HELP Key

The key labelled HELP looks up context-dependent documentation. Pressing HELP almost always displays something useful. For example, if you have typed se to the Command Processor, and then are not sure what command you actually want, pressing HELP displays:

```
These are the possible command names starting with "se":
  Select Activity
  Send ... (2)
  Set ... (20)

Command: se
```

You can use HELP in combination with other keys to get information about those keys or about Genera.

c-HELP	Shows a list of input editor commands (when typed at a Lisp Listener).
sy-HELP	Shows a list of the special function keys and the special character keys.
SELECT HELP	Shows programs and utilities that you can select using the SELECT key.
FUNCTION HELP	Shows a list of useful functions that you can invoke using the FUNCTION key.
m-HELP	Shows a helpful hint (*HackSaw*) for getting things done faster and more easily in Genera.

c-? and c-/

When you are not sure of the exact name of a command, there are two keystrokes that search the set of available commands and locate possibilities for you. c-? searches for commands whose names begin with the string you have typed so far. c-/ searches for commands whose names contain the string anywhere.

For example, in a Lisp Listener, typing start and pressing c-? yields:

```
These are the possible command names starting with "start":
  Start GC
  Start Printer
  Start Process
  Start X Screen
```

Typing start and pressing c-/ yields:

```
These are the command names containing "start":
  Restart Printer Request  Start GC       Start Process
  Restart Process          Start Printer  Start X Screen
```

Index of Special Function Keys

Introduction

This is a quick reference guide to the Symbolics keyboard special function keys. Most of these keys have the same function in any window. However, a few of them perform differently in different contexts. If you have a Sun or Macintosh keyboard, there is a mapping for finding these function keys on your keyboard. For the MacIvory keyboard, see the section "Using the Genera Application on a MacIvory". For the Sun keyboard, see the section "Symbolics UX Keyboard Templates".

ABORT Key

ABORT — When read by a program, the program stops what it is doing and returns to its "command loop". Lisp Listeners, for example, respond to ABORT by throwing back to the read-eval-print loop (top level or **zl:break**). Note that ABORT takes effect when it is read, not when it is pressed; it does not stop a running program.

c-ABORT — Aborts the operation currently being performed by the process to which you are typing, immediately (not when it is read). For instance, this forces a Lisp Listener to abandon the current computation and return to its read-eval-print loop.

m-ABORT — When read by a program, the program stops what it is doing and returns through all levels of commands to its "top level". Lisp Listeners, for example, throw completely out of their computation, including any **zl:break** levels, then start a new read-eval-print loop.

c-m-ABORT — A combination of c-ABORT and m-ABORT, this immediately throws out of all levels of computation and restarts the current process.

BACKSPACE Key

In a Lisp Listener, BACKSPACE moves the cursor back one character, as does c-B, so that you can insert additional text or edit. In Zmacs, Converse, and Zmail message windows, it inserts a backspace character into the buffer. In the main Zmail window it scrolls the current message backward (as do m-SCROLL and m-V).

CLEAR INPUT Key

Usually erases the expression you are typing. In an editor buffer CLEAR INPUT erases from the location of your cursor to the beginning of the current line. If you are at the beginning of the line, it erases the previous line.

COMPLETE Key

Completes as much as possible of partially typed commands. In Document Examiner, COMPLETE works on topic names.

m-COMPLETE after a Command Processor or Document Examiner command displays a menu of the arguments and keywords for the command you are typing. You can then specify the arguments and keywords from the menu using the mouse or the keyboard.

```
Command: Show System Modifications (for system [default All]) All
For system: All  Local  patchable systems
Author: a string
Before: a universal time
From: an integer
Matching: a string
Newest: an integer
Number: an integer
Oldest: an integer
Reviewer: a string
Since: a universal time in the past
Through: an integer
Output-Destination: a destination
<ABORT> aborts, <END> uses these values
```

See the section "Using Menus".

END Key

Marks the end of input to many programs. Single-line input can be terminated with RETURN or END. END terminates multiple-line input where RETURN is used to separate lines. When you are typing Lisp input, balanced parentheses terminate expressions and END is not used. However, if you use the input editor to yank a previous command or expression, END terminates it. See the section "Editing Your Input".

ESCAPE Key

Displays the mouse-sensitive input editor history for the current window. c-ESCAPE displays the global kill history. Sends Escape/Altmode (octal 033) in the Terminal program.

FUNCTION Key

This key is a prefix for a family of commands relating to the screen, which you can type at any time, no matter what program you are running.

Display and Hardcopy Commands

The FUNCTION commands that control screen display and hardcopying are:

RUBOUT Does nothing; press this key to cancel FUNCTION if you typed the latter by accident.

CLEAR INPUT
Discards typeahead.

REFRESH Clears and redraws all windows.

c-REFRESH Dims the screen as if the console has been idle for 20 minutes, or brightens the screen, if it has been dimmed.

A Arrests the process shown in the status line. FUNCTION - A resumes the process.

B Buries the currently selected window, if any — that is, it moves it to the bottom of the history of selected windows. This brings up the previously selected window, which is automatically selected. It takes numeric arguments to specify the window to bury:

0	or no arg buries the selected window
1	deactivates the selected window (confirm)
2	buries the mouse window
3	deactivates the mouse window (confirm)
4	buries the top level window on the screen
5	deactivates the top level window on the screen (confirm)

C Complements the entire screen. An argument of 1 means white-on-black; an argument of 0 means black-on-white.

c-C Complements the selected window, with the same argument as FUNCTION C.

m-C Complements the mouse documentation line, with the same argument as FUNCTION C.

FUNCTION E Sets the global default end of page mode. With no argument, it prompts with a menu for a new end of page mode. If an argument is supplied, it selects one of these possible modes:

Argument	*Value and Meaning*
1	Scroll. Scroll text on the viewport upward to show the next line.
2	Truncate. Continue with output beyond the end of page, but do not automatically scroll up.

	3	Wrap. Continue with output from the top of the viewport, without moving text already displayed on the viewport.

FUNCTION c-E — Sets the end of page mode for the selected window. With no argument, it prompts with a menu for a new end of page mode. If an argument is supplied, it selects one of these possible modes:

Argument	*Value and Meaning*
0	Default. When an end of page occurs, use whatever action is globally selected at that time. The global action is controlled by FUNCTION E or Set Screen Options.
1	Scroll. Scroll text on the viewport upward to show the next line.
2	Truncate. Continue with output beyond the end of page, but do not automatically scroll up.
3	Wrap. Continue with output from the top of the viewport, without moving text already displayed on the viewport.

If the window does not support an end of page mode, FUNCTION c-E just beeps.

F — Shows users logged in on other machines on your network. Arguments can be assigned to shows users logged in on various machines at your site. FUNCTION 0 F prompts you for a specific user or host to show. Giving a user name followed by /W displays the information in the user's namespace entry.

H — Shows status of network hosts. With an argument, it prompts for hosts.

M — Controls global More processing. No argument means toggle, 0 means turn off, 1 means turn on.

c-M — Controls More processing for the selected window. The arguments are the same as for FUNCTION M.

O — Selects another exposed window.

Function Q — Captures a screen image for hardcopying or inserting in a file.

Function *n* Q — (where *n* is any numeric argument, for example FUNCTION 0 Q) Displays a menu of options for capturing screen images. Note that the choices you make remain in effect for subsequent uses of FUNCTION Q.

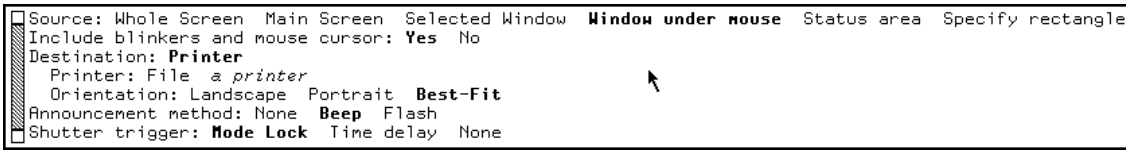


Figure 123. FUNCTION 0 Q

Source: options identify the area of the screen to capture:

Whole Screen	A bitmap image of the entire screen Note that a full-screen bitmap image includes a border around the actual screen image. If you do not want this extra whitespace around the image, select the Named Image option and edit the bitmap image using the "Refit Bounding Box Bitmap Editor Command".
Main Screen	All but the status area
Selected Window	The window of the selected activity (for example, Symbolics Concordia)
Window Under the Mouse	The window under the mouse cursor Use this option when you cannot use the mouse to specify a portion of the screen (for instance, to take a snapshot of a pop-up menu).
Status Area	The window at the bottom of the screen (including the mouse information line(s))
Specify Rectangle	Specify the area of the screen to be captured using the mouse
Window History	The history of the specified window

Shutter Trigger:

Mode Lock	Captures the image when the MODE LOCK key is pressed. This allows you to *set up* the screen a certain way, perhaps with a menu showing, or the mouse cursor in a certain place.
Time Delay	Allows you to specify a time delay before capturing the image. The default is 5 seconds.
None	Captures the image as soon as you click on [Done]. This is the default.

Selection and Notification Commands

The FUNCTION commands that control window selection and notification are:

S — Selects the most recently selected window. With an argument *n* (default is 2), it selects the *n*th previously selected window and rotates the top *n* windows in the selected window history. An argument of 1 steps through each of the previous selected windows in turn (a negative argument steps in the other direction); 0 selects a window that requires attention (for example, to report an error).

T — Controls the selected window's input and output notification characteristics. If an attempt is made to output to a window when it is not exposed, one of three things can happen:

- The program can simply wait until the window is exposed.
- It can send a notification that it wants to type out and then wait.
- It can quietly type out "in the background"; when the window is next exposed the output becomes visible.

Similarly, if an attempt is made to read input from a window that is not selected (and has no typed-ahead input in it), the program can either wait for the window to become selected, or send a notification that it wants input and then wait.

The FUNCTION T command controls these characteristics based on its argument, as follows:

no argument	If output notification is off, turns input and output notification on; otherwise turns input and output notification off. This essentially toggles the current state.
0	Turns input and output notification off.
1	Turns input and output notification on.
2	Turns output notification on, and input notification off.
3	Turns output notification off, and input notification on.
4	Allows output to proceed in the background, and turns input notification on.
5	Allows output to proceed in the background, and turns input notification off.

W — Controls the status line. With no argument, the status line is redisplayed. The arguments control the process the status line watches. The options are:

0	Gives a menu of all processes, and freezes the status line on the process you select. When the status line is frozen on a process, the name of that process appears where your user-id normally would (next to the date and time), and the status line does not change to another process when you select a new window.
1	The status line watches whatever process is using the keyboard, and changes processes when you select a new window. This is the default initial state.
2	Changes the status line so that it displays the name of the process instead of the name of the user. This also freezes the status line on that process; normally the status line switches to display a different process whenever the window system tells it to. Use this if you see an unexpected state in the status line. It will help you find out what process is in that state; you might find that you are not talking to the process you think you should be.
3	Steps the status line through all processes.
4	Steps the status line in the other direction.

Recovering From Stuck States

The following FUNCTION commands should all be used with caution.

ESCAPE
: Helps you recover from stuck states such as "Output Hold" and "Sheet Lock".

c-A
: Arrests all processes except the one shown in the status line and critical system processes, such as the keyboard and mouse processes. FUNCTION - c-A resumes all processes arrested by this command.

SUSPEND
: Gets to the cold-load stream.

c-T
: Deexposes temporary windows. This is useful if the system seems to be hung because there is a temporary window on top of the window that is trying to type out.

c-CLEAR INPUT
: Clears window system locks. This is a last resort, although not as drastic as warm booting. Use this when none of the windows will talk to you, when you cannot get a System menu, and so on.

Debugging and I/O

D	Toggle the DBG flag state.
c-D	Toggle all DBG flags. See the section "Using the Debugger".
END	Inserts an EOF (End of File) indicator into the currently selected I/O buffer.

HELP Key

The key labelled HELP looks up context-dependent documentation. Pressing HELP almost always displays something useful. For example, if you have typed se to the Command Processor, and then are not sure what command you actually want, pressing HELP displays:

```
These are the possible command names starting with "se":
  Select Activity
  Send ... (2)
  Set ... (20)

Command: se
```

You can use HELP in combination with other keys to get information about those keys or about Genera.

c-HELP	Shows a list of input editor commands (when typed at a Lisp Listener).
sy-HELP	Shows a list of the special function keys and the special character keys.
SELECT HELP	Shows programs and utilities that you can select using the SELECT key.
FUNCTION HELP	Shows a list of useful functions that you can invoke using the FUNCTION key.
m-HELP	Shows a helpful hint (*HackSaw*) for getting things done faster and more easily in Genera.

LINE Key

The function of this key varies considerably. It is used as a command by the Debugger, and sends a Line Feed character in the Terminal program. In the editor it behaves like a RETURN followed by a TAB to the indentation level appropriate to the mode of the editor. See the section "TAB Key".

LOCAL Key

This key controls local console functions on XL-family and 3600-family machines:

LOCAL-D	Makes the screen dimmer.
LOCAL-B	Makes the screen brighter.
LOCAL-Q	Makes the audio quieter.
LOCAL-L	Makes the audio louder.
LOCAL-G	Rings the bell.
LOCAL-*n* LOCAL-C	Changes the contrast of the screen. *n* is a digit between 1 and 4. 4 is greatest contrast.
LOCAL-T	(Test Mode). Sends input from the keyboard to the console serial port as ASCII characters. Since most users do not have anything connected to the serial port, this effectively usurps console input.
LOCAL-O	Exits from Test Mode.
LOCAL-ABORT	Resynchronizes the console, resetting all LOCAL settings to their initial state.

Related Lisp functions:

tv:screen-brightness
sys:console-volume

LOCAL does not work on MacIvory or UX-family machines.

NETWORK Key

This key is used to get the attention of the Terminal program. You must be connected to a host via the Terminal program before you can use this key. See the section "Using the Terminal Program".

Once connected, commands are given by pressing NETWORK and another single character.

The following commands are available:

NETWORK HELP	Displays the list of options for the NETWORK key.
NETWORK A	Sends an ATTN (in Telnet, a new Telnet "Interrupt Process").
NETWORK D	Disconnects without logging out first.
NETWORK L	Logs out of remote host, and breaks the connection.
NETWORK Q	Quits, by disconnecting and deselecting this window.
NETWORK M	Toggles More processing.
NETWORK X	Sets terminal options.
NETWORK c-Y	Sends top of kill ring.

NETWORK m-W Wipes marked text.

NETWORK : Enters an extended network command. The extended commands are:

Set Terminal Simulator Type	Set Overstriking
Set Escape Character	Send File
Set Wallpaper File	Send String

NETWORK X gives you a menu of terminal options to set:

```
Terminal command:
Escape character: Network
More processing: Yes  No
Overstrike: Yes  No
Wallpapering: Yes  No
Local Echo: Yes  No
Terminal simulator: VT100  Ambassador  Imlac  Glass TTY
<ABORT> aborts, <END> uses these values
```

PAGE Key

In Zmacs this key inserts a page separator character, which displays as <PAGE> .

REFRESH Key

Erases and redisplays the selected window.

REPEAT Key

Repeats the key pressed while the REPEAT key is held down. You can press and hold down a key and then press the REPEAT key, or you can hold down the REPEAT key and press a key. Once the repetition starts, it continues until you lift your finger from the REPEAT key. You can lift your finger from the first key and press another while still holding down REPEAT and that key starts to repeat. Pressing the REPEAT key without pressing any other key does nothing. The REPEAT key is enabled by default, but you can disable and reenable it by setting the variable **si:*kbd-repeat-key-enabled-p*** with **zl:setf**.

You can have keys repeat if they are held down without using the REPEAT key. See the section "Auto-repeat".

RESUME Key

Continues from the **zl:break** function or SUSPEND and from the Debugger. In the Terminal program this sends a Backspace character. In Zmail it resumes sending the last draft message.

RETURN Key

"Carriage return" or end of line. The exact significance of carriage return varies according to context.

RUBOUT Key

Erases the last character typed. In More breaks, RUBOUT terminates display. In Query Replace operations it means "Skip this replacement and continue."

SCROLL Key

Scrolls the display.

SCROLL	Scrolls the display forward.
m-SCROLL	Scrolls the display backward.
s-SCROLL	Scrolls the display to the right.
m-s-SCROLL	Scrolls the display to the left.
c-SCROLL	Initiates scrolling of the history for the typeout window of current window. You use this in conjunction with c-m-SCROLL to make use of the display on typeout windows, including breakpoints.
c-m-SCROLL	Used after c-SCROLL, scrolls backward through the history of the current typeout window. As with any dynamic window, previous commands and arguments are mouse sensitive and can be reexecuted or used in composing new commands.

SELECT Key

This key is a prefix for a family of commands, generally used to select a window of a specified type, such as a Lisp Listener or Zmail. The current list is:

=	Select Key Selector
C	Converse
D	Document Examiner
E	Editor
F	File System Maintenance Operations
I	Inspector
L	Lisp
M	Zmail
N	Notifications
P	Peek
Q	Frame-Up
T	Terminal
X	Flavor Examiner

SELECT c- creates a new window of the specified type.

Your applications can be assigned to a key for use with SELECT. See the function **tv:add-select-key**. Numeric keys are not valid for use with SELECT. You can customize which applications are assigned to which keys. See the section "Customizing the SELECT Key".

SUSPEND Key

SUSPEND Usually forces the process to which you are typing into a **zl:break** read-eval-print loop, so that you can see what the process is doing, or stop it temporarily. The effect occurs when the character is read, not immediately. Press RESUME to continue the interrupted computation (this applies to the three modified forms of the SUSPEND key as well). While you are in the break, elements of your history in the other window remain mouse sensitive so you can yank them into the break for experimentation.

c-SUSPEND Like SUSPEND, but takes effect immediately rather than when it is read. Press RESUME to continue the interrupted computation.

m-SUSPEND Forces the process to which you type it into the Debugger when it is read. It should type out ">>BREAK:", any proceed options, and the Debugger prompt "→". You can examine the process, then press RESUME or s-A to continue.

c-m-SUSPEND
Forces the current process into the Debugger, whether or not it is running.

SYMBOL Key

Acts as a modifier key to produce special characters. Pressing sy-HELP produces a display of special function and special character keys.

TAB Key

In general, TAB moves the cursor to an appropriate point to the right. The LINE key is related to TAB. If you change the behavior of the TAB key, the behavior of LINE can be affected.

When you are typing a Command Processor command, TAB can be used to terminate a command field, effectively making the field a quoted string. This solves the *left-substring problem*. **accept** and completion prefer long completions. For example, if you want to display the documentation for the topic "Using the Mouse", and you type

```
Show Documentation (for Topic) Using the Mouse COMPLETE
```

Completion does not find "Using the Mouse" since it is looking for the longest topic name. Pressing HELP produces:

```
These are the possible documentation topics starting with "Using the Mou
  Using the Mouse and the Keyboard on Menus
  Using the Mouse in the Debugger
  Using the Mouse in the Metering Interface
  Using the Mouse on History Elements


Press <TAB> for "Using the Mouse"

Command: Show Documentation (for topic) Using the Mouse
```

If you then press TAB, you see:

```
Command: Show Documentation (for topic) "Using the Mouse" (keywords)
```

Keys Not Currently Used

The following key currently has no function:

MODE LOCK

The following keys are reserved for use by the user (for example, for custom editor commands or keyboard macros):

CIRCLE
SQUARE
TRIANGLE
HYPER

Input Editor Commands

Input Editor Commands: c-*number*, c-Minus and c-U provide numeric arguments.

Refresh	Refresh Window
Page	Erase Typeout
m-<	Beginning Of Buffer
m->	End Of Buffer
Clear-Input	Clear Input
c-F	Forward Character
c-B	Backward Character
c-D	Delete Character
Rubout	Rubout Character
c-T	Exchange Characters
c-A	Beginning Of Line
c-E	End Of Line
c-P	Previous Line
c-N	Next Line
c-K	Kill Line

m-F	Forward Word
m-B	Backward Word
m-D	Delete Word
m-RUBOUT	Rubout Word
m-T	Exchange Words
m-U	Upcase Word
m-L	Downcase Word
m-C	Capitalize Word
c-m-F	Forward Expression
c-m-B	Backward Expression
c-m-K	Delete Expression
c-m-RUBOUT	Rubout Expression
c-m-)	Forward Up Parentheses
c-m-(	Backward Up Parentheses
c-m-U	Backward Up Parentheses
c-m-D	Forward Down Parentheses
c-m-A	Beginning Of Definition
c-m-E	End Of Definition
c-m-T	Exchange Expressions
m-(	Make ()
LINE	New Line
BACK-SPACE	Backward Character
c-L	Refresh Window
c-O	Open Line
c-Q	Quote Character
c-m-J	Set Typein Style
HELP	Display Documentation
c-HELP	Display Commands
ESCAPE	Display Input History
c-ESCAPE	Display Kill History
c-Y	Yank
m-Y	Yank Pop
c-m-Y	Yank Input
c-W	Kill Region
m-W	Save Region
c-Space	Set Mark
c-<	Mark Beginning
c->	Mark End
c-sh-Y	Yank Matching
m-sh-Y	Yank Pop Matching
c-m-sh-Y	Yank Input Matching
SCROLL	Scroll Vertical Forward
c-V	Scroll Vertical Forward
m-SCROLL	Scroll Vertical Backward
m-V	Scroll Vertical Backward
s-SCROLL	Scroll Horizontal Forward
s-m-SCROLL	Scroll Horizontal Backward
c-m-S	Save Scroll Position

c-m-R	Restore Scroll Position
s-W	Kill Ring Push Region Strings
s-G	Kill Ring Clear Region Strings
c-S	Scroll Search Forward
s-S	Scroll Search Forward
c-R	Scroll Search Backward
s-R	Scroll Search Backward
c-sh-A	Describe Arguments
c-sh-V	Describe Variable
c-sh-D	Document Symbol
m-sh-A	Lookup Function Documentation
m-sh-V	Lookup Variable Documentation
m-sh-F	Lookup Flavor Documentation
m-HELP	Show Hacksaw

Quick Summary of Mouse Functions

Mouse Cursor Shape

These are some of the more common mouse cursors:

A thin arrow pointing North by Northwest (up and to the left). This is the default mouse cursor. The mouse documentation line indicates any special commands. If the mouse is over a partially exposed window, clicking left selects that window. Clicking sh-Right gets you the System menu. The arrow appears hollow when mouse sensitivity is being computed and fills in black when the computation has completed.

A thin arrow pointing North by Northeast (up and to the right). This means the mouse is in an editor window. There are a number of editor commands on the mouse buttons. See the section "Mouse Documentation Line in Zmacs".

A slightly thicker arrow pointing North (straight up). The editor and the Command Processor use this to show that it is asking you for the name of a function or for a symbol. If you point the mouse at a function name, and stop moving it, the name is highlighted and you can click to select it.

A double-headed arrow pointing north and south (up and down) or east and west (side to side). This indicates the mouse is in a scroll bar or can be used for scrolling.

A small ×. This is used when the mouse cursor should be unobtrusive, for instance in menus.

Mouse Gestures on Dynamic Windows

Left	On a directory listing, does a Show File of the highlighted file. In Other contexts yanks the command line.
sh-Left	Like Left, but also activates. Click sh-Left on a command line to yank and activate the command.
c-Left	Marks a region; s-W pushes the marked region on the kill ring.
c-Right	Brings up the Marking and Yanking menu.
Middle	On a Lisp object does a **describe** of the object.
c-Middle	Yank the word the mouse is over. Useful for using arbitrary text to compose commands; for example, after a Show Mail command, click c-Middle on a pathname mentioned in a mail message as an argument to Show File.
Right on an object	Brings up a menu of possible operations on the object.
m-sh-Right	Gets the menu of window operations.
m-Left in Zmacs	Edit Definition. Hold down the META key and move the mouse around to see what is mouse sensitive.
m-Middle in Zmacs	Evaluate form. Hold down the META key and move the mouse to see what is mouse sensitive.

Scrolling with the Mouse

Windows display "contents" that are too big to fit entirely in the window. When this is the case, you see only a portion of the contents, and you can scroll the contents up and down using the mouse.

Dynamic windows all have scroll bars along one side. The default position for the scroll bar is along the left side. You can position the scroll bar along the right edge if you prefer. See the section "Adjusting Console Parameters". Some dynamic windows have horizontal scroll bars across the bottom.

When the mouse is moved over the scroll bar its cursor becomes a double-headed arrow. A gray area in the scroll bar indicates what portion of the window's contents is visible. The position of the gray area within the scroll bar shows the position of the visible portion of the window's contents relative to the whole. At each end of the scroll bar are small boxes.

Clicking the mouse in the box at the top of vertical scroll bars scrolls by lines. Clicking the mouse in the box at the left of horizontal scroll bars scrolls by columns. The mouse clicks are:

Left	Scroll by one line (next line) or one column (next column to the right)
Middle	Scroll to the top (first available screen) of the window or to the left-most column.

Right	Scroll to the previous line or the previous column.

Clicking the mouse in the box at the bottom of vertical scroll bars or the right of horizontal scroll bars scrolls by screens. The mouse clicks are:

Left	Scroll to the next screen (move one screen to the right).
Middle	Scroll to the end of the window (last available screen) or the rightmost screen.
Right	Scroll to the previous screen (move one screen to the left).

Clicking the mouse in the scroll bar beside the middle of window scrolls proportionally. The mouse clicks are:

Left	Next screen; position this line at the top line of the screen or this column at the left edge of the screen.
sh-Left	Previous screen; position this line at the bottom of the screen or this column at the right edge of the screen.
Middle	Position the window at the location corresponding to this percentage of its contents.
Right	Previous screen; position the line that is currently at the top line of the screen here or the column that is currently at the left edge here.

Other scrolling conventions with the mouse are:

- A fat arrow, pointing up or down. This indicates you are in a scrolling zone. Moving the mouse slowly in the direction of the arrow scrolls the window, revealing more of the text in the direction the arrow points.

- Scrolling zones often say *More above* or *More below* in small italic letters. Clicking on one of these legends scrolls the window up and down by its height, displaying the next or previous screen. When the top or bottom of the window contents is reached, so that it is not possible to scroll any farther in one direction, the legend in the scrolling zone changes to indicate this.

Selecting a Group of Objects with the Mouse

You can use the mouse to sweep over an area and select all the mouse-sensitive objects of a given presentation type in that area to use as input to a command that takes a sequence of arguments.

For example, type `Show Directory sys:examples; .`

Type `Show File` and position your mouse at the right hand side of the directory listing. Click and hold the left mouse button and drag the mouse down and to the left. All the pathnames in the rectangle you create by the diagonal motion are added as a sequence to the Show File command when you release the mouse button.

Note that you can mark groups of objects using the click and hold action, but if your mouse is over an object of the same type as the group you want to select (that is, if your mouse is over a single pathname, in this case), that single object is selected. To select a group, your mouse must initially be over a part of the screen that is not mouse sensitive.

Zmacs Quick Reference

Zmacs Help Facilities

c-ABORT	Aborts the function currently executing.
c-G	Aborts a command when entered, unselects the region, or unmerges a kill.
HELP A *string*	Shows every command containing *string* (try HELP A Paragr or HELP A Buffer).
HELP C *x*	Explains the action of any command (try HELP C c-K as an example).
HELP D *string*	Describes a command (try HELP D Query Rep).
HELP L	Displays the last 60 keys pressed.
HELP U	Offers to undo the last change to the buffer.
HELP V *string*	Shows all Zmacs variables containing *string*.
HELP W	Prompts for an extended command and shows its keybinding.
HELP HELP	Displays these HELP key functions.
HELP SPACE	Repeats the last HELP command.
SUSPEND	Starts a Lisp Listener (return from it with RESUME).

Zmacs Recovery Facilities

m-X Undo	Undoes the last command.
c-sh-U	Undo.
m-X Redo	Undoes the last undo.
c-sh-R	Redo.
c-Y	Yanks back the last thing killed.
m-Y	After a c-Y, successively yanks back older things killed.
c-sh-Y	Prompts for a string to yank.
m-sh-Y	After c-sh-Y, successively yanks back older things containing string.

Extended Commands

Extended commands (the m-X commands) put you in a small area of the screen with full editing capabilities (a *minibuffer*) for entering names and arguments. Several kinds of help are available in a minibuffer.

COMPLETE	Completes as much of the current command as possible.

HELP	Gives information about special characters and possible completions.
c-?	Shows possible completions for the command currently being entered.
END or RETURN	Completes the command, and then executes it.
c-/	Does an apropos on what has been typed so far.

Writing Files

c-X c-S	Writes the current buffer into a new version of the current file name.
c-X c-W	Writes the current buffer into a file with a different name.
m-X Save File Buffers	Offers to save each file whose buffer has been modified.

Buffer Operations

c-X c-F	Gets a file into a buffer for editing.
c-X B	Selects a different buffer (prompts; default is the last one).
c-X c-B	Displays a menu of available buffers; lines are mouse-sensitive.
c-X K	Kills a buffer (prompts; default is current buffer).
m-<	Moves to the beginning of the current buffer.
m->	Moves to the end of the current buffer.
c-m-L	Selects the most recently selected buffer in this window.

Character Operations

c-B	Moves left (back) a character.
c-F	Moves right (forward) a character.
RUBOUT	Deletes a character left.
c-D	Deletes a character right.
c-T	Transposes the two characters around point; if at the end of a line, transposes the two characters before point, ht -> th.

Word Operations

m-B	Moves left (back) a word.
m-F	Moves right (forward) a word.
m-RUBOUT	Kills a word left (c-Y yanks it back at point).
m-D	Kills a word right (c-Y yanks it back at point).
m-T	Transposes the two words around point (if only -> only if).
m-C	Capitalizes the word following point.
m-L	Lowercases the word following point.
m-U	Uppercases the word following point.

Line Operations

c-A	Moves to the beginning of the line.
c-E	Moves to the end of the line.
c-N	Moves down (next) a line.
c-O	Opens up a line for typing.
c-P	Moves up (previous) a line.
c-X c-O	Closes up any blank lines around point.
CLEAR INPUT	Kills from the beginning of the line to point (c-Y yanks it back at point).
c-K	Kills from point to the end of the line (c-Y yanks it back at point).

Sentence Operations

m-A	Moves to the beginning of the sentence.
m-E	Moves to the end of the sentence.
c-X RUBOUT	Kills from the beginning of the sentence to point (c-Y yanks it back at point).
m-K	Kills from point to the end of the sentence (c-Y yanks it back at point).

Paragraph Operations

m-[	Moves to the beginning of the paragraph.
m-]	Moves to the end of the paragraph.
m-Q	Fills the current paragraph (see HELP A Auto fill).
n c-X F	Sets the fill column to *n* (example: c-6 c-5 c-X F).

Screen Operations

SCROLL or c-V	Shows next screen.
m-SCROLL or m-V	Shows previous screen.
c-0 c-L	Moves the line where point is to the top of the screen.
c-m-R	Repositions the window to display all of the current definition, if possible.

Region Operations

c-SPACE	Sets the mark, a delimiter of a region. Move the cursor from mark to create a region. The region is highlighted. Use with c-W, m-W, c-Y and region commands, for example, m-X Hardcopy Region.
c-W	Kills region (c-Y yanks it back at point).
m-W	Copies region onto kill ring without deleting it from buffer (c-Y yanks it back at point).
c-Y	Yanks back the last thing killed.

Window Operations

c-X 2	Splits the screen into two windows, using the current buffer and the previously selected buffer (the one that c-m-L would select).
c-X 1	Resumes single window, using the current buffer.
c-X O	Moves cursor to other window.
c-m-V	Shows next screen of the buffer in the other window; with a numeric argument, scrolls that number of lines — positive for the forward direction, negative for the reverse direction.
c-X 4	Splits the screen into two windows and asks what to show in the other window.

Search and Replace

c-S *string*	"Incremental" search; searches while you are entering the string; terminate search with END.
c-R *string*	"Incremental" backward search; terminate search with END.
c-S END	Enter String Search. See the section "String Search".
c-% *string1* RETURN *string2* RETURN	Replaces *string1* with *string2* throughout.
m-% *string1* RETURN *string2* RETURN	Replaces *string1* with *string2* throughout, querying for each occurrence of *string1*; press SPACE meaning "do it", RUBOUT meaning "skip", or HELP to see all options; (see HELP C m-%).

Summary of Help Functions in Zmacs and Lisp

Both Zmacs and Lisp offer facilities for finding information either about themselves or about the current environment. In addition, Zmacs offers ways to find information about Lisp functions and variables.

This section lists the names of the functions and commands that are available, grouped according to the context in which they are available. The purpose of this section is to summarize the capabilities and to help you determine both the overall contexts for which you can find help and a particular function that might be what you are looking for.

See the section "Reference Description of Help Functions".

Zmacs Commands for Finding Out About the State of Buffers

Edit Changed Definitions (m-X)
Edit Changed Definitions Of Buffer (m-X)
List Buffers (c-X c-B)
List Changed Definitions (m-X)
List Changed Definitions Of Buffer (m-X)
List Definitions (m-X)
List Matching Lines (m-X)
Print Modifications (m-X)

Zmacs Commands for Finding Out About the State of Zmacs

Apropos (m-X)
Describe Variable (m-X)
Edit Zmacs Command (m-X)
HELP L
List Commands (m-X)
List Registers (m-X)
List Some Word Abbrevs (m-X)
List Tag Tables (m-X)
List Variables (m-X)
List Word Abbrevs (m-X)
Show Keyboard Macro (m-X)

Zmacs Commands for Finding Out About Flavors

Edit Combined Methods (m-X)
Edit Methods (m-X)
List Combined Methods (m-X)
List Methods (m-X)
Show Documentation Flavor (m-sh-F)
Show Flavor Initializations (c-sh-F)

Zmacs Commands for Finding Out About Tag Tables

Compile Changed Definitions Of Tag Table (m-X)
Find Files In Tag Table (m-X)
List Tag Tables (m-X)
Next File (m-X)
Select All Buffers As Tag Table (m-X)
Select Some Buffers As Tag Table (m-X)
Select Some Files As Tag Table (m-X)
Select System As Tag Table (m-X)
Select System Version As Tag Table (m-X)
Select Tag Table (m-X)
Tags Edit Compare Differences (m-X)
Tags Find Pattern (m-X)
Tags Multiple Query Replace (m-X)
Tags Multiple Query Replace From Buffer (m-X)
Tags Query Replace (m-X)
Tags Search (m-X)
Tags Spell (m-X)
Take Merge Choice (m-X)
Visit Tag Table (m-X)

Zmacs Commands for Finding Out About Systems

Edit Callers In System (m-X)
Edit System Files (m-X)
List Callers In System (m-X)
Multiple Edit Callers In System (m-X)
Multiple Edit Callers Intersection In System (m-X)
Multiple List Callers In System (m-X)
Multiple List Callers Intersection In System (m-X)
Select System As Tag Table (m-X)
Select System Change Buffer (m-X)
Select System Version As Tag Table (m-X)
Show All Section Changes Of System (m-X)

Zmacs Commands for Finding Out About Lisp

Describe Variable At Point (c-sh-V)
Edit Callers (m-X)
Edit Callers In Package (m-X)
Edit Callers In System (m-X)
Edit CP Command (m-X)
Edit Definition (m-.)
Edit File Warnings (m-X)
Function Apropos (m-X)
List Callers (m-X)
List Matching Symbols (m-X)
Long Documentation (c-sh-D)
Multiple Edit Callers (m-X)
Multiple List Callers (m-X)
Quick Arglist (c-sh-A)
Show Documentation (m-sh-D)
Show Documentation Function (m-sh-A)
Show Documentation Variable (m-sh-V)
Where Is Symbol (m-X)

Zmacs Commands for Interacting with Lisp

Break (SUSPEND)
Compile And Exit (m-Z)
Compile Buffer (m-X)
Compile Changed Definitions (m-X)
Compile Changed Definitions Of Buffer (m-X), m-sh-C
Compile File (m-X)
Compile Region (m-X), c-sh-C
Compiler Warnings (m-X)
Edit Compiler Warnings (m-X)

Evaluate And Exit (c-m-Z)
Evaluate And Replace Into Buffer (m-X)
Evaluate Buffer (m-X)
Evaluate Changed Definitions (m-X)
Evaluate Changed Definitions Of Buffer (m-X), m-sh-E
Evaluate Into Buffer (m-X)
Evaluate Minibuffer (ESCAPE)
Evaluate Region (m-X), c-sh-E
Evaluate Region Hack (m-X)
Evaluate Region Verbose (c-m-sh-E)
Load Compiler Warnings (m-X)
Macro Expand Expression (m-X), c-sh-M
Quit (c-Z)
Trace (m-X)

Lisp Facilities for Finding Out About Lisp

(zl:apropos *string* ***package*****)**
(arglist *function* ***flag*****)**
(describe *object***)**
(describe-area *area-name***)**
(describe-defstruct *instance* ***structure-name*****)**
(describe-package *package-name***)**
(describe-system *system-name***)**
(disassemble *function***)**
(documentation *function***)**
(flavor:flavor-allowed-init-keywords *flavor-name***)**
(inspect *object***)**
(compiler:load-compiler-warnings *file* ***flush-flag*****)**
(mexp)
(trace ***specs*****)**
(untrace ***specs*****)**
(variable-boundp *variable***)**
(what-files-call *symbol-or-symbols how***)**
(where-is *pname***)**
(who-calls *symbol how***)**

Dictionary of Command Processor Commands

These commands can be typed in most Command Processor contexts. Those preceded by a colon (:) are available only in the Debugger. The accelerators for Debugger Commands are also given.

Add Commands

Add HackSaw Command

Add HackSaw *pathname*

pathname The file to which to add the HackSaw. The default is a file for site-wide extensions.

Lets anybody add an interesting fact to a file of interesting facts. You can use this command to create your own personal file of facts.

Add Paging File Command

Add Paging File *pathname :prepend*

Enables you to add a paging file at the Command Processor prompt (in Lisp), rather than from within the FEP. If the paging file does not already exist, use the Create FEP File command to create it. See the section "Create FEP File Command".

pathname The pathname of the existing FEP file, which becomes the new paging file. The default pathname is the disk unit from which you most recently booted. For example, if you most recently booted from FEP1:>, the default paging file might look like:

```
FEP1:>.page
```

Each paging file must have a unique name.

keywords :Prepend

:Prepend {Yes, No} Yes puts the added paging file at the beginning of the list of existing paging files. This makes the newly added paging file available for immediate use. No (the default) puts the added paging file at the end of the list of existing paging files, so that it won't be used immediately.

Add Services to Hosts Command

Add Services to Hosts *service-triples specific-hosts-or-all-hosts keywords*

Adds the specified service attributes in the namespace for one or more hosts.

service-triples A service triple consists of a service, a medium, and a protocol. For more information, see the section "Concepts of Service, Medium, and Protocol".

specific-hosts-or-all A list of hosts (or "all") to which you want to add services. If you specify all, use the :Namespace, :Site, and :Type keywords.

Keywords:

:Locally — {Yes, No} The initial default is No, which updates the global namespace database (the namespace server) to reflect the changes you have made. Answer Yes to update only the local Lisp world with the changes you have made; the changes will be "forgotten" when you cold boot the local machine (unless you save the changed world).

The default for this keyword is displayed along with the object's name, and, if you are editing multiple objects in the Namespace Editor, the current setting becomes the default for subsequent objects.

:Verbose — Prints messages for each host modified.

:Namespace — Adds the services to all hosts in the namespace (use with the "all" argument).

:Site — Adds the service to all hosts in the site (use with the "all" argument).

:Type — Adds the service to all hosts of this system type (use with the "all" argument).

Here is an example adding netboot service to three hosts at once:

```
Command: Add Services To Hosts (A Service Triple) (service) netboot
(medium) slap (protocol) netboot
(A sequence of hosts or All) HARPAGORNIS, WINTER, TOWHEE
(keywords) :Locally

Adding service NETBOOT SLAP NETBOOT to hosts (locally).
Done.
```

Append Commands

Append Command

Append *(sources) source-type-and-name (forming) target-type-and-name*

Appends a selected source, which can be a file, an editor buffer, a stream, or a window, to a selected target, which can also be a file, an editor buffer, a stream, or a window.

source-type-and-name

Type in Buffer, File, Stream, or Window and then identify the particular buffer, file, stream, or window you wish to add to the target. The default is presented for File and Window. In all cases, a brief description of the target type appears. The source is unchanged by the Append operation.

target-type-and-name

Type in Buffer, File, Stream, Printer, or Window and then identify the particular buffer, file, stream, printer, or window you wish to be the target. The default is presented for File and Window. In all cases, a brief description of the target type appears. The target is changed by the Append operation.

Boot Commands

Boot Machine Command

Boot Machine *boot-file keywords*

Boots the machine, using the specified *boot-file*.

boot-file {*filename*, default} The name of the boot file to use. If no file is specified, or if `default` is specified, it uses whatever the current default boot file is in the FEP. Boot Machine verifies that the boot file exists and notifies you if it does not before halting the machine.

keywords :Delay, :Logout, :Reason, :Simulate.

:Delay {*time-interval*} An interval to wait before halting and rebooting the machine. The default is to proceed immediately if there are no active servers on your machine. If your machine has active servers, the default is to delay five minutes.

:Logout {Yes, No} Whether to log you out before halting the machine. The default is Yes, to log you out. Note that even if you do not logout (that is, use :Logout No) your machine is still shut down cleanly, because Boot Machine knows that you are halting the machine to cold boot it, unlike the Halt Machine command, which cannot make that assumption since you might want to Continue or to warm boot.

:Reason {*string*} The reason for the boot.

:Simulate {Yes, No} Whether to just show the effect of the command, without actually booting the machine. The default is No, the mentioned default is Yes.

Boot Machine warns you of any open files and asks you for confirmation that you really want to halt your machine and reboot it.

Clean Commands

Clean Directory Command

Clean Directory *pathname keywords*

Deletes and expunges all but a specified number of versions of a file in the specified directory.

pathname Pathname of the file whose excess versions you wish to clean up. A default is presented. Wildcards are accepted. To clean an entire directory, use the default *.*.*. To clean a smaller set of files use a more restrictive wildcard.

keywords :Expunge, :Keep Versions, :More Processing, :Output Destination, :Query Each

:Expunge {Yes, No, Ask}. If you specify this keyword, `Yes` expunges the directory without asking, `No` does nothing, and `Ask` queries you as to whether to expunge the directory. The default is `Ask`.

:Keep Versions {*integer*} This specifies the number of versions to keep. The default is the value of **zwei:*file-versions-kept***, which is 2 by default.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section `"FUNCTION M"`).

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query Each {Yes, No} If you answer Yes, you are informed of the files that are to be kept and deleted and asked to confirm. If you confirm, then you are asked if you want to expunge the directory. If you answer No, you are told which files are being kept or deleted and then asked if you want to expunge the directory. The default is Yes.

Clean File Command

Clean File *pathname keywords*

Deletes and expunges all but a specified number of versions of a file. This is similar to Reap Files (m-X) or purging files on other systems.

pathname Pathname of the file whose excess versions you wish to clean up. A default is presented. Wildcards are accepted. To clean an entire directory, specify *.*.*

keywords :Expunge, :Keep Versions, :More Processing, :Output Destination, :Query Each

:Expunge {Yes, No, Ask}. If you specify this keyword, Yes expunges the directory without asking, No does nothing, and Ask queries you as to whether to expunge the directory. The default is Ask.

:Keep Versions {*integer*} This specifies the number of versions to keep. The default is the value of **zwei:*file-versions-kept***, which is 2 by default.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query Each {Yes, No} If you answer Yes, you are informed of the files that are to be kept and deleted and asked to confirm. If you confirm, then you are asked if you want to expunge the directory. If you answer No, you are told which files are being kept or deleted and then asked if you want to expunge the directory. The default is Yes.

Clear Commands

Clear All Breakpoints Command

:Clear All Breakpoints *compiled-function-spec*

Clears all breakpoints in the current frame's function or in any other compiled function.

compiled-function-spec
: The name of a compiled function in which you want to clear breakpoints. (Default clears all breakpoints in the current frame's function.)

Clear Breakpoint Command

:Clear Breakpoint *compiled-function pc*

Clears a breakpoint.

compiled-function
: The name of a *compiled function* in which you want to clear a breakpoint.

pc
: The PC (program counter) at which you want to clear a breakpoint. The default is 1.

Suggested mouse operations

- To clear a breakpoint in a compiled function: Display disassembled code with the :Show Compiled Code command, point the mouse at a PC, and press c-m-Middle.
- To clear a breakpoint in a code fragment: Display the code with the :Show Source Code command, point the mouse at a code fragment, and press c-m-Middle.

Clear Output History **Command**

Clear Output History *dynamic-window keywords*

Discards the history for the specified dynamic window. This is useful for when you want to clean up and do an incremental garbage collection.

dynamic-window
: The window whose history to clear. The default is the current window.

keywords
: :Keep All Text, :Keep Marked Text, :Keep Screenfuls.

:Keep All Text
: {Yes, No} Remove presentations and graphics, but not characters. The default is No, the mentioned default is Yes.

:Keep Marked Text
: {Yes, No} Do not remove anything that is underlined. The default is Yes.

:Keep Screenfuls
: {*integer*} Do not remove anything within the last *integer* screenfuls. The default is 0, the mentioned default is 5.

Clear Sage Variable Command

No documentation available for section Clear Sage Variable Command.

Close Commands

Close File Command

Close File *file-spec keywords*

Closes the specified open files or streams.

file-spec — The pathname of the open file, or the token All. If a pathname is specified, it should be the pathname of an open file. The default is All. If All is specified, the function **fs:close-all-files** is executed.

keywords — :Mode, :More Processing, :Output Destination, :Query Each

:Mode — {Abort, Normal} The mode in which to perform the close operation. The default is Abort.

:More Processing — {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination — {Buffer, File, Kill Ring, None, Printer, Stream, Window} Enables you to direct your output. The default is the stream ***standard-output***. Note that redirecting output to a printer can be particularly useful.

:Query Each — {Yes, No} Whether to ask for confirmation before closing each file. The default is Yes.

Compare Commands

Compare Directories Command

Compare Directories *pathname1 pathname2 keywords*

Compares the two specified directories. This command compares only filenames, not the contents of the files in the directories. If the directories contain the same information, you are notified that there are no differences in the two directories. If there are differences, two lists are printed. The first list contains the names of all the files in the first directory that are not in the second directory. The second list contains the names of all the files in the second directory that are not in the first directory.

pathname1 The pathname of the first directory to be used in the comparison. The default is the usual pathname default.

pathname2 The pathname of the second directory to be used in the comparison. The default is the usual pathname default.

keywords :Ignore Versions, :More Processing, :Output Destination

:Ignore Versions {Yes or No} The default is No. If Yes, then consider files with the same name and type to be the same even if they have different version numbers.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Compile Commands

Compile File Command

Compile File *pathname keywords*

Compile the file(s) designated in *pathname*.

pathname The pathname of the file to compile. The default is the usual file default.

keywords :Binary File, :Compiler, :Load, :More Processing, :Output Destination, :Query, :Silently

:Binary File {*pathname*} The file into which to put the output. The default is *pathname*.bin for a 3600-family machine, and *pathname*.ibin for an Ivory-based machine.

:Compiler {Lisp, Pascal, Prolog, Fortran, Use-Canonical-Type} The compiler to use. The default is Use-Canonical-Type.

:Load {Yes, No, Ask} Whether to load the file after compiling. The default is No. The mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Silently {Yes, No} Whether to display the pathname of the file being compiled. The default is Yes.

:Query {Yes, No, Ask} Whether to ask for confirmation before compiling each file. The default is No. The mentioned default is Yes.

Compile System **Command**

Compile System *system keywords*

Compile the files that make up *system*.

system The name of the system to compile. The default is the last system loaded.

keywords :Batch, :Component Version, :Condition, :Copy Compile,:Include Components, :Load, :More Processing, :New Major Version, :Output Destination, :Query, :Redefinitions Ok :Silent, :Simulate, :Update Directory , :Version

:Batch {Yes, No} Whether to save the compiler warnings in a file instead of printing them on the screen. The default is No, to print them on the screen. The mentioned default is Yes.

:Component Version {*version-designator*} The version of any component system to load for the compilation. The default is Released.

:Condition {Always, New-Source} Under what conditions to compile each file in the system. Always means compile each file. New-source means compile a file only if it has been changed since the last compilation. The default is New-Source.

:Copy Compile {Yes, No, Query} For those systems where the product of the compilation is not a true binary file (notably documentation systems) and is usable on both Ivory and 3600 architectures, :Copy Compile Yes has the effect of compiling it for the other architecture. For example, if you compile a documentation system on a 3600-family machine, to "copy compile" it would have the effect of compiling it for the Ivory architecture as well. Yes does the copy compile. No does not. Query prints an explanation of what is being offered, and queries about whether to do it. The copy compile works by copying the form in the `component-dir`.

:Include Components
{Yes, No} Whether to load any component systems. The default is Yes. If :Include Components is Yes, :Component Version is used to select the appropriate version of any component systems.

:Load {Everything, Newly-Compiled, Only-For-Dependencies, Nothing} Whether to load the system you have just compiled into the world. The default is Newly-Compiled.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "`FUNCTION M`").

:New Major Version
{Yes, No} Whether to give your newly compiled version of the system the next higher version number. The default is Yes. Giving the choice No will ask you to confirm that you really want to "prevent incrementing system major version number". (Note that if your goal is to compile a system version for another machine type, you should use the :Version keyword, instead of specifying :New Major Version No. For more information, see the section "Compiling a System for Multiple Machine Types".)

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Everything, Yes, Confirm-Only, No} Whether to query before compiling. Everything means query before compiling each file. Confirm-Only means create a list of all the files to be compiled and then ask for confirmation before proceeding. No means just go ahead and compile the system without asking any questions. The default is No. The mentioned default is Everything.

:Redefinitions Ok {Yes, No} Controls what happens if the system asks for confirmation of any redefinition warnings during the compilation. Yes means assume that all requests for confirmation are answered yes and proceed. No means pause at each redefinition and await confirmation. The default is No. The mentioned default is Yes. This allows you to start a compilation that you know will take a long time and leave it to finish by itself without interruption for questions such as "Warning: *function-name* being redefined, ok? (Y or N)".

:Silent {Yes, No} Whether to suppress output to the screen. The default is No, to allow output. The mentioned default is Yes.

:Simulate {Yes, No} Print a simulation of what compiling would do. The default is No. The mentioned default is Yes.

:Update Directory {Yes, No, *version-designator*} Whether to update the directory of the system's components. The default is Yes.

:Version {Newest, *version-number*} Specifies the version number of a system. When this option is used, SCT checks the journals for the specified system and compiles a new version of the system using the same version of the source files. The new system must be compiled for a different machine type. See the section "Compiling a System for Multiple Machine Types". The default is Newest.

Compress Commands

Compress File Command

Compress File *input-files output-files keywords*

Compresses the data in *input-files* and produces *output-files*. Wildcards are allowed. If *input-files* and *output-files* are the same files, the input files are replaced by the output files.

input-files {*pathname(s)*} One or more files to compress.

output-files {*pathname(s)*} One or more files to contain the compressed data.

keywords :Copy Properties, :Create Directories, :More Processing, :Output Destination :Preamble Type, :Query, :Translation Strategy

:Copy Properties {*list of file properties*} The properties you want duplicated in the new files. The default is author and creation date.

:Create Directories
{Yes, Error, Query} What to do if the destination directory does not exist. The default is Query.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Preamble Type {Symbolics, UNIX} Type of preamble to use.

:Query {Yes, No, Ask} Whether to ask before compressing each file.

:Translation Strategy
{Text, Binary, Query, Heuristicate} Whether or not to perform character set translation. Text means to do ASCII character set translation, reading each input file as a text file. Binary means not to do ASCII character set translation, reading each input file as a binary file. Query asks, for each file, whether to treat the file as text or binary. Heuristicate attempts to guess whether the file is text or binary based on its name, as follows: The filename is broken up into *words*, where each word is separated by a non-alphanumeric character. A rightmost word of "Z" is removed (if present). Then the current rightmost word is checked against **compression::*likely-unix-binary-formats*** If a match is found, the file is assumed to be binary, otherwise it is assumed to be text.

:Translation Strategy is only useful if you are reading or writing a file with a UNIX-style compression preamble, because Symbolics-style compression preambles record the element type and character set of the compressed data. Using :Translation Strategy with a file having a Symbolics-style compression preamble is ignored with a warning.

Copy Commands

Copy File Command

Copy File *pathname* (to) *destination-spec keywords*

Makes a copy of a file.

pathname The pathname of the file you want to copy.

destination-spec The pathname of the location you want to put the copy of the file.

keywords :Byte Size, :Copy Properties, :Create Directories, :Mode, :More Processing, :Output Destination, :Query

:Byte Size {*integer*} Byte size in which to do the copy operation.

:Copy Properties {*list of file properties*} The properties you want duplicated in the new file. The default is author and creation-date.

:Create Directories
{Yes, Error, Query} What to do if the destination directory does not exist. The default is Query.

:Mode {Character, Binary, Default, Macintosh, Raw Character} The mode in which to perform the transfer. The default is Default.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No, Ask} Whether to ask for confirmation before copying each file. The default is No. The mentioned default is Yes.

Copy Flod Files Command

Copy Flod Files *keywords*

Copies FEP overlay (flod) files to a Symbolics 3600-family machine, or flod files and a FEP kernel to to an Ivory-based machine. On Ivory-based machines, Copy

Flod Files also makes sure that your FEP kernel and overlay versions are consistent with one another, and installs the previous FEP kernel as the FEP backup kernel. On XL1200 Color Systems, Copy Flod Files also copies the color system startup file. We recommend that you do not delete previous versions of the flod files and FEP kernel, because they can be useful in debugging certain problems.

keywords :Automatic, :Create Hello File, :Disk Unit, :From Directory, :Hosts, :Silent, :Version

:Automatic Whether to automatically skip copying the flod files to any hosts for which the process gets an error. The default is Yes if more than one target host is specified, otherwise No. The mentioned default in both cases is Yes.

:Create Hello File
{Yes, No, Ask} Whether or not to create a `Hello.boot` file after copying (if one does not already exist). The default is Ask.

:Disk Unit {*integer*} Disk onto which flod files will be copied. The default is 0.

:From Directory {*pathname*} Directory from which to copy files. On Symbolics 3600-family machines, the default is SYS:N-FEP;. On Ivory-based machines, the default is SYS:IFEP;.

:Hosts {*name*, All} Host(s) to which flod files will be copied. The default is your local FEP.

:Silent {Yes, No} Display files as they are copied. The default is Yes.

:Version FEP version. For example, G206, G208, V127, or I316.

Copy Microcode **Command**

Note: This command is implemented only on Symbolics 3600-family machines.

Copy Microcode *{version or pathname} destination keywords*

Installs a version of microcode.

version or pathname
Microcode version number or pathname to copy. *version* is a microcode version number (in decimal). *pathname* rarely needs to be supplied. It defaults to a file on FEP*n*:> (where *n* is unit number of the boot disk) whose name is based on the microcode name and version. (The file resides in the logical directory SYS:L-UCODE;.) The *version* actually stands for the file *appropriate-hardware*-MIC.MIC.*version* on FEP*n*:>.

destination FEP file specification. The pathname on your FEP*n*:> directory. The default is created from the microcode version.

keywords :Update Boot File

:Update Boot File
{*FEP-file-spec*, None, Query}. The pathname of the boot file you want it to update. The default is the current default boot file name.

Copy Output History **Command**

Copy Output History *destination*

destination {Buffer, File, Kill Ring, Printer, Stream, Window} Where to put the copy of the history.

Copies the history of your interaction with a Lisp Listener to *destination*. This is useful if you want to edit your interaction for saving as a text file or if you want a hardcopy of it.

Copy World **Command**

Copy World *file destination keywords*

Makes a copy of *file* (by default, a world load). This includes the specified world as well as any Incremental Disk Save (IDS) worlds on which it was built. See the section "Using the Incremental Disk Save (IDS) Facility". Copy World works from remote terminals. Copy World can also be used to copy netboot cores. You can boot a world from a remote world server with only a netboot core on your FEP. See the section "Netbooting".

file A FEP file specification; the world to copy. The default is constructed from the version of the world that you have booted.

destination A FEP file specification; the pathname for the new world. The default is a wildcard pathname assuring the correct hierarchical pathname relationship for the parent world and an IDS world.

Note: The `.ilod` file extension indicates world-load files for Ivory-based machines, just as the `.load` file extension indicates world-load files for Symbolics 3600-family machines. Files with the `.ilod` extension can be copied only between Ivory-based machines. Files with the `.load` extension can be copied only between Symbolics 3600-series machines.

After you issue the Copy World Command, Genera puts up a menu allowing you to specify the actions you want it to take:

keywords :Automatic, :End Block, :File Set, :More Processing, :Output Destination, :Query, :Start Block, :Transfer Mode, :Update Boot File.

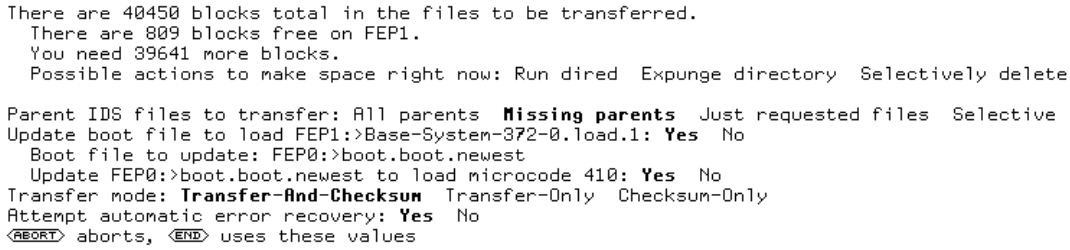


Figure 124. Copy World

:Automatic {Yes, No} Whether or not to attempt automatic error recovery. The default is Yes.

:End Block {*integer*} The number of the last block to copy from source. The default is the last block, meaning copy until the end.

:File Set {All parents, Missing parents, Just Requested Files, Selective} Which parent IDS files to transfer. The default is Missing parents.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No} Whether or not to present a menu of transfer parameters. The default is Yes.

:Show Blocks Copied {Yes, No} Whether to print block numbers for blocks finished copying, every 100 blocks. The default is No, the mentioned default is Yes.

:Start Block {*integer*} The number of the block to start copying from the source. The default is 0, meaning begin at the beginning.

:Transfer Mode {Transfer-and-Checksum, Transfer-Only, Checksum-Only} Whether to verify the integrity of the copied world. The default is Transfer-and-Checksum. You can use Checksum-Only to checksum a band that you copied previously but were unable to checksum due to network problems.

:Update Boot file {*FEP-file-spec*, none}. Boot file to update to load the new world. The default boot file for IDS or complete worlds is `boot.boot`. The default for netboot cores is `none`.

Create Commands

Create Directory Command

Create Directory *pathname*

Creates a directory for storage of files on a file system specified in *pathname*.

pathname The pathname of the directory to be created.

Create FEP File Command

Create FEP File *FEP-file-spec size*

Creates a file on a FEP directory on your machine.

FEP-file-spec The pathname of the file to create. The default is FEP:>temporary.temp.

size The size in FEP blocks of the file. You must supply this.

Use Create FEP File to do the following:

- To create an extra paging file. For example:

```
Create FEP File fep0:>aux.page 100000
```

- To allocate space into which to load a world load. For example:

```
Create FEP File fep0:>release-8-0.load 50000
```

Create File Command

Create File *pathname*

Uses the Simple editor (see the section "Edit File Command") to create a new file or to supersede a file that exists already.

pathname The pathname of the file to be created.

Create Initial FEP Filesystem Command

Create Initial FEP Filesystem *FEP-unit*

Creates an initial FEP filesystem on the given *FEP-unit*. This command is used for initializing an optical disk or a SCSI disk.

FEP-unit The number of the FEP unit.

The convention for mapping SCSI addresses to FEP unit number is that the FEP unit number is 7 greater than the SCSI address. For example, if you set up the SCSI disk drive at SCSI address 4, the FEP unit is FEP 11. (Note that if you have multiple I/O boards, and the SCSI disk drive is attached to the second I/O board, then the FEP unit is 23 plus the SCSI address.)

Create Link **Command**

Create Link *pathname target keywords*

Creates an association between one pathname and a second pathname, called the target. The first pathname is linked to the target so that any references to the first pathname actually refer to the target.

pathname The pathname you want to link from. The default is the standard file default.

target The pathname you want to link to. This pathname must exist. There is no default.

keywords :Create Directories, :More Processing, :Output Destination, :Type

:Create Directories
{Yes, Error, Query} What to do if the destination directory does not exist. The default is Query.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Type — {Read-Only, Read-Write, Create-Through, All, or Use-Default} The kind of link to create. The default is Use-Default.

Create Namespace Object **Command**

Create Namespace Object *class name keywords*

Adds a new object to the namespace database. For more information about adding objects to the namespace database, see the section "Creating a New Namespace Object".

class — {File-System, Host, Namespace, Network, Printer, Site, User} The type of object you want to create.

name — The name of the new object.

Keywords — :Copy From, :Insert Defaults, :Locally, :Property-List

:Copy From — {name of an object of the same class} This will copy the contents of the object named into the newly created object. Duplicate names are removed before the contents are inserted into the new object to avoid conflicts.

:Insert Defaults — Toggles the Namespace Editor insert-defaults mode on or off for the current namespace object. The initial mode is off.

When the insert-defaults mode is enabled for an object, the Namespace Editor inserts default values to the following indicators:

File Control Lifetime
Home Host
Lispm Name
Mail Address
Pretty-Name
Printer and Bitmap Printer
Printer Interface
Services
Site

See the section "The Insert Defaults Namespace Editor Command" for details on how default values are derived.

:Locally — {Yes, No} The initial default is No, which updates the global namespace database (the namespace server) to reflect the changes you have made. Answer Yes to update only the local Lisp world with the changes you have made; the changes will be "forgotten" when you cold boot the local machine (unless you save the changed world).

The default for this keyword is displayed along with the object's name, and, if you are editing multiple objects in the

Namespace Editor, the current setting becomes the default for subsequent objects.

:Property List — {a Lisp expression} This specifies an initial property list for the newly created object in the internal form that object properties are specified in. This keyword option will have no effect if the :Copy From keyword option is used.

Create Netboot Core Command

Create Netboot Core *world*

Creates a netboot core, a small world with just enough information in it to be able to locate and boot a world over the net. An appropriate netboot core is included on every distribution tape, so you should never need to use this command.

world — {*pathname*} The pathname of a world load file to use to create the netboot core. This world must be resident on your FEP or IFEP.

Create Spell Dictionary From Namespace command

Create Spell Dictionary From Namespace *namespace pathname*

Creates a Speller dictionary with all the user-names, first names, and last names from the list of User objects in the namespace. (For speed, the command works by accessing the files that hold the namespace database rather than by accessing the actual namespace servers.)

namespace — The namespace you wish the dictionary to represent.

pathname — The pathname of the binary dictionary file to be created.

Once you have created the file, use the namespace editor to make it a site-specific dictionary. See the section "Adding Site-specific Speller Dictionaries".

Debug Commands

Debug Process Command

Debug Process *process*

Enters the Debugger to look at *process*.

process — A process. You can press HELP for a list of all the processes currently running in your environment. See the section "Show Processes Command".

Decompress Commands

Decompress File Command

Decompress File *input-files output-files keywords*

Decompresses the data in *input-files* and produces *output-files*. Wildcards are allowed. If *input-files* and *output-files* are the same files, the input files are replaced by the output files.

input-files {*pathname(s)*} One or more files to decompress.

output-files {*pathname(s)*} One or more files to contain the decompressed data.

keywords :Copy Properties, :Create Directories, :More Processing, :Output Destination :Preamble Type, :Query, :Translation Strategy

:Copy Properties {*list of file properties*} The properties you want duplicated in the new files. The default is author and creation-date.

:Create Directories
{Yes, Error, Query} What to do if the destination directory does not exist. The default is Query.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No, Ask} Whether to ask before decompressing each file.

:Translation Strategy
{Text, Binary, Query, Heuristicate} Whether or not to perform character set translation. Text means to do ASCII character set translation, writing each resulting file as a text file. Binary means not to do ASCII character set translation, writing each resulting file as a binary file. Query asks, for each file, whether to treat the file as text or binary. Heuristicate attempts to guess whether the file is text or binary based on its name, as follows: The filename is broken up into *words*, where each word is separated by a non-alphanumeric character. A

rightmost word of "Z" is removed (if present). Then the current rightmost word is checked against **compression::*likely-unix-binary-formats***, and if a match is found, the file is assumed to be binary, else it is assumed to be text.

:Translation Strategy is only useful if you are reading or writing a file with a UNIX-style compression preamble, because Symbolics-style compression preambles record the element type and character set of the compressed data. Using :Translation Strategy with a file having a Symbolics-style compression preamble is ignored with a warning.

Define Commands

Define Site Command

Define Site *site-name*

Defines a new site.

Type the Define Site command immediately after you boot a new distribution world when you want to define a new site and namespace; it brings up a menu to create a new namespace called *site-name*; when you start this dialogue the local host is, by default, the site's:

- Primary namespace server.
- SYS host.
- Host for storing namespace database files.
- Host for bug reports.

If you want non-local host(s) to perform any of these jobs, provide their primary network addresses and operating system types in the appropriate menu slots.

During the Define Site dialogue, the namespace database files (object files, log files, changes files, and a descriptor file) are created for you on the file system of the machine you specify as the namespace server. Make sure the file system exists on a host accessible to the namespace server machine before you issue the Define Site command. For more information about the namespace database files, see the section "Namespace Database Files".

The namespace server for the new site will have these initial default attributes when the Define Site command is used (*nnnnn* represents a valid octal Chaos address):

```
System Type*: LISPM
Machine Type: 3600
Service: CHAOS-STATUS CHAOS-SIMPLE CHAOS-STATUS
Service: SHOW-USERS CHAOS NAME
Service: TIME CHAOS-SIMPLE TIME-SIMPLE
Service: UPTIME CHAOS-SIMPLE UPTIME-SIMPLE
Service: LOGIN CHAOS TELNET
Service: LOGIN CHAOS SUPDUP
Service: LOGIN CHAOS 3600-LOGIN
Service: SEND CHAOS CONVERSE
Service: SEND CHAOS SEND
Service: NAMESPACE CHAOS NAMESPACE
Service: NAMESPACE-TIMESTAMP CHAOS-SIMPLE NAMESPACE-TIMESTAMP
Service: LISPM-FINGER CHAOS-SIMPLE LISPM-FINGER
Service: FILE CHAOS NFILE
Service: FILE CHAOS QFILE
Service: CONFIGURATION CHAOS CONFIGURATION
Address: CHAOS 12345
```

See the section "Define Site Dialogue".

Delete Commands

Delete Directory Command

Delete Directory *pathname keywords*

Deletes and expunges the directory represented by *pathname*.

pathname The physical name of the directory to delete. (Using a logical pathname could result in an ambiguous directory reference.) Delete Directory asks you to confirm the deletion before doing it.

You should make sure that *pathname* represents a directory. `A:>KJones>projects>` is a directory, and

```
Delete Directory A:>KJones>projects>
```

offers to delete the `projects` subdirectory. However,

```
Delete Directory A:>KJones>projects
```

is a file name whose directory component is >KJones>; it therefore offers to delete the directory `KJones`. Similarly, the pathname

```
A:>KJones>projects.directory
```

is a file representation of a directory. It is not the directory name for use with commands that manipulate files in directories, so giving it as an argument to Delete Directory also of-

fers to delete the directory KJones.

keywords :Confirm

:Confirm {Yes, No, Each} Whether to ask before deleting each subdirectory. The default is Yes. The mentioned default is Each.

Delete File **Command**

Delete File *pathname keywords*

Deletes or marks for deletion the file *pathname*.

pathname Pathname of the file to delete. The default is the usual file default. The version defaults to newest.

keywords :Expunge, :More Processing, :Output Destination, :Query

:Expunge {Yes, No, Ask}. Whether to expunge the file. The default is No. The mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No, Ask} Whether to ask for confirmation before deleting the file. The default is No. The mentioned default is Yes.

Delete Namespace Object **Command**

Delete Namespace Object *class name keywords*

Removes information about the object *name* from the namespace.

class {File-System, Host, Namespace, Network, Printer, Site, User} The type of object you want to delete.

name The name of the object you want to delete.

Keywords: :Locally

:Locally {Yes, No} The initial default is No, which updates the global namespace database (the namespace server) to reflect the changes you have made. Answer Yes to update only the local Lisp world with the changes you have made; the changes will be "forgotten" when you cold boot the local machine (unless you save the changed world). The default for this keyword is displayed along with the object's name, and, if you are editing multiple objects in the Namespace Editor, the current setting becomes the default for subsequent objects.

If the object you are deleting is still referenced by other objects, for example, if you are deleting a host and a user still lists that host as a mail address, you cannot delete it. This protects your namespace from becoming corrupted. You get an error telling you which objects still reference the object you are trying to delete so you can edit those objects to remove the references.

```
Error: Error from Namespace on RIVERSIDE: You cannot delete HOST QUABBIN.
       It is still referenced by the following objects:
         SITE SCRC             USER LISP-MACHINE  USER WOBBLY
         USER SCH|FILE-SERVER  USER DCP           USER HASEGAWA
         USER NFEP             USER FILE-SERVER
         USER NISHIMOTO        USER ZIPPY
```

Delete Printer Request **Command**

Delete Printer Request *printer-request*

Deletes the specified print request from the print queue.

printer-request A string specifying the printer and the request you are deleting. You can select the print request with the mouse from the display of the Show Printer Status command. For more information, see the section "Show Printer Status Command". You can also specify the printer name to which you are sending requests and use "c-? and c-/"for displaying all requests. For more information, see the section "Completion in the Command Processor".

keywords :Confirm

:Confirm {Yes, No} Whether to request confirmation when you delete a request currently printing. The default is Yes.

Disable **Commands**

Disable Capabilities Command

Disable Capabilities *host capabilities*

Turns off specified *capabilities* for *host*.

host	The name of the host on which you want to disable *capabilities*.
capabilities	The capabilities you want to disable on *host*.

Disable Condition Tracing Command

:Disable Condition Tracing `c-X T`

Disables condition tracing. The `c-X T` accelerator toggles between :Disable Condition Tracing and :Enable Condition Tracing. See the section "Enable Condition Tracing Command".

Disable Network **Command**

Disable Network *network(s)*

Disconnects the local machine from the network.

Network(s)	{*network*, All} The network(s) from which to disconnect. The possibilities are the networks to which your machine is connected. The default is All, meaning disable all network service on your machine.

Disable Services **Command**

Disable Services *service-type*

Turns off service(s) on the local machine.

service-type	{All-Services, *service1, service2...*} The service to turn off. Press `HELP` for a list of available services. The default is All-services.

Dismount **Commands**

Dismount Optical Disk Command

Dismount Optical Disk *SCSI-address keywords*

Dismounts the optical disk indicated by *SCSI-address*. Once the disk is dismounted, you cannot access it again until you mount it.

SCSI-address The SCSI address of the optical disk drive.

keywords :Eject

:Eject {Yes, No} Whether to eject the optical disk after it is dismounted. The default is No. The mentioned default is Yes.

Distribute Commands

Distribute Systems Command

Distribute Systems *systems-and-versions-pairs keywords*

Writes systems to tape for distribution. If you do not specify a system, the Distribute Systems Activity window is selected for you. Distribute Systems lists the systems to write to tape, and asks if you want to perform the Distribute Systems operation. Type `Y` for Yes, `N` for No, `Q` for Quit, or `S` for Selective.

If you choose Selective, each file is listed, and you are asked if you want to distribute that particular file. You can select as many files as you want. After you enter this information, you are prompted for a tape specification, if you did not specify one already.

systems-and-versions-pairs
A list consisting of items separated by commas, each item being a system name followed by a space and a version number.

keywords :Compress Files, :Default Version, :Distribute Patch Sources, :File Types, :Full Length Tapes, :Include Components, :Include Patches, :Included Files Checkpoint, :Machine Types, :Menu, :More Processing, :Output Destination, :Query, :Source Category, :Tape Spec, :Use Cached Checkpoint, :Use Disk

:Compress Files {Yes, No} Whether to compress the files when writing them to tape. The default is No. The mentioned default is Yes.

:Default Version {Released, Latest, Newest, *version-designator*} Version of the system to distribute if not individually specified in Systems. The default is Released.

:Distribute Patch Sources
{Yes, No} Whether to include patch sources for system patches. The default is No. The mentioned default is Yes.

:File Types {Sources, Binaries, Both, Patches-Only, Default} What file types to distribute. The default leaves it to the specifications in individual **defsystem** forms.

:Full Length Tapes
{Yes, No} Write all tracks of the tape. Use this *only* if you are sure that you don't have to read the tape on a 3600 Cipher drive. The default is No. The mentioned default is Yes.

:Include Components
{Yes, No} Whether to include any component systems of the systems being distributed. The default is Yes.

:Include Patches {Yes, No, Selective} Whether to include the patch files for the systems being distributed. The default is Yes. If you include patch files and also distribute source files, the source file corresponding to the patch level, not necessarily the source used for the compilation, is the one included on the tape. For example, suppose you have a system that includes the file blue.lisp.1. You put this file in an editor buffer, modify the code, make a patch file, and then save the buffer with the altered code to blue.lisp.2. When you use Distribute Systems, blue.lisp.2 is distributed, but not blue.lisp.1.

:Included Files Checkpoint
{Patch, Release, None} Limit distributed files to those after this patch number or release name, or None (do not limit). The default is None.

:Machine Types {3600, Imach, All} Specifies whether the systems to distribute should be for 3600-family machines, Ivory-based machines, or all machine types. The default is to distribute systems for all machine types.

:Menu {Yes, No} Whether to use a menu interface to specify details of the distribution. Choosing Yes presents a Distribute Systems frame to select which files are distributed. For detailed information about this frame, see the section "Distribute Systems Activity". The default is No. The mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Everything, Yes, Confirm-Only, No} Whether to ask about distributing each file. The default is Confirm-Only. The men-

tioned default is Everything. Everything queries you about each file being distributed, and again for each system being distributed. Yes queries you about each file being distributed. Confirm-Only queries you about each system being distributed. No does not query.

For queries about individual files, the possible responses are:

Y Yes, distribute this file.
N No, do not distribute this file.
I Include the remaining files in this system.
B Bypass (do not include) the remaining files in this system.
D Directory. Show the directory containing this file.
E Edit this file.
S Source compare this file.

For queries about systems, the possible responses are:

Y Yes, distribute all files listed in this system.
N No, do not distribute any files listed.
Q Quit. Do not distribute any files listed in this system.
S Selective. Query about each file in this system individually.

:Source Category {Basic, Optional, Restricted, Optional-only, Restricted-only} Indicates which source category or categories to write to tape for distribution. The default is Basic.

:Tape Spec The specification for the tape. The default is the default drive on the local machine. For more information about tape specifications, see the section "Tape Specifications".

:Use Cached Checkpoint
{Yes, No} Use the last checkpoint gathered for this system. Using the cached checkpoint information, if there is any, saves time. But it is safe to use only if you are sure no more patches have been made since the cached information was computed. The default is No. The mentioned default is Yes.

:Use Disk On all machines other than a MacIvory, the choices are {Yes, No}. If Yes, the input is written to disk as a special file that is an image of what would be written to tape. When writing to disk, the distribution plan is not divided into parts according to any size limit. You can use this either to distribute on disk, or when you are preparing a distribution and want to see what files would be written to tape. The default is No. The mentioned default is Yes.

On a MacIvory, the choices are {Tape, Disk, Floppy}. Disk indicates the hard disk, and Floppy indicates the floppy disk. These two values also write a special file that is an image of what would be written to tape. Additionally, when a floppy disk is

used, the size limit for a "reel" is set to the capacity of the floppy (that is, 800 Kbytes). Tape means to write to tape; this is the default.

Edit Commands

Edit ACL Command

Edit ACL *directory*

Starts up a small window, editing the Access Control List for *directory*. It is used to create, edit or remove access control lists for LMFS directories. This command can be executed from any console, by the owner of *directory*. This should be coordinated with the Site Administrator, so that passwords can be assigned and any new capabilities can be set up. See the section "Access Control Model: What You Can and Cannot Protect".

The window is divided into four panes: the top displays the name of the directory, the next pane displays the existing ACLs, next is a command pane, and the bottom pane is a minibuffer for typein.

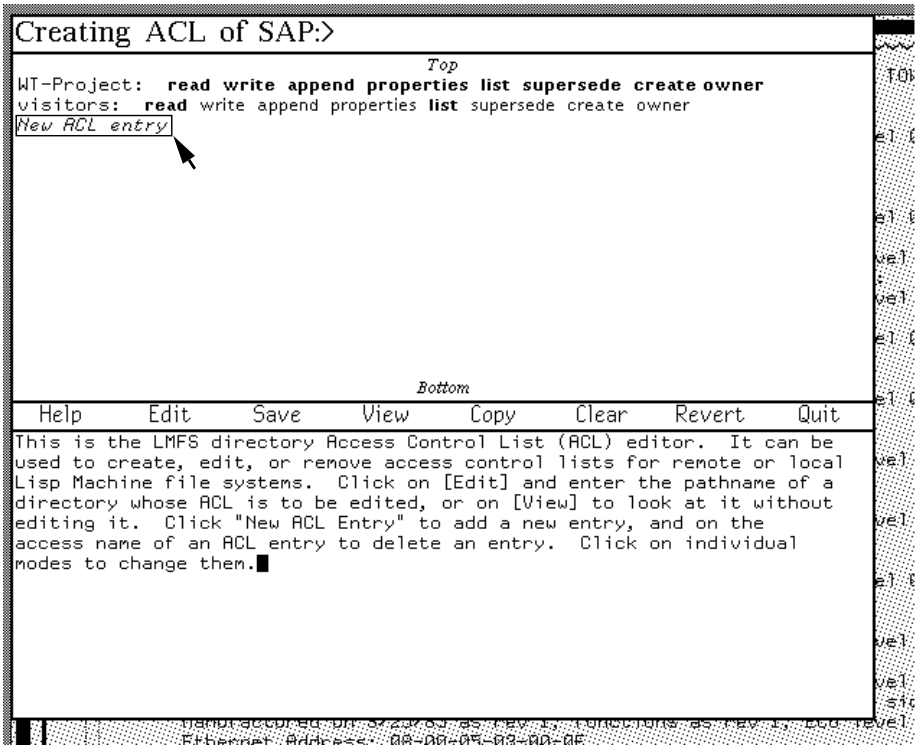


Figure 125. Access Control List Editor

[Help] Display general information about the Access Control List editor.

[Edit] Prompts for the name of a directory, and starts an edit of its ACL. If there is no ACL for that directory, the second pane will only contain *New ACL entry*. Click on this command to add a new ACL. To delete an entry, click on the access name of that entry.

Each entry consists of an access name, and a list of modes that can be enabled or disabled. Clicking on a mode toggles whether or not it is enabled. Modes that are enabled appear in boldface type, and modes that are disabled appear in roman type.

[Save] Installs the permissions for the host you are editing by changing the file properties for that directory.

[View] Prompts for the name of a directory, and displays its ACL. This is done in read-only mode, so you cannot modify the ACL using this command.

[Copy] Creates a new ACL for a directory by copying the current ACL.

[Clear] Disables all modes for the current ACL entry.

[Revert] Reverts the ACL to the modes specified in the saved file for this directory. If the ACL has never been saved, the editor asks if you want to discard the current ACL information. When you confirm, all entries are discarded.

[Quit] Return from the Edit ACL command to whatever you were previously doing.

For the example here, the capability named "WT-Project" allows all operations on the directory SAP:>. The capability named "visitors" only allows the reading and listing of files in the directory.

Edit Definition **Command**

Edit Definition *name type*

Finds the definition of type *type* for the object *name* and puts it in an editor buffer for you to edit. This is the same as the Zmacs command m-.. See the section "Edit Definition".

name Name of the object whose definition you want to edit.

type {Any,Function,**locf** function,**zl:setf** function} The type of definition for the object.

Edit Directory **Command**

Edit Directory *directory-spec keywords*

Invokes the directory editor **zl:dired** in Zmacs. See the section "Dired Mode in Zmacs".

directory-spec Pathname of the directory to edit. The default is the usual file default.

keywords :Version

:Version {All, Newest, *number*} The versions of the file to be included in the directory listing. The default is All.

Edit Disk Label Command

Edit Disk Label *unit-number*

Edits the disk label of the disk identified by *unit-number*. Brings up a menu of choices, which you can click on to change aspects of the disk label, such as the FEP kernel, backup FEP kernel, and the color system startup file (used for color systems). You can also click on choices to deinstall the current FEP kernel, deinstall the FEP backup kernel, and deinstall the color system startup file.

Just as the device PROM finds the FEP kernel by reading the disk label, it also needs to find the file where the FrameThrower color system startup programs are stored. Note that the device PROM does not understand the FEP file system; it just reads the disk label, which points to the correct file.

unit-number{*integer*, All} The number identifying the FEP disk whose disk label you want to edit. All allows you to edit the labels of all the disks attached to the machine.

Normally, the FEP kernel and FEP backup kernels are installed automatically, when you use the Copy Flod Files command. Copy Flod Files installs the new FEP kernel and installs the previous FEP kernel as the backup FEP kernel. Only in unusual debugging circumstances do you need to deinstall the FEP kernel or FEP backup kernel by using Edit Disk Label. When you deinstall the FEP kernel, the FEP backup kernel is installed as the FEP kernel. When you deinstall the FEP backup kernel, the FEP kernel is installed as the FEP backup kernel. (Note that the FEP kernel and FEP backup kernel must always be present, in order for this to work properly.) Thus, when you deinstall either the FEP kernel or FEP backup kernel, the result is that a single FEP kernel is installed as the FEP kernel and the FEP backup kernel; in other words, there is no separate FEP backup kernel. When you next use Copy Flod Files, the new kernel will be installed as the FEP kernel, which means you will again have a different FEP kernel and FEP backup kernel.

Edit File **Command**

Edit File *pathname keywords*

Enters the specified editor and reads in *pathname*.

pathname — The pathname of the file to edit. The default is the usual file default.

keywords — :Editor

:Editor — {Simple, Zmacs} The style of editor to use. The default depends on your terminal: for the main screen, it is Zmacs, but if you are using a remote terminal, it is Simple. The Simple editor provides just the input editor and is useful only for small files.

Edit Font **Command**

Edit Font *font*

Invokes the Font Editor with *font* loaded to be edited.

font — A font name. There is no default. Issuing the command with no arguments invokes the font editor with no font loaded.

Edit Function Command

:Edit Function *function* c-E

Enters the Zmacs editor to bring up the current function or any other function for editing. This command lets you look at the function's source code. This is useful when you have found the function that caused the error and want to fix the code right away. The editor command c-Z returns to the Debugger, if it is still there.

function — A stack frame that you select with the mouse or a function spec that specifies which function you want to edit. (Default edits the current function.)

Suggested mouse operations

To edit a function: Point the mouse at the function's stack frame and press m-Left.

Edit Namespace Object **Command**

Edit Namespace Object *namespace-object keywords*

Modifies an object in the namespace database.

To create a new object, see the section "Create Namespace Object Command".

namespace-object A namespace object is specified by the class of the object followed by the name of the object. For instance, to specify the printer named Asahi, you enter:

```
Edit Namespace Object printer Asahi
```

If *namespace-object* is described in more than one namespace (is "multi-homed"), Edit Namespace Object prompts for the namespace in which you want to edit this object. (Typically, a namespace object is described only in one namespace.)

Keywords :Insert Defaults, :Locally

:Insert Defaults Toggles the Namespace Editor insert-defaults mode on or off for the current namespace object. The initial mode is off.

When the insert-defaults mode is enabled for an object, the Namespace Editor inserts default values to the following indicators:

File Control Lifetime
Home Host
Lispm Name
Mail Address
Pretty-Name
Printer and Bitmap Printer
Printer Interface
Services
Site

See the section "The Insert Defaults Namespace Editor Command" for details on how default values are derived.

:Locally {Yes, No} The initial default is No, which updates the global namespace database (the namespace server) to reflect the changes you have made. Answer Yes to update only the local Lisp world with the changes you have made; the changes will be "forgotten" when you cold boot the local machine (unless you save the changed world).

The default for this keyword is displayed along with the object's name, and, if you are editing multiple objects in the Namespace Editor, the current setting becomes the default for subsequent objects.

Edit System Command

Edit System *system-or-subsystem keywords*

Reads all the files that make up a system into editor buffers for editing.

system-or-subsystem The system whose source files you want to edit.

keywords :Include Components, :Query, :Silent

:Include Components
{Yes, No} Whether to edit component systems. The default is Yes.

:Query {Everything, Yes, Confirm-Only, No} Whether to ask about editing each file. The default is No. The mentioned default is Everything or Yes, which are the same. If you select :Query Yes, you are asked about each file. Then a list of all the files you have selected is displayed and you are asked to confirm that you want to edit those files. If you select :Query Confirm-Only, you are only asked to confirm the list of files. You can answer Yes, No, or Selective. Selective allows you to answer Yes or No to each file in the list and finally displays a new list for you to confirm.

:Silent {Yes, No} Whether to suppress all terminal output. The default is Yes.

Enable Commands

Enable Capabilities Command

Enable Capabilities *host capabilities*

Turns on specified *capabilities* for *host* after checking access requirements. The host prompts for your password for *capabilities*.

host The name of the host on which you want to enable *capabilities*.

capabilities One or more of the capabilities available for *host*. Each specified capability must be already recognized by the server before access can be enabled.

Enable Condition Tracing Command

:Enable Condition Tracing *condition keyword* c-X T

Enables condition tracing. That is, this command allows you to debug an error handler when it does not work properly. For example, when you receive continuous, recursive error messages due to a defective error handler, you can use :Enable Condition Tracing to cause a trap and enter the Debugger before the condition is signalled. Once in the Debugger, you can debug and fix the handler.

You should use this command only if you code your own error handlers. If you do not code your own handlers, and suspect there is a bug in a handler, send a bug report to your Symbolics customer representative.

Any numeric argument given with this command's accelerator sets **sys:trace-conditions** to **t**. The c-X T accelerator toggles between :Enable Condition Tracing and :Disable Condition Tracing.

condition {**t**, **nil**, *conditions*} **t** enters the Debugger when any condition is signalled. **nil** turns off condition tracing previously specified by **t**. *condition* is a condition flavor, which causes entry to the Debugger when any flavor built on *conditions* are signalled.

keyword :Conditional

:Conditional {Always, Mode-Lock, Never, Once} Enables condition tracing according to certain conditions. Always: enables condition tracing in all cases. Mode-Lock: enables condition tracing only when the MODE LOCK key is held down. Never: has the effect of disabling condition tracing. Once: enables condition tracing only for the first time a condition is raised. (Default is Always.)

Enable Network **Command**

Enable Network *network(s)*

Enables (turns on) the network service for the local machine.

network(s) {*network*, All} The network(s) to enable. The possibilities are any of the networks to which your machine is connected. The default is All, meaning enable all network service on your machine.

Enable Services **Command**

Enable Services *service-type*

Turns on currently disabled service(s) on the local machine.

service-type {All-Services, *service1*, *service2*...} The service to turn on. The default is all-services. Press HELP for a list of available services.

Expunge **Commands**

Expunge Directory **Command**

Expunge Directory *pathname keywords*

Expunges files marked for deletion. When Expunge Directory (m-X) is used to expunge multiple top-level directories by wildcard reference (such as >*> or >**>) the variable **si:*expunge-notification-default*** controls whether other users at the site will be notified first. If a numeric argument is provided, the value of this variable is locally forced to **:query**. By default, the variable **si:*expunge-notification-default*** is :No.

pathname The pathname of the directory to be expunged. The default is the usual file default.

keywords :Delay After Notification :More Processing :Notify :Output Destination :Query

:Delay After Notification
{a time interval} When notifications are sent to other users, controls the length of time between notification and expunge. During the delay, the process can be aborted with Control-Abort, which both cancels the expunge and sends a notification that the expunge has been cancelled. By default, the variable **si:*expunge-notification-delay-default*** is :No.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Notify {Yes, No, Query} Controls whether other users at the site are notified when multiple top-level directories are to be expunged by wildcard reference, for example, >*> or >**>. When such wildcards are not used (or when such wildcards are only used to specify wild subdirectories of one or more specifically named toplevel directories, for example >Harry>*> or >Sally>**>), this option has no effect. By default, the variable **si:*expunge-notification-default*** is :No.

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} En-

ables you to direct your output. The default is the stream ***standard-output***. Note that redirecting output to a printer can be particularly useful.

:Query {Yes, No, Ask} Ask for confirmation before expunging the directory. The default is No. The mentioned default is Yes. :Query is useful when *pathname* contains wildcard.

Find Commands

Find HackSaw Command

Find HackSaw *one-or-more-words keywords*

Finds a particular HackSaw.

one-or-more-words {*string*} A word or words from a HackSaw to use as a search string.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Suppose you saw a fact go by, but can't remember it now that you need it. If you can remember any of the topic words likely to have been in it, Find Hacksaw will find it.

Find String Command

Find String *string keywords*

Finds files that contain one or more strings you specify, listing the line numbers and contents of the lines containing the string(s).

string The string to search for. You can specify more than one string, separated by commas. Files containing any of the strings specified are found.

keywords :Branch, :Files, :Include Components, :More Processing, :Output Destination, :Patches,:Stop If Found, :Systems

:Branch Reserved for future use.

:Files The pathnames of the files to be searched. The default is the usual file default. You must specify either this keyword or :Systems.

:Include Components
{Yes, No} Whether to include component systems of systems specified by the :Systems keyword. The default is No. The mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Patches Whether to include patch files for systems specified by the :Systems keyword. The default is Yes.

:Stop If Found {Yes, No} Whether to stop searching when the first occurrence of *string* is found. The default is No, the mentioned default is Yes.

:Systems The systems or subsystems in which to search. You must specify either this keyword or :Files.

Examples:

```
Find String (string(s) to search for) defun (keywords) :Files
  (the pathnames of one or more files [default equus:>ed>*.*.*]) equus:>ed>*.lisp
```

```
equus:>ed>flow-chart.lisp
Line #261:  (defun draw-arrow (c-points window)
Line #523:  (defun find-box (name)
Line #528:  (defun connect-boxes (name1 name2 &optional (arrow-p t) text)
Line #532:  (defun create-boxes ()
Line #625:  QUOTE, SETQ, LET and DEFUN.  Special operators
Line #1070: (defun connect-eval-boxes ()
Line #1107: (defun connect-special-boxes ()
Line #1182: (defun get-user-response (test box-list)
Line #1201: (defun print-results (form-and-args)
Line #1222: (defun introduction ()
Line #1269: (defun var-exists (var env)
Line #1279: (defun var-value (var env)
Line #1287: (defun set-var (var value which-env local-env)
Line #1307: (defun update-env-windows (env)
Equus:>ed>scoreboard.lisp.22
Line #34:   (defun draw-team-box (x y)
Line #45:   (defun draw-score-box (x y)
Line #55:   (defun batter ()
Line #65:   (defun count-balls ()
Line #74:   (defun count-strikes ()
Line #81:   (defun count-outs ()
Line #92:   (defun new-batter()
Equus:>ed>zwei-commands.lisp.1
Line #27:   (defun CENTER-MOUSE ()

22 lines matched in 3 files (5 files searched)
Find String (string(s) to search for [default "defun"]) rotate (keywords) :Systems
  (one or more system or subsystems [default SITE]) Fed

SYS:IO1;FED.LISP.NEWEST
Line #61:   V   Set sample string  ⊕  Rotate character
Line #130:                        #/⊕ :COM-ROTATE-CHARACTER-RIGHT
Line #167:          ("Rotate"
Line #169:           ((NIL :VALUE :COM-ROTATE-CHARACTER-LEFT)
Line #170:            (NIL :VALUE :COM-ROTATE-CHARACTER-180)
Line #171:            (NIL :VALUE :COM-ROTATE-CHARACTER-RIGHT))
Line #172:           :DOCUMENTATION "Rotate the character:
                             L: 90 left, M: 180, R: 90 right")
Line #2882: (DEFMETHOD (:COM-ROTATE-CHARACTER-RIGHT BASIC-FED)
                       (IGNORE &AUX (OLD-PLANE PLANE))
Line #2899: (DEFMETHOD (:COM-ROTATE-CHARACTER-LEFT BASIC-FED)
                       (IGNORE &AUX (OLD-PLANE PLANE))
Line #2915: (DEFMETHOD (:COM-ROTATE-CHARACTER-180 BASIC-FED)
```

```
                      (IGNORE &AUX (OLD-PLANE PLANE)
SYS:IO1;FNTCNV.LISP.NEWEST
Line #68:   ; ROTATE-FONT[-DESCRIPTOR], makes R(otated) fonts.
              I.e. landscape from portrait.
Line #658:  (DEFUN ROTATE-FONT-DESCRIPTOR (FD &AUX LENGTH NFD)
Line #672:      DO (ASET (ROTATE-CHAR-DESCRIPTOR CD) NFD CH))
Line #675:  (DEFUN ROTATE-CHAR-DESCRIPTOR (CD)
Line #688:  (DEFUN ROTATE-FONT (FONT-SYMBOL &AUX FD NFD NFS)
Line #690:      NFD (ROTATE-FONT-DESCRIPTOR FD)

16 lines matched in 2 files (6 files searched)
```

Find Symbol **Command**

Find Symbol *substring keywords*

Finds all symbols whose names contain *substring*.

substring The string to look for in symbol names.

keywords :More Processing, :Output Destination, :Packages, :Search Inherited Symbols, :Search Packages Used By, :Search Packages Using, :Type, :Verbose

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Packages {all, *package-name,* ...} The package(s) to search for symbols. The default is the current package.

:Search Inherited Symbols
{Yes, No} Whether to search imported symbols in the chosen packages. The default is Yes.

:Search Packages Used By
{Yes, No} Whether to search packages used by the chosen packages. The default is No. The mentioned default is Yes.

:Search Packages Using
{Yes, No} Whether to search packages that use the chosen packages. The default is No. The mentioned default is Yes.

:Type {All-Types, Class, Flavor, Function, Presentation-Type, Resource, Unbound, Variable} The type of symbols for which to look. The default is All-Types.

:Verbose {Yes, No} If verbose is Yes, the names of all packages searched are printed, whether matching symbols are found in them or not. If verbose is No, only the names of those packages where matching symbols are found are printed. The default is No, and the mentioned default is Yes.

Flush Commands

Flush Process Command

Flush Process *process*

Causes *process* to go into the state `Wait Forever`. This is one way to stop a runaway process that is monopolizing your machine and not responding to any other commands. A process that has been flushed can be looked at with the Debugger or Inspector and can be reset.

process A process. You can press `HELP` for a list of all the processes currently running in your environment. (See the section "Show Processes Command".)

Format Commands

Format File **Command**

Format File *pathname keywords*

Displays the contents of one or more files in formatted form, using the Document Examiner formatter.

pathname The file or sequence of files to be formatted.

keywords :Destination, :Page Headings

:Destination {screen, *printer*} The destination for formatted output. The default is screen. The mentioned default is the default hardcopy text printer (the value of **hardcopy:*default-text-printer***) if

that printer can handle formatted output; otherwise, the mentioned default is the last printer used to produce formatted output; otherwise, there is no mentioned default.

:Page Headings {Yes, No} Whether to print a heading line at the top of the page. The default is yes. The heading line consists of the word Page followed by the page number.

Format File is similar to the Format File (m-X) Zwei command. The file to be formatted must be a text file. It can contain the same formatting commands and environments as files acceptable to Format File (m-X). See the section "Formatting Text in Zmacs".

For example, to display a file in formatted form on the screen you might do the following:

```
Format File (file) acme-blue:>project>documentation.mss
```

To send the same file to a printer you might do:

```
Format File (file) acme-blue:>project>documentation.mss
(keywords) :Destination (a destination for formatted
output [default Our-Printer]) Our-Printer
```

Format Local SCSI Tape **Command**

Format Local SCSI Tape *keywords*

Formats a QIC-100 tape on a local SCSI drive. The drive must be local.

keywords: :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

If your machine has only one SCSI drive, Format Local SCSI Tape formats the tape in that drive. If you have more than one drive on your machine, a menu pops up that allows you to specify the drive to use.

If you do not want to bother with formatting tapes, you can purchase preformatted Gammamat 3M DC2000 mini-data cartridge (QIC-100) tapes for use with XL400 and MacIvory systems with QIC-100 tape drives.

Format Topic **Command**

Format Topic *documentation-topic keywords*

Runs the formatter on the specified topic.

documentation-topic The topic to format. A topic is the title of any documented section in any loaded documentation system, that is any topic in the candidates or bookmark panes in Document Examiner.

keywords: :Destination

:Destination {*printer*, Screen} Where to display the formatted topic. The default is the default text printer.

Halt **Commands**

Halt commands shut down some activity in such a way that you can resume it.

Halt GC **Command**

Halt GC

Turns ephemeral and dynamic garbage collection off.

Halt Machine **Command**

Halt Machine

Stops execution of Lisp and gives control to the FEP. You can now enter FEP commands, for example, to warm or cold boot the machine.

Halt Printer **Command**

Halt Printer *printer keywords*

Halts the specified printer. The printer does not print any requests until you start it using the "Start Printer Command".

printer The name of the printer you are halting.

keywords :Confirm, :Disposition, :Reason, :Urgency

:Confirm {Yes, No} Whether to ask for confirmation. The default is Yes.

:Disposition {Delete, Hold, Restart} Enables you to specify whether the system delete, hold, or restart the print request. The Delete option deletes the request from the queue. Hold retains the request in the queue but does not print it when the printer

restarts. Restart automatically prints the interrupted request from the beginning when the printer restarts. The default is Hold.

:Reason — {*string*} Enables you to specify the reason for the shutdown. This option appears in the display of the Show Printer Status command and in the Print Spooler log. The default message is "Printer *printer-name* being reset by *user-id*."

You can use the following keyword to control precisely when the printer halts:

:Urgency — {Asap, After-Current-Request, After-Next-Copy} Enables you to specify when the printer halts. Asap indicates that the printer halt and reset immediately. After-Current-Request indicates that printer halt after the current request finishes printing. After-Next-Copy indicates that the printer halt after the next copy of the request prints. After-Next-Copy is the same as After-Current-Reqest when the request is only for one copy. The default is Asap.

When you halt the printer, the Print Manager process enters a suspended state until you start it again.

Note that you can halt, start, or reset a spooled printer from any machine on the network.

Halt Process Command

Halt Process *process*

Causes *process* to stop immediately. This is the same as [Arrest] in the Peek processes menu.

process — A process. You can press HELP for a list of all the processes currently running in your environment. (See the section "Show Processes Command".)

Hardcopy **Commands**

Hardcopy File **Command**

Hardcopy File *pathname printer keywords*

Sends a file to a hardcopy device.

pathname — The pathname of the file you are printing. You can specify more than one file by separating the pathnames with commas.

printer The printer on which you are printing the file. The default is the default hardcopy text printer (the value of **hardcopy:*default-text-printer***).

keywords :Body Character Style, :Copies, :Delete, :Ending Page, :File Types, :Heading Character Style, :Notify, :Orientation, :Print Cover Page, :Running Head, :Starting Page, :Title

:Body Character Style
The character style used for printing the text of the file. You can also use this character style for merging character styles in the file. See the section "Understanding Character Styles".

:Copies {*number*} The number of copies you are printing. The default is 1.

:Delete {Yes, No} Enables you to specify whether the file deletes after printing. The default is No, not to delete.

:Ending Page {*number*} The last page you are printing. This command defaults to the last page of the file. Note that the page character or form feed character identifies page breaks. The system treats text files without page breaks as a single page, although the file can use many sheets of paper.

Press format files contain form feeds or PAGE characters. Remember that these are physical pages and do not necessarily correspond to the page numbering appearing in the heading. For example, the first page of a press file is a title page, the second page is numbered *i*, and the third page is numbered 1.

:File Types {ASCII, DMP1, LaserWriter, PostScript, Press, Text, XGP, use-canonical-type} The internal format of the contents of the file, to interpret for printing. The default is use-canonical-type, meaning that the type is determined from the extension to the file name.

:Heading Character Style
The character style used for the running head.

:Notify {Yes, No} Specifies whether to send a notification upon job completion. The default is Yes.

:Orientation {Landscape, Portrait} Specifies the paper orientation for the output. Portrait is left to right across the short dimension of the paper. Landscape is left to right across the long dimension of the paper. The default is Portrait.

:Print Cover Page {Yes, No} Specifies whether to print a cover page. The default is Yes.

:Running Head {None, Numbered} Specifies the type of running head that prints on the top of each page. The default is Numbered.

:Starting Page {*number*} The first page you are printing. This command defaults to the first page of the file. The page character or form feed character identifies page breaks. The system treats text files without page breaks as a single page, although the file can use many sheets of paper.

Press format files contain form feeds or PAGE characters. Remember that these are physical pages and do not necessarily correspond to the page numbering appearing in the heading. For example, the first page of a press file is a title page, the second page is numbered *i*, and the third page is numbered 1.

:Title {*string*} Specifies the title appearing on the cover page identifying the output. The default is "File: *pathname-you-specified* ".

Hardcopy System **Command**

Hardcopy System *system printer keywords*

Prints the source file(s) for *system* on *printer*.

system The system whose source to print.

printer A supported hardcopy device. The default is the value of **hardcopy:*default-text-printer***.

keywords :Body Character Style, :Copies, :Heading Character Style, :Include Components, :Orientation, :Query, :Running Head, :Silent, :Title, :Version

:Body Character Style
A fully specified character style. The character style in which to print the source file(s). See the section "Understanding Character Styles".

:Copies {*Number*} The number of copies to make. The default is 1.

:Heading Character Style
A fully specified character style. The character style in which to print headings. See the section "Understanding Character Styles".

:Include Components
{Yes, No} Whether to include component systems. The default is No, the mentioned default is Yes.

:Orientation {Landscape, Portrait} Orientation on the paper for the output. Portrait is left to right across the short dimension of the paper. Landscape is left to right across the long dimension. The default is Portrait.

:Query {Everything, Yes, Confirm-Only, No} Whether to ask about hardcopying each file. The default is No. The mentioned default is Everything or Yes, which are the same. If you select :Query Yes, you are asked about each file. Then a list of all the files you have selected is displayed and you are asked to confirm that you want to hardcopy those files. If you select :Query Confirm-Only, you are only asked to confirm the list of files. You can answer Yes, No, or Selective. Selective allows you to answer Yes or No to each file in the list and finally displays a new list for you to confirm.

:Running Head {None, Numbered} Type of running head on each page. The default is Numbered.

:Silent {Yes, No} Whether to suppress all terminal output. The default is No, the mentioned default is Yes.

:Title {*string*} Title to appear on the cover page. The default is "System: *system-you-specified* ".

:Version {Released, Latest, Newest, *version-designator*} The major version of the system to hardcopy. The default is Newest.

Help **Commands**

Help **Command**

Help *commands keyboard keywords*

Displays a list of Command Processor commands.

keyboard Specifies the Keyboard help category.

commands Specifies the Commands help category.

keywords :Format, :Command Table

:Format {Brief, Full, Detailed} The level of detail to show in the list. The default is Brief. This is the same as pressing the HELP key.

Brief means that only unique command names are shown in full; for the rest, the verb is shown with the number of commands that start with that verb.

Full shows all commands in abbreviated form.

Detailed displays the entire list of available commands.

:Command Table The command table to show.

Initialize Commands

Initialize Demonstration Command

Initialize Demonstration *name keyword*

Does any pre-loading specified in the demonstration definition, such as loading any required systems and running the initializer function. This command is not functionally necessary (because when a demonstration is first run, it is initialized automatically, if it has not been initialized yet), but it can save time in situations when you want to have the first run of a demonstration be fast.

name	The name of a demonstration.
keywords	:Force
:Force	{Yes, No} Specifies whether to force initialization even if it has already been done. (Even when this is Yes, required systems are not re-loaded—only the initializer action is re-run).

Initialize Mail Command

Initialize Mail

Reloads Zmail without disturbing the rest of the system. The state of your mail on the machine is lost, so you should only use this when your Zmail process is irreparably stuck.

Initialize Mouse Command

Initialize Mouse

Restarts the mouse process. This is useful if some window problem has caused the mouse to stop working.

Inspect Commands

Inspect Command

Inspect *object keywords*

Displays the components of *object*. This is similar to Show Object but it uses the Inspector, a window-oriented program for showing data structures. It allows you to do something to that object, such as inspect it, modify it, or give it as the argument to a function. You exit from the Inspector by clicking the mouse on [Exit] in the Inspector menu.

object Any Lisp object. The default is "unspecified".

See the section "The Inspector".

Kill Commands

Kill commands remove information from the world. You should use such commands with caution.

Kill Process Command

Kill Process *process*

Causes *process* to go away completely.

process A process. You can press HELP for a list of all the processes currently running in your environment. See the section "Show Processes Command".

List Commands

List commands display summary information.

List Sage Variables Command

No documentation available for section List Sage Variables Command.

Load Commands

Load File Command

Load File *pathname keywords*

Loads files into the current world.

pathname The pathname of the file to load. More than one pathname may be specified, separated by commas. The default is the usual file default.

keywords :Compile, :More Processing, :Output Destination, :Package, :Query, :Silently

:Compile {Never, New-Source, Always} The default is Never. The mentioned default is New-Source.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Package {*package-name*} The package into which to load the file. The default is the file's "home" package, as specified in its attribute list.

:Query {Yes, No, Ask} Whether to ask for confirmation before loading each file. The default is No. The mentioned default is Yes.

:Silently {Yes, No} Whether to print a line as each file is loaded. The default is No. The mentioned default is Yes.

Load HackSaw File Command

Load HackSaw File *pathname*

Loads the standard HackSaw file plus all the ones defined by **approach:*hacksaw-extension-files***. Loads HackSaws from private HackSaws files you designate.

pathname {*pathname*, All} The pathname(s) of files to be loaded. The default is All. If you give a pathname or pathnames, only those files are loaded.

This command is optional and exists to avoid delays and to load personal HackSaw files. It is designed for use from an init-file with **cp:execute-command**. The standard HackSaw files load themselves the first time Show HackSaw or m-HELP is used.

Load Patches **Command**

Load Patches *system keywords*

Loads patches into the current world for all systems, locally maintained systems, or the indicated systems.

system {All Local *system-name1, system-name2* ... } The system(s) for which to load patches. The default is All.

keywords :Dangerous Patch Action, :Excluding, :Include Components, :More Processing, :Output Destination, :Query, :Save, :Show

:Dangerous Patch Action
{Skip, Query, Load} Whether to skip loading *dangerous* patches, that is, patches that might make data structures in your world inconsistent, causing unexpected behavior. The default is Skip.

:Excluding {*System(s)*} Excludes loading patches for these systems.

:Include Components
{Yes, No} Whether to load patches for any component systems. The default is No. The mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No, Ask} Yes asks for confirmation before beginning the load patches process and again before loading each patch. Ask asks whether or not it should query before each patch, and then for confirmation before beginning the load patches process. The default is No. The mentioned default is Yes.

:Save {*pathname*, Prompt, No-Save} The file in which to save the world with all patches loaded. Omitting this keyword means do not save the world. The mentioned default is Prompt, which means save the world and then prompt for a pathname.

:Show {Yes, No, Ask} Whether to print the patch comments as each patch is loaded. The default is Yes.

See the function **load-patches**.

Load System **Command**

Load System *system keywords*

Loads a system into the current world.

system Name of the system to load. The default is the last system loaded.

keywords :Component Version, :Condition, :Include Components :Load Patches, :More Processing, :Output Destination, :Query, :Redefinitions Ok, :Silent, :Simulate, :Version

:Component Version
{Released, Latest, Newest, *version-designator*} The version of any component systems to load. Released means the version designated as released in the journal file. Latest means the most recent version recorded in the journal file. Newest means to ignore the versions in the journal file and just find the newest files. The default is the version with which the system was compiled.

:Condition {Always, Never, Newly-Compiled} Under what conditions to load each file in the system. Always means load each file. Newly-compiled means load a file only if it has been compiled since the last load. The default is Newly-Compiled.

:Include Components
{Yes, No} Whether to load component systems. The default is Yes.

:Load Patches {Yes, No} Whether to load patches after loading the system. The default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Everything, Confirm-only, No} Whether to query before loading. Everything means query before loading each file. Confirm-only means create a list of all the files to be loaded and then ask for confirmation before proceeding. No means just go ahead and load the system without asking any questions. The default is No. The mentioned default is Everything.

:Redefinitions Ok {Yes, No} Controls what happens if the system asks for confirmation of any redefinition warnings during the loading process. Yes means assume that all requests for confirmation are answered yes and proceed. No means pause at each redefinition and await confirmation. The default is No. The mentioned default is Yes. This allows you to start loading a system that you know will take a long time to load and leave it to finish by itself without interruption for questions such as "Warning: *function-name* being redefined, ok? (Y or N)".

:Silent {Yes, No} Whether to turn off output to the console while the system is loading. The default is No. The mentioned default is Yes.

:Simulate {Yes, No} Print a simulation of what compiling and loading would do. The default is No. The mentioned default is Yes.

:Version {Released, Latest, Newest, *version-designator*} Which version number to load. Released means the version designated as released in the journal file. Latest means the most recent version recorded in the journal file. Newest means to ignore the versions in the journal file and just find the newest files. The default is Released.

Note: This command only loads a system. If you want to compile and load a system, see the section "Compile System Command".

Login and *Logout* Commands

Login Command

Login *user keywords*

Logs the *user* into Genera.

user Any string. Your user ID.

keywords :Host, :Init File

:Host {Local, *any-host-name*} A particular host computer. Local as an argument to :Host is particularly useful if your namespace system is down and you wish to log in to your Machine without having it try to use the namespace database. The default comes from `home host` for your user object in the namespace database.

:Init File {Default-Init-File, File, Pathname, None} Whether to load your init file. The default is Default-Init-File. To avoid loading your init file, use :Init File None.

If someone is already logged in when you give the Login command, that user is logged out. If this happens, you see the message

```
Warning -- You are logging out from program-name
```

.

Logout **Command**

Logout *keywords*

Logs you out of Genera.

keywords :Expunge Mail, :Save Buffers, :Save Mail, :Save Spell Dictionaries,

:Expunge Mail {Yes, No} Whether to expunge modified mail file buffers before saving them to disk. The default is Yes.

:Save Buffers {Yes, No, Ask} Whether to write out modified editor buffers to disk. The default is Ask.

:Save Mail {Yes, No, Ask} Whether to write out modified mail buffers to disk. The default is Ask.

:Save Spell Dictionaries
{Yes, No, Ask} Whether to write out modified spell dictionaries to disk. The default is Ask.

Monitor **Commands**

Monitor Screen Command

Monitor Screen *host*

Allows viewing the actual contents of some other lisp machine screen. It is useful for observing the state of file servers from afar, and for helping debug someone else's machine when it is not practical to get to the machine itself.

host The name of a cooperating host. **Note**: The person whose screen is being monitored must explicitly turn on the server to allow someone else to see their screen.

Monitoring is turned off by default. The person whose screen is being monitored must explicitly turn on the server to allow someone else to see their screen. They can choose to disallow remote viewing, allow remote viewing with a warning notification, or allow remote viewing with no warning. They may also choose to make the operation of the server visible in the wholine, or to make it invisible. (The latter is useful for watching file servers, when it is important to see something in

the wholine area such as open files, or some other progress indication that appears there. See also the "Show Open Files Command".)

If the machine is not logged in, you can always monitor its screen. If the machine is logged in, however, monitoring is disallowed by default. You enable monitoring with the function **chaos:enable-monitor-screen-server**.

chaos:enable-monitor-screen-server &key *(:server-enabled* **:no***) (:wholine-appearance* **:visible***)* *Function*

Allows someone else to monitor your screen using the Monitor Screen CP command. Calling this function without any arguments disables monitoring your screen (the default). Note that, if your screen is already being monitored, the current monitoring session will *not* be terminated.

:server-enabled One of **:yes**, **:no**, or **:notify**. The default is **:no**. **:no** means no one can monitor your screen. **:yes** means anyone can monitor your screen, and you will not see a notification that they are doing so. **:notify** means that anyone can monitor your screen, but you will see a notification first when they start.

:wholine-appearance
One of **:visible** or **:invisible**. The default is **:visible**. When **:server-enabled** is **:yes** or **:notify**, **:visible** means that the wholine area will display the fact that the server is operating if someone is monitoring your screen. **:invisible** means that the wholine area will not note whether anyone is monitoring your screen.

Note that the server side of this capability has existed for many releases, though no easily-accessible user side existed. Even then, the default for the server was to disallow access.

Site administrators, and others who build worlds, should *not* enable this facility in their world-building scripts, since allowing arbitrary monitoring of others' screens is generally considered a serious invasion of privacy. Users may wish to ensure that the worlds they run do not have this turned on by default.

This is a Chaos-only protocol, meaning that only Chaosnet users can monitor screens. This means that sites which communicate with the outside world via TCP/IP, rather than Chaos (that is essentially all sites) do not have to worry about users elsewhere on the Internet monitoring any screens at their site, regardless of whether individual users enable monitoring.

Many of these capabilities are not necessary if the machine(s) of interest run some remote window system, such as the X Window System.

Screen monitoring only works if both machines are *not* embedded machines (for example, they are *not* UX-family or MacIvories). Both machines must have screens which are only one bit deep, that is, color or gray scale screens cannot be used.

To monitor a screen, you do something like the following:

```
Monitor Screen (of cooperating host [default CARDINAL]) NUTHATCH
```

To stop monitoring a screen, type any character.

Monitor System Status **Command**

Monitor System Status *keywords*

Displays a summary of your machine's status. This includes the processes that are running, any open files, and whether or not you have new mail.

keywords :Minimum Utilization, :Truncate Long Fields

:Minimum Utilization
{*percentage*} A process using less than this percentage of the machine should not be included in the summary. The default is 0.1.

:Truncate Long Fields
{Yes, No} Whether to inhibit truncation of long field contents. The default is, Yes, to truncate long process names or states.

Monitor System Status is useful when you are using a remote terminal, where Peek is not available.

Monitor Variable Command

Monitor Variable *symbol keywords*

Monitors the access of a special variable. This command arranges for a trap to be signalled when any process accesses the monitored location. This command is used to answer the question: "What program or process is reading or writing this location in memory?". This is particularly useful when there are several processes sharing some data structures, and you want to debug the interactions between the processes.

symbol The name of a symbol whose location in memory you want to monitor. Enter the name of a symbol and, optionally, its Value-Cell or Function-Cell. (See the :Cell keyword description below.)

keywords :Boundp, :Cell, :Locf, :Makunbound, :Read, :Write

:Boundp {Yes, No} Monitors the location for **boundp** operations. The default is No. The mentioned default is Yes.

:Cell {Value-Cell, Function-Cell} Specifies the cell that you want to monitor within the location. The Debugger gives you two choices: Value-Cell or Function-Cell. The default is Value-Cell.

:Locf — {Yes, No} Monitors the location for **locf** operations. (Default is No.)

:Makunbound — {Yes, No} Monitors the location for **makunbound** operations. The default is No. The mentioned default is Yes.

:Read — {Yes, No} Monitors the location for reads. The default is No. The mentioned default is Yes.

:Write — {Yes, No, Change} Monitors the location for writes. The default is Yes.

Suggested mouse operations

- To monitor a location: Point the mouse at a locative, structure slot, or instance variable and press c-m-sh-Left.
- To unmonitor a location: Point the mouse at a locative, structure slot, or instance variable that was previously monitored and press c-m-sh-Middle.

Mouse Handler Commands

The following Command Processor commands are available to provide information about mouse handlers defined in your world.

Show Handlers For Types Command

Show Handlers for Types *context-type presentation-type keywords*

Lists the mouse handlers applicable in a specified input context to a specified object type and presentation type.

context-type — The input context (*to-presentation-type* in the handler definition).

presentation-type — The presentation type of the presentation to which the mouse is pointing (the *from-presentation-type* in the handler definition).

keywords — :More Processing, :Output Destination

:More Processing — {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Mouse Handlers Command

Show Mouse Handlers *handler-name keywords*

Lists the handlers corresponding to *handler-name*, showing the input context, presentation type, and handler type (object type) to which each is applicable.

handler-name The name of a mouse handler.

keywords :More Processing, :Output Destination, :Show Defined Handlers, :Show Table Handlers

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Show Defined Handlers
{Yes No} Whether to list the handlers as described above.

:Show Table Handlers
{Yes No} Whether to show the expansion of the handlers in the handler lookup table.

Show Presentation Handlers From Type Command

Show Presentation Handlers From Type *type keywords*

Lists the handlers defined with *type* as their *from-presentation-type*.

type A presentation type.

keywords :Include t, :More Processing, :Output-Destination

:Include t {Yes No} Whether to include in the listing handlers defined on the **t** presentation type.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Mount Commands

Mount Optical Disk Command

Mount Optical Disk *SCSI-address*

Mounts the optical disk indicated by *SCSI-address*. Once the disk is mounted, if it has a FEP filesystem on it, you can access it by using its FEP unit number. Otherwise, you should use the Create Initial FEP Filesystem command.

SCSI-address The SCSI address of the optical disk drive.

The convention for mapping SCSI addresses to FEP unit number is that the FEP unit number is 7 greater than the SCSI address. For example, if you set up the optical disk drive at SCSI address 4, the FEP unit is FEP 11. (Note that if you have multiple I/O boards, and the optical disk drive is attached to the second I/O board, then the FEP unit is 23 plus the SCSI address.)

Optimize Commands

Optimize World Command

Optimize World *keywords*

Optimizes the world that is currently loaded into your enviroment by reorganizing the world to improve paging performance. Use this command after you load site-specific software or layered products into a distribution world, and before you save it.

Note: If you optimize an IDS world whose parent world was not (but should have been) optimized, your IDS world will be significantly larger than neccessary.

Use the Optimize World command on IDS worlds only if:

- The parent world was optimized, or
- The parent world didn't need optimization.

If you follow this rule, optimization will not increase the size of your IDS world.

Since distribution worlds are optimized before leaving Symbolics, if you site-configure a distribution world (but do not load any systems) you need not optimize it again. (This saves you some time in your world-building process.)

When you enter the Optimize World command, you are prompted for confirmation. Once the program has finished, a message appears. Optimization typically takes about one-half hour to execute, but this time period can vary according to the size of the world load and the total amount of main memory that's available when you execute the command.

During the time that Optimize World is running your machine does not respond to the network or keyboard; you cannot use your machine while optimization is in process.

keywords :Show

:Show Displays the progress of the optimization process on the screen.

Read Commands

Read Carry Tape Command

Read Carry Tape *keywords*

Loads files from a Carry-Tape. The loader makes no attempt to copy any file properties, including author and creation date. It copies only file contents, and provides reasonable defaults for the target file name. See the section "The Carry-Tape Dumper".

keywords :If Exists, :More Processing, :Output Destination, :Tape Spec

:If Exists {Error, Skip, Supersede} Whether to Error, Skip, or Supersede files that are already on disk. The default is Skip.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is

subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window}
Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Tape Spec The host and device for the tape. For more information, see the section "Tape Specifications".

Read Image File **Command**

Read Image File *pathname format keywords*

Reads a picture or bitmap image from a file.

pathname The pathname of the file from which to read an image.

format The format of the data in the file. These are the possible image file formats:

BIN Symbolics binary file (data only, compatible with all architectures)

Compact Bitmap
X windows "compact" bitmap

EPS/Macintosh
Postscript in the data fork, PICT in the resource fork

Gem IMG Used by Ventura Publisher

GIF CompuServe Graphics Interchange Format

Macintosh
Macintosh based on type

MacPaint MacPaint canvas (fixed size)

PC Paintbrush
PC Paintbrush format

PICT Macintosh picture record

Portable Bitmap
X windows "portable" bitmap (1's and 0's)

PostScript
Reads any PostScript images, writes EPSF compatible

TIFF Tagged Image File Format

Truevision
Truevision product family format

Utah RLE
University of Utah format Run-length-encoding

X Bitmap X windows bitmap (C code)

X Window Dump
X v.11 window dump

Xim X Image

keywords :More Processing, :Output Destination, :Rotate, :Trim

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Rotate {Yes, No} Rotate the image 90 degrees (useful for MacPaint screen images).

:Trim {Yes, No} Return the smallest image necessary to store just the ink.

Read TAR File Command

Read TAR File *tarfile root-directory-for-relative-pathnames keywords*

Reads a file in TAR format.

tarfile {*a pathname*} The TAR file to read.

root-directory-for-relative-pathnames
{*a directory*} The directory in which to put the files read.

keywords :Mode, :More Processing, :Output Destination, :Reroot Absolute Pathnames

:Mode {Binary, Query, Text} The mode in which to perform the copy. The default is Query, ask about each file. Binary means binary bytes with no character set translation. Text means text characters with UNIX character set translation.

:More Processing
{Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams.

The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Reroot Absolute Pathnames
{Yes, No} Uses the specified root directory as the "root" for absolute pathnames. The default is No. The mentioned default is Yes.

Read TAR Tape **Command**

Read TAR Tape *root-directory-for-relative-pathnames keywords*

Reads a tape in TAR format. It prompts for the tape specification.

root-directory-for-relative-pathnames
{*a directory*} The directory in which to put the files read from the tape.

keywords :Mode, :More Processing, :Output Destination, :Reroot Absolute Pathnames

:Mode {Binary, Query, Text} The mode in which to perform the copy. The default is Query, ask about each file. Binary means binary bytes with no character set translation. Text means text characters with UNIX character set translation.

:More Processing
{Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Reroot Absolute Pathnames
{Yes, No} Uses the specified root directory as the "root" for absolute pathnames. The default is No. The mentioned default is Yes.

Remove Commands

Remove Service**s** From Hosts **Command**

Remove Services from Hosts *service-triples specific-hosts-or-all keywords*

Removes the specified service attributes in the namespace for one or more hosts.

service-triples	A service triple consists of a service, a medium, and a protocol. For more information, see the section "Concepts of Service, Medium, and Protocol".
specific-hosts-or-all	A list of hosts (or "all") from which you want to remove service. If you specify all, use the :Namespace, :Site, and :Type keywords.
keywords:	:Locally, :Verbose, :Namespace, :Site, :Type
:Locally	{Yes, No} The initial default is No, which updates the global namespace database (the namespace server) to reflect the changes you have made. Answer Yes to update only the local Lisp world with the changes you have made; the changes will be "forgotten" when you cold boot the local machine (unless you save the changed world). The default for this keyword is displayed along with the object's name, and, if you are editing multiple objects in the Namespace Editor, the current setting becomes the default for subsequent objects.
:Verbose	Print messages for each host modified.
:Namespace	Add the services to all hosts in the namespace (use with the "all" argument).
:Site	Add the services to all hosts in the site (use with the "all" argument).
:Type	Add the services to all hosts of this system type (use with the "all" argument).

Here is an example of removing netboot service from a host:

```
Remove Services From Hosts (Service Triples (service) netboot
(medium) slap (protocol) netboot
(A sequence of hosts or All) HARPAGORNIS

Removing service NETBOOT SLAP NETBOOT from hosts.
Done.
```

Rename Commands

Rename File Command

Rename File *from-file to-file keywords*

from-file The pathname of the file to be renamed. The default is the usual file default.

to-file The new pathname. The default is the usual file default.

keywords :Create Directories, :More Processing, :Output Destination, :Query

:Create Directories
{Yes, Error, Query} What to do if a destination directory does not exist. The default is Query.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No, Ask} Whether to ask for confirmation before renaming. The default is No.

Report Commands

Report Bug Command

Report Bug *system name*

Sends a bug report. It starts up a bug mail window with the message header `To: BUG-`*system name*. If *system name* is omitted, `BUG-LISPM` is used. Information from your herald identifing what version of the software you are using is inserted at the beginning of the message. You can now type in your bug report and send the message.

You can control the character style of the herald information by setting the value of **dbg:*character-style-for-bug-mail-prologue***. The default value is NIL.NIL.TINY. See the variable **dbg:*character-style-for-bug-mail-prologue***.

Reset Commands

Reset Network Command

Reset Network

Turns your network interfaces off and back on again, and also resets the namespace system. Turning the interfaces off and on can be useful if your network connections appear to be stuck and nothing is being transmitted or received. Resetting the namespace system can cure problems caused by information related to the namespace having been corrupted on your local machine. One symptom of that is your host repeatedly trying to connect to another host, without success.

The Reset Network command is the same as entering **(neti:general-network-reset)** followed by **(neti:enable)**.

Reset Printer Command

Reset Printer *printer printer-request keywords*

Resets a spooled printer. You can use this command if your printer stops operating or if you have to stop the printer in the middle of a job. This command reestablishes communications with the printer and, when possible, sends a software "reset" command to the printer.

printer	The name of the printer you are resetting.
printer-request	(Optional) Resetting a printer in the middle of a print request results in the system displaying the request and asking you to confirm the reset command. You have to specify the name of the print request or select the print request with the mouse from the display of the Show Printer Status command in order to reset the printer immediately. For more information, see the section "Show Printer Status Command".
keywords	:Confirm, :Disposition, :Reason

:Confirm — {Yes, No} Whether to request confirmation that the printer is printing a request. The default is Yes.

:Disposition — {Delete, Hold, Restart} Enables you to specify whether the system delete, hold, or restart the print request. The Delete option deletes the request from the queue. Hold retains the request in the queue but does not print it when the printer restarts. Restart automatically prints the interrupted request from the beginning when the printer restarts. The default is hold.

:Reason — {*string*} The reason you are resetting the printer, which is added to the printer log. The default message is "Printer *printer-name* being reset by *user-id*."

Note that you can halt, start, or reset a spooled printer from any machine on the network.

Restart **Commands**

Restart Printer Request **Command**

Restart Printer Request *printer printer-request keywords*

Restarts a print request that you stopped before completion. For requests currently printing, the printer resets and the request prints from the beginning. For requests on hold, the printer requeues the request.

printer-request — A string specifying the request. You can select the print request with the mouse from the display of the Show Printer Status command. You can also specify the printer name to which you sent the request and use "c-? and c-/" for displaying all requests. For more information, see the section "Completion in the Command Processor".

keywords — :Extent, :Starting From

:Extent — {Entire, Copy} The extent to which you are restarting a request. Entire refers to the whole request. Copy refers to a single copy.

:Starting From — {*number*} The number of the copy after which the printer restarts. Note that you cannot use this keyword if you used the :extent keyword specifying Entire. The default is 0. *This option is currently not used.*

Restart Process Command

Restart Process *process*

Causes the process to start over in its initialized state. This is one way to get out of stuck states when other commands do not work.

process A process. You can press HELP for a list of all the processes currently running in your environment. See the section "Show Processes Command".

Restore Commands

Restore Distribution Command

Restore Distribution *keywords*

Restores data from tape to disk. This command reads the directory listing of the files on tape, and then restores the files according to the pathname and property information on the tape. For each file it restores, Restore Distribution checks to see if the file already exists in the file system. If it does not exist, it restores the file. If the file does exist, the default behavior is that Restore Distribution states that the file already exists in your file system and skips the file.

keywords :Menu, :More Processing, :Output Destination, :Skip Restoration, :Use Disk

:Menu {Yes, No} Whether to use a menu interface to specify details of the restoration. Choosing Yes presents a Restore Distribution frame to select which files are restored. The default is No. The mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Skip Restoration {Yes, No} Whether to skip restoration of files that already exist in the file system. If the file exists in the system and :Skip Restoration is No, Restore Distribution states that the file it was supposed to restore already exists in your file system, and

asks where to restore it instead. You supply a pathname. A pathname of Nowhere means skip this file. The default is Yes. The mentioned default is Yes.

:Use Disk — On all machines other than a MacIvory, the choices are {Yes, No}. If Yes, the input is read from a tape image file on disk. The default is No. The mentioned default is Yes.

On a MacIvory, the choices are {Tape, Disk, Floppy}. Disk indicates the hard disk, and Floppy indicates the floppy disk. These two values also read a tape image file from disk or floppy. Tape means to read from tape; this is the default.

Run Commands

Run Demonstration Command

Run Demonstration *name keyword*

Runs a demonstration. If the demonstration has not been initialized, it will be initialized automatically before it is first run.

name — The name of a demonstration.

keywords — :More Processing, :Output Destination, :Show Instructions, :Specify Options

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Show Instructions {Yes, No, Ask} Specifies whether to show instructions. The default is Yes.

:Specify Options {Yes, No} Specifies whether to interactively prompt for option values (if any) to control the demonstration.

Save **Commands**

Save Compiler Warnings Command

Save Compiler Warnings *pathname files-whose-warnings-to-save*

Save compiler warnings of the files *files-whose-warnings-to-save* to the specified *pathname*. *files-whose-warnings-to-save* can be All to save all warnings, or it can be a list of one or more pathnames. Among the pathnames can be the special token No File to catch warnings for no particular file.

Save File Buffers **Command**

Save File Buffers *keywords*

Saves your modified Zmacs file buffers on disk.

keywords :Query

:Query {Yes, No} The default is Yes, meaning that it asks you about each buffer before writing it out.

Save Mail Buffers **Command**

Save Mail Buffers *keywords*

Saves on disk any modified mail buffers.

keywords :Expunge, :Query

:Expunge {Yes, No} Whether to expunge each buffer before saving. The default is Yes.

:Query {Yes, No} The default is Yes, meaning that it asks you about each buffer before writing it out.

Save World **Command**

Save World (Complete or Incremental) *pathname*

Saves the current world. The system prompts for (Complete or Incremental) if Incremental Disk Save is enabled. Specify Complete to save the entire world, or Incremental (if enabled) to perform an Incremental Disk Save. The default is Complete.

pathname The pathname for the saved world. The default is the FEP file specification for the local machine, based on the version number of the current system and on whether this is a complete or incremental save.

A complete save yields a pathname of Genera-*major-minor* or System-*nnn-mmm*. An incremental save changes this default in the following way:

- "Genera-" is prefaced with "Inc-"
- "System-" is replaced by "Inc-".
- "-from-" is appended to the name.
- A shortened version of the loaded world name (the pathname that appears in the first line of the herald) is appended to the name.
- If the result is longer than 32 characters (the limit of filenames in the FEP file system), the filename is truncated and the last 4 characters within the limit are replaced with "-etc".

More information is available about saving incremental worlds. See the section "Using the Incremental Disk Save (IDS) Facility".

Scan Commands

Scan Mail Command

Scan Mail *inbox keywords*

Allows you to look over your new mail, a message at a time, without loading the file into Zmail. For each message you are shown the headers and offered the rest of the message. You can press `Y` to see the text of the message, `N` to go on to the next message, `R` to reply to the message, or `X` to exit from Scan Mail.

If you press `Y`, the message is displayed. You can stop the display by pressing `ABORT` at a `More` break or by pressing `c-ABORT` at any time. When the display is complete, you are offered the opportunity to reply to the message. You can press `Y` to reply, `N` to move on to the next message, or `X` to exit.

If you press `R`, you are prompted for the headers, and left in a simple input editor context to type your message. Pressing `END` sends the message, `ABORT` aborts the message.

inbox The mailbox to scan. The default is the inbox for your main mail file.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during the output to interactive streams. The default is Default.

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Scan Mail is useful for remote terminal use, where Zmail is unavailable, since it offers your mail one message at a time, with the option of operating on the message (unlike Show Mail, which only displays the contents of your inbox).

Select Commands

Select Activity Command

Select Activity *activity*

Selects *activity* and makes it current.

Activity can be:

Converse	Flavor Examiner	Namespace Editor
Distribute Systems	Frame-Up	Peek
Document Examiner	Inspector	Restore Distribution
Editor	Keyboard Control	Select Key Selector
FEP-Tape	Lisp	Terminal
File Server	Mail	Zmacs
File System Maintenance	Notifications	Zmail

Send Commands

Send Mail Command

Send Mail *recipient keywords*

Prompts for the text of a message and sends it as electronic mail to *recipient*.

recipient One or more electronic mail addresses of the form *user* or *user@host*. Multiple addresses should be separated by commas.

If you supply only *user*, the namespace for your site is searched to locate the proper host for that user's mail.

keywords :cc, :From, :Subject

:cc {*user*, *user@host*} One or more addresses to send a *cc*, carbon copy, of the message.

:From {*user*, *user@host*} The author(s) of the message, if different from the logged-in user.

:Subject {*string*} A line to serve as the subject for the message.

Send Message **Command**

Send Message *recipient*

Sends a Converse message to the specified recipient.

recipient *user* or *user@host*. The person to whom to send the message. If *@host* is omitted, all Symbolics machines on your network are polled to locate *user*.

Send Message prompts for text to send as a Converse message. `END` terminates and sends the message. See the section "Converse".

Set **Commands**

Set commands set status variables and user options.

Set Base **Command**

Set Base *number*

Sets the input and output bases to *number*. See the variable ***print-base***.

number A number in base 10. The default is 10.

Set Boot File World **Command**

Set Boot File World *world-load boot-file keywords*

Updates a boot file to load *world-load*.

world-load {*pathname*} The pathname of a world load.

boot-file {*pathname*} The pathname of the boot file to update.

keywords :Create

:Create — {Yes, No} Whether to create the boot file if it does not exist. The default is No. The mentioned default is Yes.

Set Breakpoint Command

Set Breakpoint *compiled-function pc*

Sets a breakpoint.

compiled-function — The name of a *compiled-function* in which you want to set a breakpoint.

pc — The PC (program counter) at which you want to set a breakpoint.

keywords — :Action, :Conditional

:Action — {Show-All, Show-Args, Show-Locals} Specifies an action to take when the breakpoint is encountered. Show-All: Displays arguments and local variables. Show-Args: Displays arguments and no local variables. Show-Locals: Displays only local variables. Give an *expression* if you want it to be evaluated in the lexical context of the frame. (Default is no action. Mentioned default is Show-All.)

:Conditional — {Always, Mode-Lock, Never, Once} Executes the breakpoint trap according to certain conditions. Always: The breakpoint is always taken. Mode-Lock: The breakpoint is taken only when the MODE LOCK key is pressed. Never: The breakpoint is never taken. Once: The breakpoint is taken only for the first time it is encountered. Give an *expression* if you want it to be evaluated in the lexical context of the frame. (Default is Always.)

Suggested mouse operations

- To set a breakpoint in a compiled function: Display disassembled code with the Show Compiled Code command, point the mouse at a PC, and press c-m-Left.
- To set a breakpoint in a code fragment: Display the code with the Show Source Code command, point the mouse at a code fragment, and press c-m-Left.

Set Command Processor **Command**

Set Command Processor *mode prompt-string keywords*

Sets the mode for the command processor.

mode — {Form-Only, Form-Preferred, Command-Preferred, Command-Only} How the Command Processor treats type-in.

Form-Only	Anything typed is taken as a Lisp form to be evaluated.
Form-Preferred	Anything typed is taken as Lisp forms unless it is preceded by a colon (:), in which case it is taken as a Command Processor command.
Command-Preferred	Anything typed is taken as a Command Processor command, unless it is preceded by a left parenthesis. If it is not a command, but a symbol with a value, it is evaluated. If it is both a command and a symbol with a value, it is treated as a command unless it is preceded by a comma (,) or any nonalphabetic character.
Command-Only.	Anything typed is taken as a Command Processor command.

The default is Command-Preferred.

prompt-string Any string. The default for command-preferred and command-only modes is `Command: `. The default for form-preferred and form-only modes is " ". To include a space in your prompt, you must enclose the string in double quotes. For example:

```
Set Command Processor Command-Preferred "Command: "
```

keywords :Noise Strings

:Noise Strings {Yes, No} Whether to print prompts (noise strings) within the command. The default is Yes. Noise strings can also be turned off by setting **si:*disable-noise-strings*** to **t**.

Set Command Processor only sets the prompt for the particular mode. You can set the prompt for all modes in your init file. See the section "Setting the Command Processor Prompt".

Set File Properties Command

Set File Properties *pathname*

Brings up a menu that allows you to change the properties of a file.

pathname The file whose properties you are modifying.

```
Change properties for S:>ellen>lispm-init.lisp.18
Generation Retention Count: a number or None
Modification Date: 2/13/87 15:07:50
Reference Date: 11/30/87 16:02:39
Creation Date: 2/13/87 15:07:50
Author: Ellen
Deleted: Yes  No
Don't Reap: Yes  No
Don't Delete: Yes  No
Don't Delete Reason: a string or None
<ABORT> aborts, <END> uses these values
```

The properties that you can change are:

Generation Retention Count
: How many versions of this file to keep unmarked for deletion. Extra copies of the file are marked for deletion.

Modification Date
: When this file was last modified.

Creation Date
: When this file was created.

Author
: Who created this file.

Deleted
: Whether or not this file has been marked for deletion.

Don't Reap
: Whether or not this file can be reaped. Reaping is deletion under program control.

Don't Delete
: Whether or not this file can be explicitly deleted with a Delete command.

Don't Delete Reason
: Why this file cannot be deleted.

Set GC Options **Command**

Set GC Options

Select options for garbage collection from a menu. The menu looks like this:
For more information about GC Options: See the section "Theory of Operation of the GC Facilities".

Set Input Base Command

Set Input Base *new-base keywords*

Sets the input bases to *new-base*. See the variable ***print-base***.

new-base
: A number in base 10. The default is 10.

Since the Input Base is closely linked to the Output Base, if you set one of them, you should set the other to the same value.

```
Command: Set GC Options
Garbage collector status:
  Ephemeral GC: Off  On
  Dynamic GC: Off  On

Garbage collector report controls:
  Report the activity of the ephemeral GC: Yes  No
  Report the activity of the dynamic GC: Yes  No
  Report space reclaimed individually for each area: Yes  No  Dynamic only  Ephemeral only
  Enable warnings, for example that the GC should be turned on: Yes  No
  Minimum interval between repetitions of a GC warning: Forever
  Minimum adequate level of free space to suppress GC warnings: 1000000
  Minimum free space to suppress warnings when ephemeral GC is on: 200000
  Ratio of free space sizes on successive GC warnings: 0.75
  Disposition of GC reports: notification
  Show Ephemeral GC progress notes: No  Foreground Only  Yes  Override
  Show Dynamic GC progress notes: No  Foreground Only  Yes  Override

Garbage collection options:
  Normal garbage collection mode: Incremental  Immediate
  Mode when free space is low: Turn GC off  Immediate
  Mode when collecting ephemeral objects: Incremental  Immediate
  Fraction of free space committed to the garbage collector: 1
  Minimum fraction of free space GC will accept: Same

Garbage collector process controls:
  Process priority for foreground operations: 5
  Process priority for background operations: 0
  Process priority for immediate garbage collection mode: 5
  Priority of GC daemon processes: 5
  Delay before warning that flipping is inhibited: 30 seconds
  Time that processes wait before inhibiting flips: 5 seconds

Scavenger performance options:
  Amount of "scavenger work" done with interrupts inhibited: 1024
  Maximum time (microseconds) with interrupts inhibited: 50000
  Number of words scavenged before turning to another region: 128
  Number of pages of Copy space to prefetch: 9
  Number of pages that refer to ephemeral objects to prefetch: 15
  Number of pages of Old space to prefetch: 5
  Number of words to look ahead for prefetchable Old space references: 2048
  Declare pages flushable from main memory after scavenging them: Yes  No

<ABORT> aborts, <END> uses these values
```

Figure 126. Set GC Options Menu

Set Lisp Context **Command**

Set Lisp Context *lisp-syntax keywords*

lisp-syntax Zetalisp or Common-Lisp.

Sets the current context to use Zetalisp or Common Lisp syntax and sets the current package **zl-user** (for Zetalisp) or **cl-user** (for Common Lisp). The default is the current Lisp syntax, so using the command with no arguments does not change anything.

Set Output Base **Command**

Set Output Base *new-base keywords*

Sets the output bases to *new-base*. See the variable ***print-base***.

new-base A number in base 10. The default is 10.

Since the Output Base is closely linked to the Input Base, if you set one of them, you should be sure to set the other to the same value.

Set Package **Command**

Set Package *package-name keywords*

Makes *package-name* the current package; in other words, the variable **zl:package** is set to the package named by *package-name*.

package The name of a package.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Set Printer **Command**

Set Printer *printer-name keywords*

Sets the default printer for hardcopy.

printer-name The name of a supported printer that can be reached by your machine.

keywords :More Processing, :Output Destination, :Output Type

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, Printer, Stream, Window} Enables you to direct the output of this command. The default is the stream ***standard-output*.**

:Output Type {text bitmap both} Enables you to specify whether a printer prints text (files and mail messages), bitmap (graphics and screen hardcopy), or both text and bitmap.

Set Process Priority Command

Set Process Priority *process value*

Adjusts the priority of *process*.

process A process. You can press HELP or use the Show Processes command for a list of all the processes currently running in your environment.

value {*number*} For Foreground processes only. The priority value to give the process. Usually values range from -1 to 30, with most normal processes having priority 0. The mouse and keyboard process usually have priority 30 and some network processes have priority 10. The garbage collector and notifications have priority 5. Use Show Processes to see all your current processes with their priorities. It is the relationship among the priorities that is important. The mouse and keyboard have a high priority so that user input is recognized and handled rapidly. Notifications should not take priority over keyboard or mouse input, but they should happen in a timely fashion. See the section "Choosing Process Priority Levels".

Set Sage Variable Command

No documentation available for section Set Sage Variable Command.

Set Screen Options **Command**

Set Screen Options

Allows you to customize your screen by selecting parameters from a menu containing all the screen options. Some of the options are global, affecting all screens. Those options that are per screen appear in the menu last, a set for each applicable screen. Thus, if you have a color screen connected to your machine, options for the Main Screen and for the Color Screen are displayed. You select options with the mouse or the keyboard. See the section "Using the Mouse and the Keyboard on Menus".

The options you can set and their values are:

Documentation line options:

Mouse documentation area height: 2
Possible values: 1 or 2.
Mouse documentation area background: **Reverse** Normal
Mouse documentation area character style:
SWISS.BOLD-CONDENSED-CAPS.NORMAL
Possible values: a character style suitable for **:screen**. See the section "Character Styles and the Lisp Listener".

Status line options:

Status area character style: FIX.ROMAN.NORMAL
Possible values: a character style suitable for **:screen**.
Status line time display: **[Mon 31 Jan 11:59:59]** 12/31/89 23:59:59 Mon 31 Jan 11:59pm
Clock colon blink half period: 1 second
Process display: **User name** Process name
Machine name: Visible **Invisible**
Progress area character style: FIX.EXTRA-CONDENSED.NORMAL
Possible values: a character style suitable for **:screen**. See the section "Character Styles and the Lisp Listener".
Progress area: Wide bar No display **Text and thin bar**

Global Screen options: (These options are not applicable to MacIvory)

Beep mode: Beep only Flash only **Beep and flash**

Interval to wait before dimming screen: 20 minutes

Dimness percentage: 0

Global window defaults:

End of screen action: **Scroll** Truncate Wrap

Amount to scroll (number of lines or screen fraction): 1

Overlap between screens when scrolling (lines): 1

Whether to prompt in the CP: **Yes** No
See the variable **si:*disable-noise-strings***.

Character style for prompts: NIL.NIL.NIL
Possible values: a character style suitable for **:screen**. See the section "Character Styles and the Lisp Listener".

Graphics Scan Conversion: **Fast** Accurate *Specific flag mask*

Background interactor window settings:

Position the interactor: **At-Right** At-Top

Fraction of the screen it covers: 3/4

Keyboard and mouse repeat controls:

(Note: on MacIvory, the Macintosh Control Panel also controls keyboard auto-repeat. These two auto-repeating controls are separate and turning them both on might cause unpredictable behavior.)

Keyboard auto-repeats when any key is held: Yes **No**
REPEAT key enabled: **Yes** No
60ths of a second before key starts repeating: 42
60ths of a second between repeated characters: 2
Scroll bar auto-repeats when mouse button is held: **Yes** No
60ths of a second before scroll bar starts repeating: 30
Minimum number of lines scrolled per second: 2
Maximum number of lines scrolled per second: 50
Maximum number of screens scrolled per second: 2

Other:

Status line present on main screen: **Yes** No

Per Screen options:

Background gray pattern: None 5.5% 6% 7% 8% 9% 10% 12% HES 25% 33% 50% 75% Black **White**
If you define your own **gray-array**, its name is added to the list or, if your array does not have a name, the item **Current** is added.

Partially exposed window gray pattern: None 5.5% **6%** 7% 8% 9% 10% 12% HES 25% 33% 50% 75% Black White

Screen background: Black **White** *(This option is not applicable to MacIvory)*

Highlighting mode for highlighting menus: Inverse Video Box
Possible values: a character style suitable for **:screen** or **nil**, the default, meaning the same character style but in inverse video.

If you find that you always want to reset certain options, you can set them in your init file by using the Lisp function **tv:set-screen-options**. See the function **tv:set-screen-options**.

Set Site **Command**

Set Site *site-name*

Configures the local distribution world to be an already existing site.

site-name {*name*, get-from-network} The name of your site.

Any further arguments are entered through an AVV menu that adjusts depending on the parameters needed. The Set Site command also fully supports multiple sites within one namespace so the site name does not have to match the namespace name, although one site in any namespace must have a name that is the same as the namespace name.

If the Set Site command is used when the local machine is already registered in a site, the current site is changed to the distribution site before changing over to the new site. This is to eliminate any problems with dangling references to the previous site.

The Set Site command enables your machine to identify all objects included in the site's namespace database. The namespace database for each site is stored in the file system accessible from a machine called the namespace server.

In order for the Set Site command to work, your machine must be a registered host in the site's namespace. If the site's namespace server is not the local host, you must know the namespace server's name and network address.

See the section "Set Site Dialogue".

Set Stack Size Command

:Set Stack Size *stack-type stack-size*

Sets the size of a stack.

stack-type The type of the stack. Enter Control, Binding, or Data. (Default is Control.)

stack-size The size of the stack. Enter a number of machine words that represents the stack size.

Set Time **Command**

Set Time *time keywords*

Sets the local time on your machine. This allows you to set the time that appears in the lower left hand corner of the status line if you need to. The default is constructed from the status line time.

time A date and time string, or the source from which to get the date and time, that is one of Network or Calendar Clock.

keywords :Set Status Line, :Set Calendar Clock

:Set Status Line {Yes, No, Ask} Whether or not to set the time in the status line. The default is Ask.

:Set Calendar Clock {Yes, No, Ask} Whether or not to set the internal backup clock of either the Symbolics machine or the embedding host computer, as the case may be. The default is Ask unless the *time* argument is Calendar Clock, in which case it defaults to No.

Set Time accepts most reasonable printed representations of date and time and converts them to the standard internal forms. The following are representative formats that are accepted by the parser:

```
"March 15, 1960" "15 March 1960" "3/15/60" "3/15/1960"
"3-15-60" "3-15" "15-March-60" "15-Mar-60" "March-15-60"
"1960-3-15" "1960-March-15" "1960-Mar-15"
"1130." "11:30" "11:30:17" "11:30 pm" "11:30 am" "1130" "113000"
"11.30" "11.30.00" "11.3" "11 pm" "12 noon"
"midnight" "m" "Friday, March 15, 1980" "6:00 gmt" "3:00 pdt"
"15 March 60" "15 March 60 seconds"
"fifteen March 60" "the fifteenth of March, 1960;"
"one minute after March 3, 1960"
"two days after March 3, 1960"
"Three minutes after 23:59:59 Dec 31, 1959"
"now" "today" "yesterday" "two days after tomorrow"
"one day before yesterday" "the day after tomorrow"
"five days ago"
```

Date strings in ISO standard format are accepted also. These strings are of the form `"yyyy-mm-dd"`, where:

`yyyy`	Four digits representing the year
`mm`	The name of the month, an abbreviation for the month, or one or two digits representing the month
`dd`	One or two digits representing the day

Following are some restrictions on strings to be parsed:

- You cannot represent any year before 1900.
- A four-digit number alone is interpreted as a time of day, not a year. For example, `"1954"` is the same as `"19:54:00"` or `"7:54 pm"`, not the year 1954.
- The parser does not recognize dates in European format. For example, `"3/4/85"` or `"3-4-85"` is always the same as `"March 4, 1985"`, never `"April 3, 1985"`. A string like `"15/3/85"` is an error. In such strings, the first integer is always parsed as the month and the second integer as the day.

Set Window Options Command

Set Window Options *window*

Displays a menu of the settable options for *window*. See the section "Using Menus".

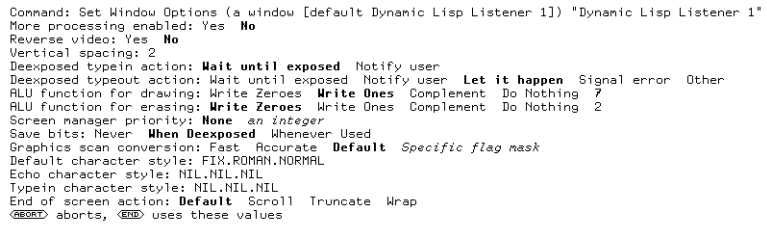


Figure 127. The Set Window Options Menu

Show Commands

Show commands allow you to request informational displays of all kinds. These displays are mouse sensitive when appropriate and can be used in composing other commands. For example, after a Show Directory display, the individual pathnames in the directory can be selected as arguments to a Show File command.

Show Additional Patches Command

Show Additional Patches *system keywords*

Shows whether or not you are up to patch level in the specified system and optionally displays the patch comments for those patches not yet loaded.

system — The system (or systems) to check for unloaded patches. The default is All, to check all systems.

keywords — :Comments, :More Processing, :Output Destination

:Comments — {Yes, No, Ask} Whether or not to show the patch comments for any unloaded patches. The default is Ask, the mentioned default is Yes.

:More Processing — {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Breakpoints Command

:Show Breakpoints

Displays all of the currently set breakpoints.

Show Callers Command

Show Callers *symbol keywords*

Tries to find all the functions in the Lisp world that call *symbol*.

symbol The name of a defined function whose callers to locate.

keywords :Called How, :More Processing, :Output Destination, :Package, :System

:Called How Lists only those callers that refer to *symbol* as a particular type of Lisp Object. The possible types are:

Condition	Flavor-Component	Macro
Constant	Function	Microcoded-Function
Defined-Constant	Generic-Function	Unbound-Function
Flavor	Instance-Variable	Variable

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Package Lists only those callers in the specified package.

:System Lists only those callers from the specified system.

See the function **who-calls**.

Show Carry Tape **Command**

Show Carry Tape *keywords*

keywords :More Processing, :Output Destination, :Tape-Spec, :Verify

Describes what is on a Carry-Tape. Once started, it does not interact in any way. See the section "The Carry-Tape Dumper".

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Tape Spec The specifications for the tape. For more information, see the section "Tape Specifications".

:Verify {Yes, No} Whether to verify the tape against the file system. The default is No.

Show Class Generic Functions **Command**

Show Class Generic Functions *class keywords*

Shows all generic functions applicable to instances of the given class.

keywords :Detailed, :Match, :More Processing, :Output Destination.

:Detailed {Yes, No} Shows arguments, indicating (with "<!>") those which can be specialized for this class. The default is No (mentioned default is Yes).

:Match {*string*} Shows only generic functions whose names contain this substring. The default is to show all generic functions.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Note that classes and methods are presented in the output in a way that enables you to click on them to perform operations on them. For example, you can click m-Left to edit a definition.

Show Class Initargs **Command**

Show Class Initargs *class keywords*

Shows the initialization arguments accepted by **clos:make-instance** of this class, and any default initial values. Also shows slots affected by initargs.

keywords :Detailed, :Direct, :Match, :More Processing, :Output Destination, :Sort

:Detailed {Yes, No} Shows affected slots and defaults. When :Detailed is No, you see the initializations from an external perspective (useful for making an instance). When :Detailed is Yes, you see the initializations from an internal perspective and get more information about how the class is constructed internally. (The mentioned default is Yes.)

:Direct {Yes, No} Shows only initializations defined in this class. With :Direct Yes, inherited initializations are not shown. The default is No, which requests all initializations defined for this class or inherited by this class. (The mentioned default is Yes.)

:Match {*string*} Shows initargs whose names contain this substring. The default is to show all initargs.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Sort (Alphabetical, Class} Sorts the display alphabetically by initarg, or by location in the class precedence list. The default is Alphabetical. Class displays initargs in a format similar to the "Show Class Superclasses Command".

Note that classes and methods are presented in the output in a way that enables you to click on them to perform operations on them. For example, you can click m-Left to edit a definition.

Show Class Methods **Command**

Show Class Methods *class keywords*

Shows methods applicable to this class.

keywords	:Direct, :Match, :More Processing, :Output Destination, :Stop At
:Direct	{Yes, No} Shows only methods applicable to instances of the given class; does not show methods inherited from superclasses. The default is No (the mentioned default is Yes).
:Match	{*string*} Shows only methods for generic functions whose names contain this substring. The default is to show methods for all generic functions.
:More Processing	{Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").
:Output Destination	{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.
:Stop At	(*superclass-name*} Only look as far upward as this superclass, which is a superclass of the argument *class* (press HELP for a list of classes in their precedence order). The default is the direct subclass of **t** from which *class* inherits (the second-to-last element of *class's* precedence list).

Note that classes and methods are presented in the output in a way that enables you to click on them to perform operations on them. For example, you can click m-Left to edit a definition.

Show Class Slots **Command**

Show Class Slots *class keywords*

Shows the slots of the given class.

keywords :Detailed, :Direct, :Match, :More Processing, :Output Destination, :Sort

:Detailed {Yes, No} Shows more detail, such as the accessors and initargs for the slots, as well as their names. The default is No (the mentioned default is Yes).

:Direct {Yes, No} Shows only direct slots (that is, those defined for this class). Slots inherited from superclasses are not shown. The default is No (the mentioned default is Yes).

:Match {*string*} Shows slots whose names contain this substring.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Sort {Alphabetical, Class} Sorts the display alphabetically by slot name, or by location in the class precedence list. If Class, each slot is displayed along with the class that provides it. The default is Alphabetical.

Note that classes and methods are presented in the output in a way that enables you to click on them to perform operations on them. For example, you can click m-Left to edit a definition.

Show Class Subclasses **Command**

Show Class Subclasses *class keywords*

Shows the subclasses of this class.

keywords :Brief, :Detailed, :Duplicates, :Initargs, :Levels, :Match, :Methods, :More Processing, :Output Destination, :Slots

:Brief {Yes, No} :Brief Yes just shows the names of the subclasses of the class. :Brief No also shows the structure. The default is No (the mentioned default is Yes).

:Detailed {Yes, No} Shows more detail, such as the accessors and initargs for the slots, as well as their names. (This option is only

useful with :Slots.) The default is No (the mentioned default is Yes).

:Duplicates {Yes, No} Shows classes everywhere they occur in the hierarchy, not just in the place from which they derive their precedence. The default is No (the mentioned default is Yes).

:Initargs {*string*} Shows initargs whose names contain this substring. This is useful, because the initarg is shown next to the class from which it is inherited. If *string* is omitted, requests all of them. If the keyword itself is not supplied, no initargs are requested.

:Levels {*integer*} Specifies the number of levels of subclasses to show. Ellipses (...) indicate subclasses of a class that are not shown because of a :Level cutoff. The default is to show all subclass levels.

:Match {*string*} Shows only the subclasses which contain this substring.

:Methods {*string*} Shows methods for generic functions whose names match this substring. This is useful, because the method is shown next to the class from which it is inherited. If *string* is omitted, requests all of them. If the keyword itself is not supplied, no methods are requested.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Slots {*string*} Shows slots whose names contain this substring. This is useful, because the slot is shown next to the class from which it is inherited. If *string* is omitted, shows all of them. If the keyword itself is not supplied, no slots are requested.

A subclass is a class that inherits from this class (directly or indirectly). This information is useful in program development or debugging, to answer the question "Which classes will be affected if I change the definition of this class?"

The output is indented to clarify the precedence order of the classes. The structure of the output is the inverse of the output of Show Class Superclasses.

Note that classes and methods are presented in the output in a way that enables you to click on them to perform operations on them. For example, you can click m-Left to edit a definition.

Show Class Superclasses **Command**

Show Class Superclasses *class keywords*

Shows the superclasses of this class in precedence order, from most specific to least specific. You can use this command to identify from which superclass a class derives part of its behavior.

keywords :Brief, :Detailed, :Duplicates, :Initargs, :Match, :Methods, :More Processing :Output Destination, :Slots

:Brief {Yes, No} :Brief Yes just lists the names of the superclasses of the class. :Brief No shows superclasses in outline form. The default is No (the mentioned default is Yes).

:Detailed {Yes, No} Shows more detail, such as the accessors and initargs for the slots, as well as their names. (This option is only useful with :Slots.) The default is No (the mentioned default is Yes).

:Duplicates {Yes, No} Shows classes everywhere they occur in the hierarchy, not just in the place from which they derive their precedence. The default is No (the mentioned default is Yes).

:Initargs {*string*} Shows initargs whose names contain this substring. This is useful, because the initarg is shown next to the class from which it is inherited. If *string* is omitted, requests all of them. If the keyword itself is not supplied, no initargs are requested.

:Match {*string*} Shows only the superclasses which contain this substring.

:Methods {*string*} Shows methods for generic functions whose names match this substring. This is useful, because the method is shown next to the class from which it is inherited. If *string* is omitted, requests all of them. If the keyword itself is not supplied, no methods are requested.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Slots {*string*} Shows slots whose names contain this substring. This is useful, because the slot is shown next to the class from which it is inherited. If *string* is omitted, shows all of them. If the keyword itself is not supplied, no slots are requested.

With :Brief No (the default), the classes are ordered from top to bottom, where the top class is the most specific class. The indentation graphically represents which classes are direct superclasses of which other classes.

With :Duplicates Yes, bracketed classes are duplicates that are included as direct superclasses, but are not ordered in this position, because of some ordering constraint. They appear in another place in the display without brackets, in their actual order. All bracketed classes have an arrow beside them. A down-arrow indicates that this class's position in the ordering is later in the display. An up-arrow indicates that this class's position in the ordering is earlier in the display (these occurrences are infrequent).

You can review the class precedence list by reading all unbracketed superclasses from top to bottom, ignoring punctuation. If :Duplicates is No, this is all that is displayed.

For information on how the order is determined, see the section "CLOS Class Precedence List".

Note that classes and methods are presented in the output in a way that enables you to click on them to perform operations on them. For example, you can click m-Left to edit a definition.

Show CLOS Generic Function **Command**

Show CLOS Generic Function *generic-function-name keywords*

Shows information about the given generic function.

keywords :Classes, :Methods, :More Processing, :Output Destination, :Sort, :Specialized

:Classes {Yes, No, By Class} Lists classes that have methods defined for this generic function. The default is No (the mentioned default is Yes). By Class is like Yes, except that it presents the class objects themselves, rather than the names of the classes.

:Methods {Yes, No, Detailed} Lists methods of this generic function. The default is No (mentioned default is Yes). Detailed shows more information.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Sort {Alphabetical, Heuristic} Sorts the methods by specializer names either alphabetically or heuristically. (Used with :Methods Yes.)

:Sort Heuristic sorts the methods in a way which takes into account the argument precedence order, the class precedence order of the specializers, the alphabetical order of the class names, and the lengths of the method qualifier lists.

:Specialized Only shows methods applicable to particular specializers (otherwise all methods are shown). For each required argument of the generic function which can be specialized, you are prompted for an object to use for that argument, or a class to use as a specializer for that argument, or a wildcard to indicate that you want to see all methods, regardless of their specializers. (Used with :Methods Yes.)

Note that classes and methods are presented in the output in a way that enables you to click on them to perform operations on them. For example, you can click m-Left to edit a definition.

Show Command Table Command

Show Command Table *command-table keywords*

Shows the structure of a command table.

command-table

keywords :Locally, :More Processing, :Multiple Columns, :Output Destination, :Show Commands

:Locally {Yes, No} Whether to suppress recursive display of information about inherited command tables. The default is No, and the mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams.

The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Multiple Columns
{Yes, No, *integer*} Whether to use multiple columns in the display. The default is No, and the mentioned default is Yes. An integer number of columns is also permitted. This is especially useful for command tables such as Global which inherit from a large number of other command tables.

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Enables you to direct your output. The default is the stream ***standard-output***. Note that redirecting output to a printer can be particularly useful.

:Show Commands {Yes, No} Whether to show the commands in the command table. The default is Yes. If No is specified, then only command table names are shown, and but not the individual commands.

The name of the command table is presented with descriptive information presented indented beneath it. Names shown in bold are command tables from which the command table inherits. Names shown in italics are commands in that command table. For example:

```
Show Command Table (named) News
  News
    Monitor AP Newswire
    Sports
      Show Scores
      Record Bet
    Weather
      Show Weather Report
      Show Wind Chill Table
```

This means that the command table News has one command of its own, Monitor AP Newswire, and inherits from two other command tables, Sports and Weather, each of which has two commands of its own.

Show Command Processor Status **Command**

Show Command Processor Status *keywords*

Displays the current mode of the Command Processor and the current prompt. It also displays the prompts for the other modes. For example:

```
The command processor's current mode is
 Form Preferred: Interprets input starting with an alphabetic
                 character as Lisp;
                 type an initial : to force command interpretation.

The prompt string is "->".

The prompt strings for other modes are:
  Command Preferred:       "Command: "
  Form Only:               ""
  Command Only:            "Command: "
```

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Compiled Code Command

Show Compiled Code *compiled-function-spec from-pc to-pc keywords* c-X D

Displays the disassembled code for a function. When you enter this command and specify a compiled-function-spec, the Debugger displays this message:

Disassembled code for (*function*):

where *function* is the name of the compiled function for which you want to see disassembled code. Immediately under this message, the Debugger lists the disassembled code instructions for this function. Each instruction — PUSH, CALL, BRANCH, and so on — is listed on its own line, numbered by the PC (program counter). PCs are numbered in octal (base 8), and numbering begins with 0.

compiled-function-spec
The name of a compiled function for which you want to see disassembled code. (Default is the function in the current frame.)

from-pc The number of the PC at which you want to begin seeing disassembled code. (Default displays all disassembled code.)

to-pc The number of the PC at which you want to stop seeing disassembled code. (Default displays disassembled code from PC 0, or from the number specified in *from-pc*, to the last PC in the disassembled code.)

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Suggested mouse operations

- To use this command with the mouse: Type in the Show Compiled Code command. When the Debugger asks you for a compiled-function-spec, point the mouse at the name of a compiled function previously displayed in the output of another command, such as Show Backtrace or Next Frame, and click Left. (You can do this only when your previous command output includes the name of a compiled function.)

- To set a breakpoint: Point the mouse at a PC in the disassembled code and press c-m-Left.

- To clear a breakpoint: Point the mouse at a PC in the disassembled code and press c-m-Middle.

Displays the disassembled code for the function associated with the current frame.

Show Compiler Warnings Command

Show Compiler Warnings *pathname keywords*

Display compiler warnings for the files specified by *pathnames*.

pathname {*pathnames(s)*, All, No File} The compiled files whose warnings to show. All shows all compiler warnings for the compilation. No File shows the warnings for no particular file. The default is All.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Crash Data Command

Show Crash Data *keywords*

keywords :Interpreted, :Output Destination

:Interpreted Interprets the data in the current world.

:Output Destination
Directs the output of this command to a specified location. The location can be the pathname of a buffer or file, a printer, stream, or window.

Obtains the most recent output from the FEP's Show Status command, and makes it available in the Lisp world.

On 3600-family machines, this information includes some hardware state, the compiled code program counter (macro PC), virtual memory address (VMA), stack pointer and frame pointer (SP and FP), and the 16 most recent microcode program counters (OPCs).

On Ivory-based machines, the Mail Bug Report subcommand of the Debug FEP Command can be used to append the stack backtrace to the crash data obtained using the Command Processor (CP) Show Crash Data command.

On both 3600-family and Ivory-based machines, the information displayed by the Command Processor (CP) Show Crash Data command is similar to the information displayed by the Show Status FEP command. However, Show Crash Data provides

an historical display; it shows the machine's status at the last crash. Show Status reveals the machine's current status (that is, its status when the command was issued).

If you want to put all of the information from the Show Crash Data command in the current editor buffer or in a mail message, use the m-X Insert Crash Data Zmacs command. This makes it easy to include information about a Lisp crash that put you into the FEP. (This data is preserved even if you cold boot the machine, but not if you reset the FEP.)

The information retrieved from the Show Crash Data command is available until the next time Lisp halts, until the FEP is reset, or the machine is powered down. Therefore, even if you are unable to warm boot your machine, you can cold boot and then make a report with the data of the last crash.

Here is a possible sequence of events:

- The machine crashes and puts you into the FEP.
- The FEP issues the Show Status command automatically.
- On Ivory-based machines, you use the Debug FEP command and its Mail Bug Report subcommand.
- Unless this is a fatal error, you use the FEP command Start to warm boot the machine. If this does not work, use the Load World and Start FEP commands to cold boot the machine.
- You use the Command Processor (CP) Show Crash Data command in order to retrieve the most recent output of the Show Status FEP Command.
- You use the m-X Insert Crash Data command in the editor to save the information in a buffer.

Show Definition Documentation **Command**

Show Definition Documentation *name type keywords*

Shows the documentation string in the definition of the Lisp object *name*.

name	{*symbol*} A defined symbol.
type	The type of the object. The default determined from the definition of the object.
keywords	:More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Demonstration Background Information Command

Show Demonstration Background Information *name keyword*

Displays background information about the demo which might help you understand where the demo came from or what it is trying to show, but which is not essential to actually running the demo. For example, if the demonstration is the Life game, this Show Demonstration Background Information might give information about who invented the Life game, the rules of the game, and other interesting information.

name The name of a demonstration.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Demonstration Instructions Command

Show Demonstration Instructions *name keyword*

Shows the essential instructions on how to use the demonstration.

name The name of a demonstration.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Demonstration Legal Notice Command

Show Demonstration Legal Notice *name keyword*

Shows any copyright notices related to the demonstration.

name The name of a demonstration.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Demonstration Names Command

Show Demonstration Names

Shows the names of demonstrations which are available in the demonstrations facility.

The output is mouse sensitive. Consult the mouse documentation line for information on the available options.

The output is partitioned into "loaded" and "unloaded". The demonstration definition resides in the file system and is not loaded until it is first referred to. Loading a demonstration loads only its definition, not its supporting code, and is therefore always fast. When a demonstration is first run, it is initialized, which may take longer.

Show Demonstration Summary Command

Show Demonstration Summary *name keyword*

Shows a brief summary of what the demonstration does.

name The name of a demonstration.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Differences Command

Show Differences *file1 file2 keywords*

Compares *file1* and *file2* and displays the differences.

file1 {*pathname*} The first file to compare.

file2 {*pathname*} The file to which to compare *file1*.

keywords :Ignore, :More Processing, :Output Destination

:Ignore {Indentation, Case-and-Style} Aspects of the file contents to ignore when searching for differences. Indentation means whitespace at the beginning of lines. Case-and-Style means capitalization and character style information.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

When Show Differences encounters a difference between two files, it displays the text from *file1* first followed by the text from *file2*. For each difference encountered, it displays last line that was the same in both files, followed by the different text up to and including the next line that matches in both files.

For example:

```
Source compare made by KJones on 9/06/87 16:28:02       -*-Fundamental-*-
of File W:>KJones>lispm-init.lisp.3
with File W:>KJones>lispm-init.lisp.4
**** File W:>KJones>lispm-init.lisp.3, Line #5, After "(load "sys:lisp;my-project")"

;;;Set up the toolkit.
**** File W:>KJones>lispm-init.lisp.4, Line #5, After "(load "sys:lisp;my-project")"
(load "tuna:>kjones>examples>decorate")

;;;Set up the toolkit.
***************

Done.
```

Show Directory **Command**

Show Directory *pathname keywords*

Displays a directory listing. The default for name, type, and version of *pathname* is **:wild**. The format of the listing varies with the operating system. For more information, see the section "Examining the Lisp Machine File System" and see the section "Interpreting Directory Listings in FSEdit".

pathname The pathname of the directory to list. The default is the usual file default.

keywords :Author, :Before, :Excluding, :More Processing, :Order, :Output Destination, :Since, :Size

:Author {*user-id*} Show those files written by this user.

:Before {*date*} Show only those files created prior to this date.

:Excluding {*file1, file2 ...*} Exclude files that do not match the indicated wildcard filenames from the directory listing. Logical pathnames and pathnames having versions specifiers of "oldest" and "newest" are not permitted as excluded *files* because the exclusion test will be done against the truename of the physical pathname, and such pathnames would never match.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Order {oldest-first, smallest-first, largest-first, newest-first, standard} Show files in this order. The default is standard, which is usually alphabetical.

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Since {*date*} Show only those files created after this date.

:Size {*integer*} Show only those files the same size or larger than *integer* blocks. The default is 0, meaning that all files will be listed, even if they are empty.

Show Disabled Services **Command**

Show Disabled Services *keywords*

Shows you which services are disabled (with the Disable Services Command).

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Disk Label Command

Show Disk Label *unit keywords*

Displays the information in the disk label of the specified disk unit.

unit {*integer*, All} The unit number of the disk whose label to show. All shows the labels of all disks connected to the machine.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during the output to interactive streams. The default is Default.

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Disk Label (unit number [default 0]) 0

Pack name: Unknown
Comment: Don't tread on me
FEP kernel: FEP0:>I325-kernel.fep.9
Backup FEP kernel: FEP0:>i317-kernel.fep.6
Color Startup File:

Show Disk Usage **Command**

Show Disk Usage *files keywords*

Displays information about the amount disk space used by *files*.

files {*pathname(s)*} The file or directories for which to display disk use.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

A line of information about the disk use of each host is also provided. For example:

```
Command: Show Disk Usage ACME:>KJones>

ACME: 8875 free, 403595/412470 used (97%, 3 partitions)
      (LMFS records, 1 = 5056. 8-bit bytes)
  ACME:>KJones>*.*.* uses 5,823 blocks in 213 files.
```

Show Distribution Directory **Command**

Show Distribution Directory *keywords*

Describes the contents of a distribution tape, or a dummy distribution file. For each file in the distribution, this command displays the file's logical pathname, and the physical pathname to which it would be restored. See Figure 19.

keywords :More Processing, :Output Destination, :Use Disk

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Use Disk On all machines other than a MacIvory, the choices are {Yes, No}. If Yes, the input is read from a tape image file on disk. The default is No. The mentioned default is Yes.

On a MacIvory, the choices are {Tape, Disk, Floppy}. Disk indicates the hard disk, and Floppy indicates the floppy disk. These two values also read a tape image file from disk or floppy. Tape means to read from tape; this is the default.

```
Distribution listing, 12/01/87 18:08:31

Systems on this tape:
  LMFS, TAPE, UTILITIES, SERVER-UTILITIES, HARDCOPY

Files on this tape:                                  Translated pathnames:
 In system LMFS:
    1  SYS:LMFS;LMFS.LISP.105                        Q:>rel-7>sys>lmfs>lmfs.lisp.105
    2  SYS:LMFS;PATCH;LMFS.SYSTEM-DIR.83             Q:>rel-7>sys>lmfs>patch>lmfs.system-dir.83
    3  SYS:LMFS;PATCH;LMFS-94.COMPONENT-DIR.1        Q:>rel-7>sys>lmfs>patch>lmfs-94>lmfs-94.component-dir.1
    4  SYS:LMFS;PATCH;LMFS-94.PATCH-DIR.1            Q:>rel-7>sys>lmfs>patch>lmfs-94>lmfs-94.patch-dir.1
 In system TAPE:
    5  SYS:LMTAPE;TAPPKG.LISP.47                     Q:>rel-7>sys>lmtape>tappkg.lisp.47
    6  SYS:LMTAPE;TAPE.SYSTEM-DIR.127                Q:>rel-7>sys>lmtape>tape.system-dir.127
    7  SYS:LMTAPE;TAPE-74.COMPONENT-DIR.1            Q:>rel-7>sys>lmtape>tape-74>tape-74.component-dir.1
    8  SYS:LMTAPE;TAPE-74.PATCH-DIR.3                Q:>rel-7>sys>lmtape>tape-74>tape-74.patch-dir.3
    9  SYS:LMTAPE;TAPE-74-1.BIN.1                    Q:>rel-7>sys>lmtape>tape-74>tape-74-1.bin.1
 In system UTILITIES:
   10  SYS:SYS;UTILITY-SYSDCL.LISP.51                Q:>rel-7>sys>sys>utility-sysdcl.lisp.51
   11  SYS:PATCH;UTILITIES.SYSTEM-DIR.55             Q:>rel-7>sys>patch>utilities.system-dir.55
   12  SYS:PATCH;UTILITIES-20.COMPONENT-DIR.1        Q:>rel-7>sys>patch>utilities-20>utilities-20.component-dir.1
                  .
                  .
                  .
```

Figure 128. Part of the Display from Show Distribution Directory

Show Documentation **Command**

Show Documentation *topic keywords*

Displays the documentation for *topic*. If *topic* is more than one word, it must be enclosed in quotation marks.

```
Show Documentation (for topic) "Getting Acquainted With Genera"
```

keywords :Destination

:Destination {screen, or *printer name*}. Where to display (print) the documentation. The default is to display the documentation on the screen. Mentioning the :Destination keyword changes the default to the default hardcopy device, if it is a supported printer, or to the supported printer most recently used for formatted output.

See the section "Document Examiner".

Show ECOs **Command**

Show ECOs *keywords*

Shows any ECOs (Engineering Change Orders) loaded into your world.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Editor Buffer **Command**

Show Editor Buffer *buffer keywords*

Displays the contents of the specified editor buffer (*buffer*).

buffer The buffer whose contents to show. Pressing HELP displays a list of all your editor buffers.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Effective Method **Command**

Show Effective Method *generic-function specializers keywords*

Lists the applicable methods of *generic-function* with the arguments described by *specializers*. For each required argument of the generic function which can be specialized, you are prompted for an object to use for that argument, or a class to use as a specializer for that argument.

keywords :Code, :More Processing, :Output Destination

:Code {Yes, No} Shows code resulting from method combination. The default is No (the mentioned default is Yes).

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Note that classes and methods are presented in the output in a way that enables you to click on them to perform operations on them. For example, you can click m-Left to edit a definition.

Show Expanded Lisp Code **Command**

Show Expanded Lisp Code *form keywords*

Shows the lisp definition of *form*. See Figure ! .

form {*macro name* None} The Lisp form whose code to expand. Specifying None causes the command to prompt you for a macro form to evaluate just as if you had used **(mexp)**. See the function **mexp**.

keywords :As If Compiler, :Constant Folding, :Expand, :More Processing, :Output Destination, :Repeat, :Whole Form

:As If Compiler {Yes, No} Does everything the compiler would do to the form. The default is No. The mentioned default is Yes.

:Constant Folding {Yes, No} Does any constant computations, arithmetic for instance, that will not change and thus can be done now. The default is No. The mentioned default is Yes.

:Expand {macros, inline-functions, style-checkers, optimizers, constants} Does macro expansion. The default is to expand inline-functions and macros.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If De-

fault, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination — {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Repeat — {Yes, No} Takes the output and operates on it again to expand it fully. The default is No. The mentioned default is Yes.

:Whole Form — {Yes, No} Expands recursively until the innermost code loop is expanded. The default is Yes.

Example:

```
Command: Show Expanded Lisp Code (defmacro form ()())

(PROGN (SI:DEFMACRO-CLEAR-INDENTATION-FOR-ZWEI 'FORM)
       (MACRO FORM (SI:.FORM. SI:.ENV.)
         (DECLARE (ARGLIST))
         (BLOCK FORM
           (TAGBODY
                    (LET* NIL
                      (AND (CDR SI:.FORM.)
                           (GO #:G3615))
                      (IGNORE SI:.ENV.)
                      (RETURN-FROM FORM
                        (PROGN NIL)))
            #:G3615 (RETURN-FROM FORM
                      (SI:DEFMACRO-ARGUMENT-ERROR 'FORM 'NIL SI:.FORM.))))))
```

Figure 129. Expansion of defmacro form

Show Expanded Mailing List **Command**

Show Expanded Mailing List *electronic-mailing-list keywords*

Shows the list of individuals who receive messages sent to a mailing list.

electronic-mailing-list — The name of a mailing list (from the file >mail>static>mailboxes.text)

keywords — :All Levels, :Matching, :More Processing, :Output Destination

:All Levels — {Yes, No} Whether to show the expansion of mailing lists that are included in the selected mailing list. The default is yes, expand all lists down to individuals.

:Matching {*string*} Shows only those names on the mailing list matching *string*.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show FEP Directory **Command**

Show FEP Directory *keywords*

Displays a description of the FEP files on the local host. The :Host keyword allows you to specify another host.

keywords :Format, :Highlight Files In Use, :Highlighting Mode, :Host, :More Processing, :Output Destination, :Type, :Unit

:Format {Normal, Detailed} How much information to include in the display. The default is Normal, meaning file name, length in blocks, and file comment (if any) are displayed for each file. Detailed means that all information, including creation date and author, is displayed.

:Highlight Files In Use
{Yes, No} Whether to indicate files that are currently in use. The default is Yes. This keyword works only when displaying the FEP file system of the local host.

:Highlighting Mode
{Bold, Arrow} How to indicate that a file is in use. Bold means display the filename in boldface, Arrow means prefix the filename with an arrow. (The arrow is useful from a remote terminal.) The default is Bold except on remote terminals, where the default is Arrow.

:Host A host on the network. The default is local.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this

command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Type {All, Boot, Fep, LMFS, Microcode, Other, Paging, World} The type of file(s) to display information about. The default is All. More than one type can be given, separated by commas.

:Unit {*disk-unit-number* All} The default is All. *disk-unit-number* is an integer, interpreted as a disk unit number on the specified host.

Show FEP Directory first displays the number of free blocks and the proportion of blocks used on each disk unit. It then displays a summary of the files on each unit grouped by file type.

Show File **Command**

Show File *pathname keywords*

Displays a file on the screen. If there is more than one screenful, it pauses between screenfuls displaying --More-- at the bottom.

pathname The pathname of the file to be displayed. The default is the usual file default.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Flavor Components Command

Show Flavor Components *flavor keywords*

Shows the order of the components of this flavor.

keywords	:Brief, :Detailed, :Duplicates, :Functions, :Initializations, :Instance Variables, :Match, :Methods, and :Output Destination. See the section "Keyword Options for Show Flavor Commands".
:Duplicates	{Yes, No} Indicates whether or not to display duplicate occurrences of flavors. The default is No.
:Brief	{Yes, No} Yes indicates that the output should not be indented to show the structure. The default is No.

The flavor components are ordered from top to bottom. The top flavor is the *most specific flavor* in the ordering. The indentation graphically represents which flavors are components of which other flavors. In the example below, **tv:minimum-window** has six direct components: **tv:essential-expose**, **tv:essential-activate**, **tv:essential-set-edges**, **tv:essential-mouse**, **tv:essential-window**, and **flavor:vanilla**.

When you use the :duplicates keyword and show the components of complex flavors, you notice special symbols in the display. For example:

```
Command: Show Flavor Components TV:MINIMUM-WINDOW :Duplicates
    --> TV:MINIMUM-WINDOW
          TV:ESSENTIAL-EXPOSE
            [TV:ESSENTIAL-WINDOW] ↓
          TV:ESSENTIAL-ACTIVATE
            [TV:ESSENTIAL-WINDOW] ↓
          TV:ESSENTIAL-SET-EDGES
            [TV:ESSENTIAL-WINDOW] ↓
          TV:ESSENTIAL-MOUSE
          TV:ESSENTIAL-WINDOW
            TV:SHEET
              SI:OUTPUT-STREAM
                SI:STREAM
          FLAVOR:VANILLA
```

Bracketed flavors are duplicates that are included by the parent flavor here, but are not ordered in this position because of some ordering constraint. They appear in another place in the display without brackets, in their correct order. All bracketed components have an arrow beside them. A down-arrow indicates that this component's position in the ordering is later in the display. An up-arrow indicates that this component's position in the ordering is earlier in the display; these occurrences are infrequent.

For example, the flavor **tv:essential-window** is a component of four other components: **tv:essential-expose**, **tv:essential-activate**, **tv:essential-set-edges**, and **tv:minimum-window** itself. Its correct position in the ordering is directly after **tv:essential-mouse**, where it appears without brackets.

You can read the order of flavor components by reading all unbracketed flavors from top to bottom, ignoring punctuation. If :Duplicates is No, this is all that is displayed.

For information on how the order is determined, see the section "Ordering Flavor Components".

Show Flavor Dependents Command

Show Flavor Dependents *flavor keywords*

Shows the names of flavors that are dependent on this flavor.

keywords	:Brief, :Detailed, :Duplicates, :Functions, :Initializations, :Instance Variables, :Levels, :Match, :Methods, :Output Destination. See the section "Keyword Options for Show Flavor Commands".
:Brief	{Yes, No} Yes indicates that the output should not be indented to show the structure. The default is No.
:Duplicates	{Yes, No} Indicates whether or not to display duplicate occurrences of flavors. The default is No.
:Levels	{All, *integer*} Specifies how many levels of indirect dependency to display. The default is all, which shows all levels. For some flavors the output can be voluminous, and it is helpful to use :Levels to pare it down.

A dependent flavor is a flavor that uses this flavor as a component (directly or indirectly). This is useful in program development or debugging, to answer the question "What flavors will be affected if I change the definition of this flavor?" For example:

```
Command: Show Flavor Dependents TV:SCROLL-WINDOW-WITH-DYNAMIC-TYPEOUT
     --> TV:SCROLL-WINDOW-WITH-DYNAMIC-TYPEOUT
           TV:BASIC-PEEK
             TV:PEEK-PANE
           TV:BASIC-TREE-SCROLL
             LMFS:AFSE-MIXIN
               LMFS:FSMAINT-AFSE-PANE
             LMFS:FSMAINT-HIERED-PANE
             TV:MOUSABLE-TREE-SCROLL-MIXIN
               TV:TREE-SCROLL-WINDOW
```

The output is indented to clarify which flavor is built on which component flavors. The structure of the output is the inverse of the output of Show Flavor Components. In this example, **tv:basic-peek** is a direct dependent of **tv:scroll-window-with-dynamic-typeout**, and **tv:peek-pane** is a direct dependent of **tv:basic-peek**.

Show Flavor Differences Command

Show Flavor Differences *flavor1 flavor2 keywords*

Shows the characteristics that two flavors have in common, and the characteristics in which they differ.

keywords :Match, :More Processing, :Output Destination. See the section "Keyword Options for Show Flavor Commands".

:Match {*string*} Displays only those generic functions or messages that match the given substring.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

This is most useful for two flavors that share many characteristics. Here is some sample output:

```
Command: Show Flavor Differences TV:ESSENTIAL-WINDOW TV:MINIMUM-WINDOW
     --> TV:ESSENTIAL-WINDOW and TV:MINIMUM-WINDOW have
         common components:
            flavors...
         TV:MINIMUM-WINDOW has other components:
            flavors...

         Differences in :ACTIVATE methods from TV:ESSENTIAL-WINDOW
                                           to TV:MINIMUM-WINDOW
            TV:SHEET before, primary,
            TV:ESSENTIAL-ACTIVATE after [added]

         Differences in handling of :BURY
            Flavor TV:ESSENTIAL-WINDOW does not handle :BURY
            Methods of TV:MINIMUM-WINDOW:
               TV:ESSENTIAL-EXPOSE wrapper, TV:ESSENTIAL-ACTIVATE
            more differences...
```

First, the common components are displayed. Second, the extra components of either (or both) flavors are displayed. Third, any differences in handling of generic functions are displayed.

In this example, **tv:minimum-window** has one method for **:activate** that **tv:essential-window** does not have: an **:after** method provided by flavor **tv:essential-activate**. The term [added] indicates that this method is defined for the second flavor but not for the first flavor. If the command had been given such that *flavor-1* was **tv:minimum-window** and *flavor-2* was **tv:essential-window**, the term would have been [deleted]. To interpret which flavor "adds" or "deletes" a method, look at the line that defines the perspective: "Differences in :ACTIVATE methods from TV:ESSENTIAL-WINDOW to TV:MINIMUM-WINDOW".

When comparing two complex flavors, the output can be voluminous. You can use :Match to pare down the output so it answers a specific question. For example:

```
Command: Show Flavor Differences DYNAMIC-LISP-LISTENER SHEET :Match
         screen
    --> information about common and different components...
        Difference in handling of :FULL-SCREEN
          Method of DW::DYNAMIC-LISP-LISTENER: TV:ESSENTIAL-SET-EDGES
          Flavor TV:SHEET does not handle generic operation :FULL-SCREEN
        another difference...
        5 local functions found with no differences:
          TV:SCREEN-MANAGE-RESTORE-AREA
          TV:SCREEN-MANAGE-CLEAR-AREA
          TV:SCREEN-MANAGE-CLEAR-UNCOVERED-AREA
          TV:SCREEN-MANAGE-CLEAR-RECTANGLE
          TV:SCREEN-MANAGE-MAYBE-BLT-RECTANGLE

        15 differing local functions were found that did not
        contain the substring "screen".
```

Show Flavor Functions **Command**

Show Flavor Functions *flavor keywords*

Shows internal flavor functions for the given flavor.

keywords	:Locally, :Match, :Output Destination, and :Sort. See the section "Keyword Options for Show Flavor Commands".
:Locally	{Yes, No} If yes, inherited internal flavor functions are not shown. The default is no, which shows all internal flavor functions defined for this flavor or inherited by this flavor.
:Match	{*string*} Displays only those internal functions that match the given substring.

Internal flavor functions are defined by **defun-in-flavor**, **defmacro-in-flavor**, and **defsubst-in-flavor**. See the section "Defining Functions Internal to Flavors".

```
Command: Show Flavor Functions TV:WINDOW
    -->  TV:ADJUST-MARGINS
         SI:ANY-TYI-CHECK-EOF
         SI:ASSURE-INSIDE-INPUT-EDITOR
         others...
```

Show Flavor Handler Command

Show Flavor Handler *operation flavor keywords*

Provides information on the handler that performs *operation* (which can be a generic function or a message) on instances of *flavor*.

keywords :Code and :Output Destination. See the section "Keyword Options for Show Flavor Commands".

:Code {Yes, No, Detailed} Specifies whether the Lisp code of the handler should be displayed. The default is No. Yes displays a template that resembles the actual code of the handler. Detailed displays the actual code of the handler. This displays some internal functions and data structures of the Flavors system. For most purposes, Yes is more useful than detailed.

If the handler is a single method (not a combined method), its function spec is given:

```
Command: Show Flavor Handler CHANGE-STATUS CELL
     --> The handler for CHANGE-STATUS of an instance of CELL is
         the method (FLAVOR:METHOD CHANGE-STATUS CELL).
         The method-combination type is :AND :MOST-SPECIFIC-LAST.
```

If the handler is a combined method, the method combination type and order of methods are displayed. In the following example, the methods used in the combined method are represented by the names of the flavors that implement them. Even in this abbreviated format, the representation of the method is mouse-sensitive.

```
Command: Show Flavor Handler CHANGE-STATUS BOX-WITH-CELL
     --> The handler for CHANGE-STATUS of an instance of
         BOX-WITH-CELL is a combined method, with
         method-combination type :AND :MOST-SPECIFIC-LAST.
         The methods in the combined method, in order of
         execution, are: CELL, BOX-WITH-CELL
```

For combined methods, :Code Yes is useful. It requests a template that resembles the actual code of the handler:

```
Command: Show Flavor Handler CHANGE-STATUS BOX-WITH-CELL :Code yes
     --> The handler for CHANGE-STATUS of an instance of
         BOX-WITH-CELL is a combined method, with
         method-combination type :AND :MOST-SPECIFIC-LAST.
         (DEFUN (FLAVOR:COMBINED CHANGE-STATUS BOX-WITH-CELL)
                  (SELF SYS:SELF-MAPPING-TABLE FLAVOR::.GENERIC.
                   &REST FLAVOR::DAEMON-CALLER-ARGS.)
           (AND call (FLAVOR:METHOD CHANGE-STATUS CELL)
                call (FLAVOR:METHOD CHANGE-STATUS BOX-WITH-CELL)))
```

Show Flavor Initializations **Command**

Show Flavor Initializations *flavor keywords*

Shows the initialization keywords accepted by **make-instance** of this flavor, and any default initial values.

keywords :Detailed, :Locally, :Match, :Sort, and :Output Destination. See the section "Keyword Options for Show Flavor Commands".

:Detailed {Yes, No} The default is No, which requests the allowed initialization keywords that can be given to **make-instance** of this flavor, including init keywords and initable instance variables. If :Detailed is Yes, any additional instance variables are also shown; these are not initable instance variables. They are initialized by default values given in the **defflavor** form. Also, any initialization methods are shown. In other words, when :Detailed is No, you see the initializations from an external perspective (useful for making an instance). When :Detailed is Yes, you see the initializations from an internal perspective and gain information about how the flavor is constructed internally.

:Locally {Yes, No} If Yes, inherited initializations are not shown. The default is No, which requests all initializations defined for this flavor or inherited by this flavor.

:Match {*string*} Requests only those initializations matching the given substring.

For example:

```
Command: Show Flavor Initializations BOX-WITH-CELL :Detailed
    -->  Instances of BOX-WITH-CELL are created in the default area
         Another area can be specified with the keyword :AREA
         Initialization keywords that initialize
           instance variables:
            :BOX-X → BOX-X
            :BOX-Y → BOX-Y
            :SIDE-LENGTH → SIDE-LENGTH, default is *SIDE-LENGTH*
            :STATUS → STATUS, default is (IF (EVENP (RANDOM 2))
                                             ':ALIVE ':DEAD)
            :X → X
            :Y → Y
         Initialization method:
            MAKE-INSTANCE method: BOX-WITH-CELL
```

Show Flavor Instance Variables Command

Show Flavor Instance Variables *flavor keywords*

Shows the state maintained by instances of the given flavor.

keywords :Detailed, :Locally, :Match, :Output Destination and :Sort. See the section "Keyword Options for Show Flavor Commands".

:Detailed {Yes, No} If Yes, the attributes of the instance variables are shown, such as their accessibility or initializations. The default is No.

:Locally {Yes, No} If Yes, inherited instance variables are not shown. The default is No, which shows all instance variables defined for this flavor or inherited by this flavor.

:Sort {Alphabetical, Flavor} If Flavor, each instance variable is displayed along with the component flavor that provides it. The default is Alphabetical.

For example:

```
Command: Show Flavor Instance Variables CELL
     --> NEIGHBORS
         NEXT-STATUS
         STATUS
         X
         Y
```

Show Flavor Methods Command

Show Flavor Methods *flavor*

Displays all methods defined for the given flavor.

Keywords :Locally, :Match, :Output Destination, :Sort, and :Using Instance Variables. See the section "Keyword Options for Show Flavor Commands".

:Locally {Yes, No} If Yes, inherited methods are not shown. The default is No, which shows all methods defined for this flavor or inherited by this flavor.

:Match {*string*} Requests only those methods for generic functions that match the given string.

:Using Instance Variable
{*name*} Requests only those methods that use the instance variable named *name*.

Each line of output contains the name of the generic function, followed by the name of each flavor that provides a method for the generic function. If the method is not a primary method, its type is also displayed.

```
Command: Show Flavor Methods BOX-WITH-CELL
    -->  ALIVENESS method: CELL
         CHANGE-STATUS methods: CELL, BOX-WITH-CELL
         COUNT-LIVE-NEIGHBORS method: CELL
         :DESCRIBE method: FLAVOR:VANILLA
         others...
```

This command is similar to Show Flavor Operations. See the section "Show Flavor Operations **Command**". The difference between the two commands is in the perspective:

Show Flavor Methods displays information from an internal perspective, answering the question: What methods are defined for this flavor, or inherited from its component flavors?

Show Flavor Operations displays information from an external perspective, answering the question: What operations (generic functions and messages) are supported by instances of this flavor?

Show Flavor Operations Command

Show Flavor Operations *flavor keywords*

Shows all operations supported by instances of the given flavor, including generic functions and messages.

keywords :Detailed, :Match, and :Output Destination. See the section "Keyword Options for Show Flavor Commands".

:Detailed	{Yes, No} If Yes, the display shows the arguments of each operation. The default is No.
:Match	{*string*} Shows only those operations matching the given substring.

For example:

```
Command: Show Flavor Operations BOX-WITH-CELL
    -->  ALIVENESS
         CHANGE-STATUS
         COUNT-LIVE-NEIGHBORS
         :DESCRIBE
         MAKE-INSTANCE
         SYS:PRINT-SELF (:PRINT-SELF)
         others...
```

One of the operations can be performed by using the generic function **sys:print-self** or sending the message **:print-self**. This operation was defined with **defgeneric**, using the **:compatible-message** option.

This command is similar to Show Flavor Methods. See the section "Show Flavor Methods **Command**". The difference between the two commands is in the perspective:

Show Flavor Operations displays information from an external perspective, answering the question: What operations (generic functions and messages) are supported by instances of this flavor?

Show Flavor Methods displays information from an internal perspective, answering the question: What methods are defined for this flavor, or inherited from its component flavors?

Show Font **Command**

Show Font *font*

Displays all characters of the font. You can get a list of the fonts loaded by pressing HELP after typing Show Font or by clicking on List Fonts in the Font Editor. You enter the Font Editor by using the Edit Font command with no arguments.

font	Font name.

Show Function Arguments Command

Show Function Arguments *function-spec keywords*

Shows arguments of the function specified by *function-spec*.

keywords :More Processing, :Output Destination. See the section "Keyword Options for Show Flavor Commands".

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show GC Status **Command**

Show GC Status *keywords*

Displays statistics about the garbage collector.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

See the section "Theory of Operation of the GC Facilities". See the section "Set GC Options Command".

Show Generic Function Command

Show Generic Function *operation keywords*

Shows information on the given *operation*, which can be a generic function or message.

keywords :Flavors, :Methods and :Output Destination. See the section "Keyword Options for Show Flavor Commands".

:Methods {Yes, No} Yes displays all methods for the generic function, and their types.

:Flavors {Yes, No} Yes displays the flavors that implement methods for the generic function.

For example:

```
Command: Show Generic Function CHANGE-STATUS
     --> Generic function CHANGE-STATUS takes arguments: (CELL-UNIT)
         This is an explicit DEFGENERIC in file SYS:EXAMPLES;FLAVOR-LIFE.
         Method-combination type is :AND :MOST-SPECIFIC-LAST.
```

Show HackSaw Command

Show HackSaw *keywords*

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Shows an interesting fact about Genera. This command is bound to M-HELP.

Show Handlers For Types Command

Show Handlers for Types *context-type presentation-type keywords*

Lists the mouse handlers applicable in a specified input context to a specified object type and presentation type.

context-type The input context (*to-presentation-type* in the handler definition).

presentation-type The presentation type of the presentation to which the mouse is pointing (the *from-presentation-type* in the handler definition).

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Herald **Command**

Show Herald *keywords*

Displays the herald message. The herald is a a multiline message displayed when you cold or warm boot, use the Command Processor (CP) Show Herald or Save World commands, or use the Disk Restore FEP command.

The herald shows you the name of the FEP file or partition for the current world load, any comment added to the herald, a measure of the physical memory and swapping space available, the versions of the systems that are running, the site's name, and the machine's own host name.

When you cold or warm boot, your machine displays a full-screen herald. When you display the herald with the Show Herald command, you see an abbreviated herald.

keywords :Detailed, :More Processing, :Output Destination

:Detailed {Yes, No, Normal} Whether or not to print the version information in full detail. Normal displays the systems which are usually of interest to users. Yes displays more systems than does Normal. No shows only machine information and release level, and no information about system versions. The default is Normal. The mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is

subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

For information about herald-related functions:

- See the function **sct:get-system-version**.
- See the function **sct:print-system-status-warning**.

Show Hosts **Command**

Show Hosts *host-spec keywords*

Asks each of the *hosts* for its status, and prints the results. If no hosts are specified, asks all hosts on the Chaosnet. Hosts can be specified by either name or octal number.

For each host, a line is displayed that either says that the host is not responding or gives metering information for the host's network attachments. If a host is not responding, probably it is down or there is no such host at that address. A Symbolics host can fail to respond if it is looping inside **without-interrupts** or paging extremely heavily, such that it is simply unable to respond within a reasonable amount of time.

host-spec A host or list of hosts (names or network addresses) or sites, separated by commas.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

For example:

```
Show Hosts Wombat
Show Hosts chaos|23557
Show Hosts Wombat,Kangaroo,Wallaby
```

The display looks like this:

```
Chaosnet host status report.  Type Control-Abort to quit.
Site  Name/Status Subnet    #-in  #-out  abort  lost  crc    ram bitc other
23557 Wombat          51 4615995  89709      0    22 8214524   07749784   919
```

Site	The Chaosnet address of the *host*, in this case 23557.
Name/Status	The name of the *host*, in this case Wombat.
Subnet	For sites with large networks, the number of the subnet on which *host* resides, in this case 51.
#-in	The number of Chaosnet packets received by *host* since it was last cold booted.
#-out	The number of Chaosnet packets transmitted by *host*.
abort	The number of packets *host* attempted to send but was unsuccessful due to network collisions.
lost	The number of incoming packets that *host* was forced to discard.
crc	Cyclic Redundancy Check. The number of packets that failed this check, because they were damaged either in transmission or by *host* when they were received.
ram	Random Access Memory. The number of hardware memory errors *host* has experienced, in this case 0.
bitc	Bit Count. The number of packets whose actual bit count did not match the bit count recorded for them.
other	Other errors.

Show IDS Children Command

Show IDS Children *file keywords*

Shows *file* (a world load file) and its IDS children. IDS children are worlds that have ben saved using the Save World Incremental command.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show IDS Files Command

Show IDS Files *keywords*

Shows the IDS files for the specified host. IDS files are world loads that have been saved using the Save World Incremental command.

keywords :Host, :More Processing, :Output Destination

:Host The name of the machine whose IDS files you want to see. The default host is the local machine.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show IDS Parents Command

Show IDS Parents *file keywords*

Shows the IDS parent worlds for *file* (a world load file). An IDS parent is the base world from which other incremental worlds are made.

file {*pathname*} The pathname of an IDS world.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Legal Notice **Command**

Show Legal Notice *keywords*

Displays the Symbolics Legal Notices, such as copyrights and trademarks.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Lisp Context **Command**

Show Lisp Context *keywords*

Displays the currently enabled lisp context (Common-Lisp or Zetalisp), the current package, and the current input and output bases. For example:

```
Common-Lisp syntax is now enabled.
Package is USER; Input base is 10; Output base is 10.
```

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is

subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Login History Command

Show Login History

Prints one line for each time the Login command has been used since the world was last cold booted. It also shows the logins done during the creation of the world load. Each line contains the name of the user who logged in, the name of the machine on which the world load was running at that time, and the date and time. This command also shows the name of an init file, if one was loaded. If you cold boot, log in, and then do Show Login History, the last line refers to your own login and all previous lines refer to logins that were done before doing Save World (or running **zl:disk-save**).

This information is useful to determine how many times a world load has been disk-saved, on what machines it was disk-saved, and who disk-saved it.

The first couple of lines do not contain any date or time, because they were made during the initial construction of the world load before it found out the current time. Names of users at other sites that are not in the local site's namespace search list are qualified with the site's namespace name and a vertical bar. The user LISPM is the dummy user used by **si:login-to-sys-host** when new world loads are created.

Show Machine Configuration **Command**

Show Machine Configuration *host keywords*

Shows the board-level hardware information about any Symbolics on the same network as your machine.

host The name of a Symbolics machine. The default is your machine.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is

subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination

{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

This information is useful for Symbolics Software Support Personnel. The display from Show Machine Configuration on a 3640 looks like this:

```
Chassis (PN 170219, Serial 70) in Chassis or NanoFEP:
    Manufactured on 10/6/84 as rev 1, functions as rev 1, ECO level 0
    Machine Serial Number: 4070
    FEP Version Number: 127
Datapath (PN 170032, Serial 2958):
    Manufactured on 1/7/85 as rev 3, functions as rev 3, ECO level 0
Extended Sequencer (PN 170299, Serial 720) in Sequencer:
    Manufactured on 5/12/86 as rev 1, functions as rev 1, ECO level 0
Memory Control (PN 170052, Serial 4102) in Memory Control or IFU:
    Manufactured on 6/11/86 as rev 5, functions as rev 5, ECO level 0
Floating Point (PN 170172, Serial 12) in FPA:
    Manufactured on 5/18/84 as rev 1, functions as rev 1, ECO level 0
Front End (PN 170062, Serial 3069) in FEP:
    Manufactured on 12/21/84 as rev 5, functions as rev 5, ECO level 0
512K Memory (PN 170002, Serial 1758) in LBus slot 00:
    Octal Base address: 0
    Manufactured on 9/28/84 as rev 2, functions as rev 2, ECO level 0
1Meg Memory (PN 170473, Serial 265) in LBus slot 02:
    Octal Base address: 4000000
    Manufactured on 10/3/85 as rev 1, functions as rev 1, ECO level 0
IO (PN 170157, Serial 849) in LBus slot 03:
    Octal Base address: 6000000
    Manufactured on 2/21/85 as rev 6, functions as rev 6, ECO level 0
512K Memory (PN 170002, Serial 1897) in LBus slot 04:
    Octal Base address: 10000000
    Manufactured on 10/15/84 as rev 2, functions as rev 2, ECO level 0
FEP Paddle Card (PN 170069, Serial 377) in FEP -- PADDLE side:
    Manufactured on 10/23/84 as rev 1, functions as rev 1, ECO level 0
IO Paddle Card (PN 170245, Serial 692) in LBus slot 03 -- PADDLE side:
    Manufactured on 6/17/85 as rev 1, functions as rev 1, ECO level 0
    Ethernet Address: 08-00-05-03-A0-15
```

Show Mail **Command**

Show Mail *pathname keywords*

Displays your mail inbox on the screen. If there is more than one screenful, it pauses between screenfuls displaying --More-- at the bottom.

pathname The pathname of the mail inbox to be read. The default is the default inbox.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Memory Error Corrections Command

Note: This command is implemented only on Ivory machines.

Show Memory Error Corrections *keywords*

With no keyword arguments, displays the number of memory errors that have occurred.

keywords: :Detailed, :More Processing, :Newest, :Output Destination

:Detailed {Yes, No} List the logged error correction codes. The default is No, the mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Newest {*a number*} The number of items to be output. The default is 20.

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

With the :Detailed keyword, displays the newest 20 error corrections, for example:

```
There have been 38 memory error corrections.
The newest 20 error corrections are listed.
The most recent memory error correction is listed first.

Memory Address    Check Bit    ECC Syndrome    Disposition
      2040256         11           145        Soft (corrected) error, physical access
          ...         ...          ...        Soft (corrected) error(s), unlogged
      1532420         11           145        Soft (corrected) error, virtual access
          ...         ...          ...        Soft (corrected) error(s), unlogged
      5646204         11           145        Soft (corrected) error, virtual access
          .
          .
          .
```

For each error listed, it also lists the kind of error and its disposition. The kind of error is either `Uncorrectable error` or `Soft (corrected) error`.

(The former appears only if you successfully used Continue in the FEP when the uncorrectable error occurred).

The disposition is one of:

`repaired by scrubber`
: meaning there was an error that could be fixed by rewriting the data.

`stuck` (***call field service***)
: meaning there is an error that cannot be fixed by rewriting, but if it is a soft error, the correct data is being returned. However, it cannot be fixed to give no errors; hence you probably have a stuck bit in your memory.

`ignored (corrected data written to disk)`
: meaning there was an error that was not worth fixing (the good data went to the disk already).

`virtual access`
: meaning an error that has not (yet) been fixed in virtual memory by the error scrubber.

`physical access`
: meaning an error that has not (yet) been fixed in physical memory by the error scrubber.

`unknown access`
: meaning either a virtual or a physical access error, but we cannot figure out which. Most likely this is due to an error that occurred in the FEP.

Show Messages **Command**

Show Messages *keywords*

Displays the contents of the specified Converse conversations.

keywords :Direction, :From, :Mention Empty Sequences, :More Processing, :Order, :Output Destination, :Query, :Recent, :Start, :Stop, :Summarize, :To

:Direction {Incoming, Outgoing, All, or Default} Whether to show incoming messages, outgoing messages, or all. The Default is Incoming.

:From {*user-or-address*} Show messages from this user or address.

:Mention Empty Sequences
{Yes, No} Whether to mention empty message sequences, for example, you have sent messages to someone but the person did not reply. The default is No, not to mention this. If it is Yes, you see "No messages from *so-and-so*".

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Order {Forward, Reverse} How to order the message presentation within each conversation. The default is Forward, that is, most recent first.

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No} Whether to ask about each conversation. The default is Yes, to ask.

:Recent {Yes, No} Whether to consider only the most recently exchanged messages in each conversation.

:Start {*number*} Number of first message to show in a conversation. If there are fewer then *number* messages in the conversation, that conversation is skipped.

:Stop {*number*} Number of last message to show in a conversation.

:Summarize {Yes, No} Whether to show the entire message or just a summary. The default is No, to show the entire message. If yes, messages are mentioned but not shown.

:To {*user-or-address*} Show messages to this user or address.

Show Monitored Locations Command

Show Monitored Locations

Displays all variables and other locations in memory that you are monitoring via the Monitor Variable command, the **dbg:monitor-location** function, and so on.

Show Mouse Handlers Command

Show Mouse Handlers *handler-name keywords*

Lists the handlers corresponding to *handler-name*, showing the input context, presentation type, and handler type (object type) to which each is applicable.

handler-name The name of a mouse handler.

keywords :More Processing, :Output Destination, :Show Defined Handlers, :Show Table Handlers

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Show Defined Handlers
{Yes No} Whether to list the handlers as described above.

:Show Table Handlers
{Yes No} Whether to show the expansion of the handlers in the handler lookup table.

Show Namespace Object **Command**

Show Namespace Object *namespace-object keywords*

Shows the information in the namespace database.

namespace-object A namespace object is specified by the class of the object followed by the name of the object. For instance, to specify the printer named Asahi, you enter:

```
Show Namespace Object printer Asahi
```

If *namespace-object* is described in more than one namespace (is "multi-homed"), Edit Namespace Object prompts for the namespace in which you want to edit this object. (Typically, a namespace object is described only in one namespace.)

keywords: :Format, :Locally, :More Processing, :Output Destination

:Format {Normal, Detailed} Whether to show fields that are empty. The default is normal, to omit empty fields. Detailed shows all fields.

:Locally Specifies whether the Show Namespace Object command queries the namespace server or uses the values in the current Lisp world.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Here is what the namespace object for a user might look like:

Showing USER KJONES in namespace ACME:

```
Lispm Name: Kjones
Personal Name: Jones, Kingsley
Nickname: King
Work Address: Building 3-701
Work Phone: 5891
Home Host: ACME
Mail Address: Kjones ACME
Birthday: 19 June
Project: Database
Supervisor: Finklestein
```

Show Notifications **Command**

Show Notifications *keywords*

Redisplays any notifications that have been received. Notifications are asynchronous messages from Genera.

keywords :Before, :Excluding, :From, :Matching, :More Processing, :Newest, :Oldest, :Output Destination, :Since, :Through

:Before A date to serve as one limit for notifications to show:

```
:before 11/1/89
```

:Excluding A string or sequence of strings to search for. Do not show notifications containing the string or sequence of strings.

:From A number to use as the first notification to show.

:Matching A string or sequence of strings to search for. Only show notifications that contain that string:

```
:matching hardcopy
```

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Newest A number of notifications to show, for instance, the ten most recent ones:

```
:newest 10
```

Using this keyword without a number is the same as `:newest 1`.

:Oldest A number of notifications to show, for instance, the ten earliest ones:

```
:oldest 10
```

Using this keyword without a number is the same as `:oldest 1`.

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Since A date to serve as one limit for notifications to show.

:Through A number to use as the last notification to show:

```
:through 17
```

Using the Show Notifications command with no keyword arguments means show all notifications in reverse chronological order (most recent first).

Show Object **Command**

Show Object *name keywords*

Show Object tries to tell you all the interesting information about any object (except for array contents). Show Object knows about areas, structures, packages, pathnames, systems, variables, functions, flavors, and resources. It displays the attributes of each. Show Object *symbol* will tell you about *symbol*'s value, its definition, and each of its properties.

name Any Lisp object.

keywords :More Processing, :Output Destination, :Type

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Type {All, Area, Structure, Package, Pathname System, Variable, Function, Flavor, Resource}. The default is All.

Show Open Files **Command**

Show Open Files *keywords*

Shows the files your machine has open for reading or writing and their state. For example:

```
Command: Show Open Files
Input file: A:>KJones>mail._ZMAIL_text.newest (100% read)
Output file:A:>sys>doc>cp>cp6.sab.newest (7647 bytes written)
Output file: HATCH:>Print-Spooler>90/04/09.Log.newest (4822 bytes written)
```

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Output History **Command**

Show Output History *dynamic-window keywords*

Displays the output history for the specified window.

dynamic-window {*window*} The window whose output history to show. Press HELP for a list of the possible windows.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Presentation Handlers From Type Command

Show Presentation Handlers From Type *type keywords*

Lists the handlers defined with *type* as their *from-presentation-type*.

type A presentation type.

keywords :Include t, :More Processing, :Output-Destination

:Include t {Yes No} Whether to include in the listing handlers defined on the **t** presentation type.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Presentation Type Command

Show Presentation Type *type keywords*

Shows the argument list, supertypes, and subtypes of a presentation type.

type A presentation type.

keywords :For Lookup, :Include Predicate, :More Processing, :Output Destination

:For Lookup {Yes No} Whether to list the types examined during mouse handler lookup. Listed supertypes for lookup are examined when *type* is the *to-presentation-type* in a handler definition; listed subtypes for lookup are examined when *type* is the *from-presentation-type* in a handler definition.

:Include Predicate
{Yes No} Whether to show, if applicable, the type reduction step of *type* to a supertype and a predicate. For example, the **symbol** presentation type is reducible to the supertype **sys:expression** and the predicate **symbolp**.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Printer Defaults **Command**

Show Printer Defaults *keywords*

Displays the current default printer(s). If you send all your hardcopy output to one printer, the command returns:

```
Default Printer (for both text and bitmap output): printer-name
```

If you use a different printer for text and screen hardcopy, the command returns:

```
Default Text Printer: printer-name1
Default Bitmap Printer: printer-name2
```

Show Printer Status **Command**

Show Printer Status *printer*

Displays the print queue for the specified printer or printers.

printer — The name of a printer or printers (separated by commas) whose print queue you are displaying. Specifying All displays the queues for all printers at your site. The default is your current printer. If your text printer and your bitmap printer are different, the command uses your text printer as the default for Show Printer Status.

The display of requests is mouse sensitive, enabling you to click on select arguments when using either the Delete Printer Request or Restart Printer Request commands.

Show Processes Command

Show Processes *keywords*

Displays all the processes currently in your environment. See Figure 31 .

Keywords — :Active, :Idle, :More Processing, :Name, :Order, :Output Destination, :Priority Above, :Priority Below, :Recent, :State, :System, :Unarrested

:Active — {*time-interval*} Shows only processes that have been active within *time-interval*. The mentioned default is "1 minute". (Obsolete, use :Recent.)

:Idle — {*time-interval*} Shows only processes that have been idle for at least *time-interval*. The mentioned default is "1 minute".

:More Processing — {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Name {*process-names-or-substrings*} Enables you to specify an existing process name, a process name substring, or a combination of the two. If you specify substrings only, the processes named by those substrings are considered for viewing (subject to the other options.) If you specify process names only, only those processes are considered. If you specify a mix, then processes named exactly and processes named by substrings are both considered.

:Order {Idle, Name, None, Percent} Sorting method for the processes display. The default is None.

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Priority Above {*integer*} Shows only processes of priority higher than *integer*.

:Priority Below {*integer*} Shows only processes of priority lower than *integer*.

:Recent {*time-interval*} Shows only processes that have run within *time-interval*.

:State {*string*} Shows only processes whose state contain *string*.

:System {Yes, No} Shows the system processes also. The default is No, the mentioned default is Yes.

:Unarrested {Yes, No} Shows only processes that are not arrested. The default is No, the mentioned default is Yes.

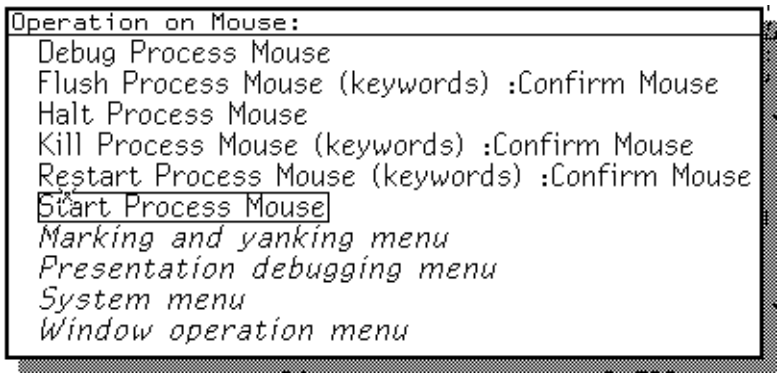
The priorities are:

F	Foreground
B	Background
D	Deadline
P	Preemptive

The process names in the display are mouse sensitive. Clicking Middle on one of them puts that process in the debugger. Clicking Right pops up a menu of operations:

```
Command: Show Processes (keywords) :Priority Above (an integer) 5
Process Name              State                  Priority  Idle     % utilization
Chaos Background          Chaos Background        F:8      1 sec    0.2%
Mouse                     Mouse                  PF:31              0.5%
Timer                     Timer Process          PD:500    forever  0.0%
Timer                     Timer Process          PD:500    2 hr     0.0%
Timer                     Timer Process          PD:500    forever  0.0%
Timer                     Timer Process          PD:500    forever  0.0%
TCP Background            TCP Background          F:8      57 sec   0.0%
Timer                     Timer Process          PD:500    forever  0.0%
Timer                     Timer Process          PD:500    1 hr     0.0%
Process Scheduler         Scheduler Wait         PD:100000          1.2%
Wait function poller      Wait function poller    F:110             1.0%
Keyboard                  Keyboard               PF:30     3 sec    0.6%
3600 Ethernet Receiver    Ethernet Packet        PF:8               1.8%
Timer                     Timer Process          PD:500             2.3%
Update Status Line        SIMPLE-PROCESS wait     D:500000          1.9%
Timer                     Timer Process          PD:500    9 min    0.0%
Timer                     Timer Process          PD:500    37 sec   0.0%
Timer                     Timer Process          PD:500             1.9%
```

Figure 130. Show Processes



Show Source Code Command

:Show Source Code *compiled-function-spec keywords* c-X c-D

Displays the source code for a function. This command works only when your code resides in an editor buffer. The output is mouse sensitive only when the function is compiled with source locators. When you specify a compiled function for which you want to see source code — for example, **myfunction** — the Debugger displays the source code for **myfunction** beneath the following message:

Source code for `MYFUNCTION:`

If **myfunction** were not compiled with source locators, the Debugger would still display the source code, but the output would not be mouse sensitive. The Debugger would display the source code only after giving you this message:

```
Function MYFUNCTION has no source locators; the code will not be sensitive.
```

compiled-function-spec
: The name of a compiled function for which you want to see source code. (Default is the function in the current frame.)

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
: {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Suggested mouse operations

When a function has been compiled using source locators — mapping source code to PCs via the editor's c-m-sh-C command — you can perform the following mouse operations:

- To use this command with the mouse: Type in the :Show Source Code command. When the Debugger asks you for a compiled-function-spec, point the mouse at the name of a compiled function previously displayed in the output of another command, such as :Show Backtrace, and click Left.

- To set a breakpoint: Point the mouse at a form (a code fragment) in the displayed source code and press c-m-Left.

- To clear a breakpoint: Point the mouse at a form (a code fragment) in the displayed source code and press c-m-Middle.

- To evaluate a code fragment: Point the mouse at a form in the displayed source code and press m-Middle.

Show Standard Value Warnings Command

Show Standard Value Warnings *keywords*

Displays more detailed information about standard variables that have been rebound. Here is an example of what the Debugger displays when you enter this command:

```
→  :Show Standard Value Warnings
The following standard values were bound:
  Rebinding CP:*COMMAND-TABLE* to #<COMMAND-TABLE User #o260252757>
  (old value was #<COMMAND-TABLE Debugger #o261747137>)
→
```

If no standard variables have been rebound, the Debugger tells you:

```
There were no standard values which required binding
```

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

See the section "Standard Variables".

Show System Components Command

Show System Components *system keywords*

Shows the systems included in *system* and graphs their dependencies. If *system* is loaded, it also shows the current system version and patch level. See Figure

! .

system A system or subsystem whose components to show.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this

command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

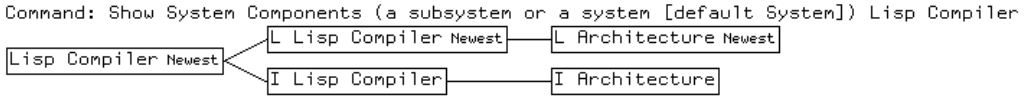


Figure 131. Show System Components Lisp Compiler

Show System Definition Command

Show System Definition *system-or-subsystem keywords*

Displays a the system definition of *system* including its current patch level, status (experimental or released), and the files that make up the system.

system-or-subsystem The system whose definition to display.

keywords :Detailed, :More Processing, :Output Destination, :Use Journals, :Version

:Detailed {Yes, No} Whether to display the information about all the component systems of the system or not. The default is No, the mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Use Journals {Yes, No} Use the information in the system's journal files instead of the information loaded into the world. The default is No, the mentioned default is Yes.

:Version {*version-designator*} What version of the system for which to display the definition. The default is :Released.

Show System Modifications **Command**

Show System Modifications *system-name keywords*

With no arguments, Show System Modifications lists the locally maintained systems present in this world and, for each system, all the modifications that have been loaded into this world. For each modification it shows the major version number (which is always the same, since a world can only contain one major version), the minor version number, and an explanation of what the modification does, as entered by the person who made it.

If Show System Modifications is called with an argument, only the modifications to *system-name* are listed.

system-name {All, Local, *system-name1, system-name2* ... } The system for which to show modifications. The default is All.

keywords :Author, :Before, :From, :Matching, :More Processing, :Newest, :Number, :Oldest, :Output Destination, :Reviewer, :Since, :Through

:Author A name. Show modifications by a particular person. For example:

```
:show modifications system :author kjones
```

would only show those modifications made by the person whose user ID is kjones.

:Before A date to serve as one limit for modifications to show:

```
:before 1/23/90
```

:From A number to use as the first modification to show.

:Matching A string to search for in the comments and show only modifications whose comment contain that string:

```
:matching namespace
```

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Newest — A number of modifications to show, for instance the ten most recent ones:

```
:newest 10
```

Using this keyword without a number is the same as `:newest 1`.

:Number — A number. Show only this particular modification. For example:

```
Show Modifications :number 6
```

would show modification number 6.

:Oldest — A number of modifications to show, for instance the ten earliest ones:

```
:oldest 10
```

Using this keyword without a number is the same as `:oldest 1`.

:Output Destination — {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Reviewer — A name. Show modifications reviewed by a particular person.

:Since — A date to serve as one limit for modifications to show.

:Through — A number to use as the last modification to show:

```
:through 17
```

Show System Plan Command

Show System Plan *system operation keywords*

Show the system plan (the order of operations) for the specified *system* under the specified *operation*.

system — The system for which to show the plan.

operation — The operation for which to show the plan. The available operations are:

All	Count-Lines-In	Kludge-Load
Compile	Distribute	Load
Copy	Edit	Load-Patches
Copy-Toolkit-C-Files	Hardcopy	Reap-Protect
Write-Toolkit-C-Files		

keywords — :Date Checking, :Detailed, :More Processing,:Output Destination, :Version

:Date Checking {Yes, No} Compare files against the file system. The default is No, the mentioned default is Yes.

:Detailed {Yes, No} Whether to describe the plans for component systems. The default is No, the mentioned default is Yes.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Version Version of the system for which to construct plans. The default is Released.

Show System Version Designations **Command**

Show System Version Designations *systems keywords*

Shows the versions of each system that have been given a name indicating their status, for example `Latest`, `Released`, `Experimental`, `Obsolete`, or any other token.

systems The name(s) of a system or systems whose version designations to show. The systems do not have to be loaded into the world.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

For example:

```
Show System Version Designations (systems [default IP-TCP]) IP-TCP, fortran
Version designations for system IP-TCP
 Latest       411
 System-350   53
 System-349   52
 System-347   49
 Release-6-g  29
 Release-6-1  29
 Release-6-0  29
 Release-5-2  23
 Release-5-1  12

FORTRAN is unknown; looking for Q:>sys>site>fortran.system.newest... reading... read.
Version designations for system Fortran
 Fortran-11  11
 Latest      404
 Released    NIL
 Fortran-10  10
```

Show Table of Contents Command

Show Table of Contents *topic keywords*

Shows the table of contents of the documentation *topic*.

topic — A topic which is documented and accessible in Document Examiner.

keywords — :Crossreferences, :Depth, :More Processing, :Output Destination, :Sources

:Crossreferences — {Yes, No} Whether to display crossreferenced topics. The default is No.

:Depth — {*integer*} How many levels deep to display. The default is to show all levels.

:More Processing — {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination — {Buffer, File, Kill Ring, None, Printer, Stream, Window}

Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Sources {Yes, No} Whether to display associated source files. The default is No. Note that in a world with a compressed documentation database, the source file information is not available.

Show TAR File Command

Show TAR File *tarfile keywords*

Shows the contents of a TAR tape.

tarfile {*a pathname*} The file to be read.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Enables you to direct your output. The default is the stream ***standard-output***. Note that redirecting output to a printer can be particularly useful.

Show TAR Tape Command

Show TAR Tape *keywords*

Shows the contents of a TAR tape. It prompts for the tape specification.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Enables you to direct your output. The default is the stream

standard-output. Note that redirecting output to a printer can be particularly useful.

Show Time **Command**

Show Time *keywords*

Displays the current day, date, and time.

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Show Users **Command**

Show Users *user-spec keywords*

Shows the users logged into *host*.

user-spec {*user*, *user@host*, *user@site*, *@host*, *@site*} The user or users to locate and the host or site to search. The default is to show all users logged in at the local site.

keywords :Format, :More Processing, :Order, :Output Destination

:Format {Brief, Normal, Detailed} How much information to display.

Brief Gives the user's login name, the host name and the idle time only.

```
kjones wombat 1:12
```

Normal Gives the user's full name and information about the world they are running in addition to the information displayed by brief.

```
kjones  Kingsley Jones   Wombat 1:12 Computer Room (x515) (Release 7.2)
```

Detailed Displays the information in the user's namespace object.

The default when only one user is specified is normal, if more than one is specified the default is brief.

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Order {Host, Idle, User} The order in which to sort the entries in the display.

Host	Sorts alphabetically by host name.
User	Sorts alphabetically by user login name.
Idle	Sorts by idle time, from active to greatest idle time.

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Start Commands

Start GC Command

Start GC *keywords*

Controls the operation of the Garbage Collector. Start GC with no keywords turns on both dynamic and ephemeral garbage collection.

keywords :Cleanup, :Dynamic, :Ephemeral, :Immediately, :Selective

:Cleanup {Yes, No, Ask} Whether or not to run GC Cleanups to attempt to free address space. The default is No. The mentioned default is Yes, which does the maximum cleanup possible. Ask queries you about each cleanup before performing it. Start GC :Cleanup without other keywords does not perform a GC or alter the mode of the background GC. See the section "GC Cleanups".

```
Command: :Start GC :Cleanup (Yes, No, or Ask [default Y
GC Cleanup Tasks
  Reset all input histories?  Yes  No
  Reset all presentation histories?  Yes  No
  Reset all editor histories?  Yes  No
  Reset LISP-TOP-LEVEL variables such as '* and '+?  Ye
  Reset interactor output histories?  Yes  No
  Clear some resources?  Yes  No
<ABORT> aborts, <END> uses these values
```

:Dynamic — {Yes, No} Enables or disables the dynamic level of incremental GC.

:Ephemeral — {Yes, No} Enables or disables the ephemeral level of incremental GC.

:Immediately — {Yes, No, In-Place, By-Area} Performs a complete garbage collection right now. The mentioned default is Yes. In-Place garbage collection does not copy objects; rather, it compacts good objects to the bottom of each region. Since there are never two copies of an object during the garbage collection, the virtual memory (paging space) required for GC is reduced. It is slower and non-interruptable, and thus is recommended only when execution speed and interaction are not important and when there is insufficient disk space for normal garbage collection.

Start GC :Immediately By-Area is the same as Start GC :Immediately Yes :Selective Yes.

Start GC :Immediately also offers to run GC Cleanups.

:Selective — {Yes, No} Specifies areas in which to collect garbage. When your address space has shrunk to where there is not quite enough free space for Start GC :Immediately to complete, :Selective suggests some areas to flip that will maximize the amount of space reclaimed without risking running out of space completely. :Selective can be used with both Start GC :Immediately Yes and Start GC :Immediately In-Place.

For more information about the process of Garbage Collection, see "Theory of Operation of the GC Facilities" and "Invoking the Garbage Collection Facilities".

Start Printer **Command**

Start Printer *printer*

Starts the specified printer printing its print queue after you halt it.

printer — The name of the printer you are starting.

Note that you can halt, start, or reset a spooled printer from any machine on the network. For more information, see the section "Halt Printer Command".

Start Process Command

Start Process *process*

Starts a process that has been halted with Halt Process.

process — A process. You can press HELP or use the Show Processes command for a list of all the processes currently running in your environment.

Unmonitor **Commands**

Unmonitor Variable Command

Unmonitor Variable *symbol keyword*

Stops monitoring one or all special variables in memory.

symbol — {*location*, RETURN} A *location* specifies one location that you want to stop monitoring. Enter the name of a symbol and, optionally, its Value-Cell or Function-Cell. (See the :Cell keyword description below.) Press the RETURN key if you want to stop monitoring all locations.

keyword — :Cell

:Cell — {Value-Cell, Function-Cell} Specifies which cell within the location you want to stop monitoring. The Debugger gives you two choices: Value-Cell or Function-Cell. (Default is Value-Cell.)

Suggested mouse operations

- To unmonitor a location: Point the mouse at a locative, structure slot, or instance variable that was previously monitored and press c-m-sh-Middle.

Undelete **Commands**

Undelete File **Command**

Undelete File *pathname keywords*

Undeletes a deleted file, if the host on which the file resides supports undelete. It prompts for the name of a file to undelete. It displays a message if the specified file does not exist.

pathname The pathname of the file to be undeleted. The default is the usual file default.

keywords :More Processing, :Output Destination, :Query

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No, Ask} Whether to ask for confirmation before removing the delete flag on the file. The default is No.

Verify Commands

Verify Distribution Command

Verify Distribution *keywords*

Reads a Distribution from tape, floppy, or disk, and compares the contents of each file in the distribution to the corresponding file in the file computer. As the files are being compared, a line is typed out for each, to note which file is currently being compared. If there is any discrepancy in length or data, a proceedable error is signalled, describing the failure to compare.

The Verify Distribution command is useful when:

- A new distribution has just been created: the tape, floppy, or disk can be checked quickly against its sources for validity.
- A distribution has just been restored at a client site: the files created can be verified quickly against the distribution on tape, floppy, or disk.

keywords :More Processing, :Output Destination, :Use Disk

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Use Disk On all machines other than a MacIvory, the choices are {Yes, No}. If Yes, the input is read from a tape image file on disk. The default is No. The mentioned default is Yes.

On a MacIvory, the choices are {Tape, Disk, Floppy}. Disk indicates the hard disk, and Floppy indicates the floppy disk. These two values also read a tape image file from disk or floppy. Tape means to read from tape; this is the default.

Write Commands

Write Carry Tape Command

Write Carry Tape *pathname(s) keywords*

Dumps a file or set of files to a Carry-Tape. See the section "The Carry-Tape Dumper". You can dump any type of file. Character files are dumped and reloaded using the Genera character set as an interchange medium. Binary files are dumped and reloaded with the proper byte size as long as either of the following is true:

- The file is of one of the system's known canonical types.
- The operating system on which the file resides knows and can supply the byte size.

pathname The pathname(s) of the file(s) to write on the tape.

keywords :Author, :More Processing, :Output Destination, :Query, :Since, :Tape Spec, :Verify

:Author Dump files written by specific user(s).

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Query {Yes, No} Whether to ask before dumping files to tape. The default is Yes.

:Since Dump files newer than the date specified.

:Tape Spec The host and device for the tape. The default is the default drive on the local machine. For more information, see the section "Tape Specifications".

:Verify {Yes, No} Whether to compare the tape against the disk after dumping. The default is No.

Write Image File Command

Write Image File *image pathname format keywords*

Saves an image in a file.

image The image to be written to a file.

pathname The file in which to save the image.

format The format of the data in the file. The possible image file formats are:

BIN	Symbolics binary file (data only, compatible with all architectures)
Compact Bitmap	X windows "compact" bitmap
EPS/Macintosh	Postscript in the data fork, PICT in the resource fork
MacPaint	MacPaint canvas (fixed size)
PICT	Macintosh picture record
Portable Bitmap	X windows "portable" bitmap (1's and 0's)

	PostScript	Reads any PostScript images, writes EPSF compatible
	TIFF	Tagged Image File Format
	X Bitmap	X windows bitmap (C code)

keywords :More Processing, :Output Destination

:More Processing {Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination {Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

Write Partition Map Command

Write Partition Map *SCSI-address keywords*

Writes a new partition map on an optical disk. This command erases any data or partitions already on the disk.

SCSI-address The SCSI address of the optical disk drive.

keywords :Partition Map Size, :Apple Driver Size, :Apple Hfs Size, :FEPFS Size

The *keywords* enable you to control the size of four partitions. The values are in number of blocks, or Rest. If you supply keywords, three of the partitions should be specified as integers, and one should be Rest (indicating that the remaining space should be used for that partition).

:Partition Map Size
The default is 40 blocks.

:Apple Driver Size
The default is 30 blocks.

:Apple Hfs Size The default is 0 blocks.

:FEPFS Size The default is Rest.

Write TAR File Command

Write TAR File *pathnames*

Writes a File in TAR format.

pathnames The pathnames of the file(s) to write.

keywords :More Processing, :Output Destination, :Relativize, :Since

:Mode {Binary, Heuristicate, Query, Text} Mode in which to perform the copy. Binary means binary bytes with no character set translation. Heuristicate means try to determine automatically per file. Query, the default, means ask for each file. Text manes text characters with UNIX character set translation.

:More Processing
{Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Relativize {Yes, No} Whether to try to write the pathname as a relative UNIX pathname, that is relative to the current UNIX directory instead of starting at root. The default is No. The mentioned default is Yes.

:Since {*universal-time-in-the-past*} Writes only files later than the specified date.

Write TAR Tape **Command**

Write TAR Tape *pathnames*

Writes a tape in TAR format. It prompts for the tape specification.

pathnames The pathnames of the files to write on the tape.

keywords :More Processing, :Output Destination, :Relativize, :Since

:Mode {Binary, Heuristicate, Query, Text} Mode in which to perform the copy. Binary means binary bytes with no character set

translation. Heuristicate means try to determine automatically per file. Query, the default, means ask for each file. Text manes text characters with UNIX character set translation.

:More Processing
{Default, Yes, No} Controls whether **More** processing at end of page is enabled during output to interactive streams. The default is Default. If No, output from this command is not subject to **More** processing. If Default, output from this command is subject to the prevailing setting of **More** processing for the window. If Yes, output from this command is subject to **More** processing unless it was disabled globally (see the section "FUNCTION M").

:Output Destination
{Buffer, File, Kill Ring, None, Printer, Stream, Window} Where to redirect the typeout done by this command. The default is the stream ***standard-output***.

:Relativize {Yes, No} Whether to try to write the pathname as a relative UNIX pathname, that is relative to the current UNIX directory instead of starting at root. The default is No. The mentioned default is Yes.

:Since {*universal-time-in-the-past*} Writes only files later than the specified date.

Login Functions and Variables

After cold booting, you are in a window named Dynamic Lisp Listener 1. You are now ready to log in. Logging in tells Genera who you are, so that other users can see who is logged in, you can receive messages, and your init file can be run. An init file is a Lisp program that is loaded when you log in; you can use it to set up a personalized environment.

If your login name is KJones, you can log in in any of the following ways. (Note that the examples are given in uppercase and lowercase, but the machine is not case sensitive. You can use all uppercase, all lowercase, or mixed case as you prefer.)

- To log into the default host machine, using your init file, type

```
Login KJones
```

- To log into your machine, without your init file, type

```
Login KJones :init file none
```

- To log into another machine "sc3", using your init file, type

```
Login KJones :host sc3
```

If the host machine you log in to is a timesharing computer system, you must have a directory and account on that system.

For information about how to write init files, see the section "Customizing Genera".

When you log out, it should be possible to undo any personalizations you have made so that they do not affect the next user of the machine. Therefore, anything done by an init file should be undoable. Thus, for every form in the init file, you should add to the list that is the value of **sys:logout-list** a Lisp form to undo its effects. The functions **login-forms** and **zl:login-setq** help make this easy.

Login **Command**

Login *user keywords*

Logs the *user* into Genera.

user	Any string. Your user ID.
keywords	:Host, :Init File
:Host	{Local, *any-host-name*} A particular host computer. Local as an argument to :Host is particularly useful if your namespace system is down and you wish to log in to your Machine without having it try to use the namespace database. The default comes from `home host` for your user object in the namespace database.
:Init File	{Default-Init-File, File, Pathname, None} Whether to load your init file. The default is Default-Init-File. To avoid loading your init file, use :Init File None.

If someone is already logged in when you give the Login command, that user is logged out. If this happens, you see the message

```
Warning -- You are logging out from program-name
```

.

zl:login *user-name* &key *:host (:load-init-file* **t***) :no-query-when-unknown :init-file-to-load* *Function*

Logs the user *user-name* in to Genera. *host* is a particular host computer. If the value of *load-init-file* is **t**, as it is by default, the user's init file is loaded from *host*. If the value of *load-init-file* is **nil** the init file is not loaded. This is the Lisp version of the Login command in the Command Processor. See the section "Login Command".

The user object in the namespace database that contains your *user-name* also contains the name of the host(s) where your mail and init files reside. Therefore, you seldom need to supply a *host* argument to **zl:login**.

You can log in as a registered user by not specifying a host, or you can log in to a specific host as a user on that host, not registered in the Genera namespace database.

If *host* requires passwords for logging in, you are asked for a password. When logging in to a TOPS-20 host, typing an asterisk before your password enables any special capabilities you might be authorized to use.

If anyone is logged in to the machine already, **zl:login** logs that user out before logging in *user-name*. See the function **zl:logout**. **zl:login** also runs the **:login** initialization list. See the section "System Initialization Lists".

When **zl:login** loads an init file, it looks for a file whose name depends on the host. See the section "Init-File Naming Conventions". Init files should be written using **login-forms** so that **zl:logout** can undo them. Usually, however, you cold boot the machine before logging in, to remove any traces of the previous user.

A typical use of **zl:login** looks like this:

```
(zl:login 'kjones)
```

A typical use of **zl:login** to avoid loading your init file looks like this:

```
(zl:login 'kjones :load-init-file nil)
```

Supplying all the arguments might look like this:

```
(zl:login 'kjones :host local :load-init-file nil)
```

The host `local` is particularly useful if your namespace system is down and you wish to log in to your machine without having it try to use the namespace database.

If you supply an unknown user id and do not specify **:host**, you are given an opportunity to specify a particular host for the current login session, and to add the user object thus created to the network database (accomplished via Edit Namespace Object or **tv:edit-namespace-object**) for subsequent logins. You can instead select the Retry option, which is useful when the namespace server did not respond to your initial **zl:login** request.

The **:no-query-when-unknown** and **:init-file-to-load** keywords are for internal use by the system, and are not intended for users.

zl:logout *Function*

First, **zl:logout** evaluates the forms on **sys:logout-list**. Then it sets **zl:user-id** to an empty string and **sys:logout-list** to **nil**. Then it runs the **:logout** initialization list and returns **t**. See the section "Initializations".

zl:user-id *Variable*

The value of **zl:user-id** is either the name of the logged in user, as a string, or an empty string if there is no user logged in. It appears in the status line.

sys:logout-list *Variable*

The value of **sys:logout-list** is a list of forms that are evaluated when a user logs out.

login-forms &body *forms* *Function*

A special form for wrapping around a set of forms in your init file. It evaluates the forms and arranges for them to be undone when you log out.

login-forms always evaluates the forms, even when it does not know how to undo them. For forms that it cannot undo, it prints a warning message.

In the following example, **login-forms** arranges for the base to be reset at logout to 10 (the default) and for **bar** either to become undefined or to get its old function definition. It would warn you about **quux** being impossible to undo.

```
(login-forms
        (zl:setq-standard-value base 8)
        (zl:setq-standard-value ibase 8)
        (defun bar (x y) (+ x y))
        (quux 3))
```

You can create functions to undo forms that **login-forms** does not recognize. To undo a given form, you put a property on the symbol that is the car of the form to undo. For example, to create a function to undo **quux**:

```
(defun (:property quux :undo-function) (form)
  `(undo-quux ,(cadr form)))
```

The value returned by an undo function is a form to be evaluated at logout time.

zl:setq-globally &rest *vars-and-vals* *Function*

Use the Symbolics Common Lisp function **symbol-value-globally** instead of this. You use **setf** with **symbol-value-globally** to set global values in your init file.