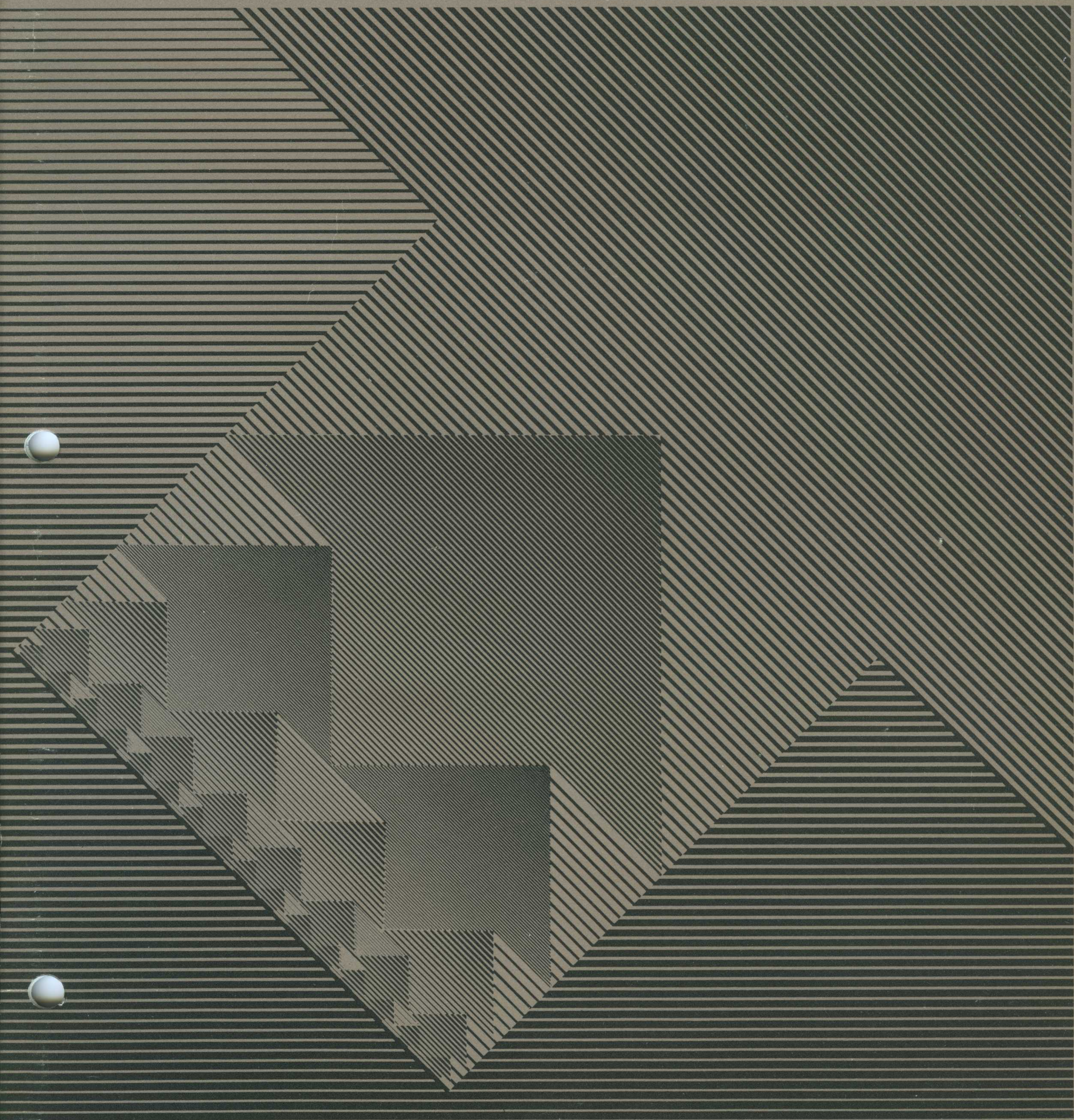
symbolics™

Release Notes for System 78



symbolics™

Release Notes for System 78

February 1982

Cambridge, Massachusetts

Prepared by Symbolics, Inc.
Written by Daniel L. Weinreb

This document corresponds to System 78.

Table of Contents

Page

Introduction 1

Notation Conventions 1

Summary of Changes 3

Changes to Take Note Of 5

Changes That May Be Safely Ignored 7

General Information 13

Introduction

These notes describe the changes in the Symbolics software release that includes System version 78. The changes are divided into two sections. The first section explains those changes that might potentially cause problems or confuse users when they start using the new system. The second section lists new features or minor changes. It is important for you to read about the changes in the first section, in order to understand problems that might otherwise arise when you start using the new software release; however, you may safely ignore the second section, or read it at your leisure.

Notation Conventions

The keys with black lettering (like SHIFT or META) are shift keys, designed to be pressed in combination with other keys. They do not themselves transmit characters. Their combinations are shown hyphenated to remind you to press them at the same time as the associated key, not before.

The keys with white lettering (like X or SYSTEM) all transmit a character. Combinations of these keys are meant to be pressed in sequence, for example, SYSTEM L means to press the SYSTEM key, release it, and then press the L key.

The CTRL and META key combinations are abbreviated with c- and m-; the SUPER, HYPER, and SHIFT keys with s-, h-, and sh-, respectively. For example, the combined keypress META-X is pronounced "meta x" and written as "m-X".
This document uses the following notation conventions:

Appearance in document	*Representing*
send, chaos:host-up	Printed representation of Lisp objects in running text.
RETURN, ABORT, c-F	Keyboard keys.
SPACE	Space bar.
`login`	Literal type-in.
`(make-symbol "foo")`	Lisp code examples.
function name *arg1 arg2*	Syntax descriptions of definitions.
Undo, Tree Edit Any	Command names in Zmacs and Zmail appear with initial letter of each word capitalized.
Insert File (m-X)	Extended command names in Zmacs and Zmail. Use m-X to invoke one.
[Map Over]	Menu items.
(mouse-R)	Mouse clicks; L=left, M=middle, R=right.

Mouse commands use notations for menu items and mouse clicks in the following ways:

Square brackets delimit a mouse command; slashes (/) separate the members of a compound mouse command. The notation indicates which button to click only when that differs from the standard. For a single menu item, always click left. For example, the following two commands are exactly the same:

> [Previous]
> [(mouse-L) Previous]

For a compound command, always click right on each menu item except the last, where you click left. For example, the following two compound commands are exactly the same:

> [Map Over / Move / Hardcopy]
> [(mouse-R) Map Over / (mouse-R) Move / (mouse-L) Hardcopy]

For all other cases, the notation shows explicitly which button to click. For example:

> [Map Over / (mouse-M) Move]

Some more examples:

- Suppose you are to click right on menu item [Map Over], then click right on menu item [Move], then click left on menu item [Hardcopy]. The notation is:
  [Map Over / Move / Hardcopy]
- Suppose you are to click left on menu item [Previous]. The notation is:
  [Previous]
- Suppose you are to click right on menu item [Map Over], then click middle on menu item [Move]. The notation is:
  [Map Over / (mouse-M) Move]
- Suppose you are to click right on menu item [Previous]. The notation is:
  [(mouse-R) Previous]

Summary of Changes

Changes to take note of:

1. System 78 requires new microcode.
2. If the **qfasl** and source of a file are in different directories, you must recompile the file.
3. Function specs have changed; **fdefine, compile, trace,** *et. al.* are affected.
4. The **:close** message to a Chaosnet stream has changed. Most programs should not be affected, but some might be.
5. The system menu has changed - it is a more powerful superset of what it used to be.
6. You may get spurious warnings about redefining functions; if you do, recompile the appropriate files.
7. There is a new compiler lockout feature; only one process may run the compiler at a time.
8. UNIX is now supported as a file system. If you use UNIX systems, you must know about the UNIX representations of standard pathname types.

Changes that may be safely ignored:

A. Streams have been changed; there is a system of flavors that you can use if you want to, though you don't have to.

B. **open** has a new argument syntax (the old one still works).

C. **with-open-stream** exists.

D. There are new high-level functions for using the Chaosnet.

E. New **:handler** function spec.

F. **si:print-readably** controls unreadable forms.

G. Changes to **process-run-function**.

H. Source files are remembered for all definitions.

I. Files with **compile-flavor-methods** ought to be recompiled if you care a lot about efficiency.

J. New features in **make-system**.

K. Some changes in low-level disk control.

L. The scavenger working set size can be controlled.

M. The editor tries to give you more information in its display.

N. Preparing now for coming new error system.

O. The Greek letter convention for the c- and m- keys is obsolete.

P. New **qsend** features.

Q. Small language changes.

R. New editor feature to turn on **Auto Fill** mode.

S. Hosts are now represented as instances.

T. A new **process-warm-boot-action** is available.

U. Minor bugs in floating point have been fixed.

V. New **no-msg-p** argument to **load**.

W. New **extra-lists** argument to **subset** and **subset-not** (**rem-if-not** and **rem-if**).

Changes to Take Note Of

1. System 78 requires microcode 836. It fails to cold-boot with the previous microcode (793). The System 74 fails to work with the new microcode also.

2. In order to interact properly with m-. in the editor, all files whose **qfasl**'s reside in a different directory than the source needs to be recompiled in System 78. The present scheme of guessing the source file pathname based on the truename of the source has proved too bug-prone, and the correct information is now recorded by the compiler.

3. Function specs have been completely reimplemented. As a side-effect of this the function spec **(foo bar)** as an abbreviation for **(:property foo bar)** is now accepted only by defining forms such as **defun, defmacro,** and **defselect**; the full form must be used with the function-manipulating functions such as **fdefine, compile, trace,** etc. Formerly the short form worked partially with these functions. If you get an error message that something is not a valid function spec, and it used to work in System 74, this is probably the reason.

4. The **:close** message to a chaos stream now behaves like the **:eof** message if not given an **abort-p** argument. The connection is also freed, so this need not be done manually. Note that the **chaos:stream** function still gives bidirectional streams, but in many cases you may only want the unidirectional flavor (see below for more details).

5. The system menu has been made more useful; the "Other" command have been integrated into the menu, as have some useful screen editor commands and some commands that are like what you can do with the SYSTEM key. If you just pop one up and look at the mouse documentation for each item, it should be clear what all of the commands mean. The function **tv:add-to-system-menu-programs-column** should be called by user programs that want to add themselves to the system menu.

6. In some obscure cases, you may get undeserved warnings about redefining system functions when you load old **qfasl** files into System 78. For example, if you had a **defstruct** with the same name as a system function, and you last compiled the file in certain system releases, this may happen. Just recompile the file in System 78 and the new **qfasl** file won't produce the warning.

7. Only one process is permitted to be running the compiler at a time. It used to be that if you tried to run the compiler in two processes at once, you got strange and mysterious errors; now the second process waits for the first one to finish. It waits in process state "**Compiler**", which you may see in the who-line if this happens to you. This means that if a process goes into the error handler during compilation and you then try to run the compiler in a second process without aborting out of the error handler in the first process, the second process waits.

8. UNIX is now a supported file system. Since UNIX filenames can only be 14 characters long, the representations of the standard pathname "types", such as **LISP** and **TEXT**, are stored in abbreviated forms in UNIX filenames. Here is a complete list of standard system types, and their UNIX representations:

System name	UNIX abbreviation
LISP	l
TEXT	tx
MIDAS	md
QFASL	qf
PRESS	pr
(PDIR)	pd
QWABL	qw
PATCH-DIRECTORY	pd
BABYL	bb
MAIL	ma
INIT	in

In System 78, UNIX pathnames whose "type" fields are these abbreviations parse as the expanded type name, but pathnames whose "type" fields are the same as a standard type name (such as "foo.lisp") do not parse at all. The reason is that if "foo.lisp" were to parse into a pathname with name "foo" and type "lisp", then it would be un-parsed back as "foo.l", which would cause very strange and undesirable behavior. This problem will be fixed in the next release.

Changes That May Be Safely Ignored

A. A major change in this system is a thorough reimplementation of streams. All old streams continue to work; however, a system of flavors has been set up to make it easier for you to write new streams. All buffered streams in the system (for example, files, network, mag tape) are now built out of these flavors, eliminating much duplicated code and the consequent inconsistencies and inefficiencies. This new basis for streams is not yet documented, but the source code may be found in the file **sys: io; stream lisp**. The **:finish** message has been changed incompatibly; it now implies a **:force-output** (which is what it ought to have done all along). There is a new message to streams, **:string-in,** which reads efficiently into an array; it and **:string-out** are the right primitives from which to build most forms of binary record I/O. As mentioned above, the **:close** message to a Chaosnet stream is now consistent with all other streams. A new message to buffered streams, **:read-input-buffer,** behaves exactly like **:get-input-buffer** but returns the values **array, index,** and **limit**. This should be used in all new code instead of **:get-input-buffer,** which will be phased out.

B. There is a new, preferred way to pass arguments to **open**. The argument list is now **(filename &rest keywords),** and the keywords should be the same things that used to be in the list of keywords that was **open's** second argument. The old-style way of calling **open** continues to work. The new style is more consistent with the way keyword arguments work elsewhere, and it more convenient for programs to use since it saves them the trouble of creating a list of arguments; they can just pass the arguments directly. Also to aid programs, there is a new keyword called **:direction** that is followed by a value that can be **:input, :output,** or **nil** (**nil** is used for **probe** openings, which don't allow data transmission at all. So a program can do

```
(open pathname ':direction (if input-p ':input ':output))
```

with-open-file has been changed analogously:

```
(with-open-file (stream pathname
                        ':direction (if input-p ':input ':output))
   ...)
```

C. There is a new special form, **with-open-stream,** which is completely analogous to **with-open-file** except that you specify a form whose value is the stream, rather than arguments to **open**. This is used with non-file streams.

D. The preferred way of interacting with the chaosnet at the highest level is now via two new functions, **chaos:open-stream** and **chaos:make-stream**. **chaos:open-stream** takes a host, contact name, and set of keyword options and returns an open stream to that host. **chaos:make-stream** takes an open connection and options and returns a stream. This is useful when the connection has been gotten via a **chaos:listen,** for example. Both functions accept the following keywords:

:direction
: **:input, :output,** or **:bidirectional**. Default is **:bidirectional.**

:characters
: Boolean. Default is **t**. If non-**nil,** character rather than binary data is to be sent.

chaos:open-stream also accepts these keywords relating to the connection itself:

:error Boolean. Default is **t**. A string is returned on an error for **:error nil**.

:timeout
Number of 60ths of a second. Default is 10 seconds.

:window-size
Fixnum. The default is like **chaos:connect** formerly.

E. A new function spec (**:handler** *flavor message*) refers to the function that is invoked when an object of flavor *flavor* is sent the message *message*. This is distinct from the **:method** function spec, which refers to the function defined only in flavor *flavor*, and possibly inherited by many other flavors.

F. A new variable **si:print-readably** enables you to specify that what you are printing is intended to be read with **read** again, and should not contain any unreadable forms. When **si:print-readably** is bound to a non-**nil** value and an attempt is made to print an unreadable object, an error is signalled. Anyone who defines his own **:print-self** messages should be aware of a new macro, **si:printing-random-object**, which must be used to preserve the functionality of **si:print-readably**. It is documented in the source.

G. **process-run-function** now does what **process-run-temporary-function** used to do. This means that the writeup in the manual is now obsolete. **process-run-temporary-function** is still around, but is obsolete and will go away eventually.

The first argument to **process-run-function** (and the two similar functions) may now be a list of alternating keywords and values rather than a string. The keywords are:

:name A string, defaulting to "Anonymous".

:restart-after-reset
Boolean, defaulting to **nil**. If it is non-**nil**, the **:reset** message kills the process.

:restart-after-boot
Boolean, defaulting to **nil**. If it is non-**nil**, a warm or cold boot kills the process.

:warm-boot-action
Same as for **make-process**, except that the **:flush** does what **:warm-boot-action nil** does, and **nil** (the default) means obey the **:restart-after-boot** keyword.

:priority
A number, defaulting to 0, which is a standard priority.

:quantum
A number of 60ths of a second, defaulting to 60.

process-run-restartable-function defaults **:restart-after-boot** and **:restart-after-reset** to **t** rather than **nil**.

Note that the names of processes, like the names of everything else, are strings. It used to work, at least to some extent, to use any Lisp object as the name of a process, and some people have been using symbols. This no longer works, as it happens.

H. The system now remembers source files for all functions, not just those named by symbols, and does not get confused by multiple definitions of different types with the same name (for instance, a function, a variable, a flavor, a defstruct, and a resource with the same name). Thus the editor m-. command is more likely to work now. The whole function spec and source file system has been reimplemented, and numerous bugs have been fixed. flavor-method-symbols (for example, **tv:basic-momentary-menu-after-deexpose-method**) no longer exist; the function spec (for example, **(:method tv:basic-momentary-menu :after :deexpose)** is now used for all purposes formerly served by the symbol).

I. Any files containing **compile-flavor-methods** should be recompiled in System 78. Old **qfasl** files still work; however, the system regards all combined-methods in them as obsolete and generates new ones at load time; thus the **compile-flavor-methods** is inutile.

J. **make-system** now offers to compile and load a new version of the file containing the **defsystem** for the system if it has changed. In fact, a new function has been provided, called **si:set-system-source-file,** that allows you to specify what file a system is defined in before that system is loaded at all. It is then loaded the first time **make-system** is done on it. This might be useful in your init file, for example. The argument to this function is the name of the system, that is the symbol that you would give to **make-system**.

Note for users of ITS: since the System system has strong opinions about file types, this only happens when the file with the **defsystem** form has a file type of **lisp,** that is an FN2 of > on ITS. Thus, if you have a file **FOO PKG,** and want to benefit from this feature, you should rename it to **FOOPKG >**.

The **:recompile** keyword to **make-system** is the same now as the **:compile** and **:reload** keywords together. This should be easier to remember.

K. The A-memory variable **sys:%disk-read-compare-enables** has been renamed to **sys:%disk-switches**. Bit 2 in this variable now controls the multiple page swapout feature. When a page is swapped out and this bit is on, the system looks to see if the next higher page in virtual memory is present in physical memory and also needs to be written on disk. If so, it is appended to the transfer. Currently the maximum size of transfer which can be built in this way is set to 20 pages. The pages in question are not actually removed from core, but just changed to "Read-write-first" status. If they are later selected for swapout before they are modified again, a disk write is saved. A new meter, **%count-disk-page-write-appends,** records the number of pages added to transfers in this way.

Bit 3 in **sys:%disk-switches** now controls the multiple page swapin feature which exists in microcode version 826 and later. When a swapin is called for, the system decides how many pages it would like to transfer with one disk op. In System 78, this computation depends the area of the page being brought in. **sys:area-swap-recommendations** is a new area number indexed area which holds the recommended swap size minus one per area. (That is, 0 says just swap the needed page, 1 try to append one page to that, etc). The user should not store in **sys:area-swap-recommendations** directly, but should use **(si:set-swap-recommendations -of-area** *area-number n***)** Currently, 17 is the maximum value of *n* that should be used. There is not much experience yet as to what are good values to put in here. **(set-all-swap-recommendations 2)** is a good way to turn on the feature and start seeing what happens. Exactly what is the best initial setting is determined thru experience. Generally speaking, the more main memory the machine has, the higher the swap sizes should be. There frequently seems to be a rather dramatic improvement in context switch times (such as entering the editor for the first time, etc.) Another new meter, **%count-disk-page-read-appends**, records the number of pages added to transfers in this way.

L. The size of the scavenger working set, formerly fixed at the low value of 12 pages, may now be set at any time by **(si:set-scavenger-ws** *value***)**. The way that this works is that main memory is divided into two pools; one pool is only used by the user program, while both the user and the scavenger contend for the other pool. The *value* is the number of pages of main memory (256 words each) in the shared pool. Setting this value higher than 12 seems to improve the performance of the GC significantly. Values of 400 seem to work well for 256K machines, larger values should probably be used for machines with more memory. Please report your experiences setting this parameter, since it is unclear at this time what the default value should be. As part of this change, the meaning of the **si:%scavenger-ws-enable** variable has been changed. It is now the dividing point in physical memory between the two pools. (As a semi-compatibility feature, however, **si:%scavenger-ws-enable** of **nil** still turns off the scavenger working set feature.) The former low value of the scavenger working set parameter may have accounted for unacceptable garbage collector paging performance in some applications.

M. When the editor is reading a command argument from the mini-buffer and some sort of command or filename completion is present, the right-hand side of the mode line says "**(Completion)**". Likewise, when a search string is being read that may contain special search characters (such as <and> and <or>), it says "**(Extended search characters)**". The version of the file last read or written is now displayed in parentheses in the mode line.

N. A new object oriented error handling and condition system will be installed in one of the next few major system releases. The major change for simple programs is that functions like **chaos:connect** and **open** which take a no-error argument and now return a string of the error will instead return an error instance. Since we imagine that there are a lot of these simple programs, and to help ease you into this change, there is a new global function **errorp** in System 78. It is the same as **stringp**. When the new system is installed, it will be different, but will still correctly recognize the error return case of the above mentioned functions.

O. The old convention of using alpha to mean `c-`, beta to mean `m-`, and so on, has been declared obsolete. The use of this convention in some `SYSTEM HELP` messages has been replaced with the new convention, namely `c-`, `m-`, and so on. The **format** ~C directive has been changed to use the new convention. The Greek letters continue to be accepted by #/ for compatibility only. If you see the system trying to type Greek at you using this convention, tell us and we will put a stop to it.

P. Sometimes, users may wish not to be interrupted with interactive messages. A new function called **chaos:qsends-off** has been created for such occasions. If you give it a string argument, the string is returned to anyone who tries to send to you. Otherwise, they just get a note saying that you are not accepting Qsends. **chaos:qsends-on** ungags you. Also, **chaos:shout** allows you to send a message to all Lisp Machines at your site.

Q. The first argument to **break** is now optional; **(break)** <=> **(break nil)**. This means that there are two easy ways to write a breakpoint into your program: **(break)** gets a read-eval-print loop, and **(err)** gets the error handler. (These are the programmatic equivalents of the BREAK and m-BREAK keys on the keyboard.)

The initial value of **readtable** is now a copy of **si:initial-readtable**. You should never change the contents of **si:initial-readtable**; this should be left pristene so that other programs can make copies of it in order to get a fresh readtable. Change **readtable** instead. (Some people who have reader macros to cause font change sequences to be ignored may be affected by this change.)

R. There is a new function called **zwei:auto-fill-if-appropriate,** analogous to **zwei:electric-shift-lock-if-appropriate**. If you do

```
        (login-setq zwei:text-mode-hook 'zwei:auto-fill-if-appropriate)
```

in your init file, then when you edit a file whose file property list has a **Mode** property of **Text,** and doesn't have a not **nil Nofill** property, the buffer is put into Auto Fill mode.

S. Hosts now have standard representations as instances. In particular, the **host** instance variable of a pathname is one. This change should not affect any user code. The variable **chaos:host-alist** is now gone, since the host table can handle multiple networks. **si:parse-host** is the way to convert a string into a host object, from which you can get the standard name or nicknames.
(si:get-host-from-address *number* **':chaos)** is the way to get a host object from a number.

T. There is a new process-warm-boot-action, **si:process-warm-boot-delayed-restart,** and it is now the default instead of **si:process-warm-boot-restart**. The effect is that random processes do not get restarted any more before the system has finished initializing itself. Processes needed to initialize the system, such as the Keyboard and Chaosnet processes, do not use this warm-boot action.

U. Some small bugs associated with floating-point numbers have been fixed; some numerical programs may get different results. The principal source of roundoff error in the reader (inaccurate powers of ten) has been corrected; many more numbers are read as the best flonum approximation, although large and small numbers still have an error of 1 least significant bit in some cases. (There are no longer any cases with an error of more than 1 LSB). A fencepost error in floating-point subtraction has been corrected; formerly if the subtrahend was exactly equal to 1.0 times the weight of the last significant bit in the minuend, the answer would be the minuend, which is incorrect.

V. The **load** function now takes a fifth argument named **2o-msg-p**. If the value of this argument is **t** (it defaults to **nil**), then **load** does not print out the message that it usually prints (that is, the message that tells you that a certain file is being loaded into a certain package).

W. The functions **subset** and **subset-not** (also known as **rem-if-not** and **rem-if**) now take a **&rest** argument, named *extra-lists*, in addition to the first two arguments. If *extra-lists* is present, each element of *extra-lists* (that is, each further argument to **subset**) is a list of objects to be passed to *predicate* as *predicate*'s second argument, third argument, and so on. The reason for this is that *predicate* might be a function of many arguments; *extra-lists* lets you control what values are passed as additional arguments to *predicate*. However, the list returned by **subset** is still a "subset" of those values that were passed as the first argument in the various calls to *predicate*.

General Information

If you load a file that tries to **setq** the variables **base** and **ibase**, and that file has a "**Base**" property in its -*- line, you will find that it doesn't work. This is because if the file has such a property, then the values of **base** and **ibase** are saved before the file is loaded and restored after it, so that they can be set to the value of the property during the loading. If you want to set the values of **base** and **ibase** in a file, don't give the file a "**Base**" property. By the way, there is no such thing as an "**Ibase**" property; the "**Base**" property controls the base used when a file is read. Names of properties are not case-sensitive.

There is a a frequently-encountered annoying problem with the editor. The Lisp-parsing algorithms always assume that any line that starts with an open-parenthesis is the beginning of a top-level form. This is important for efficiency: it saves the parser from having to start at the very beginning of the buffer to find out what level it is at. However, occasionally you may have a program that includes a line that starts with an open-parenthesis that is not the beginning of a top-level form; this sometimes happens inside multi-line strings. When your file has such open-parentheses, the parser can get confused. If the parenthesis is inside a string, you can get around the problem by preceeding the parenthesis with a slash. The Lisp reader interprets the slash as a quoting character and ignores it; the editor parser doesn't see the parenthesis as being at the beginning of a line, so it doesn't get confused.

Notes

Notes

symbolics TM

Release Notes for System 78
#99091

Design: Schafer/LaCasse
Typesetting: Cover – Litho Composition Co.
Printing: Henry Sawyer Co.