kit:

- *Medley Reference Manual, Volumes 1-3*
- *Lisp Documentation Tools*
- *Lisp Library Modules*
- *Medley for DOS User's Guide*

In addition, when you register your copy of Medley for DOS, you will receive software and documentation for ROOMS and LispUsers.

You might find it helpful to have a copy of Guy Steele's *Common Lisp: the Language* on hand for help with Common Lisp functions.

Medley is an integrated programming environment, with support for the Interlisp and Common Lisp languages, an integrated windowing system, and a large collection of utilities and programs. It offers a mature and rich programming and development environment, as well as access to a large number of applications written for Interlisp, Common Lisp, and LOOPS.

What Medley Requires

Hardware

Medley runs on IBM PCs and compatibles equipped with 80386 or 80486 processors. The other hardware components necessary to run Medley on your PC are: A mouse with a Microsoft-compatible driver, a 101-key keyboard, and either a VGA or SVGA card and monitor.

Reasonable interactive performance can be expected with 8 megabytes (MB) or more of RAM. A total of 23 MB of disk space is necessary to hold Medley; in addition, you will need disk space for Medley's swapping file, up to 32 MB.

Naturally, larger applications will benefit from more memory. Medley's maximum working set is approximately 40 MB.

Input/Output Devices

Medley gives you access to a PC's input/output devices, such as display, keyboard, mouse, and file systems.

Bitmap Display

Medley supports VGA monitors with 640 x 480 resolution and SVGA monitors with up to 1024 x 768 resolution.

Software Requirements

Medley runs on PCs running DOS 4.0 or higher.

You must have a Microsoft-compatible mouse driver installed and active for Medley to run.

For Medley to support an SVGA at better than 640X480 resolution, you must be running a VESA driver; generally, one comes with your SVGA board.

Medley and TSR Applications

Medley does not release control of the keyboard to other applications. This means that applications which rely on "hot keys" to invoke them will not work with Medley. Neither will applications which watch the keyboard, RoseSoft's PROKEY, for example.

System Components

Functionally, Medley consists of the following components:

emulator A DOS program that executes compiled Lisp code, using a special-purpose byte-code instruction set. It also provides access to your PC's I/O (display, keyboard, file system).

1

sysout   A "canned" virtual memory image of a running Lisp. Saving a sysout saves the state of Lisp; restarting one picks up where you left off. Medley comes with a basic sysout.

library Files of compiled Lisp code and data structures.

fonts   Data describing the "looks" of printed characters used by Medley's graphics, window, and hardcopy subsystems. Font directories are in two groups: display fonts and PostScript fonts.

Medley Device-Naming Conventions

Medley lets you interact with the DOS file system (including file systems mounted using PC-NFS or other network file systems) using the device {DSK}. See Chapter 5 for more detail.

Notation Conventions

Command Line Options

In example command lines, anything shown in brackets is optional; the -vga below is optional:

> C> **medley [-vga]**

If you may choose only one of several options, they are shown separated by vertical bars; in the example below you may optionally type one of -vga or -svga:

> C> **command [-vga | -svga]**

Fonts

Bold text in Courier indicates text you should type in exactly as printed.

Regular Courier text indicates what the system prints on your monitor screen. Lisp functions and variables and programs are also shown in Courier.

Text in italics indicates variables or parameters that you should replace with the appropriate word or string.

Packages

In this manual, most symbols are in the INTERLISP package (IL:). This is the default when no package qualifier is shown.

Compatibility

Medley for DOS is designed for maximum compatibility with Unix implementations of Medley. However, when moving applications to a PC note the differences in end-of-line conventions and techniques for moving files.

Sysout Compatibility

Sysouts of the same version are compatible with all machine types, with one exception. A sysout generated on a PC or Unix workstation cannot be used on a Xerox workstation.

Note: Medley 2.0 sysouts cannot handle European keyboard conventions. This capabillity has been added in Medley 2.01. If you try to run a 2.0 sysout on a 2.01 emulator the keyboard will not work properly. Use the command line option -noeurokbd to run 2.0 sysouts on a 2.01 emulator.

2

Compiled-File Compatibility

Code compiled in earlier Medley versions cannot be loaded into Medley 2.0 sysouts, nor can code compiled in Medley 2.0 be loaded onto earlier sysouts.

However, code compiled for Medley on a PC *can* be loaded into Medley on any Unix workstation, and vice versa.

End-of-Line Convention

Some care must be taken in moving files to and from non-DOS workstations, since the default end-of-line convention in UNIX is to terminate lines with the line feed (LF) character, while, traditionally, Medley has terminated lines with the carriage return (CR) character. In addition, the convention in DOS is to use the carriage return/line feed (CR/LF) combination. In particular, if you use some other file transfer mechanism, such as FTP or Kermit, be careful to transfer .TED, .DFA, and .LCO files in binary mode.

In Medley on a PC, the default end-of-line convention for all text files is the carriage return/line feed (CR/LF) combination. The default end-of-line convention for all binary files is carriage return (CR); this is because CR (ASCII 13) is used internally in the system.

Release Contents

The release distribution contains the following documentation and software.

Documentation

The Medley documentation kit for DOS contains:

- *Medley For DOS User's Guide*
- *An Introduction to Medley*
- *Medley Reference Manual, Volumes 1-3*
- *Xerox Common Lisp Implementation Notes*, Lyric Release
- *Lisp Library Modules*, Medley Release
- *Lisp Documentation Tools*, Lyric Release

You may also purchase:

- *LispUsers' Modules*, Medley Release

Software

The software release is available on 3½" floppies and consists of the following files: .\medley.exe, .\emul.exe, .\library\, .\lisp.sys, .\fonts\display, and .\fonts\ps.

Rooms and LispUsers Modules

Rooms is a software supplement to Medley. *Rooms* is a window organizer that helps you keep track of many windows.

LispUsers' Modules is software written by our users. The support for these modules comes from each module's author; Venue has no commitment to support LispUsers' modules.

You will recieve these on a separate distribution when you send your registration card to Venue.

[This page intentionally left blank]

To install Medley on a PC, you need the following:

- Release diskettes
- Medley documentation for DOS.

Getting Ready to Install Medley

Medley includes a batch file for automatic installation.

Before installing Medley, you should gather some facts about the hardware and network environment on which you will be using Medley. The following checklist will help you.

- Are you installing Medley for a single user, or will several users be sharing this copy?

  If it is for shared use, you will want to install Medley on a public directory on a shared server. For that, we recommend using \medley as the directory name.

- Do you have enough free disk space?

  You need to select a drive with enough disk space to install the software, about 23 MB. Use the DOS commands DIR or CHKDSK to verify free disk space.

- Have you selected an installation directory?

  The directory must be on a drive with enough space. Regardless of the drive, we recommend \medley as the directory name.

- Where will Medley's swapping file be?

  The space you'll need depends on how big Medley's virtual memory will be, in the range 8 to 32 MB. By default, Medley uses drive C.

Ensuring Adequate Disk Space

Medley requires 23 MB of disk space on top of your normal disk space requirements.

Medley's Swapping File

To provide virtual memory (VM), Medley uses a swapping file; its size depends on how big you let Medley's VM get (normally 32 MB). The file gets created when Medley starts, and is deleted when you leave Medley.

To compute the file's size, take the maximum VM size, minus the size of your PC's memory. For a 32 MB virtual memory and an 8 MB PC, the swapping file will be a bit over 24MB.

If you don't specify otherwise, Medley creates the swapping file on drive C. If there isn't enough space there, you'll want to put it on a drive where there is room.

Installing Medley for Shared Use

If several people will be be using Medley on different machines, it probably makes sense to install one copy and have people share it.

You will need to find one machine—probably your main file server—with enough disk space. You will also want to have the directory look the same to every user. We recommend calling the installation directory \medley.

Installing Medley Software

The whole installation takes about half an hour on a 486/33. Of that, about ten minutes goes to load the floppies and some twenty minutes for the machine to uncompress and organize the files that have been loaded.

1. Insert the diskette labeled **Installation Disk 1 of 4** into the appropriate drive (usually A, but on some machines it's B).

3. Connect to that drive by typing: **A:** (or **B:**, if appropriate)

3. Run the installation utility by typing: **install**

4. The installer asks you for the drive and path where you want to install Medley. It also asks for your serial number (located on your registration card).

5. The installer asks if you want to update the AUTOEXEC.BAT file with information about the PATH and the SWAP variables. If you answer Y, the installer updates your AUTOEXEC.BAT file as part of the installation process. If you answer N, you will either have to edit your AUTOEXEC.BAT file yourself or set the variables *every* time you start Medley.

You'll need to re-boot your machine before trying to run Medley.

Updating AUTOEXEC.BAT Yourself (optional)

The AUTOEXEC.BAT file contains commands that are run every time you boot DOS. It's located on the disk you boot from, either a floppy or the hard disk.

There are two pieces of information that belong in AUTOEXEC.BAT:

1. Medley's installation directory should be part of the PATH, so DOS can find Medley to run it.

2. The drive that contains Medley's swapping file, if it's other than the C drive.

To add this information, put the follwing lines at the end of the AUTOEXEC.BAT file:

> **PATH=%path%;** *medley-dir*
> **SET SWAP=***drive***:\SWAP.TMP**

For example, if you installed Medley on E:\medley, and want to swap on the G drive, you'd add:

> **PATH=%path%;E:\medley**
> **SET SWAP=G:\SWAP.TMP**

(If you want to swap on the C drive, you can omit the **SWAP=** line.)

You can create or edit AUTOEXEC.BAT using whatever editor you like; these instructions are for EDLIN, the DOS line editor:

1. Connect to the drive that contains AUTOEXEC.BAT (or should contain it, if it doesn't yet exist) by typing: *drive***:**

2. Type: **edlin autoexec.bat**

You will see either New file or End of input file, and a "star" (*) prompt.

3. To insert information into the file, type #i followed by carriage return.

```
End of input file
*#i
```

EDLIN will respond with a number followed by a colon (the next available line in your AUTOEXEC.BAT file): **14:**

4. Type the two new lines, ending each with a carriage return. EDLIN will give you a new line number every time you type carriage return.

```
14: PATH=%path%;E:\medley
15: SET SWAP=G:\SWAP.TMP
16:
```

5. When you are done adding information, type CTRL-C (hold down the Control key and type C). The "star" prompt returns.

6. To end the edit session, type e followed by carriage-return: ***e**

This saves the AUTOEXEC.BAT file and returns you to DOS.

To make the changes permanent, re-boot your machine. Alternatively, you can type the following two lines at the DOS prompt and the changes take effect immediately:

```
C>PATH=%path%;E:\medley
```

```
C>SET SWAP=G:\SWAP.TMP
```

[This page intentionally left blank]

Getting Ready

You need to set the DOS path to include the Medley installation directory; if Medley is to swap on a drive other than C, you need to tell DOS that, too. See the section "Updating AUTOEXEC.BAT" in Chapter 2 for details.

Running Medley

Start Medley by typing the following at the prompt:

medley [*sysout*] [**-m** *memory-size*] [**-vga** | **-vesa102** | **-vesa104**] [**-help**]

sysout is the name of a Lisp virtual memory image file (see the section "Where Medley Looks for Your Sysout" below).

The -m flag lets you control the maximum number of megabytes of memory Medley uses. *memory-size* is a number in the range 8 through 32.

The -vga, -vesa102, or -vesa104 flags let you control Medley's screen resolution. Normally, Medley uses the highest-resolution mode it can find; for VGA it is 640x480, and for SVGA it's 1024x768. Using -vga sets 640x480 resolution (even if you're running an SVGA monitor); -vesa102 sets up 800x600 resolution; -vesa104 sets up 1024x768 resolution. You can only use one of the flags.

The -help flag gives you information about the system and available options.

The First Time

Connect to the drive and directory where you installed Medley, e.g.:

```
C>E:
E>cd \medley
```

Once you are connected to the install directory, typing the follwing should start Medley on any machine: **medley lisp.sys -m 8 -vga**

Doing this starts the smallest sysout we distribute. If you have problems doing this, see "Trouble-Shooting at Start-Up Time" below.

When Medley starts, your screen will go blank for a short time; how long depends on your machine, disk, and network, but it's typically at most a couple of minutes. During this time, Medley is loading the sysout; if the sysout is on your local disk, you'll hear a lot of disk activity.

If the sysout was created on a machine with a different size display, the image will appear garbled for several seconds. As Medley starts running, it adjusts the display to the current size and the screen image unscrambles. The initial screen should look something like:
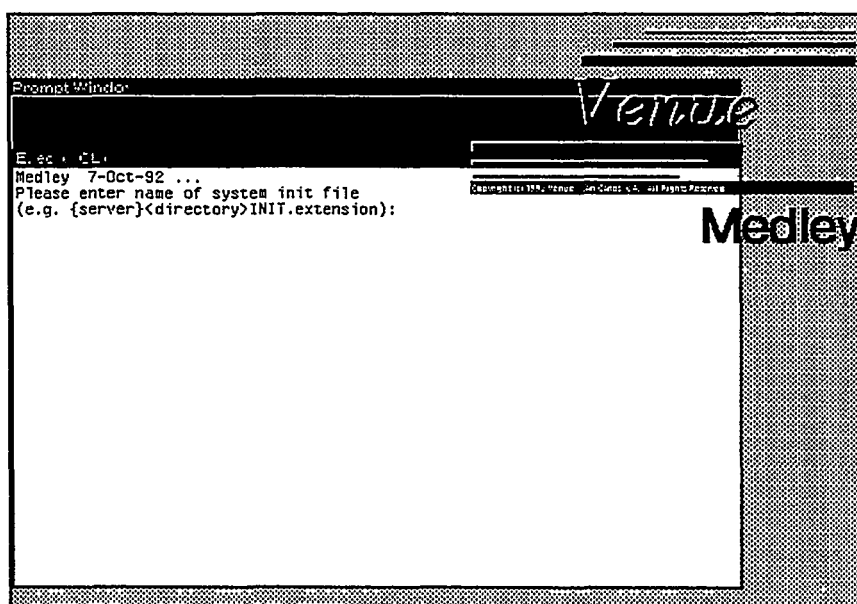
When the mouse cursor appears (as an arrow in the upper left corner of the screen), move it into the white window and click the left mouse button. The caret flashes, indicating that what you type goes to that window.

If you have a system init file prepared, type its name now, followed by carriage-return. For example, if your init is called \medley\init.lsp, type: **{dsk}/medley/init.lsp**

{dsk} tells Medley to look on the local disk (and network-mounted disks); \'s became /'s because Medley uses Unix-style directory specification.

If you don't have an init file yet, just hit carriage return, and you're ready to start using Lisp! See "Setting up a Site Init File" in Chapter 4 in this *Guide* for more information on init files.

Normal Use

Normally, you'll be working on one project across several sessions with Medley, and you'll want to save your state between sessions. When you call the Lisp function (IL:LOGOUT), Medley saves a copy of your state, usually in the file \lisp.vm.

Once you've done that, you can start Medley without specifying a sysout file, and you'll pick up where you left off. Or, if something dire happens, you can restart Medley at the point where you last saved your state.

Trouble-shooting at Start-Up Time

Medley doesn't start; what now? Here's a list of the possible error messages and suggestions for how to recover from them:

```
Bad command or file name
ERROR: Loading emulator failed: File not found.
ERROR: Loading emulator failed: Path not found.
```

You didn't set PATH properly; use the SET command to look at its value, and make sure the Medley directory is on it.

```
sysout_loader: Can't allocate x MB virtual memory
DOS Extender: Error X0135: Extended memory exhausted during application
DOS Extender: Error X0134: Insufficient extended memory
DOS Extender: Error X0130: Failure detected during program load
```

Two things may cause these errors:

1. You tried to load a sysout that's larger than the memory limit you set (e.g., loading a 15 MB sysout when you typed -m 8). You need to request more memory or omit the -m flag to get the 32 MB limit.

2. You don't have enough space for the swapping file. You need to free space on the drive you're using, or specify a different drive by typing: **SET SWAP=d:\SWAP.TMP**, where d: is the new drive.

```
ERROR: -m must be followed by a number 8-32
```

You either didn't enter a number after the -m flag, or the number was outside the range 8 to 32, inclusive.

```
ERROR: Couldn't free excess storage.
ERROR: Loading emulator failed: Invalid Function
ERROR: Loading emulator failed: Unknown problem
```

Your DOS version isn't 4.0 or greater.

```
ERROR: Loading emulator failed: Too many files open.
```

You probably have TSR's running that have files open. You need to shut one or more of them down. Even if you leave some of them running, you can't use them with Medley because they can't get to the keyboard.

```
ERROR: Loading emulator failed: Access denied.
```

Medley's directory is on a remote machine mounted and you don't have execute permission for the file emul.exe. You need to reset the file's permission to let you execute the file.

```
ERROR: Loading emulator failed: Not enough memory.
```

You probably have TSR's running that use a lot of memory. You need to shut one or more of them down. Even if you leave some of them running, you can't use them with Medley because they can't get to the keyboard.

```
ERROR: Loading emulator failed: Environment corrupt.
```

Something corrupted the environment (which you can examine with the SET command). You may need to re-boot to fix it.

```
ERROR: Loading emulator failed: EXE file corrupted.
ERROR: emul.exe is corrupted.
```
Something has smashed Medley's emulator file; you should restore it from backup or re-install it.

```
Can't find a sysout to run
```
Medley couldn't find a sysout. Check your spelling. If you spelled the sysout name correctly or if you omitted the sysout, then Medley couldn't find the file lisp.vm on the current directory. Explicitly specify the drive and directory containing Medley.

Where Medley Looks for Your Sysout

Medley searches the following places, in order, for the sysout to use:

- *command line*

  You can give the sysout name on the command line when starting Medley; for example:

      C> cd medley
      C> medley lisp.sys

- LDESRCESYSOUT

  If you don't specify a sysout name, Medley looks at the environment variable LDESRCESYSOUT for a name. For example:

      C> SET LDESRCESYSOUT=D:/my.sys
      C> medley

  looks for a file named my.sys on drive D. The slash is Medley's internal directory delimiter.

- lisp.vm

  Finally, Medley looks for the file lisp.vm in the current directory.

Where Medley Looks for Your Site Initialization File

When Medley starts, it tries to read a site initialization file. This contians Lisp code to set things like pathnames for fonts, site parameters, etc.

Medley looks for a site initialization file in the following locations:

- LDEINIT

  If the environment variable LDEINIT is set to a complete Lisp file name, Medley looks there first for the site initialization file:

      C> SET LDEINIT=C:/medley/library/my-init.lis

- \medley\library\site-ini.lis

  If LDEINIT is not set or there is no file with the given name, Medley looks for a site initialization file called \medley\library\site-ini.lis. The distribution diskette contains a sample site initialization file in the Lisp library directory \medley\library\site-ini. You can copy site-ini into \medley\library\site-ini.lis then customize it. The comments in the sample describe the parameters it sets and give guidelines for customizing it.

- {DSK}INIT.DFA, {DSK}INIT.LCO, {DSK}INIT.LIS

  Finally, Medley looks for a site initialization file on your Medley directory ({DSK}). Chapter 5, Medley File Systems, describes the {DSK} device.

If Medley can't find an initialization file in one of these places it will ask for an initialization file. If you don't want to load an initialization file, just press the ENTER key.

For further information, see "Setting Up a Site Init File" in Chapter 4 fo this *Guide,* or the *Medley Reference Manual.*

Environment Variables

You can control Medley's interaction with DOS using several environment variables. These variables are listed below, with references to sections in this *Guide* for further information.

LDEINIT	See the Site Initialization File section in Chapter 4.
LDESRCESYSOUT	See "Where Medley Looks for Your Sysout" in this chapter.
LDEDESTSYSOUT	See "Saving Your State" in Chapter 4.
LDEFILETIMEOUT	See "File System Errors" in Chapter 5.

Keyboard Interpretation

This section illustrates how Medley interprets 101-key keyboards. Medley performs its own keyboard interpretation, taking raw up/down key transitions directly from the keyboard. Internally, Medley uses its own key numbering scheme; key numbers are used by Lisp functions such as IL:KEYDOWNP and IL:KEYACTION.

101-Key Right Key Pad

101-Key Center Key Pad Interpretation

14

75	65	95	96
81	82	83	102
84	85	87	
94	69	70	76
98		13	

101-Key Right Key Pad Numbering

33		97	99	100	67		68	101	66	104		80	106	107	108		92	14	61

45	32	17	16	-1	0	2	4	53	22	8	10	59	105	15		89	62	63
34	19	18	3	48	49	51	6	23	25	11	58	29				46	90	91
56	21	20	5	35	50	52	38	9	26	43	28	44						
41	40	24	37	7	39	54	55	27	42	12	60					130		
36	31	57						47	93						129	131	132

101-Key Center Key Pad Numbering

15

[This page intentionally left blank]

Once you have installed Medley, you're ready to go! This chapter provides basic information to get you started. This is only a quick overview; *An Introduction to Medley* has much more detail, and we recommend you work through it.

Talking to Lisp—Medley Executives

When Medley first starts, the biggest window on the screen is the Lisp Executive Window, the main window where you'll run your functions. If you'll be working mainly in Interlisp, you can use the right button background menu to open an Interlisp Executive window to avoid prepending IL: to most of the symbols.
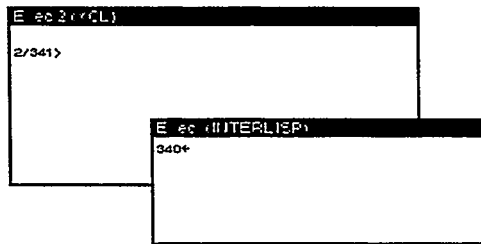
Medley Executive Windows

Executives are your interface to Lisp's READ-EVAL-PRINT loop: You type Lisp code; Medley reads it, evaluates it, and prints the results back into the Exec window.

You can have more than one Executive open at once; to move between them, move the mouse cursor over the window you want to move to, and click the left mouse button. A caret will flash in the Executive that "owns" the keyboard.

Using Medley's Window System

Medley comes with a built-in window system. It's the Interlisp Window System, designed at Xerox PARC, and it's your easiest route to high-quality user interfaces.

Controlling Windows

Position the cursor inside a window, and press and hold the right mouse button. A menu of commands should appear (do not release the right button!), like the one below. To execute one of the commands on this menu, choose the item.
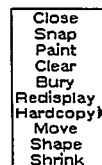
Right Button Window Menu

For example, select "Move" from this menu. The mouse cursor will become a ghost window (just an outline of a window, the same size as the one you are moving), with a square attached to one corner, like this one:
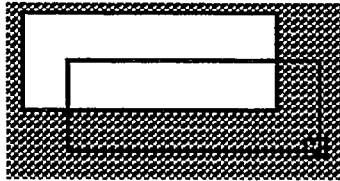
Moving a Window

Move the mouse around. The ghost window will follow. Click the left mouse button to place the window in its new location.

Choose "Shape", and notice that you are prompted to sweep out another window. Your original window will have the shape of the window you sketch out.

The right-button window menu normally offers you these options:

Close Removes the window from the screen

Snap Copies a portion of the screen into a new window

Paint Allows drawing in a window

Clear Clears the window by erasing everything within the window boundaries

Bury Puts the window beneath all other windows that overlap it (but the window will come back to the top if the caret is flashing there)

Redisplay Redisplays the window contents

Hardcopy Sends the contents of the window to a printer or to a file

Move Allows the window to be moved to a new spot on the screen

Shape Repositions and/or reshapes the window

Shrink Reduces the window to an icon of the appropriate shape for that window type, e.g.:

Example of a Shrunken Window Icon

Expand Changes an icon back to its original window. Position the mouse cursor on the icon, press the right button, and select Expand. Or, just button the icon with the middle mouse button.

The right-button menu is available in most windows, including the Executive Window. When the right button has other functions in a window (e.g., in editor windows, it extends a selection), the right button window menu is available by pressing the right button in the black border at the top of the window.

Editing Functions & Variables

Suppose you've entered a function definition (in the XCL Exec):

```
(DEFUN MY-FIRST-FUNCTION (A B)
      (IF (> A B)
           '(THE FIRST IS GREATER)
           '(THE SECOND IS GREATER)))
```

To edit the function, type: `(ED 'MY-FIRST-FUNCTION)`. Your function is displayed in an edit window, as shown below.

If there is no edit window on the screen, you will be prompted to create a window: Hold the left mouse button down, move the mouse until it forms a rectangle of an acceptable size and shape, then release the button. Your function definition will automatically appear in this edit window.

```
SEdit M. FIRST FUNCTION Package -CL-USER
(DEFUN MY-FIRST-FUNCTION (A B)
  (IF (> A B)
      '(THE FIRST IS GREATER)
    '(THE SECOND IS BIGGER)))
```
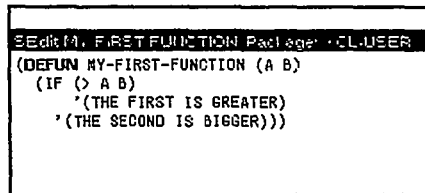
An SEdit Window

By pressing the left mouse button you can place the caret in position, and by pressing the middle mouse button you can select symbols or s-expressions. Repeated pressing of the middle button selects bigger pieces of text. You can then type new code.

When you're done editing, close the window; your function now has a new definition.

If a variable's value is a list, you can also use ED to edit it. For example,

```
(SETQ VAR1 '(THIS (IS A) LIST (1 2 3 4 5)))
(ED 'VAR1)
```

You'll get the same style edit window, and the editing commands are the same. Closing the edit window actually changes your variable's value.

Saving Your Work—Source Code

Medley can take care of source-file management for you. After you've entered some function definitions, call `(IL:FILES?)`. You'll be asked where each function, structure declaration, etc. should go. Answer with a source-file name for each. This sets up Medley's definitions to files mapping. *Caution:* If you wait too long, the questions will time out. You can just call `(IL:FILES?)` again and keep going.

You can then use the function `(IL:CLEANUP)` to actually save the definitions and compile your code.

Saving Your Work—Virtual-Memory Snapshots and SYSOUTs

To save what you're doing (so you can pick it up later), you need to save a copy of Medley's memory on a file. You can do that by calling one of the functions SAVEVM, LOGOUT, SYSOUT, or MAKESYS. This file is an ordinary DOS file (normally \lisp.vm). Medley's entire virtual memory, which may be many megabytes of data, is written out there.

(**IL:LOGOUT** *FAST*)   [Function]

> Lets you exit Medley cleanly. The parameter *FAST* indicates whether or not to save a copy of Medley's virtual memory. Before exiting, disk buffers are written, and network connections subject to timeout are closed.
>
> If *FAST* is NIL, LOGOUT first saves your virtual memory in a file. Change the file name by setting the environment variable LDEDESTSYSOUT. If this variable is not set, the file saved is \lisp.vm (i.e., lisp.vm on your directory).
>
> If *FAST* is T, Medley stops without writing the virtual memory file. *It is not possible to resume execution in the same image.*

(**IL:SAVEVM**)   [Function]

> Saves your state, but does not exit. It causes the current virtual memory image to be written to the location specified by the environment variable LDEDESTSYSOUT, if this variable is set; otherwise it is written to \lisp.vm. This allows Lisp and execution in Medley to continue after memory is saved; thus, SAVEVM operates as a sort of checkpoint of the current working state. SAVEVM can cause the following error: File System Resources Exceeded.

(**IL:SYSOUT** *FILE*)   [Function]

> Performs the equivalent of SAVEVM and then copies the saved image to *FILE*. (See Chapter 5, Medley File Systems, for further information on {DSK}.) SYSOUT can cause the following error: File System Resources Exceeded.

Cursor tracking interferes with writing out the screen bitmap as part of the Medley memory image. For this reason, Medley takes the cursor down before saving a virtual memory image as part of LOGOUT, SAVEVM, SYSOUT, or MAKESYS. When this happens, the message Saving VMem, taking mouse down appears in the prompt window, and cursor tracking is disabled.

The functions LOGOUT and SAVEVM can signal the following file errors:

```
File System Resources Exceeded
Protection Violation
File Wont Open
```

Even if some errors occur while saving a virtual memory, the old destination file is safe. Saving does not overwrite the old virtual memory file. The saving virtual memory file is named with the extension "vm-", such as lisp.vm-. The file is renamed to the specified name, such as lisp.vm, at the end of the save.

The file that LOGOUT and SAVEVM writes is normally \lisp.vm on the current drive. You can use the environment variable LDEDESTSYSOUT to override this default. For example, you might want to keep virtual memory images on \user\local. During a demonstration where you do not want the memory image saved, you can reset LDEDESTSYSOUT to nul:. You can use the DOS command SET to do this, e.g.: **SET LDEDESTSYSOUT = nul:**

When you don't have enough space to save the virtual memory, you can over-write the old VM file by setting IL:\LDEDESTOVERWRITE to T (it's normally NIL). In some cases, even if you try to overwrite, there may still not be enough space.

Note: When Medley saves your sysout, a temporary file with the string "temp" prepended to the sysout name is created. If you have a file named "tempXXXX" where XXXX are the first four characters in the name of your sysout, it will be overwritten and removed.

Setting Up a Site Init File

The users at a given site generally use the same printers, load library files from the same directory, and so on. Medley uses variables to supply defaults for such things. The obvious place to set these variables is in one common initialization file. That is the Site Init File's role.

The Site Init File is a file of Lisp expressions that is loaded when you start Medley with a fresh `LISP.SYS`.

A Sample INIT File

A sample site-init file is shown below, with comments about what's going on.

```
;; site init file. The symbols in this file are mostly in
;; the Interlisp package, so start out there:
(in-package "INTERLISP")

;; Give Medley a search path for finding files you ask for.
;; If you keep files in other places, add them to the list.
(setq directories
    '("{dsk}E:/medley/library/"
    "{dsk}E:/medley/users/"))

;; LISPUSERSDIRECTORIES must contain everything
;; DIRECTORIES does; it CAN have other entries, as well.
(setq lispusersdirectories (copy directories))

;; Look for user init files in a few other places. You only need
;; this if you keep personal init files on odd places.
(appendtovar usergreetfiles
    ("{DSK}/" user "/lisp/init." com)
    ("{DSK}/" user "/lisp/init.lsp")
    ("{DSK}/" user "/medley/init." com)
    ("{DSK}/" user "/medley/init.lsp")
    ("{DSK}/" user "/init." com)
    ("{DSK}/" user "/init.lsp") )

;; Daylight-savings time
(setq |\\BeginDST| 98)
(setq |\\EndDST| 305)

;; Copyright info
(setq copyrightflg 'default)
(setq copyrightowners '((Venue "Venue")))
(setq defaultcopyrightowner 'Venue)
(setq copyrightreserved t)
```

21

```
;; Site name
(setq XCL:*SHORT-SITE-NAME* "Venue")
(setq XCL:*LONG-SITE-NAME* "Venue")

;; paths for display fonts; list the ones that are installed
(SETQ DISPLAYFONTDIRECTORIES
  '("{DSK}E:/medley/fonts/display/pre
    "{DSK}E:/medley/fonts/display/pub/"
    "{DSK}E:/medley/fonts/display/wheel/"
    "{DSK}E:/medley/fonts/display/misc/"
    "{DSK}E:/medley/fonts/display/jis1/"
    "{DSK}E:/medley/fonts/display/jis2/"
    "{DSK}E:/medley/fonts/display/chinese/"))

;; Postscript fonts directories
(SETQ POSTSCRIPTFONTDIRECTORIES
    '("{DSK}E:/medley/fonts/ps/"))

;; Load some files this site always uses.
(FILESLOAD FILEBROWSER)
```

Variables in INIT Files

The following Lisp symbols are those most commonly set in site init files. Naturally, you can set other variables or load other code, as well.

IL:DIRECTORIES   [Variable]

> The list of paths to search for files that are not in the current connected directory. For example,
> `("{DSK}/medley/lispcore/library/" "{DSK}/LispUser/MEDLEY/")`.

IL:LISPUSERSDIRECTORIES   [Variable]

> The list of paths to search for library and LispUsers' files. Every path in this list should also be in DIRECTORIES.

IL:DISPLAYFONTDIRECTORIES   [Variable]

> A list of directories to search when the system is looking for display fonts. You should set it to a list of strings, each containing a complete pathname for font files. For example, `("{DSK}/medley/fonts/display/pres/")`.

IL:\BeginDST   [Variable]

> The day of the year on or before which Daylight Savings Time takes effect (i.e., the Sunday on or immediately preceding this day). Must be set to 98 in the USA if Lisp is to perform time computations correctly (subject, of course, to future legislation). If you are in a region where Daylight Savings Time is not observed, set the value to 367.

IL:\EndDST   [Variable]

> The day of the year on or before which Daylight Savings Time ends. Must be set to 305 in the USA.

IL:USERGREETFILES [Variable]

A list of templates to search for the place where individuals should find their personal init files. If this is not set in the site init file, no personal init file is used. The list should be similar to the following:

```
(({DSK}< USER >LISP>INIT.LCO)
({DSK}< USER >LISP>INIT)
({DSK}< USER >INIT.LIS))
```

IL:DEFAULTPRINTINGHOST [Variable]

A list of names of default printers, e.g. ((POSTSCRIPT maui))

IL:DEFAULTPRINTERTYPE [Variable]

The default printer type for PCs is POSTSCRIPT.

XCL:*LONG-SITE-NAME* [Variable]

Value of the function XCL:LONG-SITE-NAME, e.g., "Frobnitz, Baz and Lispers, Incorporated."

XCL:*SHORT-SITE-NAME* [Variable]

Value of the function XCL:SHORT-SITE-NAME, e.g., "Frobco".

Setting Up a Personal Init File

Your personal init file keeps track of the location of your home directory and windows layout; It also remembers which library files you always load.

Your personal init file is a file of Lisp expressions that is loaded and run after the site init file. You can create it either as a text file, or have Medley's File Manager help you (See Chapter 8 of *An Introduction to Medley*).

Your initialization file is normally \INIT.LCO

PC-Specific Environment Functions

System Environment Functions and Variables

The following functions interrogate the system environment. They work differently from machine to machine and operate as described below when invoked on a PC:

(IL:REALMEMORYSIZE) [Function]

Does not work (i.e., returns a meaningless value).

(CL:MACHINE-TYPE) [Function]

Returns a string identifying the type of computer hardware the system is running under. On the PC MACHINE-TYPE returns "386".

(IL:MACHINETYPE) [Function]

Identifies the generic type of Lisp machine in use. On the PC, it returns the symbol IL:MAIKO.

23

(CL:MACHINE-VERSION) [Function]

> Returns a string identifying the version of the emulator running; e.g., "Microcode version: 279, memory size: 16384".

(CL:MACHINE-INSTANCE) [Function]

> Returns NIL under DOS.

IL:LISP-RELEASE-VERSION [Variable]

> Identifies the release number within a single major release name. In Medley 2.0, IL:LISP-RELEASE-VERSION is 2.0. IL:MAKESYSNAME does not change but IL:LISP-RELEASE-VERSION *always* changes with each new sysout release. This variable did not exist in the Medley 1.0-S sysout.

IL:\MY.NSADDRESS [Variable]

> Fills in the fields of the network address with the host ID if Medley is run without the Ethernet enabled. If you use the network address as a unique identifier, you should be aware that the value could vary from session to session, depending on whether or not the Ethernet is enabled. (Refer to Chapter 14 of the *Medley Reference Manual* for further information.)

VM Functions

The following functions deal with virtual memory and operate as described below when they are invoked on a PC.

(IL:VMEMSIZE) [Function]

> Returns the number of 512-byte pages of the Medley virtual memory that are in use. It's a good estimate of the size of a SYSOUT, MAKESYS, or SAVEVM virtual memory file.

(IL:VMEM.PURE.STATE *ON/OFF*) [Function]

> Has no effect on a PC.

IL:BACKGROUNDPAGEFREQ [Variable]

> Has no effect on a PC. The virtual memory file is not modified except by an explicit (LOGOUT) or (SAVEVM).

You can control how much virtual memory Medley uses by using -m when you start a sysout.

When you use -m, the value of IL:\STORAGEFULLSTATE in the sysout you start should not be 3 or 4. Those values mean it already used more than the 8-Mbyte space in the sysout. Because of Medley's storage management architecture, the virtual memory size cannot be changed after IL:\STORAGEFULLSTATE has been set to 3 or 4. Take a look at it just before (IL:LOGOUT) if you want to specify the virtual memory size during the next start-up. For example: medley \usr\LISP.SYS -m 16. This example means 16 Mbytes of virtual space will be assigned for Lisp.

Environment Inquiry

The following functions return the values of DOS environment variables or machine parameters. The use of "UNIX" in the names is a carryover from earlier versions of Medley. On a PC, the functions behave as described below:

(**IL:UNIX-GETENV** *STRING*)   [Function]

> Returns the value of the environment variable with the given name. The argument *STRING* should be the name of a DOS environment variable. For example, (UNIX-GETENV "HOME") might return your home directory.

(**IL:UNIX-GETPARM** *STRING*)   [Function]

> Returns the value of one of a few built-in parameters. The argument *STRING* should be the name of one of the following DOS environment variables:

Variable	If running on this hardware	Returns
"MACH"	Sun-4	"sparc"
	Sun-3	"mc68000"
	RS/6000	"rs/6000"
	HP9000	"hp9000"
	DEC3100	"mips"
	PS/2	"i386"
	PC	"386"
"ARCH"	Sun-4	"sun4"
	Sun-3	"sun3"
	RS/6000	"rs/6000"
	HP9000	"hp9000"
	DEC3100	"dec3100"
	PS/2	"ps/2"
	PC	"dos"
"HOSTNAME"	All	Returns the local host name
"HOSTID"	All	Returns the local host identification number as a hexadecimal string

Display and Keyboard Functions and Variables

The following functions have no effect on a PC, and always return NIL:

> IL:CHANGEBACKGROUNDBORDER
> IL:VIDEORATE
> IL:SETMAINTPANEL
> IL:VIDEOCOLOR

The following functions generate monotones on a PC:

(**IL:BEEPON** *FREQ*)   [Function]

> Turns on the keyboard tone generator. *FREQ* is ignored.

(**IL:BEEPOFF**)   [Function]

> Turns off the keyboard tone generator.

(**IL:PLAYTUNE** *TUNEPAIRS*) [Function]

    Sounds tones, but ignores the frequencies of the values in *TUNEPPAIRS*.

(**IL:RINGBELLS**) [Function]

    Makes the machine beep several times.

This chapter discusses conventions for using files in Medley.

File Naming Conventions

In Lisp, a file name consists of a collection of fields: *host, directory, name,* and *extension.* These fields are optional. Standard Lisp syntax for a file name is: *{host}drive:<directory>name.extension*

The *directory* field can be a directory path consisting of a sequence of directory and subdirectory components. Slashes (/) are used to delimit directory names. Square brackets ([]) are not acceptable directory delimiters. *Drive* is an optional disk-drive letter, indicating which hard, floppy, or network-mounted disk to search.

Duplicated directory delimiters are treated as a single delimiter. Thus, the following two file names are equal: {DSK}A:<LISP>USERS>FOO and {DSK}A:</LISP/USERS/>FOO

Hosts that Medley Supports

The following hosts are supported by Medley and are described in more detail in the *Medley Reference Manual.*

{CORE} Creates "files" in memory; useful for temporary files.

{LPT} Creates files that are automatically sent to your printer.

{NULL} Creates a file that does nothing.

{DSK} Gives you access to the PC's file system; the rest of the chapter concentrates on it.

Using DOS Files In Medley

You can access any DOS disk directly from Lisp and it will look like Lisp's local disk, even though it may be a remotely mounted file system on a networked file server.

Many of the file devices to which Medley can talk, including PC-NFS file servers, the {CORE} device, etc., have facilities that are not directly supported by DOS. For example, many file systems have file version numbers and case insensitive file search conventions.

Medley on the PC has one distinct "host" name that can be used to access the DOS file system. This host name is provided for compatibility with existing applications and tools. It also simultaneously allows natural interaction with the DOS file system. It is {DSK}.

Common {DSK} Naming Conventions

- To include a special character (e.g., > or ;) in a file name, precede it with a single quote ('). To include a single quote in a file name, precede it with another single quote. You can quote any of these characters: <, >, ;, ~, and . (a period). The following examples show how single quote notation is used on {DSK}.

{DSK} Name In Lisp	File Name In DOS
foo'>bar.baz;1	foo>bar.baz
foo';bar.baz;1	foo;bar.baz
foo''bar.baz;1	foo'bar.baz

27

- {DSK} does not allow you to use slash (/), backslash (\), or the NUL character in file names.
- {DSK} ignores the UNIX home-directory specifier ~ (tilde).
- {DSK} cannot distinguish between a file name ending with a period (e.g., foo.) and a simple file name (e.g., foo).
- {DSK} supports the DOS directory notations . and .. in Lisp directory specifications. The period (.) corresponds to the current working directory. Two periods (..) indicates the parent of the current working directory.
- File names are returned by the system (e.g., INFILEP) in a Lisp-like form so that tools that depend on conventional file name representations (described in the *Medley Reference Manual*) work correctly (e.g., COPYFILES).

{DSK} Naming Conventions

{DSK} performs file name transformation to access files in the DOS file system according the the following conventions:

- The file's name is truncated to 8 characters and the extension is truncated to 3 characters, conforming to DOS file-naming restrictions.
- Directories are delimited by slash (/), rather than back-slash (\).
- File name searches are case-insensitive.
- {DSK} supports relative pathnames. You can specify relative pathnames by omitting a slash (/) or left angle bracket (<) as the first character in the directory field. For example:

  {DSK}test/foo.fee is relative to the top-level directory on the default DOS drive (/test/foo.fee).

  {DSK}../foo.fee is relative to the parent directory of the current DOS working directory (../foo.fee).

Version Numbering

The DOS file system does not support version numbers in file names. Because of this limitation, Medley can provide only a single back-version of each file. Opening a file for writing may overwrite the oldest version.

The newest version of every file is version 1. The older version of a file is version 0. Opening a file causes renumbering of the existing file versions.

The DOS file name for a version-1 file is the same as the Medley name. In version-0 files, the last character of the extension is a percent-sign (%). For example:

{DSK} Name In Lisp	File Name In DOS
foo.bar;1	foo.bar
foo.bar;0	foo.ba%
codefile.m;1	codefile.m
codefile.m;0	codefile.m%
doc.tedit;1	doc.ted
doc.tedit;0	doc.te%

Directories

Directory Enumeration

You cannot use the wildcard character, asterisk (*), in subdirectories for {DSK} devices. For example:

```
>(DIRECTORY '{DSK}/users/x*/foo)
   NIL
```

Directory Creation

When you write a new file on {DSK}, if the directory named in a pathname does not exist, the {DSK} device handler creates the directory automatically. This feature is provided for compatibility with other Medley implementations.

If you try to connect to a nonexistent directory (with CONN or IL:CNDIR), Medley returns the message: Nonexistent directory.

Directory Deletion

{DSK} does not support automatic directory deletion. To delete a directory you must use the DOS command RMDIR.

Open File Limit

The number of simultaneously open {DSK} files must fall within DOS's limits. In DOS, the number of open files may be set in the CONFIG.SYS file. If you try to exceed this number, you'll get the error: Too many open files.

Default Pathname

If no path is given, the {DSK} device defaults to the top-level directory on the current drive. This current working directory can be changed with the CHDIR function. The current working directory is also used to resolve the interpretation of the period (.) and double period (..) specifications at the beginning of a {DSK} pathname.

(**IL:CHDIR** *PATHNAME*)   [Function]

Changes the current working directory. For example,

```
(CHDIR "{DSK}/subdir/")
(OPENSTREAM "{DSK}./foo" ...)
```

opens the DOS file /subdir/foo.

When *PATHNAME* does not end with a slash (/), the whole *PATHNAME* is treated like a directory name:

```
(CHDIR '{DSK}/users/local/)
> "{DSK}/users/local/"
(CHDIR '{DSK}/usr/local)
> "{DSK}/usr/local/"
```

If *PATHNAME* is NIL, CHDIR tries to change the current working directory to the current connected directory. If the directory is connected to devices other than {DSK}, the error

message `Bad Host Name` appears, followed by the host name of the current connected directory.

If the *PATHNAME* does not exist, the error message, `No Such Directory` appears, followed by the system echo of the pathname.

File Attributes

This section describes how the various file attributes are treated by Lisp on a PC and what they translate to in DOS.

GETFILEINFO obtains file attributes and SETFILEINFO sets the attributes (see the *Medley Reference Manual* for details).

WRITEDATE   [File Attribute]
CREATIONDATE   [File Attribute]

Resets the date to the current time whenever the contents of a file are modified. This only works for the owner of the file. Since DOS does not naturally support more than one date for file modification, the WRITEDATE and CREATIONDATE are treated identically by Lisp functions OPENSTREAM, OPENFILE, GETFILEINFO, and the {DSK} device.

TYPE   [File Attribute]

Sets the TYPE property of files; usually TEXT or BINARY. Normally, programs will infer the type by the file extension, using the Lisp variables DEFAULTFILETYPE and DEFAULTFILETYPELIST. If no file extension is given, the value in DEFAULTFILETYPE is used. SETFILEINFO cannot change the TYPE attribute.

EOL   [File Attribute]

Returns the end–of–line convention. {DSK} uses line feed (LF) as the default EOL for text. The EOL for binary files is carriage return (CR). DOS uses the carriage return/line feed (CR/LF) combination for text. EOL uses the TYPE property to determine a file's end-of-line convention. If the TYPE property of a file is TEXT, EOL is LF (=10). If the TYPE property of a file is BINARY, EOL is CR (=13).

AUTHOR   [File Attribute]

Returns the author of the file (the login name of the user who created it). This attribu: cannot be changed.

PROTECTION   [File Attribute]

DOS does not support the notion of file protection.

SIZE   [File Attribute]

Returns the file size as the number of 512-byte pages.

Note: SETFILEINFO lets you change the SIZE attribute of I/O streams and output streams but a file cannot be modified in this way.

File Variables

This section discusses how Medley uses certain file variables in DOS.

IL:FileTypeConfirmFlg [Variable]

The file-type attribute of a file on {DSK} is determined by its extension, DEFAULTFILETYPELIST, and DEFAULTFILETYPE. Binary file extensions should be registered in DEFAULTFILETYPELIST. If you don't do this, hardcopying files on {DSK} may confuse the printer. If you try to hardcopy a file whose extension is not registered in DEFAULTFILETYPELIST, a menu appears so you can confirm the file type (text or binary). You can disable this menu by setting IL:FileTypeConfirmFlg to NIL. The default is T.

If binary file extensions are not registered in DEFAULTFILETYPELIST, copying or renaming files from {DSK} to any of Medley's other internal devices ({CORE}, {LPT}, etc.) may also cause file type confusion. You may get one of the following warning messages:

```
Extension of {DSK}foo.fee isn't in DEFAULTFILETYPELIST.
{CORE}foo.fee was copied as TEXT.
```

```
Extension of {DSK}foo.fee isn't in DEFAULTFILETYPELIST.
{CORE}foo.fee was renamed as TEXT.
```

Either of these messages can be stopped by setting IL:FileTypeConfirmFlg to NIL.

IL:DEFAULTFILETYPE [Variable]

Initially set to TEXT. Used with the file attribute TYPE.

DEFAULTFILETYPELIST [Variable]

A list of accepted file types. Initially set to:

```
((NIL . TEXT) (C . TEXT) (H . TEXT) (LIS . TEXT) (LSP . TEXT)
 (O . BINARY) (OUT . BINARY) (LCO . BINARY) (DFA . BINARY)
 (DCO . BINARY) (SKE . BINARY) (TED . BINARY) (DIS . BINARY)
 (WD . BINARY) (IP . BINARY) (RST . BINARY) (BIN . BINARY)
 (MAI . BINARY) (SYS . BINARY))
```

Used with the file attribute TYPE. Binary files, such as Sketch files or TEdit files, should have their extensions registered in DEFAULTFILETYPELIST. This is especially important as DOS does not support file types.

File System Errors

Several types of errors may occur in the Medley file system.

When a remotely mounted file system or PC-NFS service is down, or when network traffic is heavy, attempts to access a file on that file system results in the error: File access timed out.

Medley will wait until the file system responds or until a timeout occurs. If the file system is mounted with the "hard" option, the timeout is controlled by the value of the environment variable

LDEFILETIMEOUT. If the file system is mounted with the "soft" option, the timeout depends on the PC-NFS file system timeout time and the value of LDEFILETIMEOUT. Medley will wait until the shorter of these two times is exceeded.

If LDEFILETIMEOUT is not set, the default value of 10 seconds is used. The variable is inspected at boot time. A setting between 1 and 100 seconds is appropriate in most cases.

The following error messages (discussed in detail in Chapter 6 of this *Guide*) may appear when there are Medley file errors:

```
Not owner
Device error:
Protection violation
File won't open
Too Many Files Open
File too large
File System Resources Exceeded
Connectiom timed out
No Such Directory
Bad Host Name
FS-RENAMEFILE-SOURCE-COULDNT-DELETE
File not found
```

The error handling system for Medley on the PC includes the following:

- The Medley error system, described in the *Medley Reference Manual*

- A diagnostic program, URAID to handle emulator errors

Occasionally, you may encounter DOS error messages. Refer to your DOS documentation for recovery procedures when these errors occur. When running Medley on a PC, Lisp error handlers such as Teleraid and MP errors are not available.

URAID

Medley normally operates as a self-contained environment. In some unusual circumstances it may encounter a situation from which it cannot recover. When an unrecoverable emulator error is encountered, the emulator halts and enters a small debugger called URAID. URAID lets you inspect memory, or look inside the sysout file to attempt to recover from the error.

Entering URAID

Normally, the emulator automatically enters URAID when an unrecoverable emulator error occurs. If you want to enter URAID directly, there are two additional methods available:

- Use the SHIFT-CTRL-DELETE key combinations to enter URAID between opcodes. The DELETE key referred to here is the DELETE key on the upper center keypad of 101-type keyboards (*not* the right keypad). This sequence allows you to return to Lisp later.

- Use SHIFT-CTRL-NEXT for emergency interrupts only. The NEXT key is the right-hand CTRL key on 101-type keyboards. These combinations are useful for exiting from an opcode infinite loop. SHIFT-CTRL-NEXT does not necessarily enter URAID between opcodes; once you are in URAID mode, another URAID command such as "f" could cause the emulator to crash. At this point it is unlikely that you could return to Lisp. USE WITH CAUTION!

Conventions

URAID uses these display conventions:

- Numbers are displayed in hexadecimal unless otherwise noted.

- Symbols should be an uppercase string when used with a package prefix (e.g., XCL: EVAL).

- Symbols are displayed with a package prefix, but with no escape character.

- Symbols in the Interlisp (IL:) package are case-sensitive (e.g., IL:\InterfacePage); symbols in other packages are case-insensitive (e.g., XCL: EVAL).

In addition, these input conventions apply:

- Symbols may only be qualified by their home package.

- A full package name may prefix an input symbol. URAID also supports approved abbreviations of package names (e.g., XCL:, SI:, CL:, XCLC:).

  A symbol without a prefix is treated as a symbol in the Interlisp package. For instance, \InterfacePage is equivalent to IL:\InterfacePage.

• Type-in is uppercase for symbols in any package except the Interlisp package; type-in is in mixed case for IL: package symbols or symbols with no prefix.

URAID Commands

URAID has a few simple commands which you can use for diagnosis and error recovery. All URAID commands are case-sensitive.

    h Hard Reset. Attempts to recover by resetting the Lisp stack. Quits URAID and causes Lisp to resume execution. This command should *not* be used unless you are sure that execution can be resumed.

    e Exits to DOS. Medley will end.

    q Quits URAID and returns to Lisp.

Note: An error may occur while the Medley system is running uninterruptably. The following message signals this error: Error in uninterruptable system code -- ^N to continue into error handler.

Disregard the ^N command; it is not supported by URAID. Use q to continue.

Displaying the Stack

For casual users, the l command followed by several f commands generally provides the most useful information. Many of the other commands require some knowledge of the internal representation of Lisp objects and stack frames.

    c Checks all user stack contents; stack inconsistency is displayed.

  k *type* Changes the stack link that precedes the l command to be *type*, which is either a (to follow ALinks) or c (to follow CLinks). The default is to trace ALinks. ALinks follow the chain of free variable access.

  l *type* Shows the stack as a back trace consisting of a numbered sequence of frame names. The default is the user stack. The argument *type* is a single letter denoting the stack to view. The system has a number of special contexts, which are areas of stack space used by certain system routines. Legal values of *type* are:

        g (garbage collect)
        k (keyboard handler)
        m (miscellaneous)
        p (page fault)
        r (reset)
        u (user stack) - Default

    C Checks the contents by scanning all stack space in the sysout. For example:

        0x11880 BF,[ivar:0x1800]
        0x11802: FX for CL:T[ ]
        0x11816 BF,[ivar:0x1816
        0x11818: FX for IL:\TURN.ON.PROCESSES[ ]

Viewing Frames From a Stack

After displaying a particular stack with the l command, the following commands view individual frames from that stack:

f *number*   Displays the contents of frame *number* (decimal) with its basic frame, IVars and PVars. The frame is printed in two parts, a basic frame containing the function's arguments and a frame extension containing control information, the function's local (PROG) variables, and dynamic values. On the left side of the printout are the hexadecimal contents of each cell of the frame, with an interpretation, usually as a Lisp value, on the right. The following message appears as you display a frame with the f command: Press Return (To quit ESC and RET)

    To abort the printing of a frame, first press the ESC key then the RETURN key. The URAID prompt "<" reappears.

<CR>   Displays the next frame (closer to the root, or bottom, of the stack). This is the same as f *n+1*, where *n* is the number of the frame most recently viewed. Immediately after an l command, *n* is zero, so **<CR>** views the first frame.

a *symbol*   Displays the top-level value of the *symbol*.

d *symbol*   Displays the contents of definition cell for *symbol*. If it's compiled code, **d** prints a CCODEP hexadecimal address pointer, i.e., {CCODEP}0x14ccc4.

    Otherwise, it prints a Lisp definition; i.e., interpreted code returns: (LAMBDA () ...).

M   Displays TOS, CSP, PVar, IVar, PC.

m *func1 func2*   Moves the definition of *func1* to *func2*.

t *Xaddress*   Displays the type of this object.

p *symbol*   Displays the contents of *symbol*'s property list.

w   Displays the current function name and PC.

x *Xaddress[Xnum]*

    Prints *Xnum* words (16-bit) of the raw contents of the virtual memory starting at virtual address *Xaddress*. This is most useful for examining the contents of a datatype which other commands simply print as its virtual address.

@*symbol[snumber | NIL | T]*

    Sets the TOPVAL of *symbol* to the specified value. *snumber* is a signed smallp number.

<*Xaddress val*   Sets the the contents of the word (16-bits) at the *Xaddress* to *val*.

Miscellaneous

v *filename*   Saves the current virtual memory on the *filename*. This file can be examined using the functions READSYS and VRAID in the TeleRaid Lisp Library module, but *cannot* be used as a sysout file.

s   Invokes a subshell.

(*num*   Sets the print level (default is 2).

?   Displays this summary.

!   Prints the error message passed from the emulator.

Other Fatal Error Conditions

Occasionally, other emulator, operating system, or system administration errors may occur from which the URAID program cannot recover. Such error conditions include the process dying, the emulator going into an infinite loop, the keyboard being lost, or the system freezing up.

If any of these emulator errors occur, use the RESET button or CTRL-ALT-DEL to reboot your machine.

Lisp Errors

Errors While Running Medley

The following Lisp errors may occur when running Medley on a PC.

ERROR MESSAGE	CAUSE
File access timed out	Occurs if you try to access a file when the remotely mounted file system or PC-NFS service is down, or when network traffic is heavy. See Chapter 5, Medley File Systems.
File too large	Self-explanatory.
Too Many Files Open	Occurs when you exceed one of the following:
	• DOS open file limit (see Chapter 5, Medley File Systems)
	• System file resources while writing a sysout (using IL:SYSOUT)
Nonexistent directory	Occurs if you try to connect to a nonexistent directory using IL:CNDIR or CONN.
No Such Directory	CHDIR
Connection timed out	Self-explanatory.
Bad Host Name	Self-explanatory.
FS-RENAMEFILE-SOURCE-COULDNT-DELETE	Occurs if you try to rename a file in a directory for which you do not have delete permission.

Xerox Workstation-Specific Errors

These Xerox workstation-specific errors may occur if certain functions are inadvertently used on a PC.

ERROR MESSAGE	CAUSE
Floppy: No floppy drive on this machine.	Self-explanatory.
Device error: {FLOPPY}	Occurs when you try to enter a Lisp floppy function while running on a PC.
Wrong machinetype	Occurs when functions controlling Xerox disk drive device specific behavior are entered while running in DOS.

Virtual Memory Errors

ERROR MESSAGE	LISP FUNCTION RESPONSIBLE
File System Resources Exceeded	IL:SYSOUT, IL:LOGOUT, IL:SAVEVM
Protection Violation	IL:SYSOUT, IL:LOGOUT, IL:SAVEVM
File Won't Open	IL:SYSOUT, IL:LOGOUT, IL:SAVEVM

[This page intentionally left blank]

Font Directories

The following listing shows the organization of the font directories, as well as their descriptions and contents.

Directory	Description	Families	Types
./fonts/display/pres	All presentation fonts for display and user interface applications	Helvetica Gacha Times Roman	Sans serif Monospace screen font in 8, 10, 12 MRR Serif
./fonts/display/pub	All publishing fonts for character sets, foreign characters, and technical alphabets	Classic Modern Terminal	Serif; in all character sets, sizes, faces Sans serif; in all character sets, faces, but with selected sizes Monospaced, in all character sets, faces, but with selected sizes
./fonts/display/wheel	All printwheel fonts for word processing applications	BoldPS LetterGothic Titan	Proportional serif Monospaced sans serif Monospaced serif
./fonts/display/JIS1	Japanese Kanji fonts, character set 1	Classic	Point sizes 8 through 24
./fonts/display/JIS2	Japanese Kanji fonts, character set 2	Classic	Point sizes 8 through 24
./fonts/display/chinese	Chinese character fonts	Classic Modern	Point sizes 12 and 24 12 point
./fonts/display/misc	Miscellaneous fonts for nonstandard and rare applications	ClassicThin Hippo Logo Math OldEnglish Symbol Tonto	Brackets and parentheses in point sizes 16, 20, 26, and 30 Greek or Latin Xerox logo Math symbols Point sizes 10 and 18 Math symbols Thick monospaced 14 point MRR
./fonts/ps	All metric information for PostScript printers		

[This page intentionally left blank]

byte code emulator	A byte-code instruction interpreter. Executes the Interlisp-D virtual machine instruction set compatibly with microcode for msny workstations.
{DSK}	A host device name allowing users to access the DOS file system. Uses conventions similar to those used by the Xerox 1100 series workstation local disk device ({DSK}).
environment variable	A name/value pair that is passed to subprocesses. By convention, environment variable names use uppercase rather than lowercase letters, e.g., LDEDESTSYSOUT. The Medley environment variables are LDESRCESYSOUT, LDEDESTSYSOUT, LDEINIT, LDESHELL.
home directory	The working directory when a user logs in.
host access key	A special code which must be entered to install Medley on the PC.
Medley	The Venue programming environment. Supports Common Lisp and Interlisp; a library of utilities, graphics packages, applications; a complete windowing system; network protocols. Runs on Xerox and Sun workstations, and PCs.
PC-NFS	PC-Network File System; one way DOS handles remote file systems.
pathnames	In DOS, the full identifier of a file or directory within the file system tree structure.
	An *absolute* pathname gives the position, beginning with the root directory, of the file or directory in the file system hierarchy.
	A *relative* pathname locates the position of the desired file or directory from the working directory.
site initialization file	A Lisp file, used when Medley is started up. Contains standardized information about the site environment such as pointers to fonts and site parameters.

[This page intentionally left blank]

W

[

\

{

*

.