Genera Workbook

Using This Book

Preface

This is the document to read when you're ready to sit down at your Symbolics computer. It teaches you the initial skills you need to use the Genera system effectively. We present concepts you need to know, as well as providing exercises and activities that turn those concepts into practical skills.

The material covered in the first group of chapters in this workbook is a prerequisite for all Symbolics Education Services Courses. However, you can complete the second and third sections after taking your first course. Practicing the material helps you learn it faster. Bring this workbook to all courses you take with Symbolics.

We expect that you have used a computer before; we do not expect that you have used a Symbolics computer before. You need to learn this material whether or not you are a programmer. This is not a programming textbook. We make no assumptions about your programming background or future; this workbook is for both programmers and non-programmers.

Every time we use a term for the first time, we put it in *italics*. If you see a term you don't recognize, be sure to check the glossary at the end of the workbook.

This workbook uses Symbolics documentation conventions and a few of its own. If you see a word or phrase enclosed in brackets, like [Help], it is a menu item. We use two different conventions with respect to Symbolics keyboard keys: in text, the key labelled HELP looks like this:

```
HELP
```

but in examples it looks like this:

```
<HELP>
```

to differentiate it as a single keystroke from the text around it.

It is very important to have a Symbolics computer available so that you can work through the hands-on activities, called walk-throughs. These walk-throughs are important to the learning process. If you do not have access to a machine and are enrolled in one of our courses, you can use one of our machines. Call the registrar at the training center where your course is being held to arrange time to use our machines before your course.

You can use this workbook with any configured Symbolics computer running Genera 8. The machine should already be configured for your site. If you are not sure which software release your machine is running, or if you don't know whether your machine has been configured, ask the person who is responsible for the Symbolics computers at your site. You might get a warning message if you do not have an *init file* and a *user object*. For instructions on setting up an init file and a user object, contact your site administrator or see the section "Workbook: The Namespace".

We have provided references to other Symbolics documentation at the end of each chapter. The section called **Documentation References** offers topic names for additional documentation. You can use either the printed documentation or Document Examiner to find additional information about any of the topics covered in this workbook as well as other topics of interest. (See the section "Workbook: Document Examiner".).

This book is a learning aid, not a reference manual. Use it as you would a science textbook, not as you would a dictionary. It is best to read the book in order. When you come to a walk-through, go to your machine and do the exercise before you continue your reading. The exercises are written specifically to follow one another. Follow the directions carefully, typing exactly what is written — when you are supposed to type a carriage return or space, we tell you to do so explicitly. Note that hardware-specific examples in this book are 3600-based. For details on MacIvory and UX-family machine platforms, see their respective user manuals. Genera and Lisp examples are platform independant.

The exercises in this material are not case-sensitive. Therefore, type in lowercase, if you wish. We show letters in uppercase only for typographical clarity; if we want you to hold down the SHIFT key, we say so. We describe our keyboard notation in more detail later. Symbolics computers are robust and it is hard to damage the machine. Feel free to experiment. Don't be nervous, and don't worry about hurting anything.

Occasionally, we give simplified explanations of complex topics, glossing over exceptions and quirks. This is to make the material easier for you to learn the first time through. We promise that none of our simplifications will leave you with misunderstandings.

Helpful Hints

Some of the following hints will be more understandable after you have read the first group of chapters. (See the section "Workbook: Basic Topics".) We put them here for easy access, but intentionally did not define terms, as they are defined later in the workbook. Reread this section after you complete the first group of chapters. Unless we tell you otherwise, we assume that Lisp is running for all of the exercises.

1. Make sure you are logged in for all of the exercises. See the section "Workbook: Getting Started".

2. Unless we tell you to cold boot, we assume you have *not* cold booted your machine. If you have, make sure you also have logged in. See the section "Workbook: Getting Started".

3. Press RETURN only when we say to do so.

4. If you try something and it does not work, try it again. If it still does not work, go on to something else or ask someone at your site for help.

5. If you have made several mistakes and want to start an input line over, press the CLEAR INPUT key.

6. Press the REFRESH key to clear the screen.

7. If you see **MORE** at the bottom of your screen, press the SPACE bar.

8. Press m-SCROLL to scroll back over your previous interactions (called your *output history*). You can use the command Clear Output History to get rid of this history when it becomes too large.

9. All of the operations described are generic. We use specific examples to illustrate them in this workbook but, after you have completed the exercises, try any of the commands presented here on your own work.

10. In the Command Processor, pressing the SPACE bar gives you information, so press it when we tell you to do so or when you do not see what we have indicated you should see. See the section "Workbook: Getting Around the System".

11. Pressing the ABORT key gets you out of most kinds of trouble.

Basic Topics

Introduction

This section covers the basic, most essential material in this workbook.

You are expected to know this material before you come to any class offered by Education Services. If you have been working with Symbolics Genera for more than a couple of months, you might already know this material, but you should review it for any little tricks or hints you might have missed. "Workbook: Basic Topics" has five chapters covering basic machine-oriented skills:

"Workbook: Getting Started"
Teaches you how to work with the console, mouse, and Symbolics keyboard and also how to log in, log out, and cold boot Lisp. The walk-through in the cold booting section are slanted towards the Symbolics 3600-family machines. If you have an Ivory-based machine, you should refer to the *User's Manual* for your machine type for hardware-specific information.

"Workbook: Getting Around the System"
Covers selecting different activities in the Lisp environment and using the various features of the Lisp Listener, such as Help and the Command Processor.

"Workbook: Zmacs" Covers elementary file concepts and very basic commands for editing any type of file. Commands for Lisp programming are not covered in this section or anywhere else in the document.

"Workbook: Getting Familiar with Files"
Goes into pathnames in a little more depth and explains basic file and directory commands for use with the Command Processor.

"Workbook: Document Examiner"
Introduces the basic features of the online documentation and gives you practice in the use of this valuable tool.

The best way to learn the material covered in this section is to read the manual and do the walk-throughs while sitting at a machine. Practice, using your own examples, improves your skills.

Your fluency with this material is essential for any Education Services class.

Getting Started

Introduction

This chapter familiarizes you with some elementary procedures for using the Symbolics keyboard, the mouse, and the screen. It also teaches you how to cold boot Lisp.

- Workbook: The Mouse
- Workbook: The Screen
- Workbook: The Keyboard
- Workbook: FEP and Lisp Processor
- Workbook: Cold Booting
- Workbook: Walk-through for Cold Booting
- Workbook: Logging in
- Workbook: Walk-through for Logging in
- Workbook: Logging Out
- Workbook: Walk-through for Logging Out
- Workbook: Getting Started Documentation References

The Mouse

The *mouse* is a little black or gray box with three buttons, which is connected to the back of the console (Figure 1). The three buttons are known as the Left, Middle, and Right buttons, according to their positions. The Macintosh mouse has only one button. If you are using a MacIvory with the Macintosh keyboard and mouse, the single button is Left. You use the arrow keys on the keypad while clicking the single button for Middle and Right. Down-arrow is Middle. Right-arrow is Right. Try rolling the mouse around on a smooth surface. If the clock in the lower left of

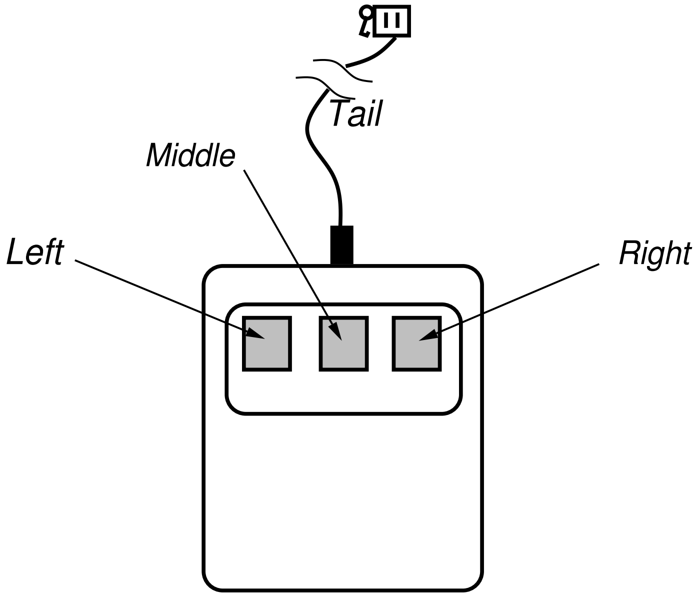


Figure 1. The Mouse

the screen is running, you can see the *mouse cursor*, a small black arrow, move on the screen as you move the mouse. In Symbolics documentation, the use of the word "mouse" often means the mouse cursor rather than the physical mouse.

Generally, you use the mouse by moving the mouse cursor to something on the screen and *clicking* (pressing and releasing quickly) one of the three mouse buttons. To get additional operations, you hold down one or more Symbolics keyboard keys while clicking a mouse button.

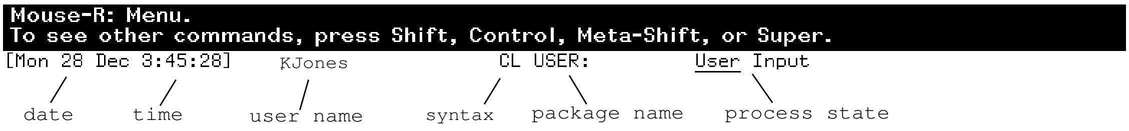
If you are just told to "click the mouse" or "click" on something, but are not told which of the buttons to use, you should click the left mouse button once.

The Screen

The very bottom line of your screen is the *status line* (Figure !). Develop the habit of looking at the status line periodically. Starting at the left, you see the date, the time, and the name of the current user. Near the center of the status line is a combined display of the *current Lisp syntax* and *current package* (typically, the syntax and package display as `CL USER:`). To the right of the package name is the *process state*. This is `User Input` when Lisp is waiting for input. When Lisp is busy, other states, such as `Run`, replace `User Input`. Genera supports typeahead, but if you do not know what is going to happen next this can cause problems, so make sure that `User Input` is showing before you type anything or click a mouse button.

As long as the clock on the status line is advancing, the Lisp Processor is still running.

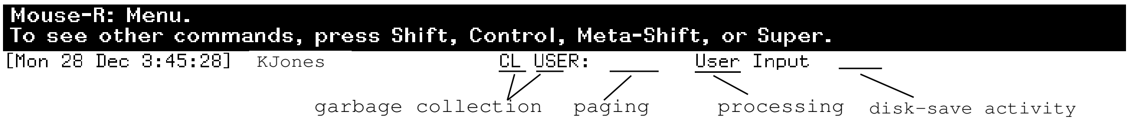
When your machine is actively processing, several *run bars* regularly appear under the process state (Figure !). These are small horizontal lines that appear when Lisp performs various operations. There are two small bars for *garbage collection*,



```
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Mon 28 Dec 3:45:28]   KJones          CL USER:        User Input
```

Figure 2. The Status Line

one for *paging*, one for *processing*, and one for *disk-save activity*. The run bars are located (in that order from left to right) under the current package name (Figure !). If you suspect that your machine is not responding to you, look at these bars. If they are flickering, your machine is doing something. Move the mouse and you see the processing bar flicker.



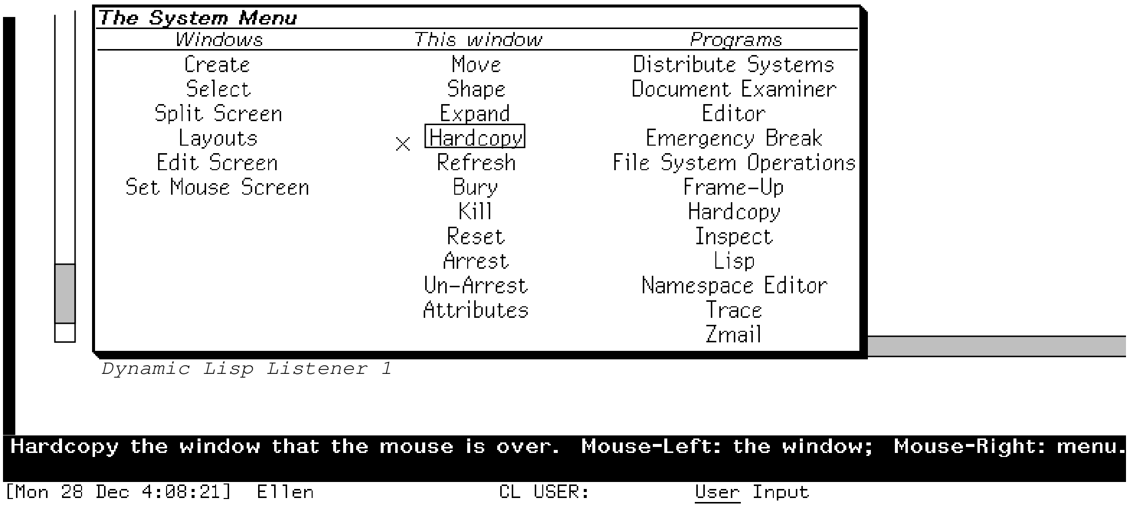
```
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Mon 28 Dec 3:45:28]  KJones          CL USER:        User Input
```

Figure 3. Run Bars

Just above the status line, you find two lines in reverse video. These are the *mouse documentation lines* (Figure !). As the name implies, these lines contain information about the mouse. Specifically, they tell you what happens when you click the mouse buttons. If there are many possible mouse clicks and results, these lines contain a lot of information. Move the mouse around the screen, crossing over the scroll bars at the left and bottom of the screen. As you move over different parts of the screen, the mouse documentation lines tell you what each mouse click does. When the mouse is over a *menu*, which is an area of the screen with a list or lists of choices displayed on it, the mouse documentation line provides information about the choices (Figure !).

The items on a menu are *mouse-sensitive*, which means that when the mouse cursor is moved over or near one of them a box appears around it. To see this, first bring up the *System menu* by holding down the `SHIFT` key while clicking Right. Move the mouse cursor over the menu items slowly and notice the changes on the mouse documentation lines (Figure 4). When you move the mouse cursor off the menu, the menu disappears.

On the left and the bottom of the screen, you can see long grayed-over rectangles with clear squares at each end. These are *scroll bars*. If you move your mouse into these scroll bars, the mouse cursor changes into a double-headed arrow and the information in the mouse documentation lines tell you what direction each mouse click scrolls the screen. The area of the scroll bar that is grayed-over changes depending on how much of the actual contents of the window are visible. Unless you have some text in the window to scroll over, you cannot scroll. In a later chapter, you will get to practice up-and-down scrolling using the scroll bars on the left of the screen. For right now, you can scroll left and right using the scroll bars at the bottom of the screen.



```
Hardcopy the window that the mouse is over.  Mouse-Left: the window;  Mouse-Right: menu.
[Mon 28 Dec 4:08:21]  Ellen              CL USER:          User Input
```

Figure 4. The Mouse Documentation Lines

Items on your screen are mouse-sensitive just as the items in a menu are. As you move your mouse over the screen, you see outline boxes appear around some of the text on the screen. You can execute commands by moving the mouse cursor over an item until the outline box appears and then clicking the mouse; we sometimes refer to this as "clicking on" an item.

If a screen is left on for a long period of time (several days), the life of the tube is shortened. To protect the tube, the screen automatically dims if you have not used the machine in a specified amount of time. We recommend that you allow the screen to dim rather than turn the console off. Pressing any key or moving the mouse brightens the screen again.

The Keyboard

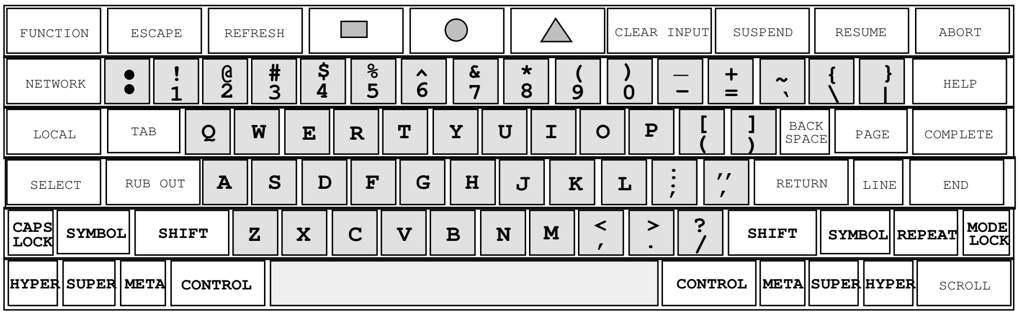


Figure 5. The Symbolics Keyboard

There are 88 keys on the Symbolics keyboard. The keyboard has unlimited rollover, meaning that a keystroke is sensed when the key is pressed, no matter what other keys are held down at the time.

The keys are divided into three groups: special function keys, character keys, and modifier keys. Special function keys and character keys transmit something. They have black labels and are typed in sequence. Modifier keys are intended to be held down while a function or character key is typed, to alter the effect of the key. They have red labels.

Function Keys FUNCTION, ESCAPE, REFRESH, CLEAR INPUT, SUSPEND, RESUME, ABORT, NETWORK, HELP, TAB, BACKSPACE, PAGE, COMPLETE, SELECT, RUBOUT, RETURN, LINE, END, and SCROLL

Character Keys a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9 : - = ` \ | ( ) ; ' , . / and the space bar.

Modifier Keys LOCAL, CAPS LOCK, SYMBOL, SHIFT, REPEAT, MODE LOCK, HYPER, SUPER, META, and CONTROL

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

CIRCLE
SQUARE
TRIANGLE
HYPER

To get information about the keyboard, press the SYMBOL and HELP keys simultaneously.

You can have keys repeat if they are held down. This feature is disabled by default, but you can enable it by setting **si:*kbd-auto-repeat-enabled-p*** to **t**.

```
(setf si:*kbd-auto-repeat-enabled-p* t)
```

The speed of repetition is controlled by **si:*kbd-repetition-interval***. See the variable **si:*kbd-repetition-interval***.

You can exempt certain keys from auto-repetition using the function **si:set-auto-repeat-p**. For example, to make SQUARE one of the keys that do not auto-repeat, you would type:

```
(si:set-auto-repeat-p #\Square nil)
```

See the function **si:set-auto-repeat-p**.

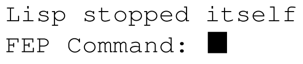
You can customize key bindings; see the section "Setting Key Bindings in Init Files".

If you have a MacIvory or a UX400S, there is a mapping from those keyboards to the Symbolics special keys. For the MacIvory, see the section "Using the Genera Application on a MacIvory". For the UX400S, see the section "Symbolics UX Keyboard Templates".

FEP and Lisp Processor

Cold booting is how you start up the Lisp Processor; you usually do it from the Front End Processor (FEP). To get into the FEP, you have to stop the Lisp Processor, using the Halt Machine command (you see how to do this in the walk-through which follows).

On the 3600-family machines and the XL400, the FEP takes control when Lisp is first powered up. On MacIvory and UX-family machines, the powering up process is slightly different. For the powering up sequence for The FEP takes care of booting Lisp and many other operations (some of which are discussed elsewhere in the workbook).See the section "Workbook: Overview of the Machine".



```
Lisp stopped itself
FEP Command: ■
```

Figure 6. FEP Prompt

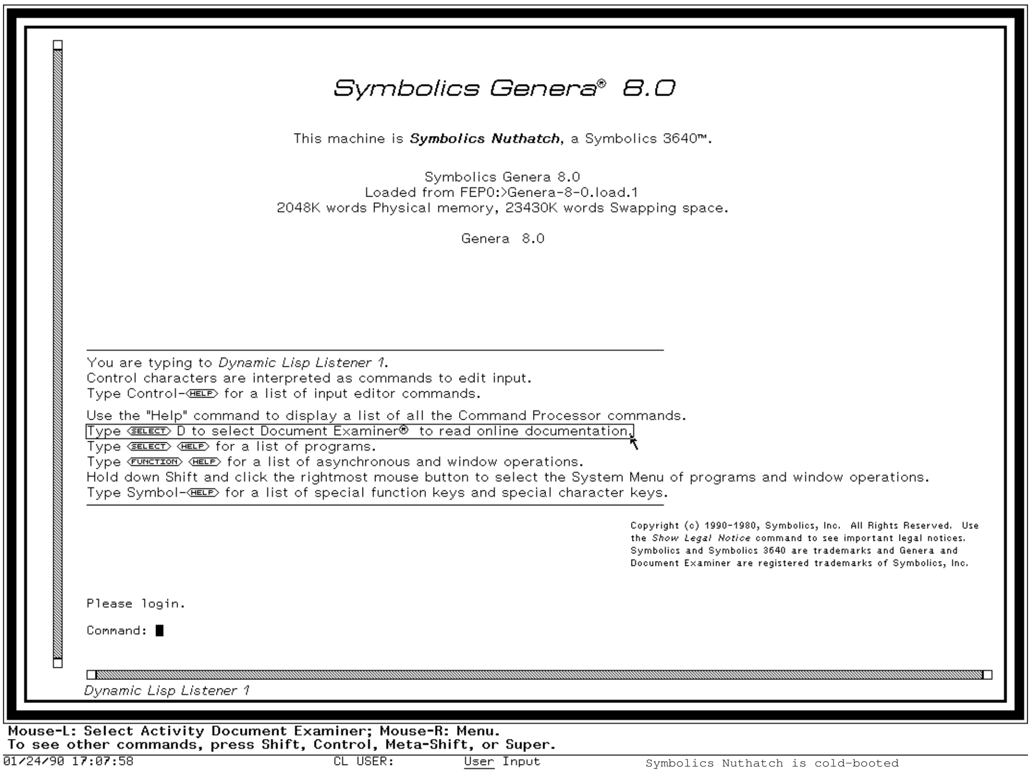
You use the same console for both the FEP and the Lisp Processor. You can tell when you're typing to the FEP because you see the prompt "FEP Command:" (Figure 6). When you are typing to the Lisp Processor, you see the Command Processor prompt (usually "Command: "). If you are in doubt, look at the clock in the lower left hand corner and at the run bars to see if there is any activity. If the clock has stopped and the run bars are static, you are in the FEP.

Cold Booting

When you use a Symbolics computer, you alter its computing environment by creating new objects or modifying existing objects. When you first sit down at a machine, you want to begin work in a clean, unaltered computing environment. Cold booting Lisp provides an environment, or *world*, which is clean. You can think of your Lisp world as a blackboard. When someone uses it, it gets filled up with what they have done. If you want to use it, you wipe it clean (cold boot) so that you begin work on a blank surface.

Look at the lower right corner of your screen (Figure 7). If you do not see the words cold booted, the world might have been modified by someone's use since Lisp was last cold booted. You cannot easily tell whether their modifications will affect your work.

When a machine is cold booted, all previous work in the environment is lost, unless it has been saved in files. Returning to the blackboard analogy, if you copy the blackboard information onto a piece of paper and put that paper in a file folder (write a file to disk), your work is saved, even if someone comes along later and erases the blackboard (cold boots). Remember to explicitly save all of your work in files, because only information in files (which are stored on disk) survive cold booting. (For information on saving files: See the section "Workbook: Zmacs".) If there is a login-name in the status line, it's a good idea to check with the previous user of Lisp before cold booting, because that user may need to save his or her work.



```
Symbolics Genera® 8.0

This machine is Symbolics Nuthatch, a Symbolics 3640™.

Symbolics Genera 8.0
Loaded from FEP0:>Genera-8-0.load.1
2048K words Physical memory, 23430K words Swapping space.

Genera 8.0

You are typing to Dynamic Lisp Listener 1.
Control characters are interpreted as commands to edit input.
Type Control-<HELP> for a list of input editor commands.

Use the "Help" command to display a list of all the Command Processor commands.
Type <SELECT> D to select Document Examiner® to read online documentation.
Type <SELECT> <HELP> for a list of programs.
Type <FUNCTION> <HELP> for a list of asynchronous and window operations.
Hold down Shift and click the rightmost mouse button to select the System Menu of programs and window operations.
Type Symbol-<HELP> for a list of special function keys and special character keys.

Copyright (c) 1990-1980, Symbolics, Inc. All Rights Reserved. Use
the Show Legal Notice command to see important legal notices.
Symbolics and Symbolics 3640 are trademarks and Genera and
Document Examiner are registered trademarks of Symbolics, Inc.

Please login.

Command: █

Dynamic Lisp Listener 1
Mouse-L: Select Activity Document Examiner; Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
01/24/90 17:07:58          CL USER:      User Input        Symbolics Nuthatch is cold-booted
```

Figure 7. A Cold-Booted Machine

As you do the walk-through for cold booting, remember the conventions that we use:

- If you see a word or phrase enclosed in brackets, like [Help], it is a menu item.
- In text, Symbolics keyboard keys look like this:

  `HELP`

  but in examples they look like this:

  `<HELP>`

  to differentiate them from the text around them.

Walk-Through for Cold Booting

In this walk-through, and many others, you might see the display `**MORE**` at the bottom left of the screen. This means that the machine wants to give you more information, but it is waiting so that you can read what is already displayed on the screen. Just press `SPACE` (or almost any other key) to make Lisp continue to display information.

1. Do all of the following exercises in order before going on to the next section.

2. Remember we assume that Lisp is running for all of these exercises. Check the clock in the lower left hand corner of the screen. If it is running, Lisp is running.

3. Press `SELECT L`. This displays a Lisp Listener window, which is described in more detail later. (See the section "Workbook: Getting Around the System".) If you are already in a Lisp Listener, this either puts you into another Lisp Listener (which is fine for these exercises), or the screen flashes to tell you that you are already in the only Lisp Listener.

4. On the screen you see the prompt (unless someone using your machine has changed the prompt string) followed by a blinking black cursor:

   ```
   Command:
   ```

5. This is the prompt for the Command Processor. Look at the status line. If there is someone currently logged in to Lisp, after checking with that user, type

   ```
   Logout<RETURN>
   ```

   to log them out. If you see `**MORE**` at the bottom left of the screen, press the `SPACE` bar.

6. If you log someone out, you are asked about saving unsaved buffers. If so, you should ask the previous user of the machine if any of his or her work should be saved, and then enter `Y` or `N`. After the Logout command completes, you see the message

   ```
   Logged out.
   ```

7. When the `Command:` prompt appears again, type

   ```
   Halt<SPACE>Machine<RETURN>
   ```

8. You are then asked:

   ```
   Do you really want to halt the machine? (Yes or No)
   ```

9. Type

   ```
    Yes<RETURN>
   ```

10. Look at the upper left corner of the screen. You see the following:

    ```
    Lisp stopped itself
    FEP command:
    ```

11. This means that you are now addressing the *Front End Processor (FEP)*, and have to use the FEP command Boot to execute the commands in the *boot file*, which is a file of FEP commands. To execute the commands in the boot file,

you type:

```
boot<RETURN>
```

12. Several things then occur on the screen. You see the microcode being loaded, the world being loaded, paging files being loaded, and so on. (Figure !). Finally, you have a screen which displays the Symbolics logo, *Dynamic Lisp Listener 1* in the lower left corner, and *your-machine-name* `is cold booted` in the right corner of the status line (Figure 7). Cold booting takes several minutes.

13. If yours is a standalone machine (not on a network with other machines), you might be asked about the date and time. If so, take the default by pressing `RETURN`.

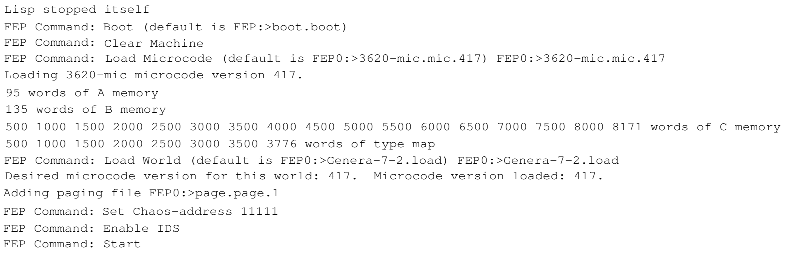
14. If your machine does not look like the picture (Figure 7), find your site administrator or someone else who can help you.

15. You are now in the *Dynamic Lisp Listener 1* window. It has a distinctive multiple border. You are being prompted with

    ```
    Please login.
    Command:
    ```

    Look at the notation in the lower right corner: *your-machine-name* `is cold booted`. Type one character, then press `RUBOUT`. Look for the notation in the lower right corner again. Is your machine still cold booted?

16. Cold boot your machine again. You should become very familiar with the booting process. You need to be able to boot easily, without having to refer to these notes.



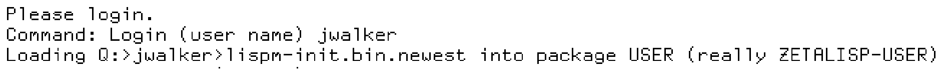
```
Lisp stopped itself
FEP Command: Boot (default is FEP:>boot.boot)
FEP Command: Clear Machine
FEP Command: Load Microcode (default is FEP0:>3620-mic.mic.417) FEP0:>3620-mic.mic.417
Loading 3620-mic microcode version 417.
95 words of A memory
135 words of B memory
500 1000 1500 2000 2500 3000 3500 4000 4500 5000 5500 6000 6500 7000 7500 8000 8171 words of C memory
500 1000 1500 2000 2500 3000 3500 3776 words of type map
FEP Command: Load World (default is FEP0:>Genera-7-2.load) FEP0:>Genera-7-2.load
Desired microcode version for this world: 417.  Microcode version loaded: 417.
Adding paging file FEP0:>page.page.1
FEP Command: Set Chaos-address 11111
FEP Command: Enable IDS
FEP Command: Start
```

Figure 8. Booting

Logging In

The initial display on a freshly booted machine always asks you to log in.

Logging in on a Symbolics computer records your name for file operations and loads the *lispm-init* file from your *home directory*. (Figure 9). This directory is where you keep your files. Your lispm-init file is an initialization file containing



```
Please login.
Command: Login (user name) jwalker
Loading Q:>jwalker>lispm-init.bin.newest into package USER (really ZETALISP-USER)
```

Figure 9. Logging In

Lisp code to customize your Lisp environment. If you do not yet have a lispm-init file, you can log in, but Lisp notifies you that the file cannot be found.

On a Symbolics computer, your login-name can be any set of alphanumeric characters (as long as it does not contain spaces or conflict with any other login-name at your site). Popular login-names are first names, last names, initials or nicknames. However, your login-name and the name of your home directory should be the same, since Genera expects this.

If you have any problems logging in see your site administrator or: See the section "Workbook: The Namespace".

Walk-Through for Logging In

1. You should be looking at Dynamic Lisp Listener 1. If not, you can press `SELECT L` to get to a Lisp Listener; if you are already in one, the screen flashes and/or beeps.

2. If the Lisp Listener is not prompting you with

   ```
   Command:
   ```

   press the `CLEAR INPUT` key to cancel whatever you may have typed. You then get the `Command:` prompt again (unless someone at your site has changed the prompt). If you see the words *Dynamic Lisp Listener* at the lower left of the screen, you are in the right place.

3. Type

   ```
   Login<SPACE>
   ```

   You are prompted for your login-name. Type your login-name followed by `RETURN` (Figure 9).

   ```
   Login <SPACE> (user name) DAVID
   Loading SYMBOLICS1:>david>lispm-init.bin into package USER
   (really COMMON-LISP-USER)
   ```

4. If you see `**MORE**` at the bottom left of the screen, press the `SPACE` bar.

Logging Out

When you are finished with the machine, you should save any work you want to keep in files, and then log out. Logging out informs Lisp that you are leaving; it queries you about any work that has not been saved.

Walk-Through for Logging Out

1. You should still be looking at Dynamic Lisp Listener 1.

2. Type

```
        Logout<RETURN>
```

3. If you see the line `**MORE**` at the bottom of your screen, press `SPACE`.

If you share this machine with other users, it is proper etiquette to cold boot after you log out, so that the environment is clean and the next user knows that it is all right to use the machine.

Documentation References

- See the section "The Mouse".
- See the section "The Screen".
- See the section "Index of Special Function Keys".
- See the section "The Front-End Processor".
- See the section "Cold Booting".
- See the section "Logging In".
- See the section "Logging Out".

Getting Around the System

Introduction

This chapter introduces you to some of the features of the Dynamic Lisp Listener window, the `HELP` key, and the `SELECT` key.

- Workbook: The Dynamic Lisp Listener
- Workbook: The `HELP` Key
- Workbook: The Command Processor
- Workbook: Walk-through for Command Arguments
- Workbook: Walk-through for Getting Help in the Command Processor
- Workbook: Selecting a New Activity
- Workbook: Walk-through for Selecting a New Activity
- Workbook: Using the Window System
- Workbook: Getting Around the System Documentation References

The Dynamic Lisp Listener

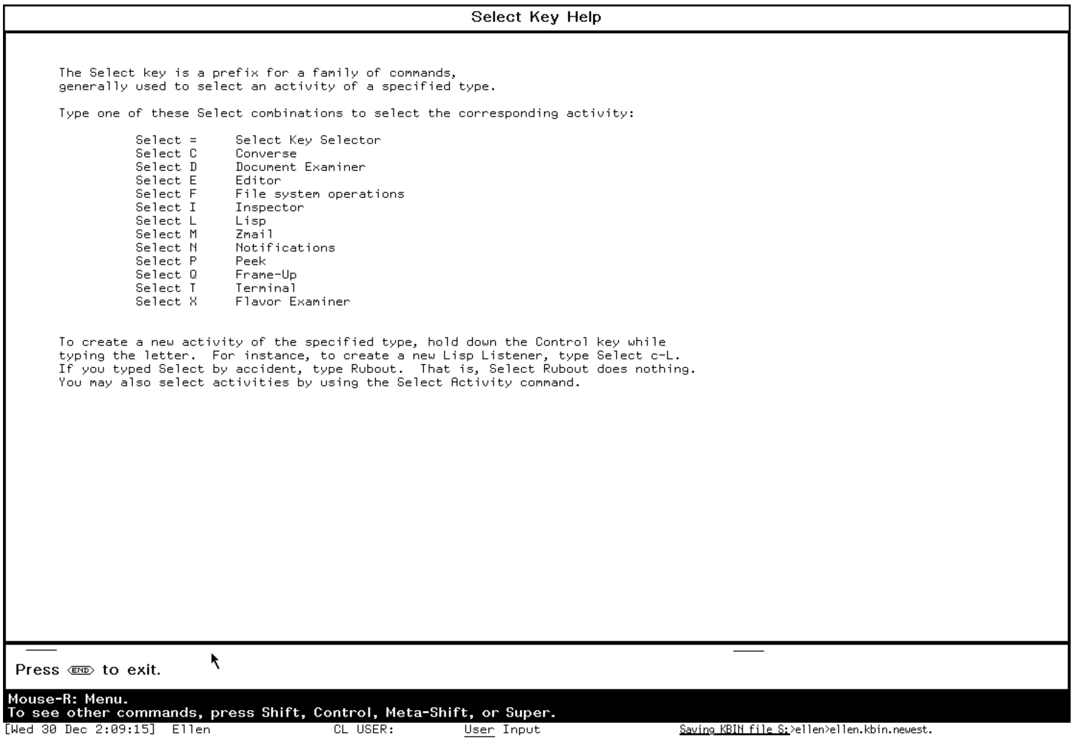
A *Dynamic Lisp Listener* is the first window you see when you cold boot Lisp. (We call the Lisp Listener "dynamic" because, like many other windows in Genera, it retains all interaction between you and the system.)

You give input to the Lisp Listener as *Command Processor commands* and pieces of Lisp code called *Lisp forms*. (For more information on the command processor: See the section "Workbook: The Command Processor".) The Lisp Listener executes the commands and Lisp forms at once.

The HELP Key

When you need information about what to do in any context, try the HELP key.

Find the HELP key (below the ABORT key at the right edge of the Symbolics keyboard). Frequently, pressing the HELP key provides online documentation designed to assist you with what you're doing in the current context.



Select Key Help

```
The Select key is a prefix for a family of commands,
generally used to select an activity of a specified type.

Type one of these Select combinations to select the corresponding activity:

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

To create a new activity of the specified type, hold down the Control key while
typing the letter.  For instance, to create a new Lisp Listener, type Select c-L.
If you typed Select by accident, type Rubout.  That is, Select Rubout does nothing.
You may also select activities by using the Select Activity command.
```

Press END to exit.

```
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Wed 30 Dec 2:09:15]  Ellen           CL USER:        User Input          Saving KBIN file S:>ellen>ellen.kbin.newest.
```

Figure 10. SELECT HELP

Here is a partial listing of situations in which pressing the HELP key provides context-related assistance.

- In a Lisp Listener, pressing the HELP key gives you an abbreviated list of Command Processor commands.

- Pressing either the FUNCTION key or the SELECT key followed by the HELP key provides on-screen documentation of commands you can execute using that key (Figure 10).

- SYMBOL-HELP (sy-HELP) does the same thing for the SYMBOL key and also provides a chart of special function keys.

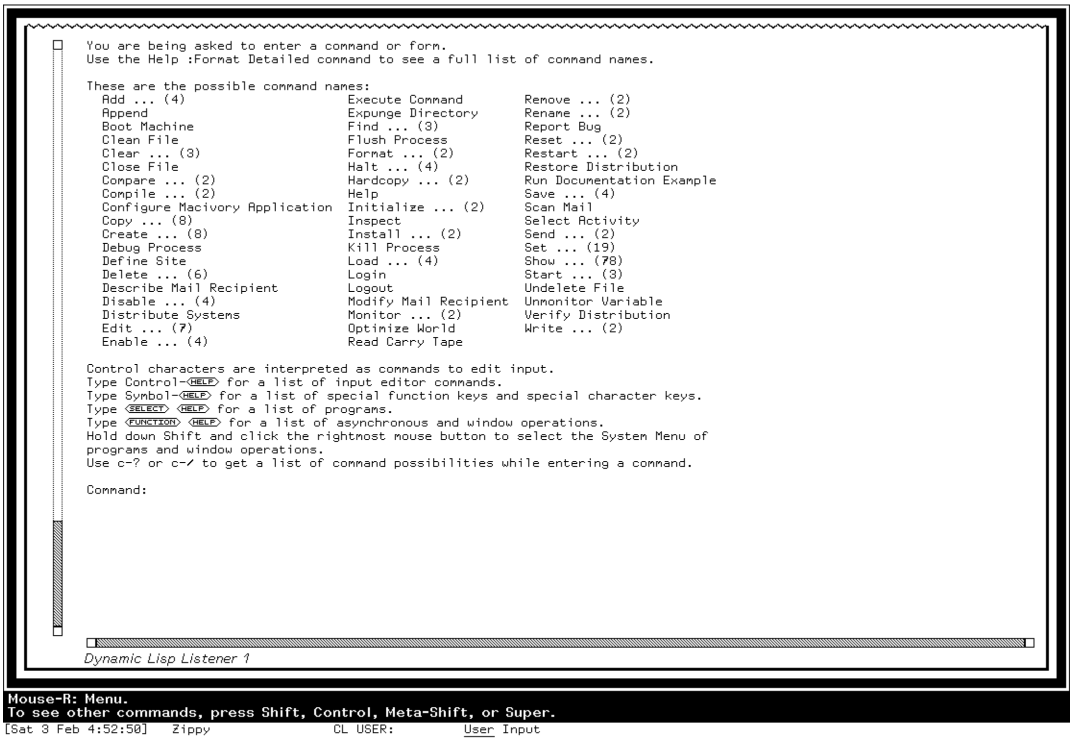
- CONTROL-HELP (c-HELP) provides a list of Input Editor commands. Using these commands is covered elsewhere: See the section "Workbook: The Input Editor".

- In the editor, pressing the HELP key returns:

  ```
  Help: Type one of A,C,D,L,V,W,Space,Help,Abort:
  ```

  If you then press the HELP key again, explanations for each of these options are displayed.

The Command Processor



```
You are being asked to enter a command or form.
Use the Help :Format Detailed command to see a full list of command names.

These are the possible command names:
  Add ... (4)                     Execute Command       Remove ... (2)
  Append                          Expunge Directory     Rename ... (2)
  Boot Machine                    Find ... (3)          Report Bug
  Clean File                      Flush Process         Reset ... (2)
  Clear ... (3)                   Format ... (2)        Restart ... (2)
  Close File                      Halt ... (4)          Restore Distribution
  Compare ... (2)                 Hardcopy ... (2)      Run Documentation Example
  Compile ... (2)                 Help                  Save ... (4)
  Configure Macivory Application  Initialize ... (2)    Scan Mail
  Copy ... (8)                    Inspect               Select Activity
  Create ... (8)                  Install ... (2)       Send ... (2)
  Debug Process                   Kill Process          Set ... (19)
  Define Site                     Load ... (4)          Show ... (78)
  Delete ... (6)                  Login                 Start ... (3)
  Describe Mail Recipient         Logout                Undelete File
  Disable ... (4)                 Modify Mail Recipient Unmonitor Variable
  Distribute Systems              Monitor ... (2)       Verify Distribution
  Edit ... (7)                    Optimize World        Write ... (2)
  Enable ... (4)                  Read Carry Tape

Control characters are interpreted as commands to edit input.
Type Control-HELP for a list of input editor commands.
Type Symbol-HELP for a list of special function keys and special character keys.
Type SELECT HELP for a list of programs.
Type FUNCTION HELP for a list of asynchronous and window operations.
Hold down Shift and click the rightmost mouse button to select the System Menu of
programs and window operations.
Use c-? or c-/ to get a list of command possibilities while entering a command.

Command:

Dynamic Lisp Listener 1
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Sat 3 Feb 4:52:50]   Zippy             CL USER:         User Input
```

Figure 11. Command Processor Commands

The *Command Processor* or *CP* lets you run many functions that are part of Genera by typing simple English-like commands. You have already used a few Command Processor commands: Login, Logout, and Halt Machine. There are over one hundred and fifty commands (Figure 11), including ones to:

- Manipulate files and directories

- Manipulate windows

- Set the date and time

Mouse-sensitivity

When you press the `HELP` key, you see the abbreviated list of Command Processor commands. This list shows you the name of the command (if only one command begins with that word) or the first word of the command and the number of commands that begin with that word. For instance, the command name Report Bug is the only command that begins with the word Report, but several dozen commands begin with the word Show.

All the commands are mouse-sensitive. If you move the mouse over a command and wait until the box appears around the command name, the mouse documentation lines tell you what happens if you click the mouse. For example, the mouse documentation lines tell you that clicking Left on Report Bug puts that command after the prompt but does not activate the command. Clicking Left on Show displays the expanded list of commands beginning with Show. If you want one of these commands put after the prompt, you have to click on the command name. Clicking Right gives you a menu of operations to perform or a menu of all the possible commands for that word.

Parsing commands

A command has three logical parts: a *name*, *positional arguments*, and *keyword arguments*. The command name may be followed by zero or more positional arguments and by zero or more keyword arguments.

Name One or more words separated by spaces. For example:

```
Show Command Processor Status

Clear Output History

Show Herald

Login
```

Positional arguments

Zero or more *required* arguments to the command. They must be supplied in the proper order. When a command requires arguments, the CP prompts you for each argument by providing some or all of these things:

- The *type* of argument required in parentheses.

- A *default* (which you may or may not elect to use).

- A *list of possibilities* that you must choose from.

An example of a Command Processor that provides the *type* of argument is

```
Login (user name)
```

Keyword arguments

Zero or more *optional* arguments to the command. Keyword arguments always follow positional arguments, although the order in which you supply the keyword arguments does not matter.

A keyword argument always begins with a colon, and it may require a value. If it does, you type the value right after the keyword. Keyword arguments can have default values just like positional arguments. You can get a list of keyword arguments when the Command Processor is prompting you with `(keywords)` by pressing the `HELP` key.

An example of a Command Processor command that uses a keyword argument is

```
Help (keywords) :Format (Brief or Detailed [default Brief])
```

It is hard to go wrong with the Command Processor. Remember that:

- You can press `CLEAR INPUT` or `ABORT` at any time to abort a command and start again.

- You can press `HELP` at any time to get assistance.

- You must press `RETURN` or `END` to make the Command Processor execute a command.

- Pressing `SPACE` gets you the default value for an argument.

Walk-Through for Command Arguments

Do **not** execute any of the commands in this walk-through; they are all for illustration. Do not press `RETURN` when you type these examples. Make sure that you log in before you start this exercise.

1. Type `SELECT L` to get to the Lisp Listener.

2. For an example of a command that provides a default for the positional argument, type

```
Show<SPACE>File<SPACE>
```

3. Press the `CLEAR INPUT` key to erase what you typed in the previous step.

4. For an example of a command that provides a list of possibilities, type

   ```
   Select<SPACE>Activity<SPACE>F<HELP>
   ```

5. Press the `CLEAR INPUT` key again.

6. Type

   ```
   Logout<SPACE>
   ```

   Notice the prompt `(keywords)`.

7. Press the `HELP` key for a description of the optional keyword arguments to the Logout command.

8. After you read the help message, type

   ```
   :Save<SPACE>Buffers<SPACE>
   ```

   to fill in a keyword argument and to see the list of possible values for this argument. (Remember that keywords are always preceded by a colon.)

9. Now you see

   ```
   Logout (keywords) :Save Buffers (Yes, No, or Ask [default Yes])
   ```

   Press `SPACE` again to get the default argument for `:Save Buffers`.

10. Press `CLEAR INPUT` to cancel the `Logout` command. Note: if you accidentally complete the Logout command, simply log in again.

Getting Help in the Command Processor

Pressing the `HELP` key to the Command Processor prompt gives you an abbreviated list of all the possible Command Processor commands.

You can get the full or detailed list by typing

```
Help :format detailed
```

or you can construct the above command using the mouse and the `HELP` key as follows:

1. Press the `HELP` key

2. Click on [Help]

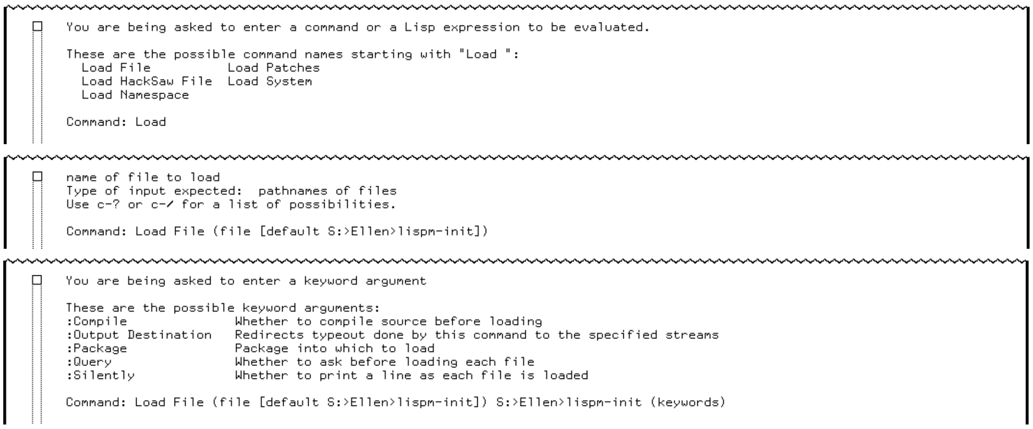
3. Press the `HELP` key

4. Click on `:Format`

5. Press the HELP key

6. Click on detailed

(This iterative building up of a command is a very powerful tool but is more appropriate in a programming context where you want to access complex Lisp expressions that are sometimes in difficult-to-type internal representations.)



```
□  You are being asked to enter a command or a Lisp expression to be evaluated.

   These are the possible command names starting with "Load ":
     Load File          Load Patches
     Load HackSaw File  Load System
     Load Namespace

   Command: Load
```

```
□  name of file to load
   Type of input expected:  pathnames of files
   Use c-? or c-/ for a list of possibilities.

   Command: Load File (file [default S:>Ellen>lispm-init])
```

```
□  You are being asked to enter a keyword argument

   These are the possible keyword arguments:
   :Compile              Whether to compile source before loading
   :Output Destination   Redirects typeout done by this command to the specified streams
   :Package              Package into which to load
   :Query                Whether to ask before loading each file
   :Silently             Whether to print a line as each file is loaded

   Command: Load File (file [default S:>Ellen>lispm-init]) S:>Ellen>lispm-init (keywords)
```

Figure 12. Pressing HELP at Three Points

Pressing the HELP key after you have typed any part of a command gives you useful information about getting help for various things (Figure 12).

You can also press the HELP key to get useful information at various points during the command-issuing process.

If you press HELP	*You see...*
While typing the characters in a command name	A list of the possible completions for the characters you have typed so far (Figure 12).
After completing a command name	A description of the positional arguments the command requires (Figure 12).
After supplying the positional arguments	A description of the optional keyword arguments for the command (Figure 12).

Note that pressing HELP never interferes with the part of a command you have typed so far. Instead, the window redraws itself, with the help information displayed above the command you are typing in (Figure12).

Walk-Through for Getting Help in the Command Processor

1. Press the HELP key.

2. Type

   ```
   Help<SPACE>
   ```

   and press the HELP key to see options for the keyword argument. Click on `:format` and type `detailed` to see the full list of command names.

3. Type

   ```
   Ha<HELP>
   ```

   Notice that the window is redrawn with the help information appearing *above* the command you are typing in. Notice that there are several commands that begin with `Ha`, one of which is the Halt Machine command that you have already used.

4. To finish entering the command, type

   ```
   rdcopy<SPACE>File<SPACE>
   ```

   This is the Hardcopy File command, which lets you print files.

5. Press the SPACE key to get the default file.

6. You are asked for the name of a printer; pressing SPACE gets the default (Figure 12).

7. Now you should be prompted for `(keywords)`. Press the HELP key again to see what the possible keyword arguments are.

8. Press the CLEAR INPUT key, since you do not really want to hardcopy a file.

9. Type

   ```
   Clear<SPACE>Output<SPACE>History<RETURN>
   ```

   to clear the text out of your window.

Selecting a New Activity

System programs in Genera are called *activities*. The Lisp Listener and the editor are two of these activities. Frequently, you want to go from one of these activities to another. There are several ways to switch between activities, but we'll just look at using the SELECT key here.

The SELECT key is in the middle of the left-hand side of the Symbolics keyboard. Here is a list of all the activities (generated when you press the SELECT key followed by the HELP key).

= **SELECT key selector.** Lets you create and change SELECT key assignments for all activities.

C **Converse**. For sending messages to users on other machines.

D **Document Examiner**. Lets you read the online Symbolics documentation.

E **Editor**. The editor is called Zmacs. Edits text (and program) files and directories.

F **File system operations**. For doing local Lisp Machine File System (LMFS) maintenance. Also allows you to browse through any hierarchical file system. The file system maintenance activity is called FSMaint or FSEdit.

I **Inspector**. Lets you browse through Lisp data structures. This is very useful when you are writing programs.

L **Lisp**. Lisp Listeners evaluate Command Processor commands and Lisp forms you type. By default, they use Symbolics Common Lisp syntax.

M **Zmail**. A mail reading and sending activity.

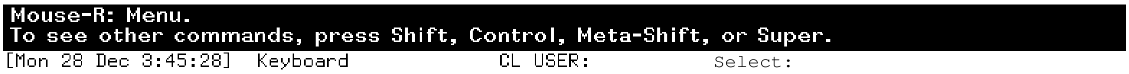
N **Notifications**. Allows you to see any notifications you have gotten.

P **Peek**. Lets you monitor and change some of the active parts of the system.

Q **Frame-Up**. The layout designer activity that allows programmers to design and create windows for their own programs.

T **Terminal**. Allows you to log in to any other machine that has a network connection to your machine.

X **Flavor Examiner**. Allows you to browse through flavor information. This is very useful when you are writing programs.



```
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Mon 28 Dec 3:45:28]  Keyboard            CL USER:          Select:
```

Figure 13. The Status Line When You Press SELECT

In order to select one of these activities, you

1. Press the SELECT key and let it go. Notice that when you press the SELECT key, the process state (in the center of the status line) now says `Select:` (Figure 13).

2. Press one of the above letters.

After selecting an activity, any characters you type go to the newly selected activity.

To get back to the activity that you were using, you can use SELECT and the letter of the old activity. To return to the Lisp Listener, press SELECT L. You can get the preceding list on your screen by pressing the SELECT key followed by the HELP key.

Note that you don't "quit" or "exit" an activity, you just select another one.

When you select a new activity, the old ones stay the way they were when you left, and they are still running when you come back. For instance, you could sit down at your machine, log in, select the editor, use it, select the Lisp Listener again, and reselect the editor, which would be in the same state as when you left it. You could then select Zmail to read your mail, reselect the Lisp Listener, and so on. None of these activities would ever "finish" and stop running; they just sit in the background waiting to be selected.

Walk-Through for Selecting a New Activity

1. Start by cold booting Lisp to ensure that your environment starts this walk-through in the correct condition. If you need to review cold booting: See the section "Workbook: Walk-through for Cold Booting". Log in. You should always log in when you start using a machine and after you cold boot.

2. Now you can use the Lisp Listener activity to do something. The Lisp Listener should be prompting you with: Command:.

3. Type

   ```
   Show<SPACE>Font<SPACE><HELP>
   ```

   to see a list of the fonts which are loaded on your system. Lisp prompts you with

   ```
   There are (some number) possible font names.  Do you want to see
   them all? (Y or N)
   ```

   Press the Y key.

4. A list of all the loaded fonts is displayed, followed by

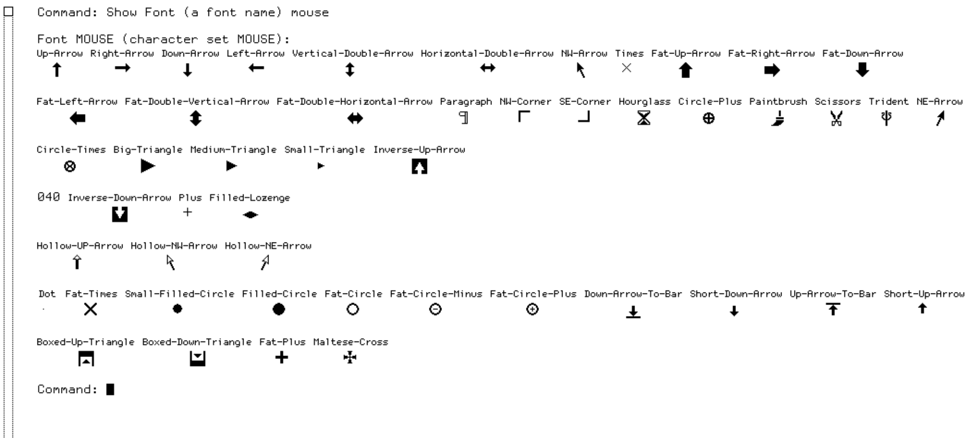
   ```
   Show Font
   ```

   Type

   ```
   MOUSE<RETURN>
   ```

   If you did this correctly you should see all the characters in the MOUSE font displayed. (Figure !).

5. Press the SELECT key, and let go. Notice that in the center of the status line it now says Select:.



```
Command: Show Font (a font name) mouse

Font MOUSE (character set MOUSE):
Up-Arrow Right-Arrow Down-Arrow Left-Arrow Vertical-Double-Arrow Horizontal-Double-Arrow NW-Arrow Times Fat-Up-Arrow Fat-Right-Arrow Fat-Down-Arrow

Fat-Left-Arrow Fat-Double-Vertical-Arrow Fat-Double-Horizontal-Arrow Paragraph NW-Corner SE-Corner Hourglass Circle-Plus Paintbrush Scissors Trident NE-Arrow

Circle-Times Big-Triangle Medium-Triangle Small-Triangle Inverse-Up-Arrow

040 Inverse-Down-Arrow Plus Filled-Lozenge

Hollow-UP-Arrow Hollow-NW-Arrow Hollow-NE-Arrow

Dot  Fat-Times Small-Filled-Circle Filled-Circle Fat-Circle Fat-Circle-Minus Fat-Circle-Plus Down-Arrow-To-Bar Short-Down-Arrow Up-Arrow-To-Bar Short-Up-Arrow

Boxed-Up-Triangle Boxed-Down-Triangle Fat-Plus Maltese-Cross

Command:
```

Figure 14. Show Font Mouse

6. Press the `E` key. It takes a few seconds this first time, because the window must be paged in, and there might be some odd things happening in the status line, but soon you see the editor window. The editor window is distinguishable by the fact that it's only about three-quarters the width of the screen. If you can see this window, you have now selected the editor activity. Notice that the part of the Lisp Listener which is not under the editor window is still visible, but it is grayed-over (referred to as *deexposed* in the documentation).

7. Now that you've selected it, you can use the editor. There isn't anything special you have to do, just start typing. Don't worry about any errors. Type in

   ```
   Old MacDonald had a farm
   ```

   Simply by typing you are using the editor. The window system is directing your input to the editor, since that is the selected activity.

8. Type `SELECT L` to return to the Lisp Listener. Notice that nothing has changed. It still has the same `Command:` prompt, and the font display is still there.

9. Give the

   ```
   Show<SPACE>Font<SPACE>HL14BI<RETURN>
   ```

   command.

10. Repeat `SELECT E` and `SELECT L` a few times. Notice that it works much faster after the first time. End up with the Lisp Listener selected.

11. By the time you have progressed this far into this document, you cannot see the section that lists all the activities accessible via the `SELECT` key. So, press `SELECT HELP` to see it on your screen.

12. Press END to get rid of the help window. You are back in the Lisp Listener.

13. Press the SELECT key once more. Look at the status line. The `Select:` means that Lisp is expecting you to enter the letter for some activity.

14. Suppose you don't really want to select another activity. Press the RUBOUT key to cancel the SELECT. Note that the `Select:` disappears from the status line.

Using the Window System

Remember that the status line (at the bottom of the screen) contains useful information about what your window processes are doing. Remember also that you do not (in general) "quit" or "exit" an activity, you just select some other activity. Some activities continue typing to their windows, even when you cannot see the window on your screen anymore. Then, when you select that activity again, there is new output on the screen. Some activities give you a *notification* when they want input or output.

Documentation References

- See the section "HELP Key".
- See the section "SELECT Key".

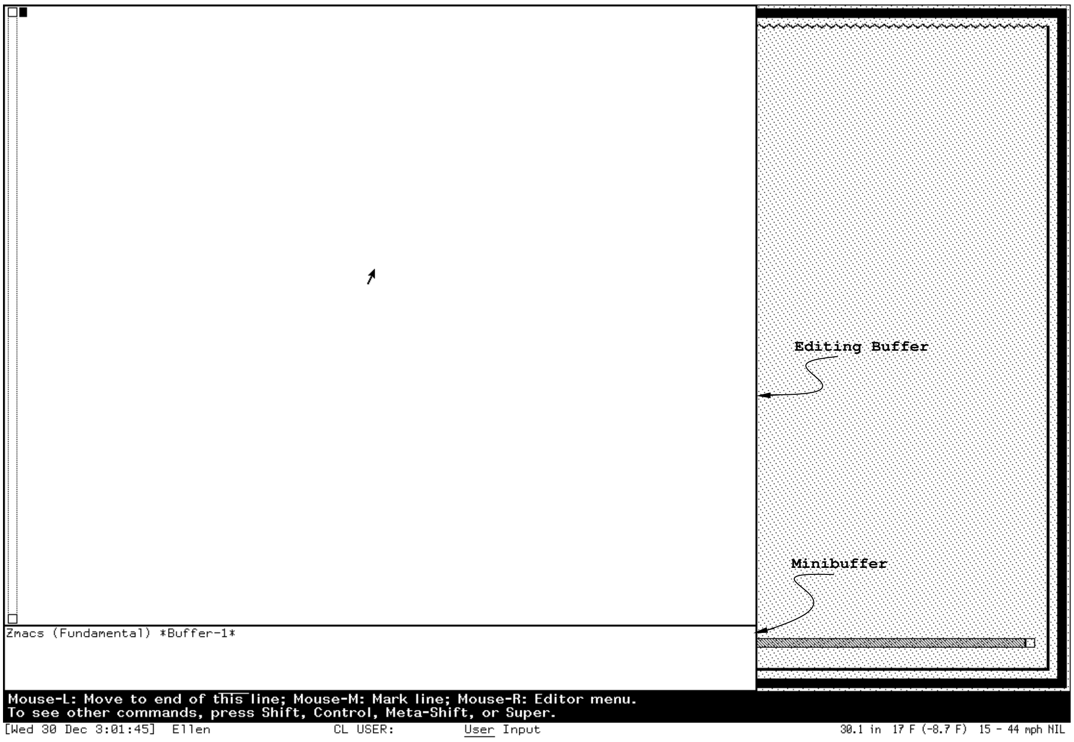
Zmacs

Introduction

This chapter teaches you how to use the Zmacs editor. It does not cover all the possible Zmacs commands, only the ones you need to begin editing effectively.

- Workbook: The Zmacs Window
- Workbook: Files and Buffers
- Workbook: Walk-through for Finding a File
- Workbook: Inserting Text
- Workbook: Basic Cursor Movement
- Workbook: Walk-through for Inserting Text and Basic Cursor Movements
- Workbook: Deleting and Modifying Text
- Workbook: Walk-through for Deleting and Modifying Text
- Workbook: Zmacs Documentation References

The Zmacs Window



```
Zmacs (Fundamental) *Buffer-1*

Mouse-L: Move to end of this line; Mouse-M: Mark line; Mouse-R: Editor menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Wed 30 Dec 3:01:45]  Ellen          CL USER:       User Input                    30.1 in  17 F (-8.7 F)  15 - 44 mph NIL
```

Figure 15. The Zmacs Window

Zmacs is a screen-oriented text editor, which means that you see the updated state of your text as you make corrections. You can use Zmacs by pressing SELECT E. When you do this, you see a window which has a large area at the top and a small area at the bottom, separated by a single horizontal line (Figure 15). The area at the top is the *Editor Window*. This is where you see the text you are working on. The area at the bottom contains the *Mode Line* and the *Echo Area*. The mode line tells you what the name of the buffer (work area) is, what *mode* you are in, whether you are looking at all of the text or only part of it, and whether you have changed the text or not, as well as a few other things. The echo area is where the commands you type appear, if they appear at all, and where you are prompted for arguments to those commands (Zmacs command arguments are much like Command Processor command positional arguments). The part of the echo area in which you are prompted for arguments is called the *minibuffer*.

Remember, the following abbreviations indicate that you should press one or more keys:

c-	CONTROL

m-	META
sh-	SHIFT
c-sh-	CONTROL-SHIFT
m-sh-	META-SHIFT

This means that c-D means press CONTROL and D together (which deletes one character, the one your cursor is on) and m-D means press META and D together (which deletes one word).

Files and Buffers

When you read a file into Zmacs, the text goes into an editor buffer. A *buffer* is a work area. Buffers are temporary; they go away when Lisp is cold booted. *Files* are relatively permanent collections of data stored on disk.

You read text from a file into a buffer, work on it in the buffer, then write it back to a file on disk.

Thus, you refer to buffers in Zmacs, not files. Zmacs allows many buffers, holding the text from different files, to be available at one time. In fact, users may have a dozen or more buffers available at any one time. The only limit is how many buffers you can keep track of.

Cold booting your machine clears out your buffers, along with everything else in the Lisp World. When you save the contents of a buffer, it becomes a file whose contents you can retrieve at any time. If you don't save it before you cold boot, it is lost forever.

It is a good idea to use the Logout command before cold booting, as it will prompt you to save your editor buffers, along with reminding you of other things to do.

File Pathnames

To perform any file operation, it is necessary to specify the file you want. Each file is identified by a unique *pathname*, which is basically a set of directions on how to find the file. A pathname has several *components*. All of these components are used to specify a file, but due to pathname *defaulting*, you rarely have to type all of them. For more information: See the section "Workbook: Pathnames". The most important pathname components are, in order:

host The machine on whose disk the file resides.

directory The directories and subdirectories (directories "under" other directories) in which the file resides. Directories are used to group files.

name The name of the file.

type The type of the file. Common types of files are text, lisp, and bin (binary). File type is also known as the *extension* of the file.

version The version number of the file. The first time you save a file its version number is 1. Each time you save the file after that, the version number is automatically incremented.

In the Genera system, these components are grouped together into a pathname according to the following pattern:

HOST:>*directory*>*sub-directory*>*sub-dir*>...>*name*.*type*.*version*

```
CHARO:>cugie>chihuahua>cup>...>frob-all-knobs.lisp.3
```

The greater-than symbol (>) is used to separate the directory component of the pathname from the host name and the file name as well as from each other when you use subdirectories. We recommend that you do not use spaces in your directory names or your file names, but that you use the minus-sign symbol (-) instead.

In saying pathnames out loud, the > is called "down", as in "cugie down chihuahua down cup" and so forth.

Finding and Saving Files

```
Find file [default H:>sys>doc>doc.lisp.newest]:
```

Figure 16. Prompting in the Minibuffer

All of the prompts and defaults appear in the Zmacs minibuffer at the bottom of the window (Figure 16).

`c-X c-F` *Find File*. This command prompts you for a file to find and copy into a buffer. A default pathname is offered. If the file does not exist, this command creates an empty buffer with the specified file name (Figure 16).

```
c-X c-F <pathname><RETURN>
```

`c-X c-S` *Save File*. This command saves the current buffer to the file associated with the buffer. If there is no file associated with the buffer, it prompts you for the pathname the first time the buffer is saved and saves all subsequent versions of the buffer under that pathname, with updated version numbers.

```
c-X c-S <pathname><RETURN>
```

`c-X c-W` *Write File*. This command saves the current buffer to a file, prompting you for a pathname. It offers a default pathname. This command allows you to rename a file or write a file to a different directory. When you use it, the name of your buffer is also changed.

```
c-X c-W <pathname><RETURN>
```

Walk-through for Finding a File

1. Find out the name of the file server at your site before you start doing these exercises. The name of the file server is the first, or "Host:" part of a pathname.

2. Press `SELECT E` to select the editor. You should be in `*Buffer-1*`. If you have not had occasion to cold boot since the last walk-through, the line of text

   ```
   Old MacDonald had a farm
   ```

   is still there.

3. Type `c-X c-F`.

4. In the echo area, you should see the words `Find File` and a default pathname (Figure 16). You are being prompted to enter the pathname of the file you wish to find. Type

   your-file-server`:>`*your-login-name*`>zmacs-test.text<RETURN>`

   We are using *your-file-server:* instead of *HOST:* to remind you to make this specific to your site. This file should not exist, so Zmacs should create an empty buffer with this pathname as its name. However, `*Buffer-1*` has not disappeared. It is no longer visible to you, but you could reselect it. Zmacs just accumulates buffers, unless you explicitly delete one. Now you are ready to begin editing text.

Inserting Text

Zmacs is always ready to accept an insertion except when you are actually typing commands. When you type a character, it is inserted in the buffer at the current cursor position. When the cursor appears to be on top of a letter, it is logically between that character and the character before it. So, when you begin typing, the characters are inserted between the character under the cursor and the preceding character. Zmacs is a screen-oriented editor, so you can always see everything in the buffer.

Basic Cursor Movement

You can type Zmacs commands at any time during an editing session. These commands can have different effects depending on where you are in the buffer. This is certainly true of the movement commands, which move the cursor from its current position to a position that is some logical unit (a character, a word, a sentence, a region, a Lisp form) of text away (Figure 17). You cannot move around in an empty buffer, as there are no units of text on which to operate.

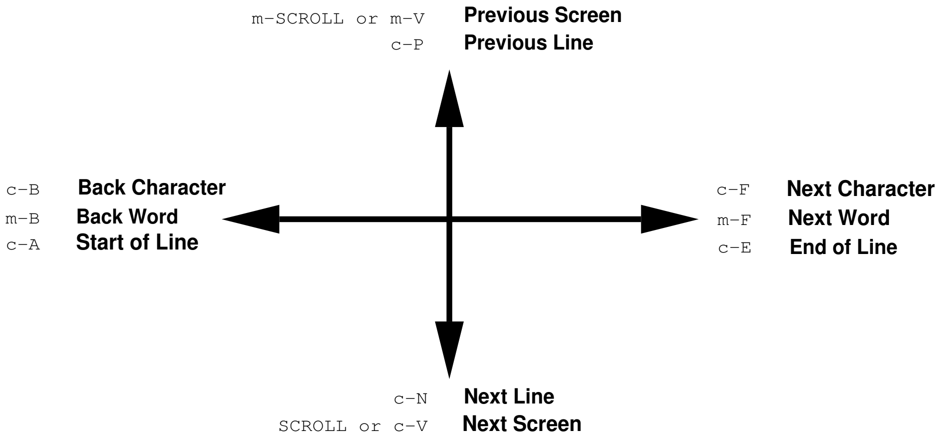


Figure 17. Diagram of Movement Commands

The Mouse

You can also move the cursor with the mouse. Just point the mouse at the position in the buffer where you want the cursor to be and click Left. The blinking cursor moves to that position.

When you are entering a command, you can click with the mouse in the minibuffer to see a menu of completions for whatever you have typed, command names or pathnames.

Movement

Forward

c-F Moves cursor forward one character.

m-F Moves cursor forward to the space at the end of the next word.

c-E Moves cursor to the end of the line.

c-N Moves cursor down to the next line.

m-sh-> Moves cursor to the end of the buffer.

c-V Moves cursor to the beginning of the next screen of text.

Backward

c-B Moves cursor backward one character.

m-B Moves cursor backward to the beginning of the word.

c-A Moves cursor to the beginning of the line.

c-P Moves cursor up to the previous line.

m-sh-< Moves cursor to the beginning of the buffer.

m-V Moves cursor to the end of the previous screen of text.

Note that some of the these commands move the cursor to characters and some move it to blank spaces.

Searching

Searching is a different form of movement. Instead of moving by logical units, searching moves the cursor to the next occurrence of a given string of characters in the buffer. Zmacs uses a form of searching called *incremental searching*. This means that, as you type in the characters you are looking for (you are prompted for this string in the minibuffer), the cursor moves either forward or backward in the buffer.

Thus, if you are searching for the string `the`, when you type the `t`, the cursor moves to the first occurrence of a `t` in the buffer, searching forward or backward (depending on which searching command you used) from the current cursor position. Then, when you type the `h`, the cursor moves to the first occurrence of `th`. Finally, when you type the `e`, the first occurrence of `the` is found. Sometimes you find `the` before you enter the entire string if it's the first occurrence of either `t` or `th`.

If the occurrence of the string which you find is not the one you want, just type the search command again (`c-S`), and the cursor moves to the next occurrence of the string. You can search either forward and backward. Forward searching leaves you at the end of the string you ask for, backward searching leaves you at the beginning of the string. You can terminate a search by pressing `END`.

`c-S` Searches forward in the text for the string you specify. The cursor moves in the text as you type in the minibuffer.

`c-R` Searches backward in the text for the string you specify. The cursor moves to the beginning of the found string.

`c-G` Returns the cursor to the starting position.

Walk-through for Inserting Text and Basic Cursor Movements

1. Before you can move around in the buffer, it would be helpful to have some text to move the cursor through. Type in (don't worry about mistakes)

```
"Will you walk a little faster?" said a whiting to a snail,
"There's a porpoise close behind us, and he's treading on my tail.
See how eagerly the lobsters and the turtles all advance!
They are waiting on the shingle - will you come and join the dance?
Will you, won't you, will you, won't you, will you join the dance?"
```

2. This could use a title, so move to the beginning of the buffer with `m-sh-<`.

3. Type

```
<TAB>The Lobster-Quadrille<RETURN><RETURN>
```

Notice how the insertion works. The cursor remains on the " throughout, as the characters are inserted before it.

4. Press `c-F` a few times.

5. Press `c-N`. Notice that the cursor moves to the same column in the next line.

6. Press `c-A` to move to the beginning of the line.

7. Press `c-F` twice.

8. Press `m-B`.

9. Press `m-F` a few times.

10. Press `c-E`.

11. Press `c-P` a few times. Notice that when you move to a shorter line, the cursor moves to the last column that actually has a character in it, rather than the same column as the cursor was in.

12. Move to the beginning of the word `turtles`. (Don't use the searching commands for this.)

13. Press `c-S`. When you are prompted for a string (in the minibuffer), type `a`.

14. Now type `n`, `c`, and `e` slowly, watching the cursor as you do so.

15. Press `c-S` twice more.

16. Now press `c-R` twice. Notice that the first time, the matching string that is found is the one you are already on, but the cursor moves to the beginning of it. This is because forward searching positions the cursor at the end of the string it finds, so the first occurrence of the string before the cursor (which is what reverse search looks for) is the one you just found with forward search. Also notice that you did not have to type in a new string, as you were still in the middle of the old search.

17. Press `END` to terminate the search.

18. Now press `c-S` again.

19. When you are prompted for a search string, press `c-S` again. Notice that the search string you used before is used by default.

20. Press `c-P` to terminate the search. Any of the basic movement commands (and several others) terminates a search.

21. Try out the movement commands until you are comfortable with them. It is a good idea to have a definite goal or result in mind, then try a string of commands to see if the result is what you wanted.

Deleting and Modifying Text

	Forward	Backward	Begin	End	Delete Forward	Delete Backward	Transpose
Character	c-F	c-B			c-D	RUBOUT	c-T
Word	m-F	m-B	m-B	m-F	m-D	m-RUBOUT	m-T
Lisp Form	c-m-F	c-m-B	c-m-A	c-m-E	c-m-K	c-m-RUBOUT	c-m-T
Line	c-N	c-P	c-A	c-E	c-K	CLEAR INPUT	c-X c-T
Sentence	m-E	m-A	m-A	m-E	m-K	c-X RUBOUT	

Figure 18. Chart of Zmacs Commands

Text can be deleted character-by-character or in larger units (Figure 18). Any unit larger than a character which is deleted can be *yanked* (brought) back. This is useful if you accidentally delete too much. To yank deleted text back, press c-Y.

Global kill ring

A useful part of the window system is the *global kill ring*. This is a storage place for all units larger than a character that have been deleted. Global means that it is available throughout the window system. It is especially useful when you want to move some text from one place to another. For instance, if you have typed some text into one buffer and want to move it to another buffer, you can delete the text and yank it back. You can also move text or Lisp forms from one window to another in this way. You can yank examples from Document Examiner and run them in the Lisp Listener, or send them in Zmail.

Everything more than a character that you have deleted is stored on the kill ring until you reboot. You can yank back the last thing killed with c-Y. After c-Y, successively pressing m-Y brings back earlier killed items. You can also search through the kill ring by pressing c-sh-Y and supplying a string to search for. After c-sh-Y, successively pressing m-sh-Y brings back earlier killed items containing the same string.

Undoing in Zmacs

Almost anything you do in Zmacs can be undone. Not only can you yank back deleted text, but you can undo any action that affects the appearance of your text. Type in or delete some text and then try pressing c-sh-U to undo what you have done. If it turns out you don't really want to undo it, press c-sh-R to redo what you undid.

Deleting

Forward

c-D Deletes the character that the cursor is on top of.

`m-D` Deletes the word that the cursor is on top of (from the current cursor position to the end).

`c-K` Deletes to the end of the line exclusive of the carriage return. Use another `c-K` to delete the carriage return.

`m-K` Deletes the sentence that the cursor is on top of (from the current cursor position to the period).

Backward

`RUBOUT` Deletes the character to the left of the cursor.

`m-RUBOUT` Deletes the word to the left of the cursor (from the current cursor position to the beginning).

`CLEAR INPUT` Deletes from the current cursor position to the beginning of the line.

`c-X RUBOUT` Deletes the sentence to the left of the cursor (from the current cursor position to the beginning).

Replacing

Sometimes you want to change several occurrences of one string to some other string. Zmacs has two replacing commands. Both of these commands work from the current cursor position to the end of the buffer. *Replace String* automatically replaces all occurrences of a string with another string you specify. *Query replace* waits at each occurrence of the string being replaced for you to tell it whether to replace the string or not.

`c-sh-%` This command prompts you to enter two strings: first, the string to replace; second, the string to replace it with. It replaces all occurrences of string1 with string2 from the cursor to the end of the buffer.

`c-sh-%`*<string1>*`<RETURN>`*<string2>*`<RETURN>`

`m-sh-%` This command prompts you to enter two strings as well. It replaces string1 with string2, querying you before each replacement for all occurrences from the cursor to the end of the buffer. Press `SPACE` to execute the replace or the `RUBOUT` key to skip executing that replace and move to the next candidate. You can also press the `HELP` key for more possibilities.

`m-sh-%`*<string1>*`<RETURN>`*<string2>*`<RETURN>`

Transposing

`c-T` Transposes the character the cursor is over with the character preceding the cursor.

`m-T` Transposes the word following the cursor with the word preceding the cursor.

`c-X c-T` Transposes the line the cursor is over with the preceding line.

Walk-through for Deleting and Modifying Text

1. Go to the top of the buffer with `m-sh-<`. You can also get to the top of the buffer using the scroll bars on the left side of the window and then clicking the mouse where you want the cursor to be.

2. Press `c-sh-%`. When you are prompted for the string to replace, type

   ```
   will<RETURN>
   ```

3. When you are prompted for the string to replace `will` with, type

   ```
   would<RETURN>
   ```

   Notice that all the `will`s in the buffer are changed to `would`s. Notice also that you are told that five replacements have been made. (If you have made any typing mistakes, you might not have exactly five replacements.)

4. Now press `m-sh-%`. Type

   ```
   would<RETURN>will<RETURN>
   ```

   for the command arguments.

   Notice that the cursor moves to the end of the first occurrence of the string `would` and stops. Press `SPACE`. Notice that `would` is changed to `will` and the cursor moves to the end of the next occurrence of `would`.

5. Press `RUBOUT`. Notice that this occurrence of `would` is not changed, but the cursor moves on to the next occurrence.

6. Press `SPACE` until you are informed that the query replace is finished.

7. Move the cursor to the `b` in the word `lobsters`.

8. Press `m-RUBOUT`.

9. Press `m-D` three times.

10. Press `c-Y`. Notice that all the text you killed has come back.

11. Press `c-D` three times.

12. Press `c-Y`. Notice that the three characters you just deleted do not come back. Instead, the last text you killed in units larger than a character appears, because any unit larger than a character that is deleted can be *yanked* back.

13. Press c-sh-U. The text from the kill ring disappears and the three characters you deleted come back. Press c-sh-R. The undo is redone. Press c-sh-U. It is undone again.

14. Press c-N.

15. Press c-K. Notice that the entire line does not disappear, just the part from the cursor to the end of the line.

16. Press c-K again to remove the carriage return.

17. Press c-Y to bring the line and carriage return back.

18. Press m-B.

19. Press c-T.

20. Press c-T again.

21. Press c-A.

22. Press m-T.

23. Press m-T again.

24. Move to some point in the text and press c-SPACE.

25. Move around in the text, using any of your cursor control commands. Starting from where you pressed c-SPACE, the text is marked out. You have selected a region.

26. Press c-W and the marked region disappears. Press c-Y. The region is back. Use this technique to move text from one place to another.

27. Mark another region. This time press m-W. The region marking disappears but the text is still there.

28. Move your cursor a little bit. Press c-Y. The text you marked before appears right at your cursor. Use this technique to copy text from one place to another. Press c-sh-U. The extra copy disappears again.

29. Make any other changes you care to make in the text.

30. Press c-X c-S to save the file. Notice that Zmacs notifies you when the file has been written to disk.

31. Press c-X c-S again. Notice that the file is not written out, since no changes have been made since the last time you wrote it out. Zmacs tells you (No changes need to be written.)

32. Press `c-sh-U`. Nothing is undone. Zmacs tells you `No changes have been made to this buffer.`

33. Press `c-X c-W`. You are prompted for a pathname. Type

    your-file-server:>*your-login-name*>`zmacs-test-2.text<RETURN>`

    Note that you can write the file even when you have not made changes. Also, the name of your buffer has changed to be the same as the new file name.

34. Press `c-X c-F`. When you are prompted for the pathname, type

    your-file-server:>*your-login-name*>`zmacs-test.text<RETURN>`

    Notice that you are put into a buffer that has the same contents as the one you were in, since the two files are the same, but the new buffer is associated with the file you just found.

    If you try to find a file that you are already editing, using `c-X c-F` does not get a copy of the file from disk, but puts you into the buffer you already have associated with that file.

Documentation References

- See the section "Entering Zmacs".
- See the section "Creating and Saving Buffers and Files".
- See the section "Using the Zmacs Editor".
- See the section "Inserting Text in Zmacs".
- See the section "Moving the Cursor in Zmacs".
- See the section "Deleting and Transposing Text in Zmacs".

Getting Familiar with Files

Introduction

This chapter introduces files and file handling. It covers pathnames as well as common file and directory operations.

- Workbook: The File System
- Workbook: Pathnames
- Workbook: File and Directory Operations in the Command Processor
- Workbook: Walk-through for File and Directory Operations
- Workbook: Getting Familiar with Files Documentation References

The File System

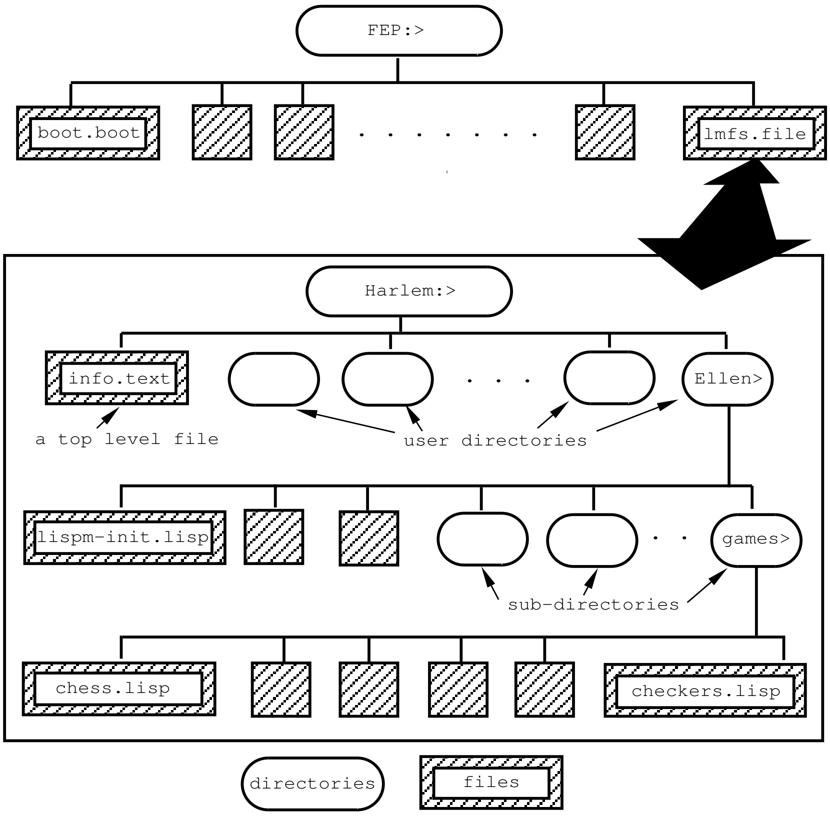


Figure 19. File System structure

A *file* is a section of the disk on which you store data. Storing the data in a file makes it easy to access it again when you want it. If you do not store your data in a file, it disappears when you cold boot. All sorts of data can be stored in files: program text, compiled programs, pictures, and letters to Mom, to name a few.

On the Symbolics computer, there are two kinds of files, *FEP files* and *LMFS files* (Figure 19). FEP files are stored directly on the disk. One of these FEP files is called *lmfs.file*. *All* the LMFS files are stored in this file. The data stored in lmfs.file are LMFS files. Your data are stored in LMFS files, never directly in FEP files. This chapter shows you how to use Lisp Machine File System (LMFS) files; for a discussion of FEP files: See the section "Workbook: Overview of the Machine".

Within lmfs.file, your files are organized into *directories* (Figure 19). Within a directory, files can again be organized into *subdirectories*. Thus, the file system is *hierarchical* and *tree-structured*. Hierarchical means that one directory can be said to be "below" or "above" another directory. Tree-structured means that the file system can be thought of as looking like a tree, with a root (lmfs.file), then branches (directories) which split into more branches (subdirectories), and finally leaves (files).

Pathnames

In the Zmacs chapter, you learned about the components of a pathname. (See the section "Workbook: Zmacs".) Fortunately, most file commands provide you with a *default pathname*. When the system provides a default pathname, you do not have to specify any of the components of the pathname that you want which match those of the default pathname. Thus, if the default pathname is

```
SYMBOLICS1:>Rocco>old-hack.lisp
```

and you want the file

```
SYMBOLICS1:>Rocco>new-hack.lisp
```

all you have to type for the pathname is

```
new-hack
```

The host (*SYMBOLICS1*), directory (*Rocco*), and type *(lisp)* are all supplied from the default pathname (Figure !). Remember that your input *always* overrides the default.

We are using the host (*SYMBOLICS1*) as our file-server for these examples, just as we used the host (*Towhee*) as our file-server in the picture (Figure 19). For the exercises later in this chapter, you need to use the host at your site that we refer to as (*your-file-server*). All of these are hosts that contain a LMFS file.

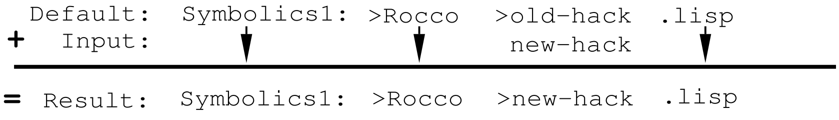

Default: Symbolics1: >Rocco >old-hack .lisp
+ Input: new-hack
= Result: Symbolics1: >Rocco >new-hack .lisp

Figure 20. Pathname Defaulting

If, instead, you want the file

```
SYMBOLICS1:>Randee>interesting-program.lisp
```

you have to type

```
>Randee>interesting-program
```

The host and type are still provided by the default (Figure !). (The > is referred to as "down," so the pathname above is called "down Randee down interesting-program.")

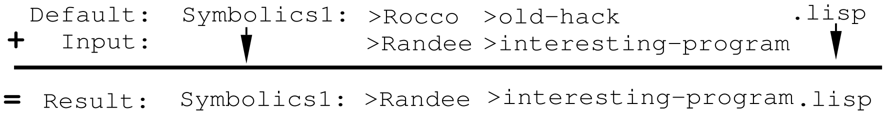

Default: Symbolics1: >Rocco >old-hack .lisp
+ Input: >Randee >interesting-program
= Result: Symbolics1: >Randee >interesting-program.lisp

Figure 21. More Pathname Defaulting

If you have the following default:

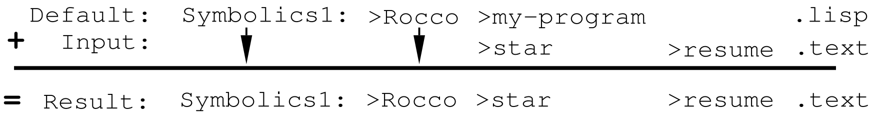
```
SYMBOLICS1:>Rocco>my-program.lisp
```

but you want a file from a subdirectory, such as:

```
SYMBOLICS1:>Rocco>star>resume.text
```

then you type:

```
star>resume.text
```



```
  Default:  Symbolics1: >Rocco >my-program          .lisp
+   Input:                     >star        >resume .text
= Result:   Symbolics1: >Rocco >star        >resume .text
```

Figure 22. Still More Pathname Defaulting

Notice that you type no > before the directory *star* (Figure 22). This says that the directory *star* is a subdirectory of *Rocco*, rather than being a directory at the same level, the way *Randee* was. Also notice that you had to specify the file type, as well as the file name, because you wanted a file of a different type from that provided by the default.

In addition to names and types, pathname components also have a version number. This is incremented by 1 every time the particular file is saved. If you do not specify a version number, then the default is the most recent version of the file (if the file system supports version numbers — not all of them do). None of our examples specify version numbers, because the only time you should specify one is when you do not want the most recent version of a file. If you do specify a version number and then try to save the file, the editor tries to save the file with that specific version number, thus trying to write over an existing file and causing an error.

Wildcards

A pathname may contain asterisks (*). These are called *wildcards*. Wildcards are used in place of components to mean "all the possible values of this component." One special use of wildcards is for specifying directories and subdirectories. One asterisk where the directory component should be means "all directories at this level." Two asterisks for a directory component means "all directories and subdirectories." For instance, to refer to all the files on a host named CAMBRIDGE, you type:

```
CAMBRIDGE:>**>*.*.*
```

To refer to all the files with the type .text on a host, you type:

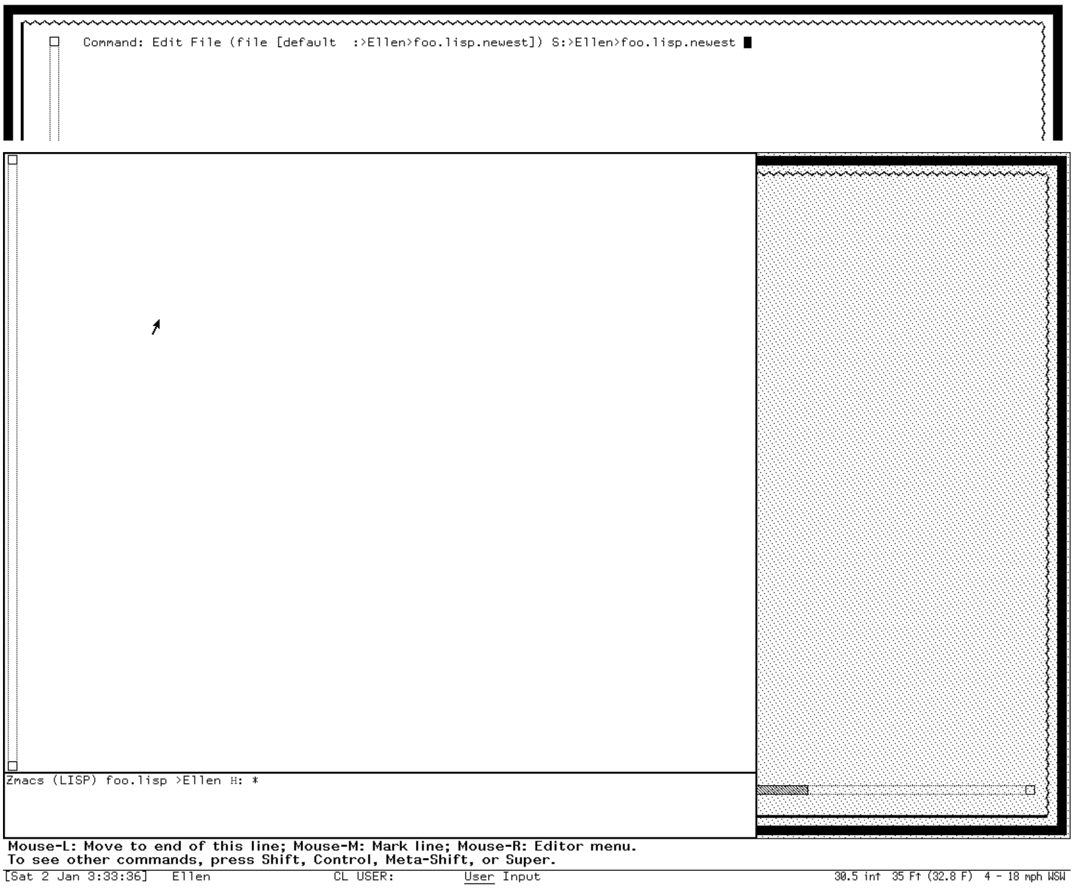
```
CAMBRIDGE:>**>*.text.*
```

File and Directory Operations in the Command Processor

You can use several programs to perform a variety of operations on files and directories. In this section, we're going to talk only about the basic Command Processor commands for handling files and directories. Later chapters discuss additional ways of operating on files and directories. See the section "Workbook: More Files". See the section "Workbook: Additional Files".

As you know, Command Processor commands are typed wherever you see the `Command:` prompt, usually at a Lisp Listener window. They generally take keywords, and you can find out what the possible keywords are by pressing the `HELP` key when you are prompted with `(keywords)`. The pathname arguments for these commands default to the last pathname you specified in the Command Processor.

File Operations



```
Command: Edit File (file [default  :>Ellen>foo.lisp.newest]) S:>Ellen>foo.lisp.newest
```

```
Zmacs (LISP) foo.lisp >Ellen H: *
```

```
Mouse-L: Move to end of this line; Mouse-M: Mark line; Mouse-R: Editor menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Sat 2 Jan 3:33:36]    Ellen              CL USER:          User Input                30.5 in↑  35 F↑ (32.8 F)  4 - 18 mph WSW
```

Figure 23. The **Edit File** Command

Edit File <*pathname*>

Automatically selects the Zmacs window, creates a new buffer for the specified file, and brings that file into the buffer (Figure 23). If the file does not exist, an empty editor buffer is created. The Zmacs command `c-X c-S` saves this buffer, creating a new file with the specified file name. If the file does exist, the contents are copied into the buffer so that you can edit them.

Delete File <*pathname*>

Undelete File <*pathname*>

If your file-server is a Symbolics computer, a file you delete does not actually disappear. Instead, the file is "marked for deletion," and thus can be undeleted if you change your mind. A deleted file is not permanently erased until its directory is *expunged*. This is known as *soft deletion*.

Both of these commands offer a default pathname and take the keyword :Query. This keyword can have a value of Yes, No, or Ask.

- If you press `RETURN` without typing the keyword the default is No, meaning "don't ask, just do it."

- If you type the keyword, but don't type a value, just press `RETURN`, the default is Yes, meaning "ask before deleting or undeleting each file," and you are prompted to enter `(Y or N)` before each file is deleted or undeleted.

You can delete and undelete directories with these commands by specifying a pathname with *.directory* as the file type. For instance, to delete the directory `SYMBOLICS1:>temp>`, you would type

```
Delete File SYMBOLICS1:>temp.directory<RETURN>
```

A directory cannot be deleted unless it is empty, which means that all the files in it have to have been deleted and then the directory has to have been expunged. You need to expunge a directory only once; all the deleted files disappear.

Show File <*pathname*>

Shows you the contents of a file on your screen. If you want to edit it, use the Edit File command.

Directory Operations

Show Directory <*pathname*>

Shows you a directory listing. If you wish to edit a directory, use the Edit Directory command.

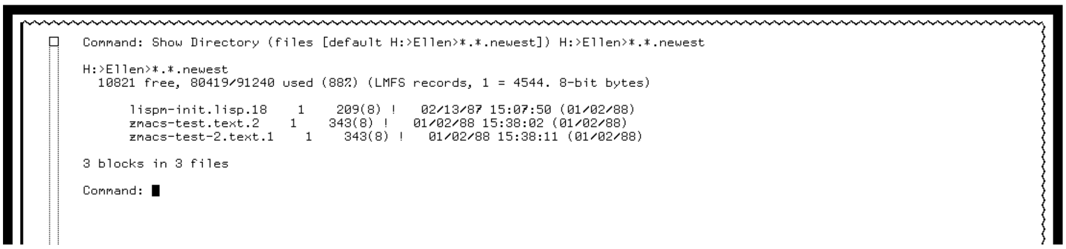
Edit Directory <*pathname*>

Takes you into Dired (the Directory Editor in Zmacs) with the directory specified. Read the section on Dired before using this command: See the section "Workbook: The Directory Editor - Dired".

Create Directory <*pathname*>

Creates the directory specified by <*pathname*>. The higherlevel directories must already exist if you are trying to create a subdirectory.

Expunge Directory <*pathname*>
Expunges the directory specified, causing all the files marked for deletion in that directory to disappear permanently.



```
Command: Show Directory (files [default H:>Ellen>*.*.newest]) H:>Ellen>*.*.newest

H:>Ellen>*.*.newest
  10821 free, 80419/91240 used (88%) (LMFS records, 1 = 4544. 8-bit bytes)

      lispm-init.lisp.18    1    209(8) !   02/13/87 15:07:50 (01/02/88)
      zmacs-test.text.2    1    343(8) !   01/02/88 15:38:02 (01/02/88)
      zmacs-test-2.text.1    1    343(8) !   01/02/88 15:38:11 (01/02/88)

3 blocks in 3 files

Command: █
```

Figure 24. The **Show Directory** Command

Walk-through for File and Directory Operations

This walk-through generates a lot of text. If you wish to clear your screen, press the REFRESH key. If you see the line **MORE** at the bottom of your window, press SPACE.

1. Go to a Lisp Listener window using SELECT L.

2. Type

   ```
   Show<SPACE>Directory<SPACE>
   ```

3. The default pathname should be *your-file-server*:>*your-login-name*>*.*.*. If it is, press SPACE; otherwise, type in your directory's pathname.

4. Press <RETURN>.

5. A listing of the files in your directory should appear (Figure 24). Notice the files zmacs-test.text and zmacs-test-2.text that you created and saved in the Zmacs walk-throughs.

6. Type

   ```
   Delete<SPACE>File<SPACE>
   ```

7. The default pathname should have the correct host and directory components, so just type

   ```
   zmacs-test-2.text<RETURN>
   ```

8. Type the Show Directory command again. Notice that the file `zmacs-test-2.text` now has a `D` before the file name, which indicates that the file has been marked for deletion.

9. Type

   ```
   Show<SPACE>File<SPACE>zmacs-test-2.text<RETURN>
   ```

   Notice that you can't look at the contents of the file, even though you see it in the directory listing. (If you do see the contents of the file, you are probably looking at an earlier version of the file, so check the version number.)

10. Type

    ```
    Undelete<SPACE>File<SPACE>
    ```

11. The default should be `zmacs-test-2.text`, so accept it by pressing `SPACE`, then press `RETURN`. However, if you have more than one copy of this file, you have to specify the version number of the deleted version.

12. Type the Show File command again. Notice that you can now see the contents of the file.

13. Type the Show Directory command once more. Notice that the `D` has disappeared from before the file name.

14. Type

    ```
    Create<SPACE>Directory<SPACE>your-file-server:>your-login-name>star><RETURN>
    ```

    This creates a new subdirectory under your top-level directory. If you do not save any files into this directory, you can delete it using the Delete File command. If you do save files into the directory, all the files must be deleted and the directory expunged before you can delete the directory.

15. Type

    ```
    Delete<SPACE>File<SPACE>
    ```

16. The pathname argument should be

    ```
    your-file-server:>your-login-name>star.directory
    ```

    Type in as much of this as you have to and press `RETURN`.

17. Type the Show Directory command again. Make sure that the directory pathname is *your-file-server*:>*your-login-name*>`*.*.*`. Notice that the file `star.directory` is marked as deleted.

18. Type

    ```
    Expunge<SPACE>Directory<SPACE><SPACE><RETURN>
    ```

19. Type the Show Directory command again. Notice that the `star` directory has disappeared.

Documentation References

- **Pathnames**
  See the section "Pathnames".
- **File and Directory Operations**
  See the section "Dictionary of Command Processor Commands".

Document Examiner

Introduction

This chapter introduces the basic features of Document Examiner that you need to look up topics of interest in the online documentation database.

- Workbook: Document Examiner Overview
- Workbook: The Document Examiner Frame
- Workbook: Basic Document Examiner Commands
- Workbook: Walk-through for Basic Document Examiner Commands
- Workbook: More Document Examiner Commands
- Workbook: Walk-through for More Document Examiner Commands
- Workbook: Bookmarks
- Workbook: Document Examiner Documentation References

Overview

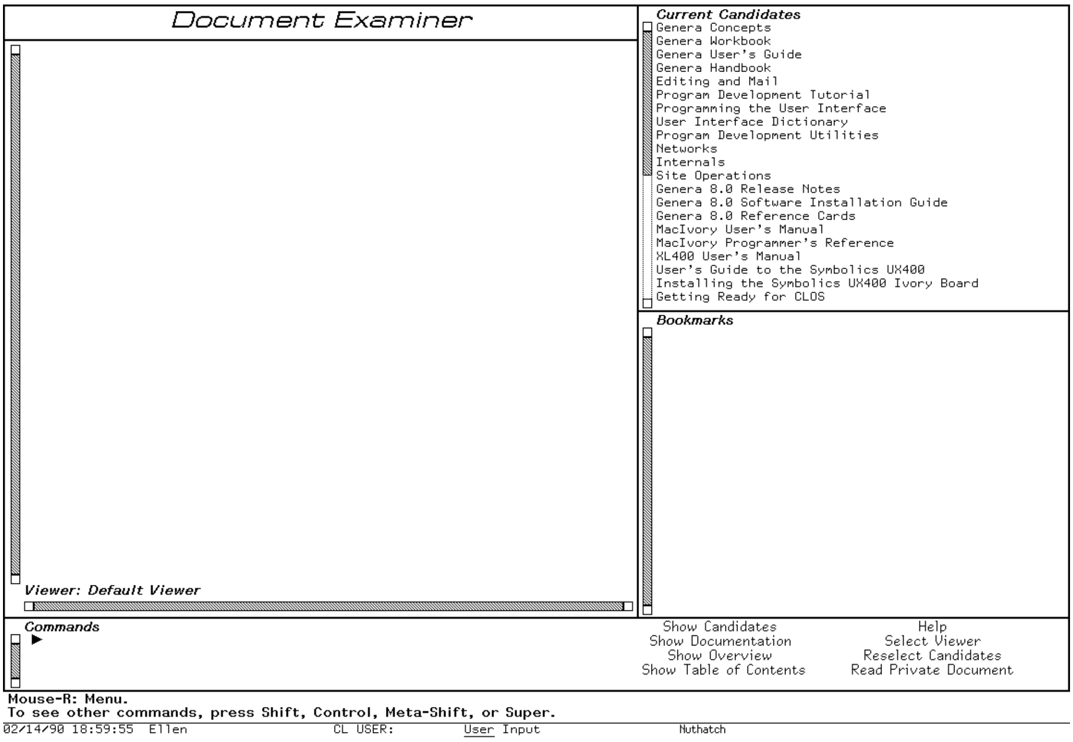
The entire Symbolics documentation set is available in *Document Examiner*, which allows you to:

- Search for and view specific topics or associated ones.
- Learn where a topic is discussed in the printed documentation.
- See what other topics are discussed in the same chapter.

- Use the mouse to select a different topic from the text to learn more about it, then get back to the original topic via an automatic bookmark.

Note: If you cannot find information about a given topic or get an error message while trying to read a topic into Document Examiner, it could mean that the documentation files are not installed at your site or that the translation files are incorrect. See your site administrator for more information about the availability of documentation.

The Document Examiner Frame



```
Genera Concepts
Genera Workbook
Genera User's Guide
Genera Handbook
Editing and Mail
Program Development Tutorial
Programming the User Interface
User Interface Dictionary
Program Development Utilities
Networks
Internals
Site Operations
Genera 8.0 Release Notes
Genera 8.0 Software Installation Guide
Genera 8.0 Reference Cards
MacIvory User's Manual
MacIvory Programmer's Reference
XL400 User's Manual
User's Guide to the Symbolics UX400
Installing the Symbolics UX400 Ivory Board
Getting Ready for CLOS
```

```
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
02/14/90 18:59:55  Ellen              CL USER:      User Input              Nuthatch
```

Figure 25. The Document Examiner Frame

Select Document Examiner by pressing `SELECT D` (if you haven't done so already). You see the *Document Examiner frame* (Figure 25), which is divided into four *panes*:

- **Viewer** - When you request a view of documentation on a particular topic, the text is displayed in the *Viewer* pane. It is the largest pane of Document Examiner.

- **Current Candidates** - When you ask Document Examiner to find all sections of documentation that relate to a given *topic* (the subject of your inquiry — it can be a phrase, a word or even part of a word), the section names are listed in the

Current Candidates pane. You can then click the mouse on a specific section name and see an overview of the topic or have the text brought into the viewer. When you first enter Document Examiner, the titles of the books in the Symbolics documentation set are listed in the Current Candidates pane (Figure 25).

- **Bookmarks** - Every time the text of a topic is brought into a viewer, the section name is listed in the *Bookmarks* pane. This does not happen when you just show an overview of a topic. You can always click Left on one of these *bookmarks* to have a topic redisplayed. This pane is initially blank.

- **Commands** - When you want Document Examiner to perform some action, type a command in the *Commands* pane (all Symbolics keyboard input is displayed in this pane) or click on one of the command options on the right side of the Commands pane. All command options can be typed as well as selected with the mouse.

Note: The mouse is a useful partner to Document Examiner. Always look at the mouse documentation lines before clicking to learn the effect of clicking Left, Middle, or Right.

Basic Document Examiner Commands

In this section (and throughout the Symbolics documentation) a word or phrase in brackets, such as [Help], indicates a menu choice.

- [Help] - You can get help by clicking Left on [Help] in the Commands pane, typing the Help command, or by pressing the HELP key. These ways of getting help list the various Document Examiner commands, including how to scroll, search for text, make hardcopies and move around in the documentation database.

  If you want detailed help on Document Examiner, click Middle on the [Help] option in the Commands pane. Text explaining Document Examiner in detail appears in the viewer.

- [Show Candidates] - You can locate all the sections that discuss a particular topic by searching with [Show Candidates] and supplying a word to base a search on. [Show Candidates] generates a list of *candidates*, which are sections of documentation with that word in the title or as one of the indexing keywords associated with it. This candidates list appears in the Current Candidates pane. The walk-through exercise describes the effects of various mouse clicks on how [Show Candidates] searches for documentation regarding a particular topic. (See the section "Workbook: Walk-through for Basic Document Examiner Commands".)

  If you move the mouse over one of the choices in the Current Candidates pane and read the mouse documentation lines, you see that not only can you read that topic into the viewer, but you can also show an overview of the topic or select from a menu of options. All this is discussed later.

Once a topic is read into the viewer, parts of its text are *mouse-sensitive* (a box is drawn around the topic that the mouse is over, indicating that you can use the mouse in some way) just like the Lisp Listener. You can perform operations on these mouse-sensitive topics exactly as though they were candidate choices.

Walk-through for Basic Document Examiner Commands

1. Enter Document Examiner. (`SELECT D`)

2. Press the `HELP` key.

3. Look at the commands that do not initially fit in the viewer by pressing `SCROLL` to move to the next screen. Watch the viewer's *scroll bar*, adjacent to its left margin, move too. The scroll bar works the same as the ones in (for example) the Lisp Listener and Zmacs do.

4. Try `m-SCROLL` to move back to the previous screen.

5. Now try `SCROLL`. What did that do? Press `m-SCROLL` until you reach the beginning of the topic.

6. Move the mouse over the [Show Candidates] command in the Commands Pane and look at the mouse documentation lines.

   Clicking either Left or Right "Shows topic names associated with a word."

7. Click Left on [Show Candidates] and enter the topic

   ```
   finding
   ```

   and press `RETURN`.

8. Dozens of candidates are listed in the candidates pane. You'll have to use the scroll bar to see them all. (Move the mouse to the little square box at the bottom of the scroll bar and check the mouse documentation line to see how to scroll the pane.) Some have the word "finding" as part of the title, others have "find", some read as if they have something to do with finding, but don't have the word in their titles, and some are just mysteries.

   The [Show Candidates] menu item (which works the same as the Show Candidates command) takes the word you have given it and looks through the entire documentation database for that word and a series of variations on it. It looks in both the titles of each documentation section and in the keyword list for each section. The keyword list is supplied by the documentation writer to help you find the topic even if you don't know its name.

Each of the topics listed has something to do with finding, as indicated by its title or keywords. Sometimes it is you the user doing the finding, sometimes it is a program finding something, and sometimes it is a tool that assists you or the program in finding something.

9. Now that we've found our topics, what can we do with them? Move the mouse to the Current Candidates pane and position the cursor over one of the candidates. Look at the mouse documentation lines: You can read a topic into the viewer, show an overview, or put in a bookmark.

10. Move the mouse over `Looking Up Documentation`. You'll find it several screens into the list, which is in alphabetical order. Click Right to see a menu of the operations you can perform on this candidate. Click Left on [Show Documentation]. Lo and behold, it's more about Document Examiner, from another book. Now click Middle on the same topic name. You'll see an overview that tells you where the topic fits in and what its keywords are. (See Figure !). Much of the overview display is also mouse-sensitive. Doing an overview of a topic before you bring it into a viewer is a good idea. You can make sure that you want that topic and not a related one.

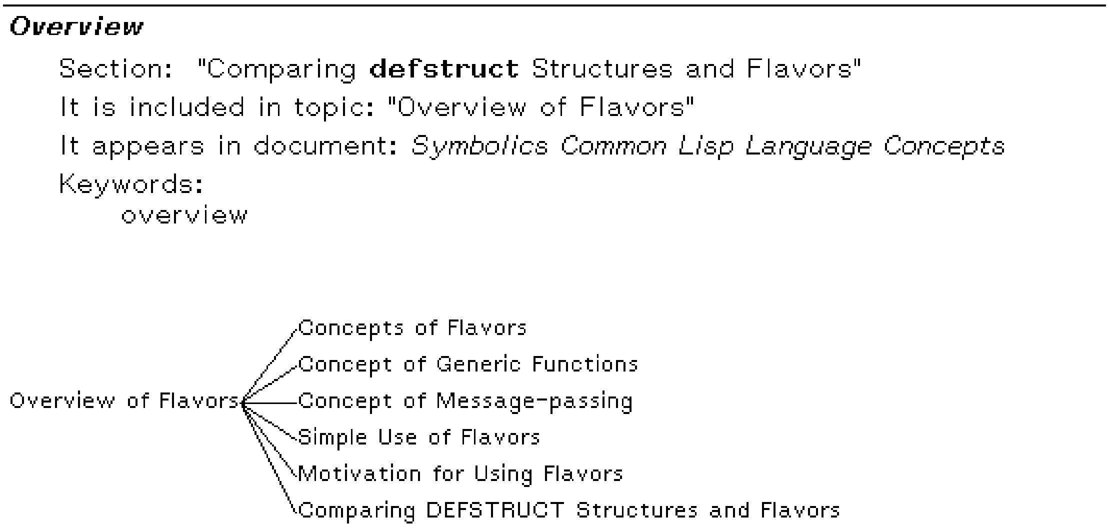
11. Now that you know how to use the [Show Candidates] option, try to use it to look up documentation on the topics covered in the workbook. Although [Show Candidates] works with two words, its usually better to start with one word and then narrow it down with two if you have to. For more about using [Show Candidates], see the section "Workbook: More Document Examiner".

More Document Examiner Commands

- [Reselect Candidates] - Clicking on [Reselect Candidates] allows you to choose, via a menu, any list of candidates that you have created using the [Show Candidates] menu option or the Show Candidates command. The reactivated list of candidates you choose appears in the Current Candidates pane.

- [Show Documentation] - Clicking on [Show Documentation] brings a topic directly into the Viewer pane without first listing candidates in the Current Candidates pane. To use this command, you have to know the exact topic you are looking for. Names of Lisp functions, special operators, and variables are often topic names.

- [Show Overview] allows you to see the overview (Figure !) of a topic that you type in. The overview appears in the Viewer pane, describing the topic's place in the documentation. The section, parent topic, and document volume in which it appears is given, as well as a list of *keywords* (other topics that, when searched for, include this topic in their list of candidates).



Overview

Section: "Comparing **defstruct** Structures and Flavors"
It is included in topic: "Overview of Flavors"
It appears in document: *Symbolics Common Lisp Language Concepts*
Keywords:
overview

Overview of Flavors
Concepts of Flavors
Concept of Generic Functions
Concept of Message-passing
Simple Use of Flavors
Motivation for Using Flavors
Comparing DEFSTRUCT Structures and Flavors

Figure 26. An Overview

Below the description is a map of where this topic appears in relation to other topics in the same chapter (Figure 26). These other topics are mouse-sensitive as well. You press `SPACE` to remove the overview window. You can also get an overview by clicking Middle on a candidate. Overviews are very useful and make it much easier to find out if the section contains exactly what you want to know.

- [Show Table of Contents] displays a table of contents (Figure !) for a topic. Any documentation topic — a volume in the documentation set, a chapter or section, or a language object — can produce a table of contents. The table of contents includes only subordinate topics, while the overview includes superior and equal topics as well.

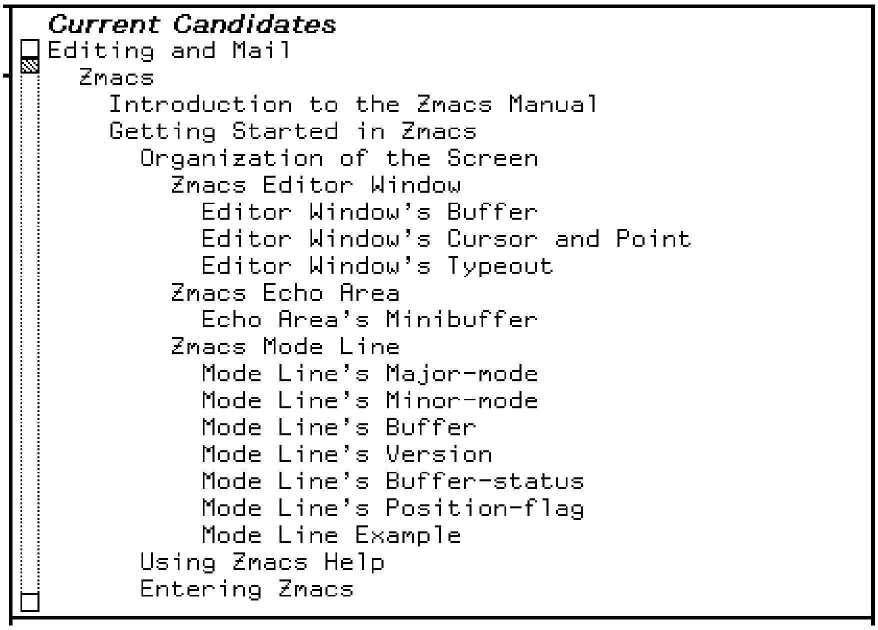
For all these menu options there are also Document Examiner commands with the same names.

You can also use the documentation facilities in Zmacs. Two Zmacs commands you might find useful right away are `m-X` Show Documentation and `m-X` Show Candidates. The `m-X` Show Candidates command gives you a mouse-sensitive list of candidates in the editor buffer.

In addition, the Command Processor offers two commands, Show Documentation and Show Overview.

Walk-through for More Basic Document Examiner Commands

1. Let's look at the "finding" candidates again. (If you do not still have the candidates lists from the last walk-through, make a [Show Candidates] query using `finding` again or retrieve the old list by using [Reselect Candidates].)



```
Current Candidates
Editing and Mail
  Zmacs
    Introduction to the Zmacs Manual
    Getting Started in Zmacs
      Organization of the Screen
        Zmacs Editor Window
          Editor Window's Buffer
          Editor Window's Cursor and Point
          Editor Window's Typeout
        Zmacs Echo Area
          Echo Area's Minibuffer
        Zmacs Mode Line
          Mode Line's Major-mode
          Mode Line's Minor-mode
          Mode Line's Buffer
          Mode Line's Version
          Mode Line's Buffer-status
          Mode Line's Position-flag
          Mode Line Example
      Using Zmacs Help
      Entering Zmacs
```

Figure 27. Table of Contents for "Text Editing and Processing"

2. Move the mouse over the sub-topic `More HELP Commands For Finding Out About Zmacs` (below `Looking Up Documentation`) and click Middle to get an overview. Move the mouse over the graphic outline. Each entry in the graph is mouse-sensitive. You also see the keywords, if any, associated with this topic.

3. At the bottom of the graph, `Example of a Search String for HELP A` is mouse-sensitive. Many items in documentation are mouse-sensitive in this way. They are usually in bold print or in quotations. Wherever you see a mouse-sensitive documentation title, you can click Left to show the documentation, click Middle to show the overview, or click Right to see a menu that allows you to do these things and others, including making a hardcopy.

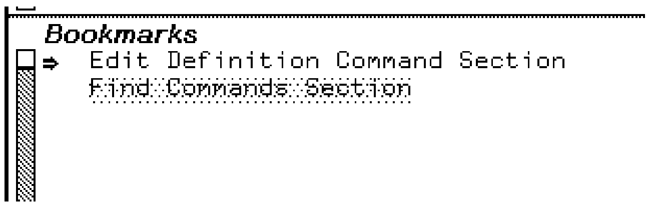
4. If you know exactly what topic you would like to see, use the [Show Documentation] option. Click on [Show Documentation], then type:

   ```
   what is a world load
   ```

   and press `RETURN`.

5. The topic goes directly into the Default Viewer and a bookmark is inserted. No candidates are listed.

Bookmarks



Bookmarks

⇒ Edit Definition Command Section

Find Commands Section

Figure 28. Bookmarks

Every time you read a topic into the viewer, the section name is placed into the Bookmarks pane (Figure 28). An arrow at the left margin of the Bookmarks pane points to the section currently in the viewer. Move the mouse over to one of the bookmarks and look at the mouse documentation lines. Various mouse clicks can read a previous section into the viewer, get an overview, or discard the section completely. You can also bring up a menu to perform these operations and others, such as hardcopying the topic.

You can also place a section name in the Bookmarks pane without actually reading the text into the viewer. There are several ways to do this. Move the mouse on top of a topic in the Current Candidates pane and notice in the mouse documentation line that one choice is to put in a bookmark. You can do the same with any of the words that are mouse-sensitive in the Viewer pane.

When you read a section into the Bookmarks pane, the section name is covered by a gray strip (Figure 28). This indicates that, although the section name is in the Bookmarks pane, the actual text has not been read into the viewer. After you read in the text, the gray strip disappears.

Show Candidates Command

Show Candidates is useful when you're not quite sure what you're looking for. Show Candidates is designed for searching. Supply one or more words to Show Candidates and it will search the entire documentation database for documentation topics with the same or similar word or words in the title, or in the list of keywords (similar to index entries) associated with each topic.

By default, Show Candidates uses a heuristic (or "smart") searching strategy. The advantage of heuristic matching is that you don't have to think about the form of the word to supply as a lookup request; the heuristic approach finds singulars and plurals, gerunds, negations, and so on, because the search is based on the *stem* of your lookup request.

For example: The stem of "initialize" is "initial". Show Candidates `initialize` matches any topic names and keywords containing "initialize", "initialization", "initialized", "initial", ":INITIAL", "[Initialize]", and so forth.

For more information, see the section "Searching Through Documentation".

Documentation References

Once you're more familiar with the system it is well worth your while to read Document Examiner documentation (using Document Examiner, of course!). If you have a solid understanding of how this program works, you'll be more inclined to use it and benefit from it.

- **Overview**
  See the section "Introduction to Document Examiner".
- **The Document Examiner Window**
  See the section "Document Examiner Window".
- **Basic Document Examiner Commands**
  See the section "Looking Up Documentation".
- **Bookmarks**
  See the section "Putting Topics Aside for Future Reading".

Intermediate Topics

Introduction

"Workbook: Intermediate Topics" expands on many of the topics covered in "Workbook: Basic Topics". This section does not teach essential material that you must know before attending a Symbolics Education Services class. What it does cover is material that makes your work on a Symbolics computer much easier. "Workbook: Intermediate Topics" has four chapters explaining more interesting and useful commands to build on the skills that you developed in "Workbook: Basic Topics":

"Workbook: More Getting Around"
Details additional features available in the Lisp Listener and introduces the Input Editor, the Command History, and the System menu. We also provide more information on the Command Processor.

"Workbook: More Zmacs"
Provides additional information on Zmacs features like help and completion. Using regions to move text is also explained.

"Workbook: More Files"
Builds on the information about files and directories you have already seen in the previous section. Directory listing format and pathnames for other machines are also described.

"Workbook: More Document Examiner"
Expands on the material in the previous section by providing information on using private documents and additional viewers to increase the ease with which you use the software.

More Getting Around

Introduction

This chapter presents several features, such as completion, the Input Editor, and the Command History, that make it easier for you to use the Command Processor. The System menu is also discussed.

- "Workbook: Completion in the Command Processor"
- "Workbook: Walk-through for Completion"
- "Workbook: The Input Editor"
- "Workbook: Walk-through for the Input Editor"
- "Workbook: The Command History"
- "Workbook: Walk-through for the Command History"
- "Workbook: Useful Keys"
- "Workbook: Creating More Than One of an Activity"
- "Workbook: Walk-through for Creating More Than One of an Activity"
- "Workbook: The System Menu"
- "Workbook: Editing the Screen Using the System Menu"
- "Workbook: Walk-through for Editing the Screen"
- "Workbook: Window Operations Using the System Menu"
- "Workbook: Summary of Activities in the System Menu"
- "Workbook: More Getting Around Documentation References"

Completion in the Command Processor

In general, it is not necessary for you to type an entire command name to the Command Processor; you can use *completion*. You can also edit what you are typing by using the Input Editor, and bring back things you have already typed by using the Command History. These features, which are described in this chapter, make it much easier and faster for you to use Genera. Most of these features are available everywhere in Genera, but we describe their use only in the Command Processor here.

Completion is a feature provided so that you do not have to type more of a command than is necessary to uniquely specify that command. Very often, you have to type only a few letters of a command's name. There are two kinds of completion: *partial* completion and *token* completion.

Partial completion When you are typing either a command name or a keyword argument name, pressing SPACE "fills in" or "completes" the remainder of the word you are currently typing and all previous words. If there is more than one possible completion, the Command Processor fills in letters up to the point where the possible completions diverge.

Token completion When you are typing either a command name or a keyword argument name, pressing COMPLETE fills in as much of the entire command name you are typing as is unique.

You can also use the mouse-sensitivity of the Lisp Listener that was discussed in "Workbook: Getting Around the System" to avoid typing the entire command name. You can click sh-Left on some command names to activate those commands. Check the mouse documentation lines to see if this feature is implemented on a particular command. You can use m-COMPLETE to get a menu of possible arguments for a command that you have typed in. These arguments are mouse-sensitive, so you click on them to select them.

There are several other keystrokes that give you useful information when you have partially typed a command:

HELP or c-? Either of these keystrokes shows you the possible completions of the command name.

Right This mouse click gives you a menu of possible completions.

c-/ This keystroke shows you all the commands which contain the partial strings you have typed.

Walk-through for Completion

1. Select a Lisp Listener.

2. Type

```
L<SPACE>
```

   The Command Processor fills in o after the L and then stops, since you could be typing any one of Load, Login, or Logout.

3. Press HELP to see all the possible completions.

4. Click the right mouse button to see a menu of completions.

5. Now type

   ```
   F<SPACE>
   ```

   Notice that the Command Processor completes `Lo` to `Load` and `F` to `File`, since that is the only possible completion of `Lo<SPACE>F`.

6. Press `CLEAR INPUT` to erase this command.

7. Type

   ```
   Se<SPACE>O
   ```

8. Press `c-/` to see the commands that contain the strings "Se" and "O".

9. Press the `COMPLETE` key. Notice that it fills in

   ```
   Set Output Base (number base [default 10])
   ```

10. Press `CLEAR INPUT` to erase this command.

11. Type

    ```
    Se<SPACE>O
    ```

    again.

12. Press `SPACE`. Notice that the Command Processor fills in only

    ```
    Set Output
    ```

    even though you have seen that `Se O` is enough to uniquely specify a command.

13. Press `SPACE` again. Notice that the command is now completed.

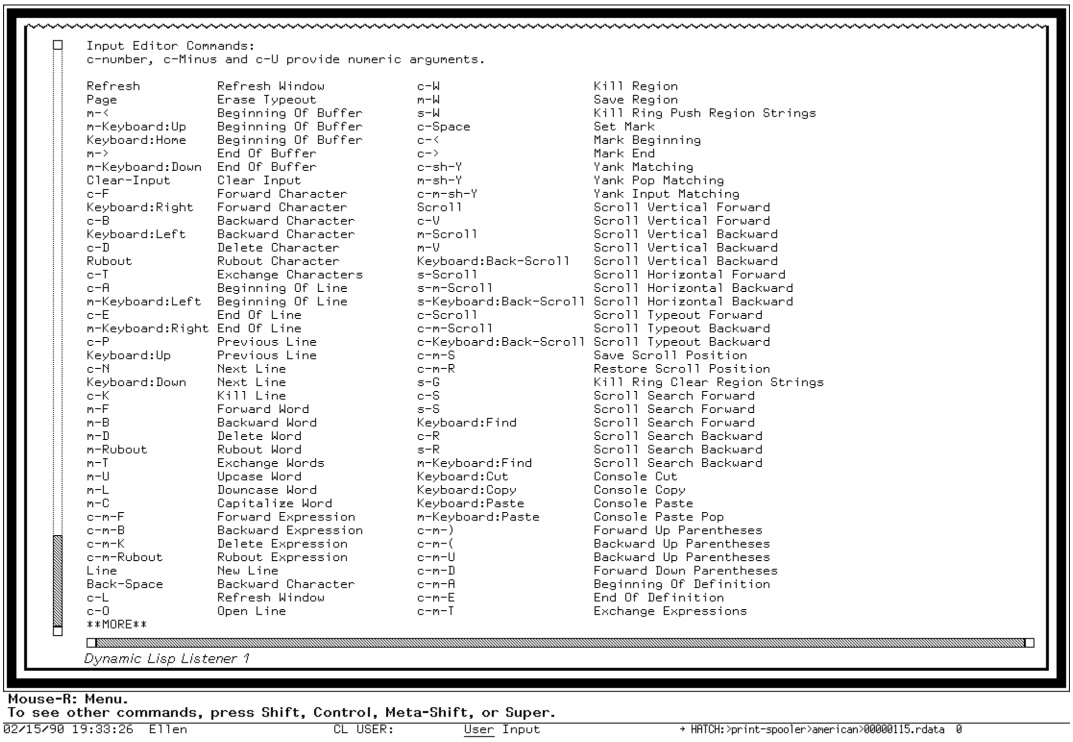
14. Press `CLEAR INPUT` to erase this command.

The Input Editor

A useful feature in the Lisp Listener is the *Input Editor*. As you type characters, the Input Editor saves them in an *input buffer* so that if you change your mind, you can edit the characters. There are several possible commands for editing a line you are typing in to the Command Processor. These commands are a subset of Zmacs commands. You can get a complete list of these commands by typing `c-HELP` (Figure 29).

Walk-through for the Input Editor

In this walk-through, and many others, you might want to clear your screen so that the results of what you do are easy to see. Any time you want to scroll the



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

Refresh          Refresh Window         c-W                     Kill Region
Page             Erase Typeout          m-W                     Save Region
m-<              Beginning Of Buffer    s-W                     Kill Ring Push Region Strings
m-Keyboard:Up    Beginning Of Buffer    c-Space                 Set Mark
Keyboard:Home    Beginning Of Buffer    c-<                     Mark Beginning
m->              End Of Buffer          c->                     Mark End
m-Keyboard:Down  End Of Buffer          c-sh-Y                  Yank Matching
Clear-Input      Clear Input            m-sh-Y                  Yank Pop Matching
c-F              Forward Character      c-m-sh-Y                Yank Input Matching
Keyboard:Right   Forward Character      Scroll                  Scroll Vertical Forward
c-B              Backward Character     c-V                     Scroll Vertical Forward
Keyboard:Left    Backward Character     m-Scroll                Scroll Vertical Backward
c-D              Delete Character       m-V                     Scroll Vertical Backward
Rubout           Rubout Character       Keyboard:Back-Scroll    Scroll Vertical Backward
c-T              Exchange Characters    s-Scroll                Scroll Horizontal Forward
c-A              Beginning Of Line      s-m-Scroll              Scroll Horizontal Backward
m-Keyboard:Left  Beginning Of Line      s-Keyboard:Back-Scroll Scroll Horizontal Backward
c-E              End Of Line            c-Scroll                Scroll Typeout Forward
m-Keyboard:Right End Of Line            c-m-Scroll              Scroll Typeout Backward
c-P              Previous Line          c-Keyboard:Back-Scroll Scroll Typeout Backward
Keyboard:Up      Previous Line          c-m-S                   Save Scroll Position
c-N              Next Line              c-m-R                   Restore Scroll Position
Keyboard:Down    Next Line              s-G                     Kill Ring Clear Region Strings
c-K              Kill Line              c-S                     Scroll Search Forward
m-F              Forward Word           s-S                     Scroll Search Forward
m-B              Backward Word          Keyboard:Find           Scroll Search Forward
m-D              Delete Word            c-R                     Scroll Search Backward
m-Rubout         Rubout Word            s-R                     Scroll Search Backward
m-T              Exchange Words         m-Keyboard:Find         Scroll Search Backward
m-U              Upcase Word            Keyboard:Cut            Console Cut
m-L              Downcase Word          Keyboard:Copy           Console Copy
m-C              Capitalize Word        Keyboard:Paste          Console Paste
c-m-F            Forward Expression     m-Keyboard:Paste        Console Paste Pop
c-m-B            Backward Expression    c-m-)                   Forward Up Parentheses
c-m-K            Delete Expression      c-m-(                   Backward Up Parentheses
c-m-Rubout       Rubout Expression      c-m-U                   Backward Up Parentheses
Line             New Line               c-m-D                   Forward Down Parentheses
Back-Space       Backward Character     c-m-A                   Beginning Of Definition
c-L              Refresh Window         c-m-E                   End Of Definition
c-O              Open Line              c-m-T                   Exchange Expressions
**MORE**
Dynamic Lisp Listener 1

Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
02/15/90 19:33:26  Ellen              CL USER:       User Input           → HATCH:>print-spooler>american>00000115.rdata  0
```

Figure 29. Input Editor Commands

screen to a clear state, press REFRESH or use the Clear Output History command to clear the history.

1. Select a Lisp Listener.

2. Type (just as you see it)

   ```
   lgoin<SPACE>
   ```

3. You should see the message

   ```
   The input read, lgoin , was not a command name or a
   symbol with a value.  Press RUBOUT to correct your input.
   ```

   Actually, you can use other Input Editor commands here as well as RUBOUT. Type c-B until the cursor is on top of the o.

4. Type c-T. Notice that the g and o switch places.

5. Press c-E RUBOUT SPACE. Now the command completes to

   ```
   Login
   ```

   and tells you that you have to specify a user name.

6. Press CLEAR INPUT, since you should already be logged in. Notice that the message about the user name goes away.

7. Type

   ```
   S<SPACE>C<SPACE>P<SPACE>
   ```

   Notice that this completes to

   ```
   S Command Processor
   ```

   This is because there are two commands, Set Command Processor and Show Command Processor Status, that could be the completions of what you typed. (Press the HELP key to see these possible completions.)

8. Press m-B twice to move back to the beginning of the word Command.

9. Press c-B to move the cursor to the space following the S.

10. Type

    ```
    e<RETURN>
    ```

    Notice that the RETURN signals the Input Editor and the Command Processor that you have finished typing the command, so it completes and executes, using the default values for mode and prompt string.

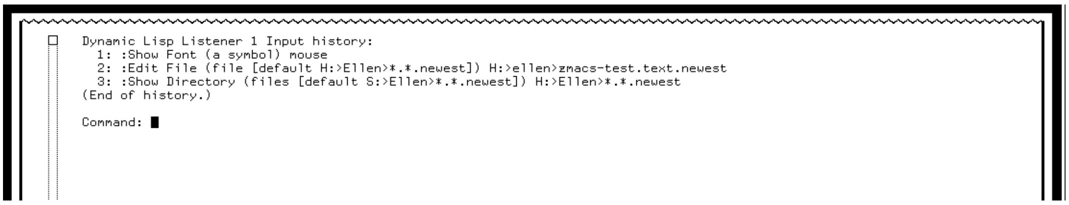
The Command History

Lisp Listeners have scrollable *histories*. This means that you can use the scroll bars to look at output on the window that has scrolled off the visible part of the screen. Much of this output is mouse-sensitive. If you want to clear the screen by scrolling all the output off the visible part of the screen, press the REFRESH key. If you want to clear the whole history, use the command Clear Output History.

You can use c-S to search forward and c-R to search backward for a particular string, just like in Zmacs. The search string appears in the status line.

Each Lisp Listener has a separate history. The history is the record of all the commands or forms that have been typed to that Lisp Listener since you cold booted Lisp. You can use this *input history* after you have executed the Clear Output History command, because the search commands described above do not work after you have cleared the output history. You can see this history by pressing the ESCAPE key (Figure !). The most recent commands appear first. If you have typed many things to your Lisp Listener, not all of them appear, just the twenty most recent ones. To see the entire history, press c-0 ESCAPE. To see the *n* most recent commands, press c-*n* ESCAPE. If several consecutive commands were the same, only the most recent one is shown in the history list.

You can bring back (*yank*) the most recent command from the history by pressing CONTROL-META-Y (c-m-Y). To yank the next most recent command, follow c-m-Y with m-Y. Thus, to yank the third most recent command, you would press c-m-Y m-Y m-Y. Also, to yank command number *n*, you would press c-*n* c-m-Y. For instance, to yank the command listed as #5 in the history list, you would press `c-5 c-m-Y`.



```
Dynamic Lisp Listener 1 Input history:
  1: :Show Font (a symbol) mouse
  2: :Edit File (file [default H:>Ellen>*.*.newest]) H:>ellen>zmacs-test.text.newest
  3: :Show Directory (files [default S:>Ellen>*.*.newest]) H:>Ellen>*.*.newest
(End of history.)

Command: █
```

Figure 30. The Command History

Walk-through for the Command History

1. Select a Lisp Listener.

2. Log in, if you haven't already.

3. Press the `ESCAPE` key. Notice how many commands are listed. There may be only one, the `Login` command that you just used. If all the commands are listed, it says `(End of history.)` at the bottom of the list. If not, it says `(`*n* `more items in history.)`

4. Type the command

   ```
   Show Font MOUSE<RETURN>
   ```

5. Yank back the `Show Font MOUSE` command with `c-m-Y` and press `RETURN`.

6. Press the `REFRESH` key to clear your screen.

7. Now use the scroll bar to scroll back your output.

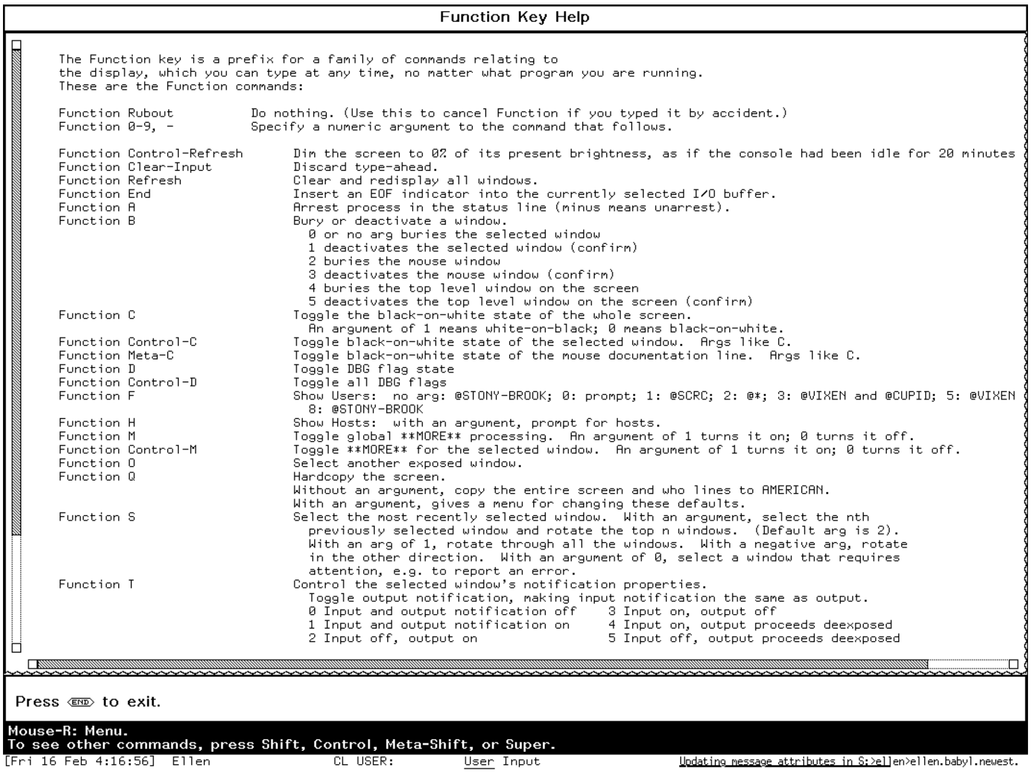
8. Press the `ESCAPE` key again. Notice that now the top item in the history is

   ```
   1: Show Font MOUSE
   ```

   Also notice that the command `Show Font MOUSE` is not listed twice. This is because command #2 was the same as command #1, so it is only listed once.

Useful Keys

`ABORT` Aborts the operation currently in progress. It returns to the command level in such programs as the Lisp Listener, Zmacs,



```
Function Key Help

     The Function key is a prefix for a family of commands relating to
     the display, which you can type at any time, no matter what program you are running.
     These are the Function commands:

     Function Rubout          Do nothing. (Use this to cancel Function if you typed it by accident.)
     Function 0-9, -          Specify a numeric argument to the command that follows.

     Function Control-Refresh      Dim the screen to 0% of its present brightness, as if the console had been idle for 20 minutes
     Function Clear-Input          Discard type-ahead.
     Function Refresh              Clear and redisplay all windows.
     Function End                  Insert an EOF indicator into the currently selected I/O buffer.
     Function A                    Arrest process in the status line (minus means unarrest).
     Function B                    Bury or deactivate a window.
                                     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)
     Function C                    Toggle the black-on-white state of the whole screen.
                                     An argument of 1 means white-on-black; 0 means black-on-white.
     Function Control-C            Toggle black-on-white state of the selected window.  Args like C.
     Function Meta-C               Toggle black-on-white state of the mouse documentation line.  Args like C.
     Function D                    Toggle DBG flag state
     Function Control-D            Toggle all DBG flags
     Function F                    Show Users:  no arg: @STONY-BROOK; 0: prompt; 1: @SCRC; 2: @*; 3: @VIXEN and @CUPID; 5: @VIXEN
                                     8: @STONY-BROOK
     Function H                    Show Hosts:  with an argument, prompt for hosts.
     Function M                    Toggle global **MORE** processing.  An argument of 1 turns it on; 0 turns it off.
     Function Control-M            Toggle **MORE** for the selected window.  An argument of 1 turns it on; 0 turns it off.
     Function O                    Select another exposed window.
     Function Q                    Hardcopy the screen.
                                   Without an argument, copy the entire screen and who lines to AMERICAN.
                                   With an argument, gives a menu for changing these defaults.
     Function S                    Select the most recently selected window.  With an argument, select the nth
                                     previously selected window and rotate the top n windows.  (Default arg is 2).
                                     With an arg of 1, rotate through all the windows.  With a negative arg, rotate
                                     in the other direction.  With an argument of 0, select a window that requires
                                     attention, e.g. to report an error.
     Function T                    Control the selected window's notification properties.
                                     Toggle output notification, making input notification the same as output.
                                     0 Input and output notification off    3 Input on, output off
                                     1 Input and output notification on     4 Input on, output proceeds deexposed
                                     2 Input off, output on                 5 Input off, output proceeds deexposed

Press END to exit.

Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Fri 16 Feb 4:16:56]  Ellen              CL USER:        User Input              Updating message attributes in S:>ellen>ellen.babyl.newest.
```

Figure 31. Function Help

File System Editor, and so forth. It takes effect when the program reads it, rather than immediately.

- c-ABORT is like ABORT, except that it takes effect immediately.

- m-ABORT causes the process to return to the topmost command loop, aborting all computation and other command loops that might be on the stack. It takes effect when read, rather than immediately.

- c-m-ABORT is like m-ABORT, except that it takes effect immediately.

CLEAR INPUT Erases all characters typed since the last command prompt.

COMPLETE Completes (as far as possible) the characters entered to match a command. If the characters typed do not uniquely identify a command, the Command Processor completes the command to the point where the possibilities diverge. For example, Lo does not complete, since a user might want either the command

Load, Login, or Logout.

END

Enters the current command line or form. This is particularly useful when yanking a previously typed Lisp form. If a command line is yanked and then edited, the END key allows you to enter the entire line, even if the cursor is positioned at the beginning or middle of the line.

FUNCTION

Allows you to do interesting things with windows and processes. Type FUNCTION HELP (Figure 31) to see a listing of all the actions that can be taken by pressing the FUNCTION key in conjunction with other Symbolics keyboard keys. For example, FUNCTION-4-T allows output on a deexposed window.

REFRESH

Clears the window, sometimes redrawing the contents. For instance, in a Lisp Listener, pressing REFRESH scrolls the window so that the part you can see is empty, but in Zmacs, pressing REFRESH redraws the contents of the window, clearing out only the echo area.

REPEAT

Causes a key being pressed at the same time to repeat. Normally, no matter how long you hold a key down, only one of that character is sent to Lisp. With the REPEAT key down, the character is sent until you stop pressing the REPEAT key.

RETURN

Ends the current line at the cursor position and moves the cursor down one line. If you enter a complete command, the Command Processor executes it. If you enter a string that is not recognized, the Command Processor notifies you of this and invites you to modify your input. If your input can complete to more than one command, Lisp waits for you to add enough to specify a unique command.

SYMBOL

The character set contains more characters than there are Symbolics keyboard keys, even when they are combined with the modifier keys. SYMBOL allows you to access special characters. Press SYMBOL-HELP (Figure !). A three-part listing appears.

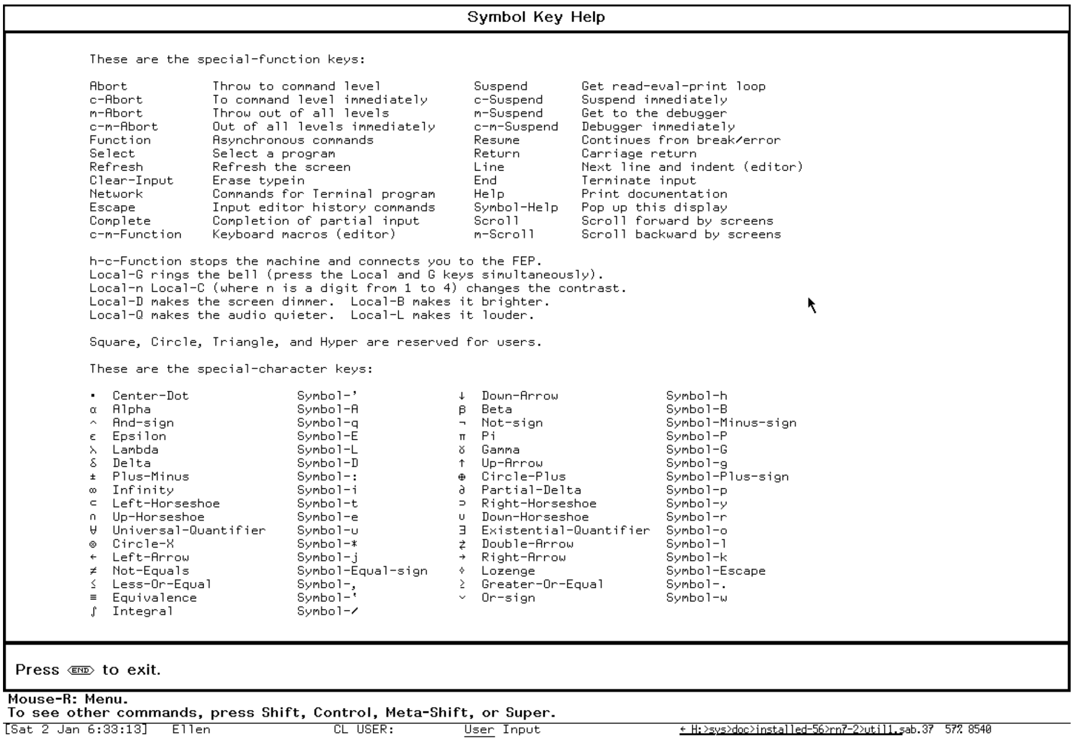
- The first part briefly summarizes the action of each function key.
- The second part documents the LOCAL key.
- The third part documents the SYMBOL key.

SPACE

As well as sending the character " " when you are typing a command, the SPACE key completes the current word (and all previous words) of that command. This is often the fastest way to enter commands. For example, typing se<space>fi<space><space> in the Command Processor causes:

```
Set File Properties (existing file [default SYS:DOC;USER;WB1.TEXT])
```

to be displayed.



```
                              Symbol Key Help

These are the special-function keys:

Abort          Throw to command level          Suspend       Get read-eval-print loop
c-Abort        To command level immediately    c-Suspend     Suspend immediately
m-Abort        Throw out of all levels         m-Suspend     Get to the debugger
c-m-Abort      Out of all levels immediately   c-m-Suspend   Debugger immediately
Function       Asynchronous commands           Resume        Continues from break/error
Select         Select a program                Return        Carriage return
Refresh        Refresh the screen              Line          Next line and indent (editor)
Clear-Input    Erase typein                    End           Terminate input
Network        Commands for Terminal program   Help          Print documentation
Escape         Input editor history commands   Symbol-Help   Pop up this display
Complete       Completion of partial input     Scroll        Scroll forward by screens
c-m-Function   Keyboard macros (editor)        m-Scroll      Scroll backward by screens

h-c-Function stops the machine and connects you to the FEP.
Local-G rings the bell (press the Local and G keys simultaneously).
Local-n Local-C (where n is a digit from 1 to 4) changes the contrast.
Local-D makes the screen dimmer.  Local-B makes it brighter.
Local-Q makes the audio quieter.  Local-L makes it louder.

Square, Circle, Triangle, and Hyper are reserved for users.

These are the special-character keys:

•  Center-Dot               Symbol-'            ↓  Down-Arrow               Symbol-h
α  Alpha                    Symbol-A            β  Beta                     Symbol-B
^  And-sign                 Symbol-q            ¬  Not-sign                 Symbol-Minus-sign
ε  Epsilon                  Symbol-E            π  Pi                       Symbol-P
λ  Lambda                   Symbol-L            γ  Gamma                    Symbol-G
δ  Delta                    Symbol-D            ↑  Up-Arrow                 Symbol-g
±  Plus-Minus               Symbol-:            ⊕  Circle-Plus              Symbol-Plus-sign
∞  Infinity                 Symbol-i            ∂  Partial-Delta            Symbol-p
⊂  Left-Horseshoe           Symbol-t            ⊃  Right-Horseshoe          Symbol-y
∩  Up-Horseshoe             Symbol-e            ∪  Down-Horseshoe           Symbol-r
∀  Universal-Quantifier     Symbol-u            ∃  Existential-Quantifier   Symbol-o
⊗  Circle-X                 Symbol-*            ⇄  Double-Arrow             Symbol-l
←  Left-Arrow               Symbol-j            →  Right-Arrow              Symbol-k
≠  Not-Equals               Symbol-Equal-sign   ◊  Lozenge                  Symbol-Escape
≤  Less-Or-Equal            Symbol-,            ≥  Greater-Or-Equal         Symbol-.
≡  Equivalence              Symbol-'            ∨  Or-sign                  Symbol-w
∫  Integral                 Symbol-/

Press END to exit.
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Sat 2 Jan 6:33:13]   Ellen             CL USER:        User Input              ← H:>sys>doc>installed-56>rn7-2>util1.sab.37  57% 8540
```

Figure 32. Symbol Help

Creating More Than One of an Activity

No doubt you noticed a paragraph in the SELECT HELP display that discussed creating new activities. You can create, for instance, another Lisp Listener. This is very useful when you are running something in one window, and you want to use that activity for something else.

Walk-through for Creating More Than One of an Activity

(The description in this walk-through assumes that you haven't tried using the SELECT and CONTROL keys yet.)

1. Select Dynamic Lisp Listener 1. (You should know how to do this by now.)

2. Now, with Dynamic Lisp Listener 1 selected, press SELECT L.

The screen should flash or beep, because you already have the Dynamic Lisp Listener selected. We're now going to create *another* Dynamic Lisp Listener.

3. Press the SELECT key.

4. Hold down one of the CONTROL keys.

5. With the CONTROL key down, press L.

   Now you should see a new activity (and a new window) named Dynamic Lisp Listener 2. This activity can do anything Dynamic Lisp Listener 1 can do. Now, however, you have two of them. You might do this to keep a certain display on one, while issuing commands to another.

   Note: You can't create another Zmail window.

   With two Dynamic Lisp Listeners, you might ask, "How do I get to the other one?"

6. Press SELECT L.

   This brings you back to Dynamic Lisp Listener 1.

7. Press SELECT L again.

   This brings you back to Dynamic Lisp Listener 2. Using the SELECT key and a letter key repeatedly rotates through all the existing windows of the kind associated with that letter.

The System Menu

The System menu (Figure !) is a facility for doing common operations. You can get to it by holding the SHIFT key down and clicking Right. The System menu appears where the mouse was. To make it disappear (and not do anything) just move the mouse off the menu. You use the System menu to do three kinds of operations:

1. Selecting an activity.

2. Changing a window.

3. Dealing with window activities in general.

The left column of the System menu (Figure 33) is a list of operations on windows in general. You can use options in this column to create new windows and change the sizes and locations of windows.

The middle column (Figure 33) is a list of operations on the current window.

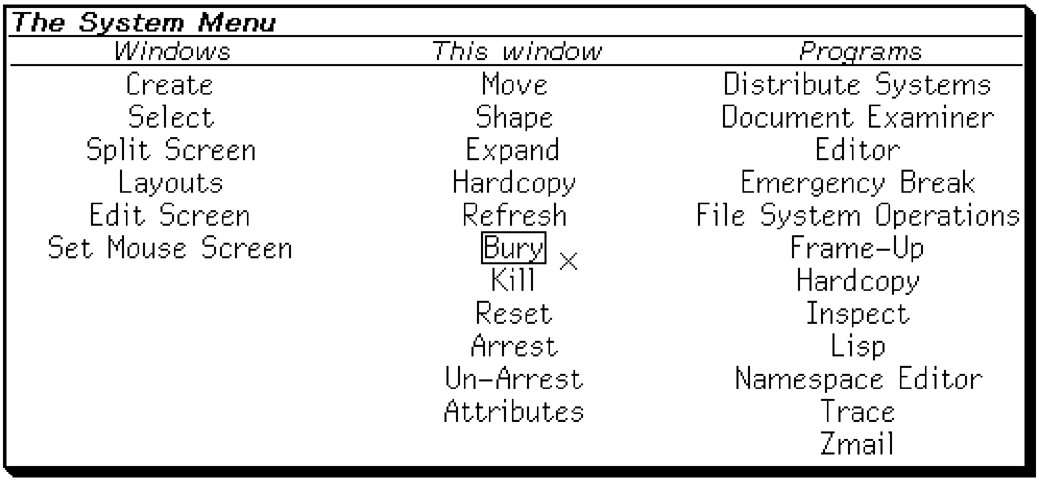


Figure 33. The System Menu

The right column (Figure 33) is a list of activities that can be selected. Some of these are available via the SELECT key.

Editing the Screen Using the System Menu

The left column of the System menu contains options for editing the screen.

- Clicking on the [Create] option gives you a menu of types of windows. When you select one of these, a window of that type is created.

- The [Edit Screen] option allows you to do many interesting things to your windows. Among these are:

  - Create new windows (just like [Create]).

  - Reshape windows, specifying the corners with the mouse.

  - Expand windows, either one window or all of them. Windows are expanded to fill the space that is not already occupied by an exposed window.

  - Move windows around the screen.

  - Expose windows that are partially visible.

  - Bury windows that are partially visible.

- The [Split Screen] option lets you select any number of windows, and splits the screen evenly among them.

- Once you have redesigned the screen using [Edit Screen] or [Split Screen], you can use the [Layouts] option to save and recall that particular configuration of windows. Clicking on [Layouts] gives you a menu of configuration names and an option to save the current configuration. If you click on a configuration name, that configuration is exposed. If you choose to save the current configuration, you are prompted to enter a name for it. This name is then available on the menu of configurations.

Walk-through for Editing the Screen

1. Select the editor.

2. Bring up the System menu (hold down `SHIFT` and click Right).

3. Click on [Edit Screen].

4. Click on [Create]. Select [Lisp] as the type of window to be created.

5. The mouse cursor now looks like the upper left corner of a rectangle. Put the mouse cursor over the gray area of the screen and click Left.

6. The mouse cursor is now the normal arrow shape, pointing at the lower left corner of a rectangle. Move the mouse cursor around a little, then click Left while the rectangle is fully in the gray area of the screen. You now have a new Lisp Listener, and the Edit Screen menu has reappeared.

7. Click on [Expand window].

8. The mouse cursor is now a circle with a cross in it. Move this cursor over the new Lisp Listener and click Left. Notice that the window expands to fill all of the gray area of the screen, and the Edit Screen menu reappears.

9. Click on [Reshape].

10. Move the cursor over the editor window and click Left.

11. Reshape the editor window to be narrower.

12. Click on [Expand all]. Notice that both the editor window and the Dynamic Lisp Listener expand to fill up the gray area.

13. Now that you've got this configuration, click on [Exit].

14. Bring up the System menu and click on [Layouts].

15. Click on [Save This]. When you are prompted for a name, enter something like "Lisp and Editor".

16. Press SELECT D.

17. Bring up the System menu and click on [Layouts] again.

18. Click on the "Lisp and Editor" configuration. Notice that both the windows are exposed when you do this.

Window Operations Using the System Menu

[Move] Allows you to move the window that the mouse is over around the screen.

[Shape] Allows you to reshape the window that the mouse is over. This is like the [Reshape] option on the Edit Screen menu.

[Expand] Expands the window that the mouse is over to fill the available space. This is like the [Expand] option on the Edit Screen menu.

[Hardcopy] Hardcopies the window that the mouse is over (if you have a bitmap printer at your site).

[Refresh] Refreshes the window that the mouse is over.

[Bury] *Buries* the window that the mouse is over. Burying a window means that it is put (logically) underneath all the rest of the windows.

[Kill] Asks for confirmation, then kills the window that the mouse is over. In general, you should not do this to any window that you did not create yourself.

[Reset] Asks for confirmation, then resets the process associated with the window that the mouse is over.

[Arrest] *Arrests* the process associated with the window that the mouse is over. Among other things, arresting a process means that it stops I/O and all computation until it is un-arrested.

[Un-Arrest] Un-arrests the process associated with the window that the mouse is over.

[Attributes] Allows you to change the attributes of the window that the mouse is over. Some of the attributes you might want to change are whether or not more processing is enabled and whether or not the window is shown in reverse-video.

Summary of Activities in the System Menu

[Lisp] A Lisp Listener. Available on SELECT L.

[Edit] Zmacs. Available on SELECT E.

[Inspect] The Inspector. Available on SELECT I.

[Mail] Zmail. Available on SELECT M.

[Trace] Not really an activity, but simply a menu interface to the **trace** debugging facility. This is useful for programmers.

[Emergency Break] Not really an activity, this provides a mechanism for evaluating Lisp forms without using the window system. Rarely used, it may be useful for programmers.

[Frame-Up] The Layout Designer. This allows you to design *constraint frames* for programs. Frames are windows that have subwindows, such as the Zmacs window.

[Namespace Editor] The Namespace Editor. This allows you to modify the namespace database to add or change parameters about users, hosts, or printers. Not available on a SELECT key.

[Hardcopy] Not really an activity, this is simply a menu interface to the file-printing facility.

[File System] The File System Maintenance window. Available on SELECT F.

[Font Editor] The Font Editor. This separately loadable system allows you to create and modify your own fonts. Not available on a SELECT key.

Documentation References

- See the section "Entering Commands".
- See the section "Editing Your Input".
- See the section "The Input Editor Program Interface".
- See the section "Command History".
- See the section "Index of Special Function Keys".
- See the section "Using the System Menu".

More Zmacs

Introduction

This chapter explains more about Zmacs and teaches you more Zmacs commands so that you can edit more effectively.

- "Workbook: More Zmacs Help"

- "Workbook: More Zmacs Scrolling"
- "Workbook: More Zmacs Regions"
- "Workbook: More Zmacs Walk-through for Scrolling and Regions"
- "Workbook: More Zmacs Modes"
- "Workbook: More Zmacs Extended Commands"
- "Workbook: More Zmacs Completion"
- "Workbook: More Zmacs Other Useful Commands"
- "Workbook: More Zmacs Walk-through for Other Useful Commands"
- "Workbook: More Zmacs Documentation References"

Help

To get help in Zmacs, press the HELP key. The possible choices are listed in the minibuffer at the bottom of the window. If you press HELP again, the definitions of the commands appear at the top of the editor window (Figure 34).

Some of the more useful help options are:

A Apropos. Allows you to enter a string, which Zmacs uses as a search key. Zmacs returns a list of all the commands whose names contain that string.

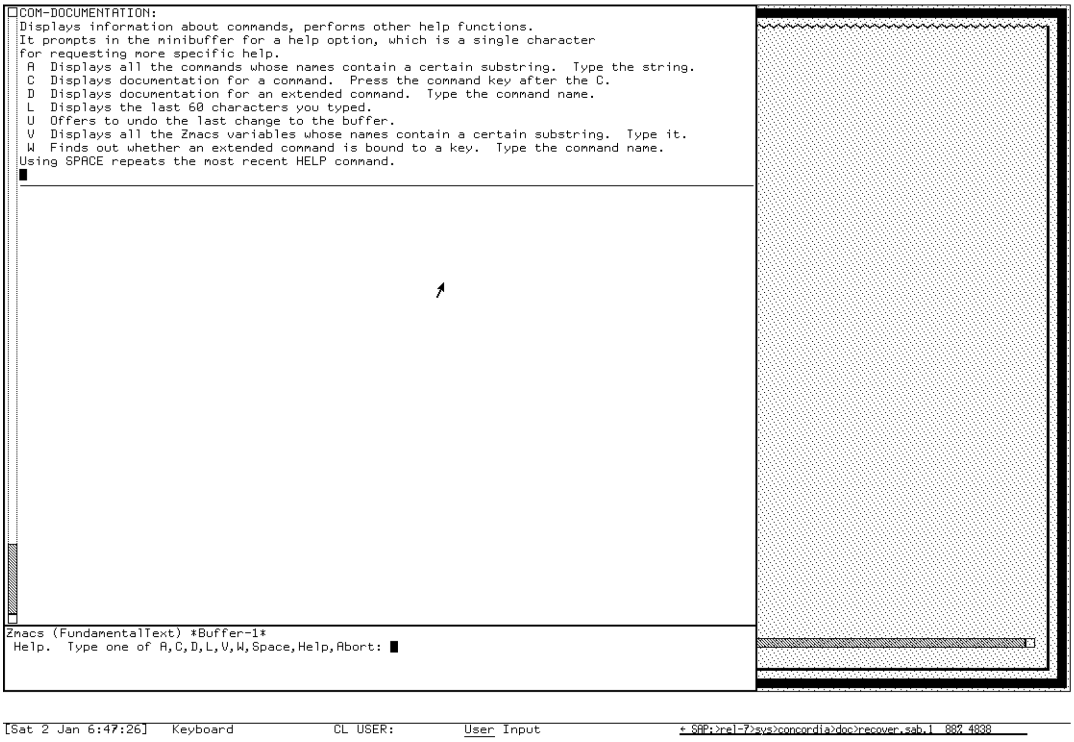
C Displays documentation for a command. Just type the command (c-A, for instance) and Zmacs tells you what it does.

D Displays documentation for an extended command.

U Undo. Allows you to undo the last change to the buffer. This is very helpful no matter whether the change you made was large (such as a global replace) or small (such as a centering command).

You can also get help by giving keystroke commands and then pressing the HELP key. c-X HELP give you a list of the single keystroke commands and CONTROL key combinations. m-X HELP gives you a listing of the over 100 possible m-X commands. c-m-X HELP gives you a listing of the over 200 possible c-m-X commands.

Scrolling



```
COM-DOCUMENTATION:
Displays information about commands, performs other help functions.
It prompts in the minibuffer for a help option, which is a single character
for requesting more specific help.
 A  Displays all the commands whose names contain a certain substring.  Type the string.
 C  Displays documentation for a command.  Press the command key after the C.
 D  Displays documentation for an extended command.  Type the command name.
 L  Displays the last 60 characters you typed.
 U  Offers to undo the last change to the buffer.
 V  Displays all the Zmacs variables whose names contain a certain substring.  Type it.
 W  Finds out whether an extended command is bound to a key.  Type the command name.
Using SPACE repeats the most recent HELP command.

Zmacs (FundamentalText) *Buffer-1*
 Help.  Type one of A,C,D,L,V,W,Space,Help,Abort:

[Sat 2 Jan 6:47:26]    Keyboard          CL USER:        User Input            ← SAP:>rel-7>sys>concordia>doc>recover.sab.1  882 4838
```

Figure 34. Help in Zmacs

You can scroll (move the text up and down) within your Zmacs buffer by using the mouse or the Symbolics keyboard.

Keyboard

`SCROLL, c-V`	Displays the next screenful of text.
`m-SCROLL, m-V`	Displays the previous screenful of text.
`c-L`	Redraws the screen, moving the cursor and the text around it to the center of the window.

Mouse

You can scroll the text in your buffer using the mouse and the scroll bar, just as you do in the Lisp Listener. In addition, if you move the mouse slowly towards the top or bottom of the text window, in any corner, an arrow pointing up or down and surrounded by a circle and a box appears, with the mouse cursor becoming x-shaped. Holding the left mouse button down then scrolls the window, line by line, in that direction.

Regions

Regions are an effective way to deal with an arbitrary amount of text. You can create a region by *marking* it. A marked region is underlined, so you can always tell if you have a region. You can put the marked region into the global kill history by using a *kill* command. All the windows in the system look into the same global kill history, so you can kill anything bigger that a character and put it into the global kill history. You can pull anything from the global kill history into a window by using the *yanking* commands.

Marking

Mouse

Position the mouse at the beginning of the text you wish to mark. While holding down the left button, drag the mouse to the end of the section of text you wish to mark.

Keyboard

`c-SPACE`	Marks the beginning of the region. Use basic cursor movement commands to go to the end of the region.
`c-sh-<`	Marks from the current cursor position to the beginning of the buffer.
`c-sh->`	Marks from the current cursor position to the end of the buffer.
`c-X H`	Marks the entire buffer.

Killing

`c-W`	Deletes the marked region.
`m-W`	Copies the marked region into the kill history, without removing the text from your buffer.

Yanking

`c-Y`	Yanks the most recent region or item out of the kill history and places it at the cursor.
`m-Y`	Cycles through the kill history, yanking the previous item. This keystroke works *only* after you have done a `c-Y`.

Walk-through for Scrolling and Regions

1. Press SELECT E to select the Zmacs editor.

2. Find the file (c-X c-F) more-text.text. This file should not exist, so you should get an empty buffer.

3. Type in:

   ```
   "The time has come," the Walrus said,
           To talk of many things:
   Of shoes - and ships - and sealing wax -
           Of cabbages - and kings -
   And why the sea is boiling hot -
           And whether pigs have wings."
   ```

4. Press c-X H to mark the whole buffer. Notice that you can tell which text is marked because it is all underlined.

5. Press m-W to place that text on the kill ring, without actually removing it from the buffer.

6. Move to the end of the buffer by clicking the mouse in the blank area below where the text appears.

7. Press RETURN twice.

8. Press c-Y. The text that you marked should appear below the original text.

9. Move the cursor to the word ships in the third line of the second copy of the paragraph.

10. Press c-SPACE.

11. Move the cursor forward a few characters with c-F. Notice that a line is dragged along behind the cursor as it moves.

12. Press c-P. Notice that the line now extends the other way from the ships following the cursor.

13. Press m-B three times.

14. Press c-W to kill the underlined text.

15. Move to the beginning of the buffer with m-sh-<.

16. Press c-Y three times. Notice that you have three copies of the text you killed.

17. Press `m-Y`. Notice that the last copy of the killed text was replaced with the text you killed originally. `m-Y` cycles back through the list of kills.

18. Press `c-N` five times.

19. Press `c-sh-<`. Notice that the text from the cursor to the top of the buffer is marked.

20. Press `m-W`.

21. Move to the end of the buffer with `m-sh->`.

22. Press `c-Y` four times. Notice that your buffer scrolls as it is filled. Also notice that it says `[More above]` in the mode line.

23. Press `m-SCROLL`. Watch how the text moves and where the cursor appears as you do the next few commands.

24. Press `SCROLL`.

25. Press `m-V`.

26. Press `c-V`.

27. Press `c-L`. Notice that the text around the cursor is now centered in the buffer.

28. Move the mouse cursor towards the left edge of the buffer until part of the scroll bar turns gray. Notice that the gray section corresponds to the text that is now showing in the buffer.

29. Leave the mouse over the scroll bar and watch the scroll bar while you press `m-V`.

30. Notice that the gray section moves up the scroll bar as you move up in the text of the file.

31. Notice the line extending across the buffer from the mouse cursor. Watch the scroll bar and click Left. Notice that the line of text which was near the mouse cursor has moved to the top of the buffer. Notice also that the gray section has moved lower, as you moved lower in the text.

Modes

Zmacs has many *modes*, which are ways of customizing the editor. When you first select the editor, you are in *Fundamental Mode*. You have been working in *Text*

Mode so far, since if the extension of a file is `.text`, Zmacs assumes that it should use Text Mode unless some other mode is specified. Text Mode is for English text. It sets the tab stops for indentation.

There are two kinds of modes, major and minor. Most major modes are for programming languages, except Text Mode and Fundamental Mode. A buffer has only one major mode. Minor modes are options to make the editing environment more customized. A buffer can have more than one minor mode at a time. One useful minor mode is *Auto Fill Mode*. This causes text to be automatically *filled*, so that as you type, carriage returns are inserted when you type past the end of the line. To turn it on, type `m-X Auto Fill Mode`. To turn it off, type the command a second time.

Extended Commands

There are more Zmacs commands than can be represented with control- and meta-keystrokes. You can get another set of commands by pressing `m-X` and then typing the command name (and possibly arguments). These commands are called *meta-X* commands or *extended commands*. Some useful extended commands are listed below.

Mode Commands

`m-X` `Auto Fill Mode`
Causes your text to be automatically filled.

`m-X` `Lisp Mode` Puts you into a desirable environment for editing Lisp code.

`m-X` `Text Mode` Puts you into a desirable environment for editing text.

Buffer Commands

`m-X` `List Buffers`
Shows you a list of all your buffers. The buffers are mouse-sensitive, so you can click on one to select it or perform other operations, such as saving it. (You can also use `c-X c-B` to list your buffers.)

`m-X` `Kill or Save Buffers`
Shows you a menu of all your buffers, with options to kill or save each one. This is useful when you have modified many buffers and you wish to save them all at once, rather than going through and saving each one.

`m-X` `Revert Buffer`
Gets the most recent version of the buffer from disk. This is useful if you have made major changes to a file, then change your mind. Remember that the current information in the buffer is lost when you use this command.

Adding Existing Text

m-X	Insert Buffer	Inserts specified buffer into current buffer beginning at the cursor.
m-X	Insert File	Inserts specified file into current buffer beginning at the cursor.

Hardcopy Commands

m-X	Hardcopy Buffer	Prompts you for the name of a buffer to send to the printer. The default is the current buffer.
m-X	Hardcopy File	Prompts you for the name of a file to send to the printer. The default is the file associated with the current buffer.
m-X	Show Printer Status	Prompts you for the name of one or more printers, and displays a list of the jobs that are currently in the printer queue(s).

Completion

Extended commands support completion, just like Command Processor commands. When you use the extended commands, you see the word `Completion` in the upper right corner of the minibuffer. This means that you don't have to type the whole command, just enough to insure uniqueness. For instance, if you want to list your buffers, you type:

```
    m-X L<SPACE>B
```

then press the `RETURN` key. You see the completed command name `List Buffers` in the minibuffer.

Zmacs also supports pathname completion. If you press the `COMPLETE` key in the middle of typing a pathname, Zmacs attempts to complete the pathname.

Other Useful Commands

c-O	Open line. Inserts a carriage return after the cursor.
c-m-L	Selects the previous buffer. This is useful for switching between two buffers.
m-A	Moves the cursor to the beginning of the sentence.
m-E	Moves the cursor to the end of the sentence.

`m-K` Kills from the current cursor position to the end of the sentence.

`m-C` Capitalizes a word. Capitalizes the character under the cursor and lowercases the rest of the word.

`m-L` Lowercases a word, from the current cursor position to the end of the word.

`m-U` Uppercases a word, from the current cursor position to the end of the word.

`m-S` Centers the current line.

`m-Q` Fills paragraph by adjusting right margin.

`c-X K` Kills a buffer. Offers a default and a new buffer to select.

`c-X B` Selects a buffer. Offers a default buffer to select.

`c-X c-B` Lists current buffers and their associated files. The list is mouse-sensitive, so you can click on your choice to select that buffer. `m-X` List Buffers does the same thing.

Walk-through for Other Useful Commands

1. Make sure you are still in an editor buffer. (If not, press `SELECT E`.) Your text from the earlier walk-through should still be there.

2. Hold down the `META` key and press `X`.

3. Type the extended command `Text Mode` (you need only type `Tex M`) and press `RETURN`. Answer `Y` to the question. The buffer is now in Text Mode. Move to the top of the buffer, and notice that a line has appeared that says `-*- Mode: Text -*-`.

4. Press `c-X c-B` to list your buffers. Notice that the buffer you are currently in is at the top of the list. Remember which buffer is next in the list. This is the buffer you were in before you selected the current buffer.

5. Press `c-m-L` to see this previous buffer.

6. Press `c-m-L` again to get back. Note that this command is a *toggle* command in that it chooses between two things.

7. Make sure you are still in the buffer `more-text.text`. If you are not, press `c-X c-B` to list the buffers, move the mouse over that buffer name, and click Left on it. This selects the buffer `more-text.text`.

8. Go to the top of the buffer (after the line which says `Mode: Text`) and press `m-K`. Notice that the text disappears from the cursor to the first period.

9. Press m-K again.

10. Press m-C three times. Notice that the words are capitalized.

11. Press c-B twice.

12. Press m-U. Notice that the word is only uppercased from the cursor to the end of the word.

13. Press m-B.

14. Press m-C. Notice that the word starts with a capital, but all the other capital letters have been lowercased.

15. Press c-O.

16. Press m-S.

17. Press m-E three times.

18. This is just a mishmash of text, so you shouldn't save it. Press c-X K.

19. When you are prompted for the buffer to kill, just press RETURN. This defaults to the current buffer.

20. Since you have not saved this buffer, the editor prompts you, asking if you want to save the text before killing the buffer. Type

    ```
    No<RETURN><RETURN>
    ```

21. Press c-X c-B. Notice that the buffer more-text.text is no longer in the list of buffers.

22. Press SPACE to get rid of the buffer list.

Documentation References

- See the section "Using Zmacs Help".
- See the section "Finding Out About Zmacs Commands".
- See the section "Working with Regions in Zmacs".
- See the section "Zmacs Major Modes".
- See the section "Built-in Customization Using Zmacs Minor Modes".
- See the section "Manipulating Buffers and Files in Zmacs".
- See the section "Zmacs Command Completion".
- See the section "Changing Case and Indentation in Zmacs".

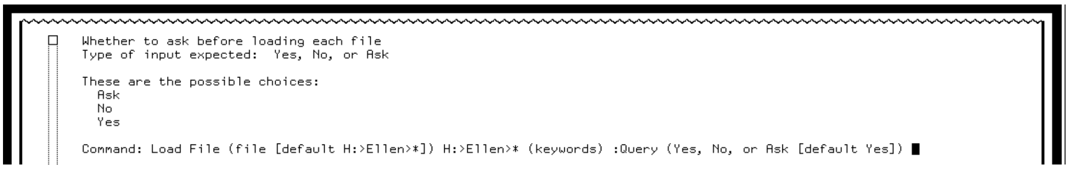
More Files

Introduction

This chapter shows you more Command Processor commands for manipulating files. It also explains how to interpret directory listings and how to use pathnames for other machines.

- "Workbook: More Command Processor File Commands"
- "Workbook: Walk-through for More Command Processor File Commands"
- "Workbook: Directory Listings Explained"
- "Workbook: Pathnames on Other Machines"
- "Workbook: More Files Documentation References"

More Command Processor File Commands



```
□  Whether to ask before loading each file
   Type of input expected:  Yes, No, or Ask

   These are the possible choices:
     Ask
     No
     Yes

   Command: Load File (file [default H:>Ellen>*]) H:>Ellen>* (keywords) :Query (Yes, No, or Ask [default Yes]) █
```

Figure 35. The :Query keyword

`Rename File` *<from-pathname> <to-pathname>*

Changes the name of a file from *<from-pathname>* to *<to-pathname>*. The directory in which the file resides can also be changed in this way, by specifying a new directory in the *<to-pathname>*. You may rename a group of files by using * as a wildcard. For example, suppose you had three files:

```
symbolics1:>rocco>my-program.lisp
symbolics1:>rocco>my-game.lisp
symbolics1:>rocco>my-tool.lisp
```

and you wanted to change the "my" to "new". You could do it by using a wildcard:

```
Rename File symbolics1:>rocco>my-*.lisp new-*.lisp
```

The Rename File command does not copy the file; if you wish to copy the file, use the Copy File command.

This command takes a *:Query* keyword (Figure 35), which can have a value of *(Yes, No, or Ask).* The value defaults to *No* if you do not type in the keyword, meaning don't ask about renaming each file, just do it. If you give the keyword and press `RETURN`, the keyword defaults to *Yes*, meaning ask before renaming each file.

`Copy File` *<from-pathname> <to-pathname>*

Causes a new copy of the file to be created and put in *<to-pathname>*. A copy of the file still exists in *<from-pathname>*. You can copy more than one file at a time by using * as a wildcard.

This command takes a *:Query* keyword, which can have a value of *(Yes, No, or Ask).* The value defaults to *No* if you do not type in the keyword, meaning don't ask about copying each file. If you give the keyword and press `RETURN`, it defaults to *Yes*, meaning ask before copying each file.

`Hardcopy File` *<pathname> <printer>*

Hardcopies (prints) a file. You are prompted to enter the name of a printer. To hardcopy a file, you *must* have a printer hooked up to a machine on your network.

This command takes many keywords, including ones to specify the number of copies to print, whether or not to print page headings, and the starting and ending pages to print.

`Save File Buffers<RETURN>`

Saves all unsaved Zmacs buffers that are associated with files.

This command takes a *:Query* keyword that can have a value of *(Yes, No, or Ask).* The value defaults to *No* if you do not type in the keyword, meaning don't ask about saving each file buffer. If you give the keyword and press `RETURN`, it defaults to *Yes*, meaning ask before saving each file buffer.

`Load File` *<pathname>*

Causes the specified file to be *loaded*, which means that you can use any of the functions and variables defined in the file. You can load more than one file by using * as a wildcard. If you do not specify an extension in the pathname, Load File looks first for a file with the specified name and the extension `.bin`. If there is no `.bin` file, it looks for `.lisp`. If you wish to edit or view this file, you must use the Edit File or Show File command.

This command takes a *:Query* keyword, which can have a value of *(Yes, No, or Ask).* If you do not give the keyword, it defaults to *No*, meaning don't ask whether to load each file or not, just do it. If you give the keyword and press `RETURN`, it defaults to *Yes*, meaning ask before loading each file.

Walk-through for More Command Processor File Commands

1. Select Zmacs and type

   ```
   c-X c-F
   ```

   your-file-server:>*your-login-name*>`best-seller.text`

2. To automatically fill the text you are about to type, type

   ```
   m-X Auto<SPACE>Fill<SPACE>Mode<RETURN>
   ```

3. Type in

   ```
   I was walking down Broadway to the office, just like I do
   every morning. Suddenly, a scream split the air, and I was off
   on the adventure of a lifetime.
   ```

4. Save this buffer (`c-X c-S`).

5. Take a break from your productive writing session. Select a Lisp Listener.

6. On second thought, a great opening paragraph like the one you just wrote has enough potential for *two* novels. Type

   ```
   Copy<SPACE>File<SPACE>best-seller.text<SPACE>hard-boiled
   <RETURN>
   ```

   to make a copy of the file `best-seller.text`, naming the new file `hard-boiled.text`.

7. You are obviously on an inspirational roll. Return to the editor and begin editing your new file by typing

   ```
   c-X c-F
   hard-boiled<RETURN>
   ```

8. Change the last line of text to

   ```
   the greatest adventure in the history of crime.
   ```

9. Switch back to the previous buffer with `c-m-L` or `c-X B <RETURN>`.

10. Change the last line to

    ```
    the longest hardware debugging session
    in the history of computing.
    ```

11. A great writer like you can't worry about keeping buffer names straight; you're an artist. Select a Lisp Listener and save the latest versions of all your edited files. Type:

    ```
    Save<SPACE>File<SPACE>Buffers<SPACE>:Q<SPACE>Yes<RETURN>
    ```

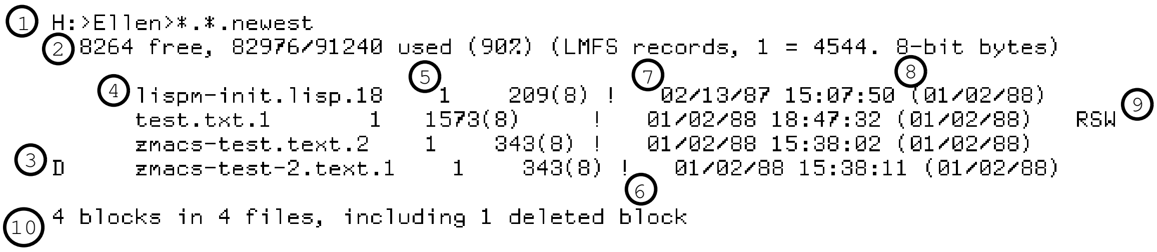
12. Answer `Y` to each prompt. When you type `No` for the value of the *:Query* keyword, *all* modified buffers associated with a file are written to their appropriate files, without your being prompted.

13. The latest versions of your two files, `hard-boiled.text` and `best-seller.text`, are now saved in your directory. `best-seller` is a somewhat presumptuous filename, so type

    ```
    Rename<SPACE>File<SPACE>first-novel<RETURN><RETURN>
    ```

    That's more like it.

Directory Listings Explained



```
H:>Ellen>*.*.newest
 8264 free, 82976/91240 used (90%) (LMFS records, 1 = 4544. 8-bit bytes)

      lispm-init.lisp.18    1    209(8) !   02/13/87 15:07:50 (01/02/88)
      test.txt.1        1   1573(8)    !   01/02/88 18:47:32 (01/02/88)   RSW
      zmacs-test.text.2    1    343(8) !   01/02/88 15:38:02 (01/02/88)
D     zmacs-test-2.text.1    1    343(8) !   01/02/88 15:38:11 (01/02/88)

4 blocks in 4 files, including 1 deleted block
```

Figure 36. Directory Listing

You can get a directory listing by using the Command Processor command Show Directory, the directory editor, *Dired*, or the File System Editor, *FSEdit*. The following list explains what each portion of the directory listing means (Figure 36).

1. The name of the directory.

2. The amount of space left for file storage, in blocks.

3. This column contains a `D` if the file has been deleted but not expunged.

4. The name of the file.

5. The size of the file, in LMFS records, followed by bytes.

6. This column contains an exclamation point if the file has not been *backed up*. Backing the file up means that it is saved on tape, so that if something happens to the disk, it is still possible to retrieve the file from the tape.

7. The date and time the file was created.

8. The date the file was last accessed.

9. The user who created the file (this field is empty when you are the person who created the file, in your directory).

10. The total number of blocks and files in the directory.

Pathnames on Other Machines

Files do not have to reside in the Lisp Machine File System for you to be able to access them. You can get files from any machine that is *networked* (has a data connection) to yours. If the machine is not a Symbolics computer, the pathname has essentially the same components (although many machines do not have automatic version numbers), but they are put together in different forms. Below are a few of the possible host and pathname combinations.

- LMFS

```
SYMBOLICS1:>WTP>cards>solitaire.lisp.10
```

- UNIX

```
UNIX-VAX:/usr/wtp/cards/solitaire.l
```

- VAX/VMS

```
VMS-VAX:sys$usr:[wtp.cards]solitaire.lsp;10
```

Genera understands pathnames on many different systems. If you know how to specify a pathname on a machine which is networked to your Symbolics machine, you can use any of the file operations, specifying the pathname in the format appropriate for that machine and file system.

Documentation References

- See the section "Dictionary of Command Processor Commands".
- See the section "Interpreting Directory Listings in FSEdit".
- See the section "Pathnames on Supported Host File Systems".

More Document Examiner

Introduction

This chapter covers additional features of Document Examiner, enabling you to access greater amounts of information more efficiently.

- "Workbook: Private Documents"

- "Workbook: Additional Viewers"

- "Workbook: Walk-through for Additional Document Examiner Features"
- "Workbook: Other Available Commands/Hardcopying"
- "Workbook: More Document Examiner Documentation References"

Private Documents

Document Examiner allows you to create private documents containing topics that you read often. Each time you enter Document Examiner you need only read in your private document to have access to just the information that you have previously saved.

Private documents are handy for storing information on a particular topic or set of topics so you can have them available in Document Examiner without having to hunt them up anew each time. Private documents work particularly well with additional Document Examiner viewers.

There are four things you can do with a private document:

- Save it, which is creating or updating it.
- Read it, which is reading all the text into a viewer.
- Load it, which is loading all the bookmarks into a viewer, but not the text. Then you can load text selectively by clicking on the bookmarks.
- Hardcopy it.

All four of these functions are available as commands, or from the menu you see by clicking Right on [Read Private Document]. If you read or load a private document, you are prompted for the name of the viewer in which to place the text. If you specify a name that does not exist, a viewer with that name is created.

- **[Read Private Document]** — If you read in the private document, the actual text for all of the topics is read into the viewer. This takes a noticeable amount of time, especially if you read in several large topics.
- **[Load Private Document]** — If you elect to load a private document, the topic names are brought into the Bookmarks pane, but the text is *not* read into the viewer. The titles in the bookmarks section are covered with a gray stripe to indicate that the topics have not yet been read into the viewer. Read them in as you need them. This saves time.
- **[Save Private Document]** — You create a private document by saving all the bookmarks in the current viewer. You will probably want to create a new viewer and make sure you have the right set of bookmarks in it. You can create a new viewer by clicking on [Select Viewer] and naming a new viewer.

- **[Hardcopy Private Document]** — This makes a hardcopy of the entire private document on the default printer for your system.

Additional Viewers

We talked a little about additional viewers when we talked about private documents, but there are many other uses of extra viewers, not the least of which is the help they give you in keeping track of the text you're interested in.

You can have one viewer with nothing but Zmacs topics in it, another with CP commands, and a third with all the Lisp functions having to do with time. You can have as many viewers as you want. Extra viewers and private documents fit together nicely.

The first viewer you see in Document Examiner is the Default viewer, which is created the first time you select Document Examiner. There is another viewer called the Background viewer, where you'll find all the documentation topics you've viewed using the Show Documentation command in the Lisp Listener or the editor.

There are three things you can do with a viewer:

- Select it, that is, go to it, or create it and go to it.

- Remove it, which is to get rid of it.

- Hardcopy it.

All three of these functions are available as commands or from the menu you see when you click Right on [Select Viewer].

- **[Select Viewer]** — To make a new viewer (or choose a viewer that already exists) you click Left on [Select Viewer] (or click Right and then select it from the menu), then enter the viewer's name. It's a good idea to give short, descriptive names to each viewer, thereby simplifying viewer selection. If you can't remember the name of the viewer you want, press `c-?` at the viewer-name prompt to get a mouse-sensitive list of available viewers.

- **[Remove Viewer]** — Prompts you to type the name of the viewer you wish to discard.

- **[Hardcopy Viewer]** — Prompts you for the name of the viewer you wish to hardcopy. Everything in the viewer is hardcopied.

Walk-through for Additional Document Examiner Features

1. Select Document Examiner if you haven't already. Refer to the earlier chapter on Document Examiner if necessary. See the section "Workbook: Document

Examiner". If you don't already have text in the viewer, read a topic (any topic) into the Default Viewer by clicking Left on a candidate. (**Note:** If the Current Candidates pane contains the names of books in the documentation set, don't click Left! It will take too long to read the entire book in. Instead, use the Show Documentation command to read in a briefer topic.)

2. Create a new viewer. Click Left on [Select Viewer] and type

   ```
   new-viewer
   ```

   and press `RETURN`.

3. Notice that all panes are cleared, the table of contents for the documentation set is displayed in the Current Candidates pane, and the label `new-viewer` appears at the lower left corner of the Viewer pane. In other words, it is a fresh Document Examiner viewer.

4. Begin creating a private document by reading a topic into the Viewer. Click on [Show Documentation] and type

   ```
   A<SPACE>Sam<SPACE>I<COMPLETE><RETURN>
   ```

   The topic `A Sample Init File` appears in the Bookmarks pane, along with the appropriate text in the Viewer pane.

5. Include another bookmark, this time regarding *spelling*. Click Left on [Show Candidates] and type

   ```
   spelling
   ```

6. When the topics appear in the Current Candidates pane, put the `Spell Word` candidate in the Bookmarks pane. Do this by clicking Right on the topic name and selecting [Bookmark] from the menu that appears.

7. Save the two bookmarks as your private document. Click Right on [Read Private Document] and select [Save Private Document] from the menu and specify a pathname in your directory for the private document file. Then write the file out by pressing `RETURN`. You now have a private document file in your directory.

8. Go back to the Default Viewer. Click Left on [Select Viewer] and press `c-?`. A list of viewers appears at the top of the current viewer. Move the mouse to `Default Viewer` and click Left. The Default Viewer still contains the same text as before.

9. Read in your private document. Click Left on [Read Private Document] and type the name you previously entered, followed by `RETURN` to read in your private document. Of course, if it is offered as the default, all you have to do is press `RETURN`.

10. You must also specify which viewer the text is read into. Type

```
new-viewer
```

and press `RETURN`.

11. The text should soon appear in the Viewer pane that you created. Anytime you want to read documentation on the topics in your private document, you can simply read the private document file into a viewer, and there you are.

12. Finally, just for kicks, click on [Select Viewer] and select the Background Viewer. If you have displayed any documentation using the Show Documentation command in the Lisp Listener or the editor, you'll find it here.

Other Available Commands/Hardcopying

The Commands pane accepts many more commands than those options listed on the right of the pane. You can see a summary of available commands by pressing the `HELP` key. You can see detailed Document Examiner information by clicking Middle on [Help]. (You can also type `Show Candidates document examiner`.)

Some of the more useful Document Examiner commands include:

- Beginning of Topic
- End of Topic
- Hardcopy Private Document
- Hardcopy Documentation
- Hardcopy Viewer

Beginning of Topic and **End of Topic** — These commands display the first and last full screen of text, respectively, of the current topic in the Viewer (the current topic is indicated by an arrow in the Bookmarks pane). You can also move around by using the scroll bar, but you should understand that the scroll bar is based on *all* the documentation in the viewer, not just the topic you're looking at.

Hardcopy commands — These commands, respectively, print the documentation of a private document's topics, a specific topic, or the topics in a particular Viewer. After entering one of these commands, you are prompted for the pathname of the private document, a specific topic, or the name of a Viewer.

Before using *any* Hardcopy command:

- Make sure a printer is available to your Symbolics computer.
- Check the length of the document you wish to print (avoid hardcopying long documents — use the printed documentation volumes).

Documentation References

- ***Private Documents***
  See the section "Creating a Private Document".
- ***Adding Viewers***
  See the section "Organizing Topics in Document Examiner".
- ***Document Examiner Command Summary***
  See the section "Document Examiner Window".

Advanced Topics

Introduction

"Workbook: Advanced Topics" discusses topics that not every user of Genera needs to know. For instance, a portion of the chapter on Zmacs describes formatting text, a topic that is not applicable to everyone's needs. The way to progress through this section is to choose the topics which interest you.

"Workbook: Additional Zmacs"

Introduces numeric arguments for Zmacs commands, which cause them to do something some specified number of times. It also explains more buffer operations such as appending and comparing. For those of you who write documents, there is a brief introduction to formatting text. We also talk about the Speller.

"Workbook: Additional Files"

Details two more ways to manipulate files and directories. *Dired*, short for the Directory Editor, is a program which you access in Zmacs. Dired allows you to treat directory listings as text files. *FSEdit*, short for the File System Editor, is associated with its own window and is menu-driven. Both of these programs allow you to do the same kind of file manipulation that is possible with the Command Processor.

"Workbook: Additional Window Features"

Reinforces some basic conceptual knowledge about windows and how they work. It also covers making modifications to the Command Processor and making your own windows with the System Menu, as well as additional methods of selecting activities.

"Workbook: The Namespace"

Explains how to add a user to the namespace database. This is only the very briefest of introductions to this software.

"Workbook: Overview of the Machine"
Describes the basic anatomy of a Symbolics computer. It also explains the Front End Processor and its role in booting the machine.

Additional Zmacs

Introduction

This chapter introduces some additional features of Zmacs. These features enhance your use of Zmacs, but are not necessarily needed by every user.

- "Workbook: Numeric Arguments"
- "Workbook: Additional File and Buffer Operations"
- "Workbook: Splitting the Screen"
- "Workbook: Formatting Text"
- "Workbook: Zmacs Speller"
- "Workbook: Additional Zmacs Documentation References"

Numeric Arguments

Many Zmacs commands take *numeric arguments*. These are numbers you can specify before you give a Zmacs command that affect the result of that command. In general, numeric arguments specify how many times you want a particular action to happen. For example, pressing c-3-D is the same as pressing c-D three times. In each case, you delete three characters. Often, however, numeric arguments change the action of a command in some other way, such as sending output to a printer instead of the screen, for example.

To specify a numeric argument, hold down any of the modifier keys (c-, m-, s-, h-) and type the number. The numeric argument appears in the echo area if you do not type the command immediately.

Another way of specifying a numeric argument is by using c-U. Typing c-U is like typing a numeric argument of 4. If you type c-U more than once, your numeric argument is 4 to the *n*th power, where *n* is the number of times you type c-U.

Numeric arguments can be negative. Just type a minus sign before you type the number. For instance, to specify a numeric argument of -32, you would type c-- c-3 c-2. You don't really have to press the CONTROL key three times; just press CONTROL and hold it down while you press the -, the 3, and the 2.

Numeric arguments most often specify how many times to execute a command. For example, if you type `m-2 m-B` you move back two words. Negative arguments usually mean to move or act in a way opposite to normal. For example, typing `c-- c-3 c-N` moves you *up* three lines, even though `c-N` is the command to move down a line. Numeric arguments are also good when you want many of the same character to be inserted in your buffer. To get a line of sixty asterisks, you could just type `c-6 c-0 *` and sixty asterisks would be inserted at the current cursor position, just as if you had typed all sixty of them, one by one.

Additional File and Buffer Operations

Renaming

`m-X` Rename Buffer Allows you to rename a buffer. The new name can be any string. Using this command removes any association with a file that the buffer might have already had.

Saving and Killing

`m-X` Save File Buffers

Asks you about saving each buffer associated with a file.

`m-X` Kill Some Buffers

Tells you about the status (modified or not, and so on) of each buffer and asks whether or not to delete it. If you say to delete a buffer that has been modified, it asks you if you want to save the buffer first.

`m-X` Kill Or Save Buffers

Gives you a menu with choices for each buffer of [Save], [Kill], [Unmodify], and [Hardcopy]. Choosing [Save] writes the buffer out to its associated file. [Kill] kills the buffer but not its associated file. Even if it has been modified, you aren't prompted to save it. [Unmodify] does not remove the modifications to a buffer, but marks the buffer as unmodified. [Hardcopy] prints the file.

Appending and Prepending

`c-X A` Append to Buffer. Prompts for a buffer name and adds the marked region to the end of that buffer.

`m-X` Append To FilePrompts for a file name and adds the marked region to the end of that file.

`m-X` Prepend To File

Prompts for a file name and adds the marked region to the beginning of that file.

Copying

m-X Copy File — Prompts for the name of the file to copy and for a file name to which to copy it.

Comparing

m-X Source Compare

Compares two files or buffers. You are prompted for the type (F or B) and name of each, and the results of the comparison are displayed over your buffer. The results are also put into a buffer called *Source-Compare-*n**, where *n* is the number of times you've done a source compare.

m-X Source Compare Merge

Compares two files or buffers. You are prompted for the type (F or B) and name of each. Unlike m-X Source Compare, this command produces a new version that reconciles the differences. When a difference is found, you are prompted twice.

The first time, you enter an option specifying what to do about the difference. Entering 1 here keeps the text from the first version, 2 keeps the text from the second version, and SPACE keeps the text from both versions.

The second time, you are prompted to confirm the change you made. Pressing SPACE makes the change; pressing RUBOUT cancels the change.

Creating

c-X B — You can create a new buffer by giving a new buffer name and pressing c-RETURN.

```
c-X B<name>c-<RETURN>
```

If you want to write the buffer out, check carefully to see what defaults c-X c-S gives you, or use c-X c-W to specify exactly what pathname you want for the contents of the buffer.

Splitting the Screen

In Zmacs, you can have many buffers, but so far you have only been able to see one of them at a time. There are several commands available for splitting the text window so that you see two buffers (or more) at once. The most general of these is

```
m-X Split Screen
```

This command gives you a menu of all your current buffers, and options for creating new buffers and reading in files, and three action options: [Do It], [Undo], and [Abort]. As you click on the various buffers, a small window shows you what the finished split-screen buffer will look like. If you click on [Undo], the last thing you added to the split-screen goes away. If you click on [Do It], the configuration shown becomes your current buffer configuration. If you click on [Abort], the menu goes away and nothing happens.

Note: If you click on [Abort] or [Undo] after creating a new buffer or reading in a file, that new buffer still exists, even though you have canceled the split-screen or split-screen element.

Formatting Text

This section is for people who want to write "pretty" manuscripts. There are many more formatting commands besides those discussed here. Formatting text makes it easier to read. For more information: See the section "Formatting Text in Zmacs".

How to do it

Two steps are required for the production of formatted text.

- Write the text in an editor buffer with formatting instructions embedded in the text.

- Display the text on screen or on a printer using the appropriate extended command.

Formatting Instructions

Formatting instructions all begin with an @. There are two possibilities for the next part of the instruction.

Short Text

Let's say that you want to change the font of a word or short phrase. You would use this form:

> @i(word or short phrase)
> @b(short phrase or word)

which would give you, after the appropriate extended command:

> *word or short phrase*
> **short phrase or word**

- @i indicates that you want *italics*, while @b is for **bold** type.

- The parentheses () enclose the affected **text**. You can use other symbols to indicate the limits of the text, such as [] or <>.

Long Text

Suppose you wish to use a format command on a large amount of text. You can use `@begin` and `@end` as delimiters. Look carefully at this example, and notice that the syntax has changed.

```
@begin(b)
I stepped back from the creaking floorboard, but it was too
late.  "Hey, what's that?" Goon #1 said.  Before Goon #2 could
answer, I popped him in the beezer.  Goon #1 whirled and came
toward me, looking for more of the same.  So I sapped him.
@end(b)
```

All the text between @begin(b) and @end(b) is displayed in bold face, because of the (b).

How to See the Results of Your Efforts

Use the extended command `m-X` Format Buffer to see the formatted text on the screen. The above text would look like this:

> **I stepped back from the creaking floorboard, but it was too late. "Hey, what's that?" Goon #1 said. Before Goon #2 could answer, I popped him in the beezer. Goon #1 whirled and came toward me, looking for more of the same. So I sapped him.**

If you have an LGP2 or LGP3 printer, you can get a hardcopy by using

```
c-U m-X Format Buffer
```

and specifying the appropriate hardcopy device.

Zmacs Speller

The Zmacs Speller is a small, simple set of tools to help you spell words right. The Speller can examine a single word, a Zmacs region or buffer, or a file or group of files, for spelling errors. When it encounters a word not listed in one of its dictionary files, it alerts you with a message and a menu of possible choices. A large dictionary of common words is included as part of the Zmacs editor. You can add other dictionaries of special spellings, as used by individuals or groups of users on your system, to the list.

To the Speller, any word not in the dictionaries is misspelled by definition. Not all words or alternate spellings are in the basic dictionary.

For the Speller, a "word" is a sequence of characters that are either letters or apostrophes. This is not the same definition of "word" that is used by Zmacs "word" commands like `m-F`.

When the Speller encounters a word not in the dictionary, a warning appears in the minibuffer:

> "*the-unknown-word*" `is unknown and possibly misspelled.`

and a menu gives you a series of possible choices:

[Prompt] Lets you type in the correct spelling.

[Accept once] Lets you accept this spelling of the word this one time.

[Accept] Lets you accept this spelling for the duration of this command.

[*Possible alternatives***]**
Lets you substitute one of the alternatives in the dictionary for the misspelled word.

Spelling

m-$ Checks the spelling of the current word. Place the cursor either on the word you want checked or just to the right of the word. If a region exists, m-$ does Spell Region.

m-X Spell Word Prompts for a word and checks it. In the minibuffer you see whether or not the word is spelled correctly; words that are similar to the misspelled word are listed. Use this to check the spelling of words that aren't in your buffer.

m-X Spell Region Checks the spelling of a marked region.

m-X Spell Buffer Checks the spelling of the current buffer.

Documentation References

- See the section "Numeric Arguments".
- See the section "Comparing Files and Buffers in Zmacs".
- See the section "Zmacs Window Commands".
- See the section "Formatting Text in Zmacs".
- See the section "Zmacs Speller".

Additional Files

Introduction

This chapter introduces two ways of handling files and directories in addition to Command Processor commands. These are the Directory Editor and the File System Editor. This chapter also describes properties of files and directories.

- "Workbook: The Directory Editor - Dired"

- "Workbook: Walk-through for Dired"

- "Workbook: The File System Editor - FSEdit"
- "Workbook: Walk-through for FSEdit"
- "Workbook: File Properties"
- "Workbook: Directory Properties"
- "Workbook: Additional Files Documentation References"

The Directory Editor - Dired

Zmacs contains a tool for editing directories, called *Dired*. To get to Dired, you can type

```
Edit Directory <pathname><RETURN>
```

in the Command Processor.

There are also two Zmacs commands for editing directories:

- `m-X Dired<RETURN>`*<pathname>*`<RETURN>`
- `c-X D`

`c-X D` puts you in Dired for the directory associated with the current buffer.

Help

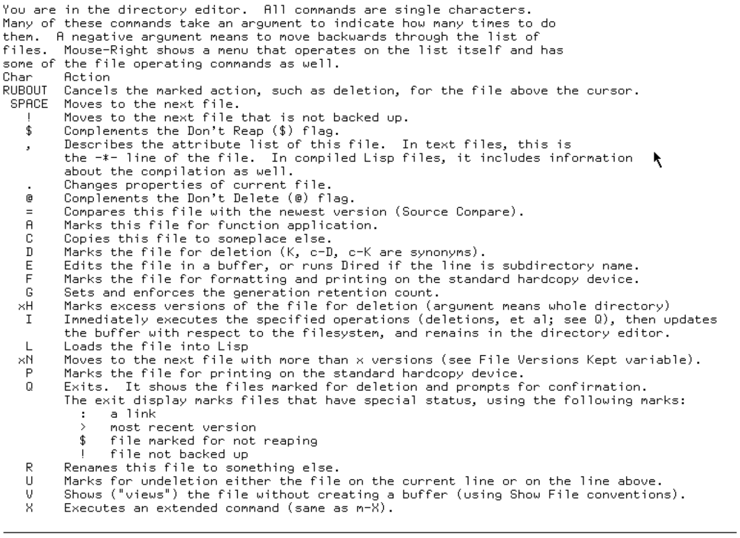
Once you are in Dired, you can get a list of commands by pressing the `HELP` key or the `?` key (Figure !).

When you are prompted for an argument in the minibuffer, you can also press the `HELP` key, to make a summary of the operation in progress appear on the screen. This tells you what you are being prompted for and what command you are executing.

Some commands take effect only when you press `Q` or `I` to exit Dired. When you press `Q` or `I`, a list of the files to be deleted, undeleted, and printed appears, followed by the query:

```
OK? (Y, E, N, Q, or X)
```

- `Y` causes files to be marked as deleted or undeleted and prints files that are marked for printing.
- `E` deletes or undelete the files and then expunges the directory and prints the files.



```
You are in the directory editor.  All commands are single characters.
Many of these commands take an argument to indicate how many times to do
them.  A negative argument means to move backwards through the list of
files.  Mouse-Right shows a menu that operates on the list itself and has
some of the file operating commands as well.
Char    Action
RUBOUT  Cancels the marked action, such as deletion, for the file above the cursor.
 SPACE  Moves to the next file.
   !    Moves to the next file that is not backed up.
   $    Complements the Don't Reap ($) flag.
   ,    Describes the attribute list of this file.  In text files, this is
        the -*- line of the file.  In compiled Lisp files, it includes information
        about the compilation as well.
   .    Changes properties of current file.
   @    Complements the Don't Delete (@) flag.
   =    Compares this file with the newest version (Source Compare).
   A    Marks this file for function application.
   C    Copies this file to someplace else.
   D    Marks the file for deletion (K, c-D, c-K are synonyms).
   E    Edits the file in a buffer, or runs Dired if the line is subdirectory name.
   F    Marks the file for formatting and printing on the standard hardcopy device.
   G    Sets and enforces the generation retention count.
  xH    Marks excess versions of the file for deletion (argument means whole directory)
   I    Immediately executes the specified operations (deletions, et al; see Q), then updates
        the buffer with respect to the filesystem, and remains in the directory editor.
   L    Loads the file into Lisp
  xN    Moves to the next file with more than x versions (see File Versions Kept variable).
   P    Marks the file for printing on the standard hardcopy device.
   Q    Exits.  It shows the files marked for deletion and prompts for confirmation.
        The exit display marks files that have special status, using the following marks:
          :   a link
          >   most recent version
          $   file marked for not reaping
          !   file not backed up
   R    Renames this file to something else.
   U    Marks for undeletion either the file on the current line or on the line above.
   V    Shows ("views") the file without creating a buffer (using Show File conventions).
   X    Executes an extended command (same as m-X).
```

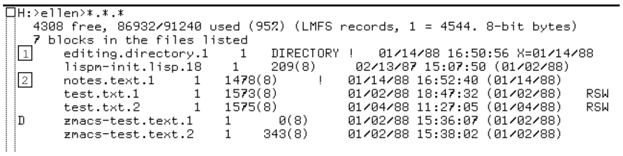
Figure 37. HELP Display in Dired

- N returns you to Dired without doing any deletions, undeletions, or printing operations.

- Q or X exits Dired without deleting, undeleting, or printing any files.

Dired Commands

The following commands should be typed with the cursor on the same line as the file or directory on which you wish to operate. Move the cursor through the buffer with the mouse or with c-P and c-N, just as you would in the regular editor. Additionally, in a Dired buffer, SPACE moves the cursor down one line and RUBOUT moves it up one line.



```
□H:>ellen>*.*.*
   4308 free, 86932/91240 used (95%) (LMFS records, 1 = 4544. 8-bit bytes)
   7 blocks in the files listed
 1    editing.directory.1    1   DIRECTORY !   01/14/88 16:50:56 X=01/14/88
      lispm-init.lisp.18    1    209(8)     02/13/87 15:07:50 (01/02/88)
 2    notes.text.1      1   1478(8)     !   01/14/88 16:52:40 (01/14/88)
      test.txt.1        1   1573(8)         01/02/88 18:47:32 (01/02/88)   RSW
      test.txt.2        1   1575(8)         01/04/88 11:27:05 (01/04/88)   RSW
 D    zmacs-test.text.1    1      0(8)      01/02/88 15:36:07 (01/02/88)
      zmacs-test.text.2    1    343(8)      01/02/88 15:38:02 (01/02/88)
```

Figure 38. Dired Listing

If the cursor were on line 1, all Dired commands would refer to the directory editing> (Figure 38).
If the cursor were on line 2, all Dired commands would refer to the file notes.text.1 (Figure 38).

`D` In Genera, which supports soft deletion, marks the file to be deleted.

`U` Unmarks the file for deletion, if the file is marked for deletion.

`E` Brings the file on the current line into an editor buffer and selects that buffer. If the cursor is on a line with a directory pathname, this command creates another Dired buffer displaying that directory's contents.

`L` Loads the file into your world. This means that the code in the file is executed as if you had typed it to the Lisp Listener, but a copy of the file is not put in an editor buffer.

`R`*<to-pathname>*`<RETURN>`

Renames the current file from the pathname on that line to the name you specify in *<to-pathname>*. You can also change the directory in which the file resides in this way, by specifying a new directory in the *<to-pathname>*. After executing this command, only one copy of this file exists, that named *<to-pathname>*.

`C`*<to-pathname>*`<RETURN>`

Copies the file to *<to-pathname>*. Two copies of the file exist when this command has been executed, one in its original location, and a new copy in *<to-pathname>*.

`F` Marks the file for formatting; the file should contain the basic Zmacs text formatting commands. (For more information: See the section "Workbook: Formatting Text".) When you exit Dired, you are asked to confirm that the marked files should be formatted on your local hardcopy device.

`P` Marks the file for printing. When you exit Dired, you are asked to confirm that the marked files should be printed on your local hardcopy device.

`V` Lets you view the file or directory specified on the line containing the cursor. You cannot do anything but look at the file or directory this way. If you wish to edit the file or directory, use `E`.

`Q E` or `I E` Upon exiting Dired with `Q` or `I`, typing `E` to the given prompt expunges the directory, provided that you have made changes to the deleted or undeleted status of any of the files in the directory.

Note: Once you have created a Dired buffer, it remains in Zmacs and can be selected just like any other buffer. However, the information in the Dired buffer is not automatically updated. To update the information, once you have selected the buffer, type

```
    m-X Revert Buffer<RETURN>
```

You are prompted with

```
Buffer to revert: [default *Dired* HOST:>directory>*.*.*]:
```

Just press RETURN to take the default.

Walk-through for Dired

1. Press SELECT E to enter the editor.

2. Type

   ```
   m-X Dired<RETURN>
   ```

3. When you are prompted for the directory to edit, type

   your-file-server:>*.*.*<RETURN>

   A listing of the files and directories on your file server appears.

4. Press HELP. Read the help information.

5. Press SPACE to make the help information go away.

6. Using Zmacs cursor motion commands, move the cursor to the line that says

   your-login-name.directory

   (The line may not be visible in the window, in which case you should scroll the screen until it becomes visible.)

7. Press E. This should put you in Dired editing your home directory.

8. Move the cursor to the line that says

   ```
   zmacs-test-2.text
   ```

9. Press D. Note that a D immediately appears beside the file name.

10. Type SELECT L.

11. Type

    Show Directory *your-file-server*:>*your-login-name*>*.*.*

    Notice that the file zmacs-test-2.text is not marked as deleted. This is because Dired does not mark files as deleted until you exit.

12. Select the editor again.

13. Press Q.

14. Respond Y to the prompt. Now the file is deleted.

15. Reselect the Dired buffer.

16. Move the cursor to the line that says

    `zmacs-text-2.text`

    again.

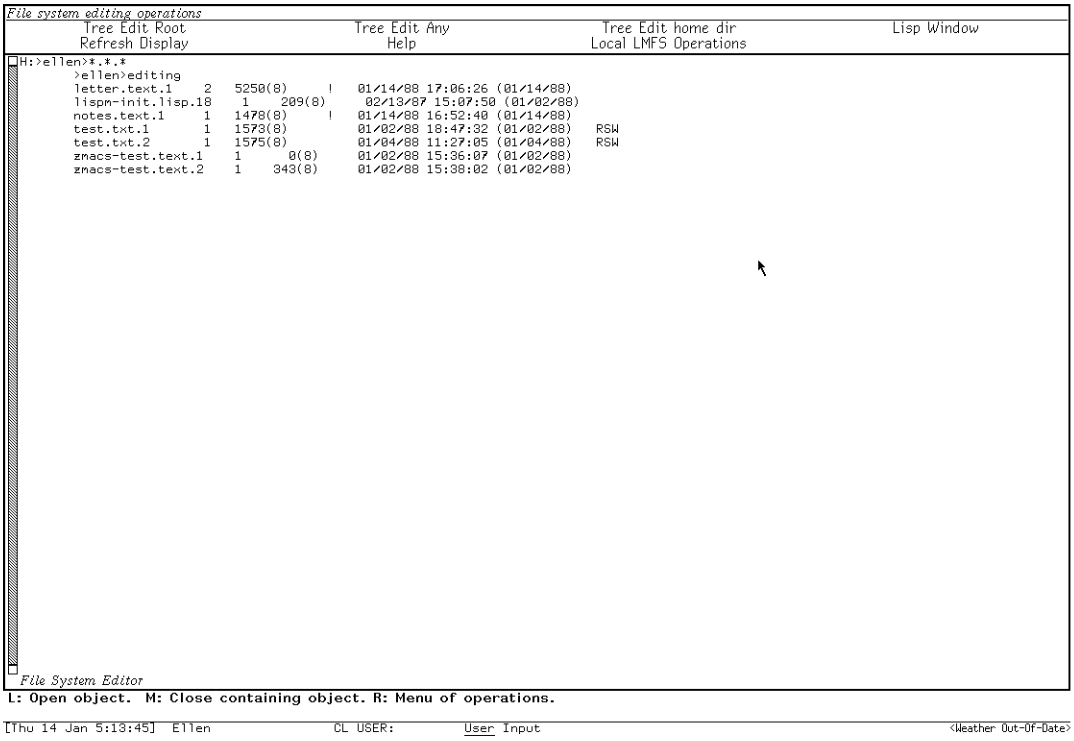
17. Press `U`.

18. Press `Q`.

19. Respond `Y` to the prompt. The file has been undeleted.

20. Try out the other Dired commands. See which ones take effect immediately and which ones take effect only when you exit Dired.

The File System Editor - FSEdit



```
H:>ellen>*.*.*
     >ellen>editing
     letter.text.1     2   5250(8)     !   01/14/88 17:06:26 (01/14/88)
     lispm-init.lisp.18  1    209(8)      02/13/87 15:07:50 (01/02/88)
     notes.text.1      1   1478(8)     !   01/14/88 16:52:40 (01/14/88)
     test.txt.1        1   1573(8)         01/02/88 18:47:32 (01/02/88)   RSW
     test.txt.2        1   1575(8)         01/04/88 11:27:05 (01/04/88)   RSW
     zmacs-test.text.1   1      0(8)       01/02/88 15:36:07 (01/02/88)
     zmacs-test.text.2   1    343(8)       01/02/88 15:38:02 (01/02/88)
```

```
L: Open object.  M: Close containing object. R: Menu of operations.
[Thu 14 Jan 5:13:45]  Ellen          CL USER:        User Input                <Weather Out-Of-Date>
```

Figure 39. The File System Editor

FSEdit is a program that helps you examine directories and the files they contain.

You can access FSEdit by pressing `SELECT F` or clicking on [File System] in the System menu. Notice that there is a menu at the top of the screen and that the `Command:` prompt appears in the window below the menu (Figure 39). Although you can type Command Processor commands to an FSEdit window, you generally do that from a Lisp Listener.

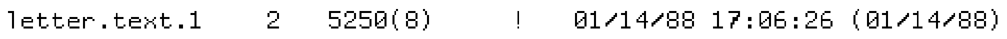
Let's start learning how to use FSEdit by examining your directory. Move the mouse over [Tree Edit home dir] and read the mouse documentation line. Here's the difference between the clicks:

- Left - Edit your directory on the file-server machine.

- Middle - Edit your directory on the machine you are working on. You get a notification if you do not have a directory on this machine.

- Right - You are prompted for a machine name; after you enter it, you can edit your directory on that machine.

You click Left in order to get your home directory on the file-server. On each line you see the name of a file, along with other information. A typical file entry is shown in Figure !.



```
letter.text.1    2   5250(8)     !   01/14/88 17:06:26 (01/14/88)
```

Figure 40. A Typical File Entry

The information given next to the file name is number of blocks (amount of storage), number of bytes and byte size, creation date and time, and date of last access. The exclamation point indicates that the file has not been backed up to tape.

Things You Can Do With Files

If you are just beginning on the machine, you probably do not have many files. You should at least have a lispm-init.lisp file, as well as several others you created when learning Zmacs. Move the mouse cursor over one of the file names and read the mouse documentation line. The clicks have the following meanings:

- Left - *Open / Close* Clicking Left displays the file on the screen.

- Middle - *Delete / Undelete* Clicking Middle marks the file for deletion if it is undeleted, or undeletes it if it is marked for deletion.

- Right - Clicking Right brings up a menu; it's useful to learn what some of the choices are.

File Menu Operations

[Delete] Marks a file for deletion. When you do this, a `D` appears to the left of the file name. If you bring up the menu again on this file you see that one choice is now [Undelete]. These work the same as the Command Processor commands Delete File and Undelete File.

Although you cannot access deleted files, you can *always* get them back until you *expunge* your directory. This procedure is explained later in this section. Be aware that, if disk space becomes scarce, your system administrator might expunge the entire file system, making your deleted files disappear. Therefore, do not mark a file for deletion unless you're certain that you no longer need it.

[Show] Lets you quickly see the contents of a file without having to read it into the editor. This works the same as the command Show File.

[Rename] Lets you rename a file and prompts you for a new file name. This works the same as Rename File.

[Edit File] Retrieves the file, puts the file into a new Zmacs buffer, and selects the Zmacs window. It works the same as Edit File.

[Hardcopy] Hardcopies the file on your default printer. This works the same as the Hardcopy File command.

[Load] Loads a binary file. If the file is not a binary file, it loads a compiled version of the source code file. This works the same as Load File.

Things You Can Do With Directories

After some time, you might become responsible for several directories on your file-server, but practice using these commands on your own directory for the present. Click Left on [Tree Edit home dir], then put the mouse on your directory name to see the box around it. Now read the mouse documentation line.

- Left - *Open / Close* opens the directory and lists the files that it contains. Clicking on the directory name a second time removes the listing.

- Middle - *Delete / Undelete* marks the directory for deletion, if it is undeleted, or undeletes it if it is marked for deletion.

- Right - Clicking Right brings up a menu. It's useful to learn what some of the choices are.

Directory Menu Operations

[Delete] Marks this directory as deleted. Note: Before you can delete a directory, all the files in the directory must be both deleted and expunged. If the directory is already deleted, the option is [Undelete].

[Open] Opens the directory and lists the files it contains. This is the same as clicking Left on the directory.

[Expunge] Removes the files marked as deleted in this directory. This is the same as Expunge Directory.

[Create Inferior Directory]
Creates a new subdirectory that is inferior to this directory. This is the similar to Create Directory.

[Rename] Renames this directory, prompting you for a new directory name.

Walk-through for FSEdit

1. Press `SELECT F` to select the File System Editor.

2. Move the mouse over [Tree Edit Root] and click Left. A display of the directories on your file-server appears.

3. Move the mouse over the line that says

   >*your-login-name*

   (If this line is not visible, move the mouse to the left edge of the screen and scroll the display until it is.)

4. Click Right to see the menu of directory operations.

5. Click Left on [Open]. The files in your directory should appear. If you have any subdirectories, they are listed at the beginning of the display under your home directory.

6. Move the mouse over

   ```
   zmacs-test.text.1
   ```

7. Click Right to see the menu of file operations.

8. Click Left on [Show].

9. Press `SPACE` to make the contents of the file disappear from the screen.

10. Click Left while still on the file.

11. Notice that the display of your directory closes up.

12. Try other mouse clicks and menu items on both files and directories.

File Properties

Files have several properties that you can see or change. Some of these are: Generation (version) Retention Count, Author, Creation, Modification, Reference Dates, and Protection Flags. Most file properties are not important to beginning or even moderately advanced users, but two of them might be useful immediately.

The *Generation Retention Count* specifies how many versions of a file to keep in a directory. The *n* most current files are kept, with the rest marked for deletion. This is generally useful, as most files go through several versions, and the old versions would just be taking up space.

The *Don't Delete* protection flag is generally set to *No*, but it can be changed to *Yes* if you wish to protect an important file from being accidentally deleted. (If the flag is set to *Yes,* and someone tries to delete the file, an error is signalled.) This flag can later be set back to *No* when you wish to dispose of the file.

Editing properties

The commands listed below cause a menu to appear. You can click on a specific field in this menu to change its value. To change the Generation Retention Count, click on the value following the colon, then type in an integer. This is the number of versions of a file left undeleted. To change the Don't Delete flag, click on either *Yes* or *No*, depending on which value you want it to have. When done editing, click on `Done` to update the file's properties or `Abort` to abort.

There are several ways to get the File Properties editing menu:

- In Zmacs, use the extended command `m-X` Change File Properties.
- In Dired, press the **.** (period) key while the cursor is next to the file whose properties you wish to edit.
- In FSEdit, click Right on the pathname of a file, then click on the [Edit Properties] menu item.

Showing properties

These commands just display a list of properties and their values:

Zmacs:

```
m-X Show File Properties<RETURN><pathname><RETURN>
```

FSEdit:

```
[Show Properties]
```

Directory Properties

Directories also have several properties that you can see or change. Most directory properties are not important to beginning or even moderately advanced users, but three of them might be useful immediately.

The *Auto Expunge Interval* specifies how often to expunge the directory automatically. This defaults to never, so that your deleted files don't vanish when you don't expect them to. However, it can be set to any time period (specified as a number and then a unit such as day, hour, or week); the system then automatically expunges the directory every two days, six hours, one week, or whenever you have specified.

The *Default Generation Retention Count* specifies how many versions of a file to keep in a directory. When a new file is created in the directory, the generation retention count of that file is set to this default value. The *n* most current files are kept, with the rest marked for deletion. This is generally useful, as most files go through several versions, and the old versions would just be taking up space.

The *Don't Delete* protection flag is generally set to *No*, but it can be changed to *Yes* if you wish to protect an important directory from being accidentally deleted. (If the flag is set to *Yes,* and someone tries to delete the directory, an error is signalled.) This flag can later be set back to *No* when you wish to get rid of the directory.

Editing properties

All these commands cause a menu to appear. You can click on fields in this menu to change the values there. To change the Auto Expunge Interval and Default Generation Retention Count, click on the value following the colon, then type in a value. To change the Don't Delete flag, click on either *Yes* or *No*, depending on which value you want it to have. For the commands that require a <*pathname*> as an argument, specify the extension as *.directory*. When done editing, click on `Done` to update the directory's properties or `Abort` to abort.

There are several ways to get the File Properties editing menu:

- In Zmacs, use the extended command `m-X` Change File Properties.
- In Dired, press the **.** (period) key while the cursor is next to the directory whose properties you wish to edit.
- In FSEdit, click Right on the pathname of a directory, then click on the [Edit Properties] menu item.

Showing properties

To show directory properties, use the same commands you use to show file properties. The only difference is that you specify the *.directory* extension in pathnames to show the properties of a directory.

Zmacs:

```
m-X Show File Properties<RETURN><pathname><RETURN>
```

FSEdit:

```
[Show Properties]
```

Documentation References

- See the section "Dired Mode in Zmacs".
- See the section "Using FSEdit".
- See the section "File Properties".
- See the section "Zmacs File Manipulation Commands".
- See the section "Accessing Directories".

Additional Window Features

Introduction

This chapter introduces you to some of the basic concepts of the window system and shows you more window features.

- "Workbook: Window System Basic Concepts"
- "Workbook: Programs"
- "Workbook: Selecting Activities Using the System Menu"
- "Workbook: Summary of Activities on the System Menu"
- "Workbook: Walk-through for Selecting Activities Using the System Menu"
- "Workbook: the Select Submenu of the System Menu"
- "Workbook: Selecting Activities Using the Mouse"
- "Workbook: Making Your Own Windows"
- "Workbook: Walk-through for Making Your Own Windows"
- "Workbook: Adjusting the Command Processor"
- "Workbook: Walk-through for Adjusting the Command Processor"
- "Workbook: Flashy Scrolling"
- "Workbook: Trouble-shooting"
- "Workbook: Additional Window Features Documentation References"

Window System Basic Concepts

You now know the things necessary to use the window system. In this chapter, we introduce the underlying concepts of the window system, some additional features of the window system, and some ways to get out of trouble if something goes wrong.

What is the Window System?
The window system is a collection of software that provides an interface between users and the Genera software. All input and output goes through the window system. The window system software controls both Symbolics keyboard and mouse input and both graphic and textual output to the screen.

Why is there a Window System?
Genera provides the window system so that you have as much flexibility as possible in switching among different programs with maximum convenience. By interfacing each system program with the window system, we simplify the work of switching among the programs to merely switching between windows.

What is a window? A *window* is a rectangular place where a program can receive input and send output. Many windows can exist at once, but most of them are not visible at any given time. In this way, many programs can input and output without interfering with each other. Programs never use the whole screen for output; they only use windows. A single window may, however, be large enough to fill the screen. A line of text that is wider than the window either wraps around or truncates (depending on the window), but never extends beyond the edge of the window. It is the window system's job (broadly speaking) to ensure this kind of behavior.

Programs

In this workbook, we do not talk about using windows in your own programs, only about how to use and switch among the existing system programs.

What is a program? This involves a number of other issues.

Multiprocessing Genera is a multiprocessing environment. You have probably used multiprocessing machines before. All time-sharing machines are multiprocessing. Multiprocessing is the ability of the computer to do many operations apparently simultaneously. On a time-sharing computer, all users seem to get their own individual computer. On a Symbolics computer, you are the only user, but you can be doing many different things apparently simultaneously.

Each system program is an independent *process*. A process is like a *job* or *fork* on time-sharing computers. The process of a system program terminates only when you turn off the machine. The editor never goes away, nor do the files in it, unless you specifically delete and expunge them. Just because a window is not visible doesn't mean that it's gone away It is merely covered up. Even when you log out, most of what you have done remains as part of your Lisp World until the machine is powered down or cold booted.

Windows Many system programs have their own window or windows. Many programs use a window called a *frame,* which is a window that has several subwindows, called *panes*.

Activities The combination of a process and a window makes an *activity*. An activity is a program that you can choose to use for a while. After a time, you may choose to use a different combination of a process and a window.

Selecting Activities Using the System Menu

Note: You can bring up the System menu by clicking sh-Right anywhere *except* while the mouse is on a menu.

To select the activity specified by one of the menu items, move the mouse onto that item and click Left on it.

When you click Left on the item, the System menu disappears, and the activity that you specified is selected. The important thing to know about switching activities using the System menu is that it's exactly the same as switching activities using the SELECT key. Use whichever technique seems more convenient.

Summary of Activities on the System Menu

[Lisp] A Lisp Listener. Available on SELECT L.

[Edit] Zmacs. Available on SELECT E.

[Inspect] The Inspector. Available on SELECT I.

[Mail] Zmail. Available on SELECT M.

[Trace] Not really an activity, but simply a menu interface to the **trace** debugging facility. This is useful for programmers.

[Emergency Break] Not really an activity, this provides a mechanism for evaluating Lisp forms without using the window system. Rarely used, it may be useful for programmers.

[Frame-Up] The Layout Designer. This allows you to design *constraint frames* for programs. Frames are windows that have subwindows, such as the Zmacs window. Available on SELECT T.

[Namespace Editor] The Namespace Editor. This allows you to modify the namespace database to add or change parameters about users, hosts, or printers. Not available on a SELECT key.

[Hardcopy] Not really an activity, this is simply a menu interface to the file-printing facility.

[File System] The File System Maintenance window. Available on SELECT F.

[Font Edit] The Font Editor. This allows you to create and modify your own fonts. Not available on a SELECT key.

[Document Examiner]
Document Examiner. Also available on SELECT D.

Walk-through for Selecting Activities Using the System Menu

1. Press SELECT L. You should be in a Lisp Listener now.

2. Hold down one of the SHIFT keys and click Right. This should make the System menu appear, right around where the mouse cursor was. Let go of the SHIFT key.

3. The right-hand column in the System menu lists activities that you can select. Clicking Left on any of these selects that activity. Move the mouse slowly over each item on the menu and notice that the mouse documentation lines changes to indicate what clicking Left on that item would do.

4. Click left on [Edit].

5. You should now be in the editor. Notice that what you were doing in the editor walk-through is still there.

6. Bring up the System menu. Don't worry if the editor menu appears instead of the System menu (this happens if you do not use sh-Right or do not click Right twice fast enough). Just move the mouse off it to make it disappear, and try again.

7. Click Left on [Lisp]. You should now be back in the Lisp Listener.

8. Use the right-hand column in the System menu to switch back and forth between activities. If you click on [Emergency Break], you'll have to press RESUME before you can do anything else.

The Select Submenu of the System Menu

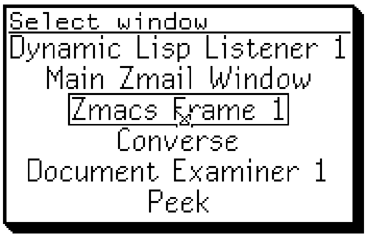


Figure 41. The Select Submenu

Another way to select an activity is via the [Select] item in the left-hand column of the System menu (Figure 41). If you click on this item, the System menu disappears and another menu pops up. This menu has as its items all the existing, selectable windows (more or less). This isn't a very commonly used mechanism, since you can select all the system programs either from the right-hand column of the System menu or with the SELECT key. However, if you have created windows of your own that are not available with either of those two methods, [Select] can be very useful.

Selecting Activities Using the Mouse

The last way to select an activity can only be used when you can see at least part of the window that the activity uses. If you can't see the window at all, then you must use one of the other mechanisms. If you click Left on a window that is partially visible, it comes up to the top of the pile (become fully visible) and becomes the selected activity. This technique also works for selecting one of two completely visible windows.

Making Your Own Windows

It is possible (in fact, easy) in Genera to create your own windows, as you have already seen. You can have many windows of the same type as well as of different types, and you can easily divide the screen into more than one window. Easy ways to accomplish this are provided by the System menu.

Walk-through for Making Your Own Windows

The left-hand column of the System menu has the heading Windows. We are going to use the menu to try different ways of reconfiguring the screen. As we proceed, watch the mouse documentation lines.

1. Cold boot your machine before starting this exercise.

2. Press `SELECT E` to select Zmacs.

3. Let's create a new Lisp Listener window on the right-hand side of the screen.

   a. Call up the System menu.

   b. Click Left on [Create].

   c. You are now being prompted for the kind of window you want to create. Click Left on [Lisp].

   d. Note that the mouse cursor is now shaped like the upper left corner of a rectangle. Position the cursor just inside the upper left corner of the screen area that is not used by the editor window. If you click Left, the upper left corner of the Lisp Listener that you are creating is located at the cursor location. If you click Right, the upper left corner of the Lisp Listener that you are creating is aligned with adjacent boundaries.

   e. Click right to position the upper left corner of your window.

   f. Now the mouse cursor is the lower right corner of a rectangle you see on the screen. Position the mouse cursor at the lower right corner of your screen.

   g. Click Right to position the lower right corner of your window.

   It was not necessary to create the window in an "unused" area. However, if we want the contents of each of the windows to be fully visible, then we don't want the windows overlapping.

4. Click Left on [Edit Screen] in the System menu.

5. Click on [Move Window], [Reshape], [Move Multiple], or [Move Single] to change an existing window.

6. If more than one window is visible on the screen, the mouse cursor assumes the shape of a circle with a cross in it. Locate the cursor over the window which you wish to move or alter and click appropriately.

7. Follow instructions in the mouse documentation lines. Continue editing the screen until you wish to exit (in which case click on [Exit] in the Edit Screen menu).

Adjusting the Command Processor

You can use the Set Command Processor command to change the Command Processor *mode* and the Command Processor *prompt*.

The Command Processor mode

The Command Processor has four *modes* which determine how it interprets what you type. The modes are:

Command-Only
: The Command Processor interprets every word you type as a command. You cannot type Lisp forms.

Command-Preferred
: This is the default mode.

If the first character you type is *alphabetic*, the Command Processor checks first to see if what you are typing is a known command and then checks to see if it is a known variable. Otherwise, it assumes you are typing a Lisp form.

If you want the Command Processor to look for a variable first, type a comma before the word.

```
login                ;the Login command

,login               ;the Lisp variable login
```

Form-Preferred
: If the first character you type is a colon, the Command Processor assumes you are typing a command. Otherwise, it assumes you are typing a Lisp form.

This means that all commands must be preceded by a colon in Form-Preferred mode.

```
:login              ;the Login command

login               ;the Lisp variable login
```

Form-Only
: The Command Processor interprets everything you type as a Lisp form. It is impossible to type commands. (To type commands again, type the Lisp function `(cp-on)`.)

The Command Processor Prompt

The default prompt in Command-Preferred mode is the string `Command:` (the default for Form-Preferred is an empty space). You can change this by specifying a new prompt (between double quotes) as the second argument to the Set Command Processor command. For example,

```
Set Command Processor Form-Preferred ">"<RETURN>
```

changes the Command Processor mode to Form-Preferred and the prompt to a right angle-bracket (`>`)

Walk-through for Adjusting the Command Processor

1. Type

```
Se<SPACE>C<SPACE>P<SPACE>
```

2. You are now being prompted for a mode. Type

   ```
   C<SPACE><SPACE>
   ```

   The Command Processor complains that `"C" is ambiguous.`

3. Press `RUBOUT` several times to get rid of the `C`.

4. Type

   ```
   C-P<SPACE>
   ```

   Notice that it now completes to `Command-Preferred`.

5. Type

   ```
   "You rang? "<RETURN>
   ```

   You have now changed your prompt, but the mode remains the same.

6. If you wish to change back to the original prompt, type

   ```
   Set Command Processor Command-Preferred "Command: "<RETURN>
   ```

   If you wish to change your prompt to something else, just specify that string instead of `"Command: "`. Make sure that your mode is Command-Preferred when you are finished, however, as that is the mode you should be in as you work through this document.

Flashy Scrolling

Windows or menus that display *More above* or *More below* have flashy scrolling implemented. This means that if you bump the mouse cursor against the sensitive part of the top edge when *More above* is displayed, the mouse cursor shape changes to a thick single-headed arrow pointing up. Each time the thick cursor is bumped against the sensitive part of the top edge, one line is scrolled downward. Similarly, if *More below* is displayed, bumping against the bottom window edge results in upward scrolling.

Editor windows do not display the *More above* or *More below* messages, but flashy scrolling is implemented on them. (Actually, you should see a line in the minibuffer of your editor that says `[More above]`, `[More below]`, or `[More above and below]`, if you are not looking at the entire buffer.)

Windows that display *More above* and *More below* skip backward and forward by an entire screenful when you click left on the respective mouse-sensitive areas. Which parts of the top and bottom are sensitive depends on how the window properties were defined; usually windows have regions near the the top and bottom of the scroll bar that are sensitized for flashy scrolling.

Trouble-shooting

Doing any one of the following operations could put you in the Debugger, a program that traps errors. You can tell that you are in the Debugger if you see **Trap:**, a message, and a right arrow (→). Read the Debugger message carefully, as it contains useful information. If you do not understand the message, just pressing the ABORT key generally fixes the problem. If ABORT does not fix the problem, try the solutions listed below.

Things That Can Get You Stuck

- Killing windows.
- Overlapping windows.
- Performing operations on unexposed windows.
- Aborting out of menus.
- Exposing a window that is larger than or outside the screen itself.

Things That Can Get You Unstuck

- FUNCTION ESCAPE causes the window that is trying to display something new to be exposed. This command is useful for the conditions Output Hold and Sheet Lock.
- FUNCTION 0 S brings up any window that is trying to display an error.
- FUNCTION c-T clears temporary window system locks (use with caution). This command is useful for Sheet Lock.
- FUNCTION c-CLEAR INPUT clears all window system locks (use with caution). This command is useful for Sheet Lock.
- The condition (no window) occurs when the window system is confused as to which is your current window. Selecting another window should clear this up.
- If you click on the System menu option [Arrest], the word Arrest appears in the status line. Bring up the System menu again and click on [Unarrest].
- If you find yourself in the *cold load stream*, the Window System is in trouble. Read the information displayed by the Debugger carefully and take appropriate action. Usually, ABORT is the right thing to do. You might need to press ABORT several times, but eventually you should be back in the window you started in.
- If you were thrown into the cold load stream because the window system is locked, usually Unlock all window system locks? is one of your options. You should answer Yes.

Documentation References

- *Window System*
  See the section "Window System Concepts".
- *System Menu*
  See the section "The Mouse and Menus".
- *Using the System Menu*
  See the section "Using the System Menu".
- *Turning the Command Processor on and off*
  See the section "Turning the Command Processor On and Off".
- *Unhanging Yourself - Trouble Shooting*
  See the section "Recovering From Errors and Stuck States".

The Namespace

Introduction

This chapter explains a few things about the namespace database and shows you how to add yourself to that database.

- "Workbook: The Namespace and Logging in"

- "Workbook: Adding Yourself to the Namespace"

- "Workbook: Optional Namespace User Attributes"

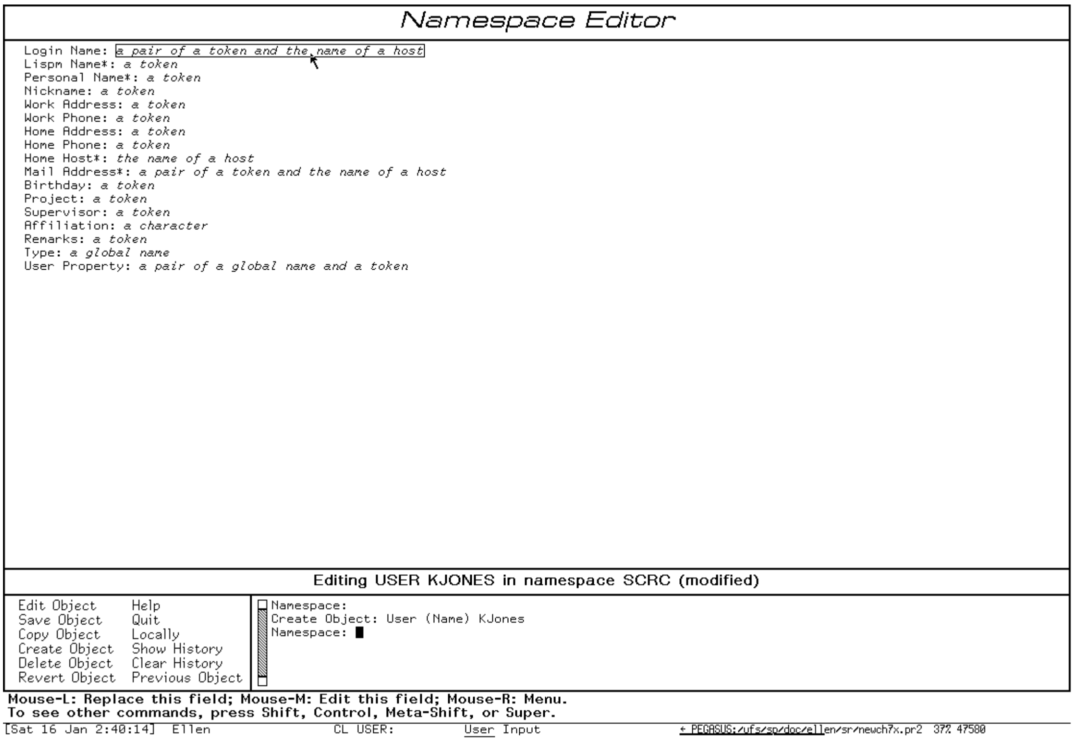
- "Workbook: The Namespace Documentation References"

The Namespace and Logging In

The association of your name with a particular *file-server* is made by the database known as the *namespace*. There is one namespace database at your site, shared by all users.

The namespace contains information about users, hosts, printers, and other objects. In particular, the information about users includes what *home host* is associated with what *login-name*. Your login-name is the name by which Genera knows you. Your home host is the machine on which you keep your files. Lisp looks for your lispm-init file in the directory *file-server:>login-name>*. This directory is your home directory. If you do not appear in the database, you are invited to add yourself, in the form of a *user object,* when you first log in. You need do this only once.

Adding Yourself to the Namespace



```
Login Name: a pair of a token and the name of a host
Lispm Name*: a token
Personal Name*: a token
Nickname: a token
Work Address: a token
Work Phone: a token
Home Address: a token
Home Phone: a token
Home Host*: the name of a host
Mail Address*: a pair of a token and the name of a host
Birthday: a token
Project: a token
Supervisor: a token
Affiliation: a character
Remarks: a token
Type: a global name
User Property: a pair of a global name and a token
```

Editing USER KJONES in namespace SCRC (modified)

```
Namespace:
Create Object: User (Name) KJones
Namespace: █
```

Mouse-L: Replace this field; Mouse-M: Edit this field; Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.

```
[Sat 16 Jan 2:40:14]  Ellen          CL USER:        User Input          ← PEGASUS:/ufs/sp/doc/ellen/sr/newch7x.pr2  37% 47580
```

Figure 42. The Namespace Editor

To add yourself to the namespace, you use the *Namespace Editor*. If you have difficulty logging in because Lisp does not recognize your login-name, you are asked if you wish to add yourself to the namespace database. If you say yes, you are put into the Namespace Editor automatically, so you can edit your user object. (Each entity known to the namespace — a printer, a Symbolics computer, you — is represented by an *object* in the namespace. Your user object is the namespace's representation of you.)

The pieces of information about a user are known as *attributes*. You must enter four of these attributes to add yourself to the namespace. The four required attributes are marked with asterisks (∗) after their names (Figure 42).

You can also access the Namespace Editor with the Command Processor command Edit Namespace Object, specifying a *class* (such as `user`) and object (such as a login-name). For example, if your login-name is `davido` and you want to edit your namespace object, you would type:

```
Edit Namespace Object user davido
```

You can use the menu options in the Namespace Editor to read in, modify, and save objects. In this chapter, we only talk about adding yourself to the namespace when you can't log in. You should not experiment with adding other objects to the namespace unless you are sure you know what you are doing.

A third way to access the Namespace Editor is from the System menu. If you select it this way, you must click on [Edit Object] in the Namespace Editor's menu to edit an object.

Workbook: Walk-through for Adding Yourself to the Namespace

1. Check to see whether you have typed your login-name properly. If not, press the ABORT key and try to log in again. Otherwise, it is a good idea to ask your site administrator to help you log in. If you do not have a site administrator available, follow the steps below. You should only need to do this once.

2. When you try to log in, if you are not part of the user database or if you have mistyped your login-name, you see the following:

   ```
   The user named "tess" was not found.:
   Do you want to log in as tess on some
   specific host? (Y, N, or R)
   ```

   You should type the letter Y.

3. You then see:

   ```
   Host to log in to:
   ```

4. You should type the name of your *home host*, the machine on which your files are stored. All the hosts that are known by the namespace database have names. This user typed

   ```
   SYMBOLICS1<RETURN>
   ```

5. You see:

   ```
   No init file: The directory >TESS does not exist.
   For SYMBOLICS1:>tess>lispm-init.bin.newest
   Do you wish to add tess to the user database? (Y or N)
   ```

   You should type the letter Y. You are now presented with a new window (Figure 42). This is the Namespace Editor; *inexperienced users should not experiment with it.* Here you are told by the namespace database exactly how to add a *user object* for yourself to the namespace database.

   The cursor is in the lower right-hand part of the window. The upper part of the window contains attributes for users. The required attributes are marked by an asterisk (*). Note that of these four, the first three already have values:

   Lispm Name*: tess
   Home Host*: SYMBOLICS1
   Mail Address*: ***Pair:*** tess SYMBOLICS1
   Personal Name*: *Token*

You must add the *Personal Name* attribute. To do this, move the mouse until the mouse cursor is close to the word *Token* following `Personal Name:` . The cursor changes into a rectangle surrounding *Token*. Click Left. The Namespace Editor prompts you, in the lower part of the menu, for a personal name. Type your name, followed by RETURN. Your name now appears in the upper part of the menu, next to `Personal Name:` .

6. Now move the mouse cursor to the [Save Object] option in the Namespace Editor menu in the lower left part of the screen, and click Left. (You can also type the Save Object command in the lower right part of the screen.) This saves the user object in the database.

7. After the message announcing that the object has been saved appears, move the mouse to the [Quit] option and click Left, or enter the Quit command in the command pane. The window disappears, leaving you in Dynamic Lisp Listener 1. You have now added yourself to the namespace, and you can log in normally.

Optional Namespace User Attributes

In the Namespace Editor, you saw many more user attributes than you were required to type in. These are *optional* attributes, but many of them are interesting or useful. For instance, you can enter your work address and phone number, your home address and phone number, your supervisor's name, and your birthday. Other people can check in the namespace database to find this information about you. If you do not want some piece of information to be available to other people, you should not enter it into the namespace.

Documentation References

- ***The Namespace System*** and ***Names and Namespaces***
  See the section "Introduction to the Namespace Database".
- ***Updating the Namespace Database***
  See the section "Using the Namespace Editor".

Overview of the Machine

Introduction

This chapter familiarizes you with the "basic anatomy" of a Symbolics 3600-family computer. If you have an Ivory-based machine, you should refer to the *User's Manual* for your machine type for hardware-specific information. We discuss the parts of the machine and the way files are distributed on the disks. This chapter does not contain any walk-throughs.

- "Workbook: Basic Parts of the Machine"
- "Workbook: Parts of the Processor"
- "Workbook: About the Fep"
- "Workbook: FEP Files Versus LMFS Files"
- "Workbook: Hello Files"
- "Workbook: The Fep and Files"
- "Workbook: A Closer Look At Booting"
- "Workbook: Overview of the Machine Documentation References"

Basic Parts of the Machine

Symbolics computers are single-user machines; all the resources of the computer are available to each user.

There are many single-user computers on the market. All of them have a central processing unit (CPU), physical or random access memory (RAM), and a disk of some kind. They also have some kind of display capability, usually a monitor or an ordinary TV screen. Your Symbolics computer has these basic parts as well. The CPU, memory and disk are all contained in the *processor cabinet*. Much of this capability is wired into circuit boards. The part that you typically look at is called the *console*. This includes the *monitor*, the *keyboard*, and the *mouse*.

Parts of the Processor

The processor unit contains the following major parts:

Lisp Processor The Lisp Processor is the CPU for the Symbolics computer. This board is the heart of the Symbolics computer.

Front End Processor
The Front End Processor (FEP) is a separate 32-bit computer (M68008). It takes control whenever the Lisp Processor stops running and when the machine is first powered up. You use the FEP to start up the Lisp Processor. This operation is called *booting*. The FEP is also used for other operations that need to be done before the Lisp Processor is turned on.

Physical Memory Every Symbolics computer has at least one board of physical memory. This physical memory is used only by the Lisp Processor, not by the FEP.

Hard Disk Every Symbolics computer has at least one hard disk with (currently) at least 140 (unformatted) megabytes of storage. This storage is broken up into free space and files. There can be two separate file systems. One, which you occasionally need to look at, is maintained by the FEP (FEPFS). The other, which contains user files, is maintained by the Lisp Machine File System (LMFS).

Other things At least one Symbolics computer at your site should have a cartridge tape drive), so that you can back up your file system and also so that you can receive updates of the software. If your machine has a color monitor, the processor box also contains some other specialized boards that are not described here. The processor box is connected to the console by a video cable. Several machines can be interconnected by network cables.

About the FEP

Things to note:

- The FEP starts when you power on the machine.

- There is very little for users to do in the FEP. Most of the time you just use the FEP to start the Lisp Processor.

- The FEP does many low-level operations. Be careful. Do not use any commands that you do not completely understand. The only FEP commands you need right now are the *Hello* and *Boot* commands. The Hello command bootstraps the FEP, after which you can use the Boot command to execute a file of FEP commands known as a *boot file*. Executing the Boot command starts the Lisp Processor.

- If the Lisp Processor is not running, but your machine is on, you must be in the FEP.

- The FEP gives you access to the FEP file system and other parts of your Symbolics computer, even when the main Lisp Processor is not running.

FEP Files versus LMFS Files

Each disk on the Symbolics computer has a separate file system called the FEP file system. The disk is divided into *FEP files*. One of these files is the *boot.boot* file, which you use to cold boot your machine. Another is a file called *lmfs.file*. The entire user file system, where you store your files, is stored within lmfs.file. It is possible to manipulate this file in all the usual ways. For example, you could delete it. However, **you should never delete the lmfs.file**. There are special interfaces to manipulate this file (and related files) in the File System Maintenance program, which is accessed through the Lisp Processor.

The Lisp Machine File System (LMFS) takes care of LMFS (user) files. The LMFS is designed to be robust and to have the sort of user-level features that you want. The FEP file system is designed to be simple, so that the FEP can understand it. Take advantage of the features of the LMFS; do not ever store your files in the FEP file system.

Each Symbolics computer has a separate FEP file system for each disk, but cannot have more than one LMFS. It is possible for a LMFS to be contained in more than one FEP file, each of which is called a *partition*. A partition can be on any disk on the machine. All of the partitions together make up a single LMFS.

At many sites, one computer on the network (the *file-server*) has a lot of disk space and stores user files for all users. This helps centralize the backup procedures. Therefore, you often do not have a LMFS on your local machine (as opposed to your file-server); it's not required.

Even if the above situation describes your site, you might choose to have a local LMFS just in case the file-server is not available when you need to store some data in a file.

Hello Files

Whenever your machine is powered up, you need to give the FEP command Hello. This causes the commands in the file Hello.boot to be executed. Executing these commands makes it possible for you to use the standard FEP commands, such as Boot; these commands are not available until after you have executed the hello file.

The FEP and Files

The FEP is also responsible for the allocation of space on the disk into files. This mechanism is called the *FEP File System*. When the Lisp Processor is not running, you can only *look* at the files in the FEP file system — you can't delete, modify or create any. The Lisp Processor must be running for you to do any operations on these files. Only the site administrator should alter or delete FEP files.

The FEP file system contains several kinds of files:

- *.load* - World load files.
- *.mic* - Microcode files.
- *.page* - Paging files.
- *.boot* - Boot files.
- *.flod* - FEP load files. Never touch these files in any way other than as specified in Symbolics documentation. They contain the software for FEP commands.

- *.file* - Lisp Machine File System (LMFS) partition files. Never touch these files in any way other than as specified in Symbolics documentation. They contain user files and directories. Manipulating these files (the `.file` files) without knowing what you're doing can result in permanent loss of your data.

A Closer Look at Booting

The FEP accepts a collection of commands defined only for the FEP. The booting operation requires a sequence of several commands. Usually these commands are put into a *boot file*, so that you do not have to type them all in each time you cold boot. To execute a boot file, you use the FEP's Boot command. This causes the commands in the boot file to be executed in order, just as if you had typed them to the FEP by hand.

In addition, it is sometimes necessary (usually only after you've just powered up the machine) to scan command tables and Hardware Initialization Tables. This is done with the FEP's Hello command, which runs a special boot file, usually called Hello.boot. You will usually be prompted to enter the Hello command, but if a FEP command you expect to see is apparently not present or if you are having trouble booting, you may need to enter the Hello command.

Conventional booting involves several FEP commands. You can enter these by hand, but in most cases you'll want to have them in a boot file, usually called Boot.boot. In either case, use the order presented here.

1. *Clear Screen*. This clears the screen and makes the booting process look neater.

2. *Clear Machine*. This erases anything from the previous time the Lisp Processor was run. You must clear the machine before loading the microcode.

3. *Declare Paging-files*. At least one FEP file is allocated for use as *paging space* (virtual memory). On some computers, you can't control how much disk space is used for virtual memory, but a user or site administrator can (and must) on a Symbolics computer. The default paging file is FEP0:>page.page, but you can use most of the free space on any disk as paging files. In general, the more paging, the better.

4. *Mount FEP1*. In a machine with two disk drives, this command mounts FEP1: so the environment knows about the disk if it is not referenced in the boot files, such as by having a paging file declared on it.

5. *Load Microcode*. Microcode is software that is loaded into the Lisp Processor to internally customize the way that the Lisp Processor works. A different microcode exists for each of the different hardware configurations.

6. Specify a world load file to run. A world load file is a snapshot of a running Lisp environment from some time in the past.

   On a running Symbolics computer, you're *always* running in the context of one of these world loads, except when you're in the FEP. You "run" a world for some time before stopping and booting again.

   Loading the world specifies which world load you're going to start running.

   There are two ways of specifying which world you want to run. Many users have multiple boot files, each of which runs a different world. The two basic choices are to use either the *Load World* command or the *Netboot* command, but not both.

   - *Load World.* To use this command at least one FEP file on your disk must be a *world load* file. The Load World command names this file and prepares to load it.

   - *Netboot.* To use this command you must have a netboot core file on your disk. Netbooting is a means of loading a world from another machine. The netboot core file on your machine enables the Netboot command to find that world and prepare to load it over the network.

7. *Set Chaos-address.* If your machine is *networked* (has data connections) to any other machine, it needs an *address* to distinguish it from the other machines on the network. This command lets you tell the machine what its address is.

8. *Start.* This starts the Lisp Processor running either the world from your disk specified in the Load World command or the world from another machine specified in the Netboot command.

After you boot, the Lisp Processor is running. All normal operations, such as editing files or compiling and running programs, are done using the Lisp Processor. Besides booting, only a few irregular operations are executed by the FEP.

Documentation References

- ***Using the FEP***
  See the section "Overlay (Flod) Files and the FEP".
- ***The FEP File System***
  See the section "FEP File Systems".
- ***Cold Booting***
  See the section "Cold Booting".

Pocket Guides

Pocket Guide for Logging In

To Log In

Type

```
    Login user-name<RETURN>
```

to the Command Processor.

To Log Out

Type

```
    Logout<RETURN>
```

to the Command Processor.

To Cold Boot

1. Log out
2. Type

   ```
   Halt Machine<RETURN>
   ```

3. Answer yes to the question, "Do you really want to halt the machine? (Yes or No)"
4. Type either

   ```
   Boot<RETURN>  or Boot >boot-file-name.boot<RETURN>
   ```

Pocket Guide for Selecting Activities

SELECT Key Options

=	Select Key Selector
C	Converse.
D	Document Examiner.
E	Editor (Zmacs).
F	File system maintenance (FSMaint or FSEdit).
I	Inspector.
L	Dynamic Lisp Listener.
M	Zmail.
N	Notifications.
P	Peek.
Q	Frame-Up.
T	Terminal.

X	Flavor Examiner.

Programs on the System Menu

[Lisp]	A Lisp Listener. Available on `SELECT L`.
[Edit]	Zmacs. Available on `SELECT E`.
[Inspect]	The Inspector. Available on `SELECT I`.
[Mail]	Zmail. Available on `SELECT M`.
[Trace]	Not really an activity, but simply a menu interface to the **trace** debugging facility.
[Emergency Break]	Not really an activity, this provides a mechanism for evaluating Lisp without using the window system.
[Frame-Up]	The Layout Designer. Available on `SELECT Q`.
[Namespace Editor]	The Namespace Editor.
[Hardcopy]	Not really an activity, this is simply a menu interface to the hardcopy facility.
[File System]	The File System Maintenance window. Available on `SELECT F`.
[Document Examiner]	Document Examiner. Available on `SELECT D`.

Pocket Guide for Input Editor Commands

Input Editor commands

`HELP`	Display documentation for the current command.
`c-HELP`	Display listing of all Input Editor commands.
`c-F`	Move forward over one character.
`c-B, BACKSPACE`	Move back over one character, without deleting it.
`c-D`	Delete the character under the cursor.
`RUBOUT`	Delete the character before the cursor.
`c-T`	Transpose the character under the cursor with the character preceding the cursor.
`c-A`	Go to the beginning of the line.
`c-E`	Go to the end of the line.
`c-K`	Delete all characters from the current cursor position to the end of the line.
`m-F`	Move forward over one word (or part of a word).
`m-B`	Move back over one word, without deleting it.
`m-D`	Delete the word (or part of a word) starting at the current cursor position.
`m-RUBOUT`	Delete the word (or part of a word) before the cursor.

Pocket Guide for the Keyboard

In the Command Processor

ESCAPE	Shows recent command history.
c-0 ESCAPE	Shows entire command history.
END	Signals that you have finished typing a command.
RETURN	Signals that you have finished typing a command.
SPACE	Asks the Command Processor to complete the word you just typed as well as the previous words, as far as possible.
COMPLETE	Asks the Command Processor to complete the entire command you are typing, as far as possible.
CLEAR INPUT	Throws away all characters you have typed since the last prompt.

Some Useful Keystrokes

FUNCTION	With another key, lets you do useful things to do with software.
FUNCTION HELP	Provides a list of things you can do with the FUNCTION key.
ABORT	Aborts the operation currently in progress. Takes effect when read.
c-ABORT	Like ABORT, but takes effect immediately.
m-ABORT	Causes process to return to topmost command loop. Takes effect when read. c-m-ABORT takes effect immediately.
SYMBOL	Allows you to access special characters.
SYMBOL-HELP	Gives you a list of the special characters and special function keys and documents the LOCAL key.
REPEAT	Makes keyboard keys auto-repeat.
SELECT	With a character, moves you to another activity.
SELECT HELP	Gives you a list of the activities you can select with the SELECT key.
LOCAL	Lets you do useful things with the console hardware, such as making the screen bright or dim.
HELP	Generally gives some help in any context.
c-HELP	Gives you a list of Input Editor commands.
HELP HELP	In the editor, explains the help options.

Pocket Guide for Zmacs

File Operations

c-X c-F	Find or Create Buffer
c-X c-S	Save Buffer (to a file of the same name)
c-X c-W	Write Buffer (to a file of a different name)

	Forward	Backward	Begin	End	Delete Forward	Delete Backward	Transpose
Character	c-F	c-B			c-D	RUBOUT	c-T
Word	m-F	m-B	m-B	m-F	m-D	m-RUBOUT	m-T
Lisp Form	c-m-F	c-m-B	c-m-A	c-m-E	c-m-K	c-m-RUBOUT	c-m-T
Line	c-N	c-P	c-A	c-E	c-K	CLEAR INPUT	c-X c-T
Sentence	m-E	m-A	m-A	m-E	m-K	c-X RUBOUT	

Figure 43. Chart of Zmacs Commands

Searching and Replacing

c-S	Search forward
c-R	Search backward
c-sh-%	Replace string1 with string2
m-sh-%	Replace string1 with string2, querying the user

Buffer Operations

c-X c-B	List Buffers
c-X B	Select Buffer
c-X K	Kill Buffer

m-X Hardcopy Buffer

m-X Kill or Save Buffers

m-X Insert Buffer

Marking and Yanking and Region Operations

Mark region	Hold left mouse button and drag to mark region
c-SPACE	Start marking region here
c-W	Kill Region
m-W	Copy Region into kill history
c-Y	Yank last item from kill history
m-Y	Yank previous item from kill history
c-sh-Y	Yank item from kill history matching string
m-sh-Y	Yank previous item from kill history matching string
c-m-Y	Yank previous command

c-m-sh-Y Yank previous command matching string

Getting Help in Zmacs

HELP Prompts " Type one of A,C,D,L,U,V,W,Space,Help,Abort:"

A Displays all the commands whose names contain a certain substring. Type the string.

C Displays documentation for a command. Type the command accelerator after the C.

D Displays documentation for an extended command. Type the command name.

L Displays the last 60 characters you typed.

U Offers to undo the last change to the buffer.

V Displays all the Zmacs variables whose names contain a certain substring. Type it.

W Finds out whether an extended command is bound to a key. Type the command name.

SPACE Repeats the most recent HELP command.

ABORT Returns you from the prompt.

c-/ After pressing m-X, type part of a command name, then press c-/ to get a mouse-sensitive list of all the commands whose names *contain* that string.

c-? After pressing m-X, type part of a command name, then press c-? to get a mouse-sensitive list of all commands whose names *start with* that string.

Click Right Type part of an extended (m-X) command and click Right and you'll get a list of all commands whose names *start with* that string.

HELP Type in part of a command name and press HELP for a mouse-sensitive list of commands whose names start with that string.

COMPLETE Type part of a command name and press COMPLETE or SPACE. The system completes as much of the command as it can figure out.

m-X Show Documentation
Displays any system documentation from the documentation database.

`m-X` Show Candidates
Displays a mouse-sensitive list of candidates from the documentation database whose names or index entries match words you supply.

Pocket Guide for File Operations

File Operations

	CP	*Dired*	*FSEdit*
Create	Edit File	--	--
Delete	Delete File	`D`	[Delete]
Undelete	Undelete File	`U`	[Undelete]
Edit	Edit File	`E`	[Edit File]
Load	Load File	`L`	[Load]
Rename	Rename File	`R`	[Rename]
Copy	Copy File	`C`	--
Hardcopy	Hardcopy File	`P`	[Hardcopy]
Show	Show File	`V`	[Show]

Directory Operations

	CP	*Dired*	*FSEdit*
Create	Create Directory	--	[Create Inferior Directory]
Delete	Delete File	`D`	[Delete]
Undelete	Undelete File	`U`	[Undelete]
Edit	Edit Directory	`E`	--
Expunge	Expunge Directory	`Q E`	[Expunge]

Show	Show Directory	I E V	--

Pocket Guide for Document Examiner

Menu Mouse Clicks

[Show Candidates]	Left: Heuristic Search for Candidates
	Press `m-COMPLETE` for menu
	`Matching: Exact Heuristic Substring Initial`
	`Multiple-word-order: Adjacent Any`
	Enter `END` or `ABORT` to accept or discard changes.

[Show Documentation]	Left: Show Documentation
[Show Overview]	Left: Show Overview
[Show Table of Contents]	Left: Show Table of Contents
[Help]	Left: Brief Command Summary Middle: Show Full Document Examiner Documentation
[Select Viewer]	Left: Select Viewer Right: Menu: [Hardcopy Viewer] [Remove Viewer] [Select Viewer]
[Reselect Candidates]	Left: Choose among all candidate lists
[Read Private Document]	Left: Read Private Document Right: Menu [Hardcopy Private Document] [Load Private Document] [Read Private Document] [Save Private Document]

Remember, there are command equivalents for each of these mouse clicks.

Glossary

- *Activity* - A system program, usually with its own window or windows.
- *Attributes* - Pieces of information about something.

- *Boot file* - A file of FEP commands that does the things necessary to start the Lisp Processor.

- *Buffer* - A work area, usually in Zmacs.

- *Clicking a mouse button* - Pressing and releasing the button quickly.

- *Cold booting* - The process of clearing out the machine and bringing in a fresh, unmodified world to use.

- *Command Processor (CP)* - The feature that parses and executes commands in a Lisp Listener.

- *Command* - A way of telling the machine to do something.

- *Completion* - The process of supplying characters to those a user enters for a command, up to the point where there is more than one possible command which could fit the current set of characters.

- *CP mode* - The way the CP interprets typed input.

- *CP prompt* - The string the CP displays when it prompts you for input.

- *Current package* - The place where the machine first looks up variables, functions, and so on. Only relevant to Lisp programmers.

- *Deexposed* - A deexposed window is one which is not completely visible on the screen.

- *Default* - A default is a value for an argument that is supplied by the system, often the last value that argument had. You can choose to take a default value or not to.

- *Directory* - A group of files and directories.

- *Echo Area* - The part of the window where the commands you type are displayed, in Zmacs.

- *Editor Window* - The part of the window where the contents of a buffer are displayed, in Zmacs.

- *Extended command* - A Zmacs command that is an entire word or phrase, typed after typing m-X. Extended commands support completion.

- *FEP file* - A file maintained by the FEP file system.

- *File* - A section of the disk on which data is stored. Also, the collection of data itself.

- *File-server* - A host on the network that is used to store many users' files.

- *Filling* - Breaking text at the last word boundary before a certain column.

- *Frame* - A window that has one or more subordinate windows, called panes.

- *Front End Processor (FEP)* - A special, supplementary computer that is used for fundamental system services, such as cold-booting.

- *Global kill ring* - The place that contains anything larger than a single character that has been killed in any window. The global kill ring can be accessed from all windows, facilitating movement of text from one activity to another.

- *Hello file* - A file of commands that bootstrap the FEP after the machine is powered up.

- *Hierarchical* - A system in which everything can be said to be either below, above, or on the same level as each other thing.

- *History* - A list of commands you have typed since your machine was cold-booted.

- *Home directory* - The main directory in which you keep your files, which is generally distinct from the home directories of the other users of the machine.

- *Home host* - The machine on whose disk most of your files are kept, usually your file-server.

- *Host* - Any computer, not necessarily a Symbolics computer.

- *Incremental Searching* - Finding a string of characters in a buffer, in Zmacs, moving along in the buffer as each character is typed in.

- *Input Editor* - The feature that collects the characters you type, so that you can edit the current input and review or reuse earlier commands and forms.

- *Keyword arguments* - Optional named arguments to CP commands.

- *Kill* - A kill command puts text into the kill ring. It does not necessarily remove the text from the buffer, although it generally does.

- *Lisp form* - A piece of Lisp code.

- *Lisp Listener* - A window that accepts Command Processor commands and Lisp forms.

- *Lisp stopped itself*, *when you're talking to the FEP*, *when you're in the FEP*, *when the Lisp Processor is not running* are all equivalent. All refer to the state of the machine in which the FEP processor is running, but the Lisp Processor is not.

- *Lispm-init file* - An optional file of Lisp code that you store in your home directory which is run when you log in.

- *LMFS file* - A file maintained by the LMFS, stored in `lmfs.file`.

- *Login-name* - The name by which the machine knows you and which distinguishes you from the other users of the machine.

- *Marking* - Creating a region. A marked region is underlined.

- *Menu* - A small, usually temporary window that contains a list of choices, one (or more) of which can be selected using the mouse.

- *Minibuffer* - The part of the window where you are prompted for command arguments, in Zmacs.

- *Mode Line* - The line that contains information about the status of the current buffer, in Zmacs.

- *Mode* - In Zmacs, a way of customizing the behavior of the editor to your current application.

- *Modifier keys* - Keys that are held down while another key is pressed.

- *Mouse cursor* - The small black arrow (or sometimes another character) on the screen. The mouse cursor moves when the mouse is moved.

- *Mouse documentation line* - The line above the status line, usually in reverse video. This line shows the meanings or effects of pressing the mouse buttons.

- *Mouse* - A small box with three buttons that is attached to the console and controls the position of the mouse cursor on the screen. Also used to refer to the mouse cursor.

- *Mouse-sensitive* - Things that are mouse-sensitive are highlighted with a rectangle when the mouse is over them.

- *Namespace*, *Namespace Database* - The place where information about users and hosts is stored.

- *Networked* - When two or more machines have data connections to each other, they are networked.

- *Netbooting* - Booting your machine by loading a world located on a file-server, rather than on your machine.

- *Notification* - A message from a machine on your network which is not connected to the activity that you are running.

- *Numeric arguments* - Arguments to Zmacs and Input Editor commands that cause them to execute more than once and/or in a different manner.

- *Partial completion* - Completion of only the words that have been typed so far, even if an entire command has been uniquely specified.

- *Pathname component* - A part of a pathname, such as a file name or a directory name.

- *Pathname* - The way in which you uniquely specify a file and the place where it is stored.

- *Positional arguments* - Required arguments to CP commands that must all be typed, in the right order.

- *Process state* - A brief (usually one- or two-word) description on the status line of what the machine is doing.

- *Process* - A separate computation that keeps running until the system is halted, but is only active when necessary.

- *Run bars* - Lines that flicker under the current package name and process state and indicate what the machine is doing.

- *Scroll bar scrolling* - Moving text up and down in a window, using the mouse.

- *Scroll bar* - A line or area, usually at the side of a window, that shows where you are relative to the entire contents of the window and how much of those contents you are seeing.

- *Scrolling* - Moving text up and down in a window, using the keyboard or the mouse.

- *Selecting* - An easy way for a user to switch among activities.

- *Status line* - The bottom line of the screen, where information relating to the current state and activity of your machine is displayed, including the date, time, and login-name.

- *Subdirectory* - A directory contained in another directory.

- *System menu* - The main menu for the system. This menu is available by clicking shift right anywhere in the window system.

- *The local machine* - The particular Symbolics computer you are using, as opposed to any other host that you can access over the network.

- *The machine* - Either the entire Symbolics computer, the Lisp Processor, or the basic software included in the world load file.

- *The system* - Either the entire Symbolics computer, the Lisp Processor, or the Genera software included in the world load file.

- *Token completion* - Completion of as much as possible of the command, no matter how many words been typed.

- *When the machine is cold-booted* - When the Lisp Processor has started running but no one has done anything to the machine yet.

- *Window system* - A collection of software that provides an interface between the user and the lower-level Genera software.

- *Window* - A rectangular area of the screen that can be used for input and output.

- *World* - An entire Lisp environment that has previously existed. The FEP command Load World makes a particular world load file be the world that you are running. Running a world load means that your environment starts out exactly the same as the environment that was saved into the world load file. Sometimes a distinction is made between the running world and a world load file, since the running world changes as you do things in it, becoming different from the world you originally loaded.

- *Yanking* - Bringing back a previously typed section of text.

- *Zmacs* - Genera's screen-oriented text editor.