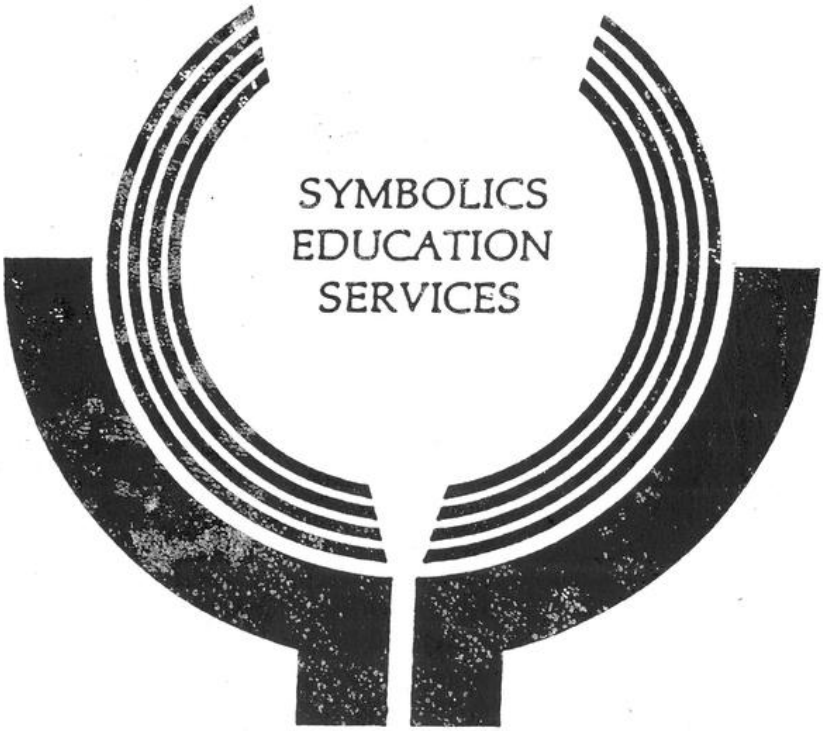
Using Your Symbolics Computer: A Workbook



symbolics inc.

Using Your Symbolics Computer: A Workbook

Printed in the United States of America.

Using Your Symbolics Computer: A Workbook

October 1986

This document corresponds to Release 6.1 and later releases.

Symbolics, Inc.
4 New England Tech Center
555 Virginia Road
Concord, MA 01742

Text written and produced on Symbolics 3600™-family computers by the Education Services Group of Symbolics, Inc.
Text masters produced on Symbolics 3600™-family computers and printed on Symbolics LGP2 Laser Graphics Printers.

Printed in the United States of America.

Printing year and number: 88 87 86 9 8 7 6 5 4 3 2 1

Table of Contents

List of Figures

Using Your Symbolics Computer: A Workbook

1 Introduction

This is the first document you should read in order to learn how to use Symbolics 3600-series computers. It is designed to teach you the skills you need to use your Symbolics 3600-series computer quickly and effectively. We present concepts that every user needs to know, and we provide exercises and activities to turn those concepts into practical skills.

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 intended for both programmers and non-programmers. The material covered in Section 1 of this workbook is a prerequisite for all Symbolics Education Services courses. However, you can complete the second and third sections after taking a course. Practicing the material will help you learn it faster.

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.

It is very important to have a 3600-series computer available so that you can work through the workbook's hands-on activities. These activities form an important part of the learning process. If you do not have access to a machine, call the registrar at either the Cambridge or San Francisco Training Centers.

You can use this book with any Symbolics 3600-series computer running Release 6.0 or 6.1. The machine should already be configured for your site. If you are not sure what 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.

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 machine exercise, go to your machine and do the exercise before you continue 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.

There are no case-sensitive exercises in this material. You might wish to type in lowercase; it's easier. We show letters in uppercase only for typographical clarity; if we want you to hold down the SHIFT key, we say so. Example: c-F means to hold down just the CONTROL key while you type a lowercase f. But

c-sh-F means to hold down the CONTROL and SHIFT keys while you press the F key. We will describe our keyboard notation in more detail later. Symbolics computers are robust and hard to mess up. Feel free to experiment with the machine. 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.

2 Helpful Hints

1. Make sure that you are logged in for all of the exercises.

2. Unless we tell you to *cold boot*, we assume that you have *not* cold booted your machine. If you have, make sure that you have logged in.

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. You can press the CLEAR INPUT key to discard anything you've typed on a line, and the REFRESH key to clear the screen.

6. If you see **MORE** at the bottom of your screen, press the SPACE bar to continue the display.

7. In the *Command Processor*, pressing SPACE gives you information, so press it when we tell you to do so or when you do not see what we tell you that you should see.

8. Press the ABORT key to get out of most kinds of trouble.

Section 1

1. Introduction

This section covers the basic, most essential material in the workbook. You are expected to know this material before you come to any class offered by Education Services. If you have been working on a Symbolics computer for more than a couple of months, you might already know this material, but you should review it for any little tricks or hints that you have missed.

Section 1 has five chapters covering basic skills:

- The first chapter, **Getting Started**, teaches you how to work with the console, mouse, and keyboard, and also how to log in, log out, and cold boot your machine.

- The second chapter, **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.

- The third chapter, **Zmacs**, covers elementary file pathname concepts and very basic commands for editing any type of file.

- The fourth chapter, **Getting Familiar with Files**, goes into pathnames in a little more depth and explains basic file and directory commands for use with the Command Processsor.

- The final chapter, **The Document Examiner**, introduces the basic features of the Document Examinter 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 for any Education Services class. Your fluency with this material is essential.

2. Getting Started

2.1 Purpose

This chapter gets you started using the Symbolics console, including the mouse, the keyboard, and the screen. It also tells you how to cold boot a Symbolics machine. Cold-booting makes a machine ready for use. Remember that your Symbolics machine is a standalone computer. The terminal is its console and most work that you do on it affects you alone and not any other user. If you are accustomed to a time-sharing system, this might take some getting used to.

2.2 Contents

- **The Mouse** is an auxiliary input device used for moving from window to window, for pointing at particular locations on the screen, and for entering commands. It has three buttons whose functions change according to the context. A **mouse documentation line** on the bottom of the screen tells you what the mouse will do in the current context.
- **The Screen** is the major output device. It includes a great deal of information in the **status line** and the **mouse documentation line**. Some areas of the screen are **mouse-sensitive**, which means that they show a box when the mouse is moved over them. When you see that box, check the mouse documentation line for what you can do. The screen also shows you one or more program windows.
- **The Keyboard** is not a conventional computer terminal keyboard. There are a number of keys that change the behavior of other keys, like the CONTROL and ESCAPE keys on conventional terminals. There are also keys that allow you to step outside the regular character set (SYMBOL key), that let you stop and start the current process (SUSPEND and RESUME keys), and get assistance from the system (HELP and COMPLETE) keys.
- **The Front End Processor and the Lisp Processor** work together. The Front End Processor (called the FEP) performs much of the "bookkeeping" for the system, while the Lisp Processor is the system's central processing unit (or CPU).
- **Cold-Booting** is the way you get a "pure" Lisp world to work with and how you start up the Lisp processor.
- **Walk-through for Cold-Booting**
- **Logging In** allows you to do some automatic customization of your world and also sets up your access to your mail and files. Unlike time-sharing systems, you don't always need to log in to use the Lisp machine.

- **Walk-through for Logging In**
- **Logging out** leaves a machine free for another user, but does not eliminate all the changes you have made. Only cold-booting does that.
- **Walk-through for Logging Out**
- **Documentation References**

2.3 The Mouse

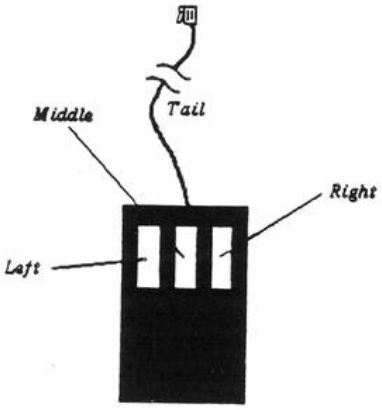


Figure 2.1 The **mouse** *is an adjunct to the keyboard for entering commands and moving the cursor. The action of the buttons changes according to the context you're in.*

The *mouse* is a little black box that is connected to the back of the console. It has three buttons that are known as the left, middle, and right buttons, according to their positions. Try rolling the mouse around on a smooth surface. On the screen, you see the *mouse cursor*, a small black arrow, move along with it. 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) a mouse button either once or twice in rapid succession. There are six possible mouse clicks: Left, Middle, Right, Left twice, Middle twice, and Right twice (L, M, R, L2, M2, and R2, respectively). To get L2, M2, or R2, you can either click the appropriate button twice quickly, or hold down the SHIFT key while clicking the button.

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

2.4 The Screen

The very bottom line of your screen is the *status line* (Figure 2.2). 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 the name of the *current package* (a series of characters followed by a colon, generally `USER:`). To the right of the package name is the *process state*. This is `Tyi` (TYpe In) when the machine is waiting for input. When the machine is busy, other states (for example, `Run` or `Chaos In` replace `Tyi`. Symbolics machines support typeahead, but if you do not know what is going to happen next this can cause problems, so make sure that `Tyi` is showing before you type anything or click a mouse button.

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

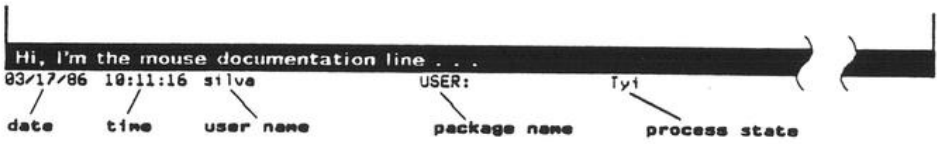


Figure 2.2 The line with the date and time in it is called the **status line.** *It's your best immediate information on what the machine is doing.*

While using your machine, you will notice that there are five *run bars* that regularly appear near (under) `Tyi` (Figure 2.3). These are just small horizontal lines that appear when the machine performs various operations. They are not always visible. There are two bars for *garbage collection*, one for paging, one for processing, and one for disk input/output (I/O). The run bars are located, in that order from left to right, beginning under the current package name (Figure 2.2). If you suspect that your machine is not responding to your input, look at these bars. If they are flickering, your machine is doing something. Move the mouse to see the processing bar flicker.

Just above the status line is a line in reverse video (Figure 2.2). This is the *mouse documentation line.* It tells you, at any time, what happens if you click the mouse buttons. Sometimes this line contains a lot of information, when there are many possible mouse clicks and results. 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. The choices on each menu are *mouse-sensitive*, which means that when you move the mouse cursor over

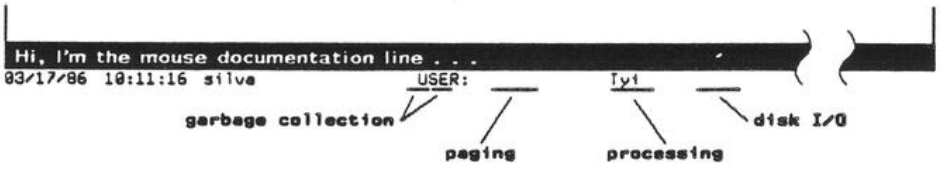


Figure 2.3 Five **run bars** *under the status line tell you what is running. The one marked* processing *tells you the computer is running. The others tell you what else is running (and perhaps slowing you down).*

them, they become enclosed in a rectangle, and initiate some action when you click on one of them.

To see this, first bring up the *System menu* by holding down the SHIFT key while clicking right (see Figure 2.4). Move the mouse cursor over the menu items slowly and notice the changes on the mouse documentation line. Move the mouse cursor off the menu; the menu disappears.

2.5 The Keyboard

The keyboard is composed of a center group of dark keys similar to a traditional typewriter keyboard, a space bar at the bottom, and light-colored keys around the edges. See Figure 2.5.

There are two groups of non-standard keys. One group, called *modifier keys*, are meant to be held down while other keys are pressed. The documentation specifies this type of keystroke as, for example, c-X, which is shorthand for CONTROL-X. This means that you are to hold the CONTROL key down while you press the X key. All the keys (except CAPS LOCK) that are meant to be used in this way have *dark* lettering on them. They are located in the lower left and right corners of your keyboard (Figure 2.5). The modifier keys are SHIFT, SYMBOL, CONTROL, META, SUPER, HYPER, and REPEAT.

The other non-standard keys are used *sequentially* with other keys. These keys have *light* lettering. In the documentation, they look like this: SELECT E. This means you are to press the SELECT key, then release it and press the E key (it does not have to be uppercase E).

In the following sections, the modifier keys are shown in lowercase letters and the keyboard keys that follow modifier keys are in uppercase letters. We are using this convention for clarity and do not mean that you should type letters that follow

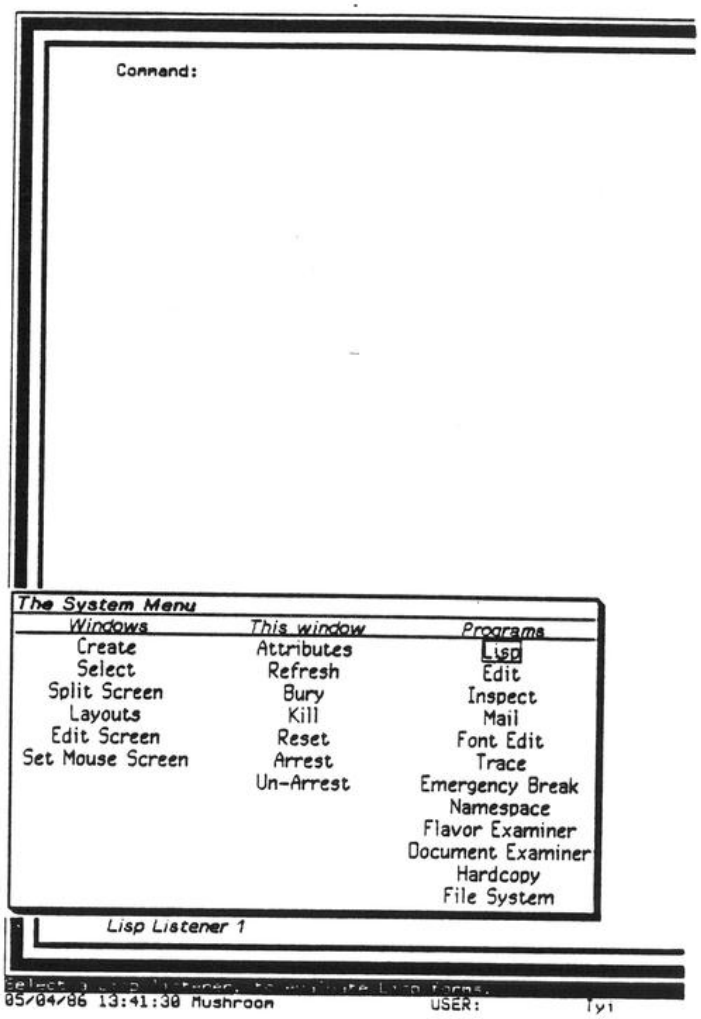


Figure 2.4 What the mouse buttons will do is shown in the **mouse documentation line.** *Left, middle, and right buttons on the mouse cause different actions. The actions (and the mouse documentation line) change according to the context.*

a modifier in capital letters. In fact, if you do type these letters in uppercase, the commands not work. When we want you to use a capital letter, we indicate that by using the abbreviation for the SHIFT key.

Figure 2.5 The **keyboard** *is diff rent from a conventional terminal keyboard. A number of keys – not just the* CONTROL *and* ESCAPE *keys – can cause changes in the behavior of other keys.*

The following abbreviations are used for modifier keys:

c-	CONTROL
m-	META
s-	SUPER
h-	HYPER
sy-	SYMBOL
sh-	SHIFT
c-sh-	CONTROL-SHIFT
m-sh-	META-SHIFT
sy-sh-	SYMBOL-SHIFT
c-m	CONTROL-META

If a screen is left on for a long period of time (several days), the life of the tube is shortened. To avoid this, dim the screen when you are finished working if the machine is not going to be used for some time. To dim the screen, hold down *both* the LOCAL key (this is one exception to the white-lettered key rule) and the D key until the screen is dark. (Some consoles have a brightness knob underneath the main part of the monitor for brightening and dimming, while some consoles do not dim at all.) To brighten the screen, hold down both LOCAL and B until the screen is comfortably bright. It is recommended that you dim the screen rather than turn the console off.

2.6 The Front End Processor (FEP) and the Lisp Processor

Cold booting is how you start up the Lisp Processor (the central processing unit for the Symbolics computer). You cold boot from the Front End Processor (FEP). To get into the FEP, you have to stop the Lisp Processor, using the Halt Machine command. (The walk-through that follows shows how to do this.)

The FEP takes control when the machine is first powered up or whenever the Lisp Processor stops running. The FEP takes care of booting, managing the FEP file system(s), and many other operations.

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:` or `Fep>` (see Figure 2.6. When the Lisp Processor is running, you usually see a prompt like this: `Command:`. One way to tell if the Lisp Processor is running is to look at the clock in the lower left corner of the screen. If the clock is advancing, then the Lisp Processor is running.

Figure 2.6 If the system stops unexpectedly, you'll see one of these prompts from the FEP. Depending on the situation, the Continue, Start, or Boot commands will get your system going again. If you see the prompt on the left, ask your Symbolics representative if your system needs a FEP upgrade.

2.7 Cold booting

When you first sit down at a machine, you want to begin work in a clean computing environment. *Cold booting* the machine provides a clean environment, or *world*. 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 2.7). If you do not see the words `cold-booted`, then the world has been modified by someone's use since the machine 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 that has not been saved in files is lost. 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. (The chapter on **Zmacs** tells you how to save files.) It is therefore a good idea to check with the previous user of the machine before cold booting, if

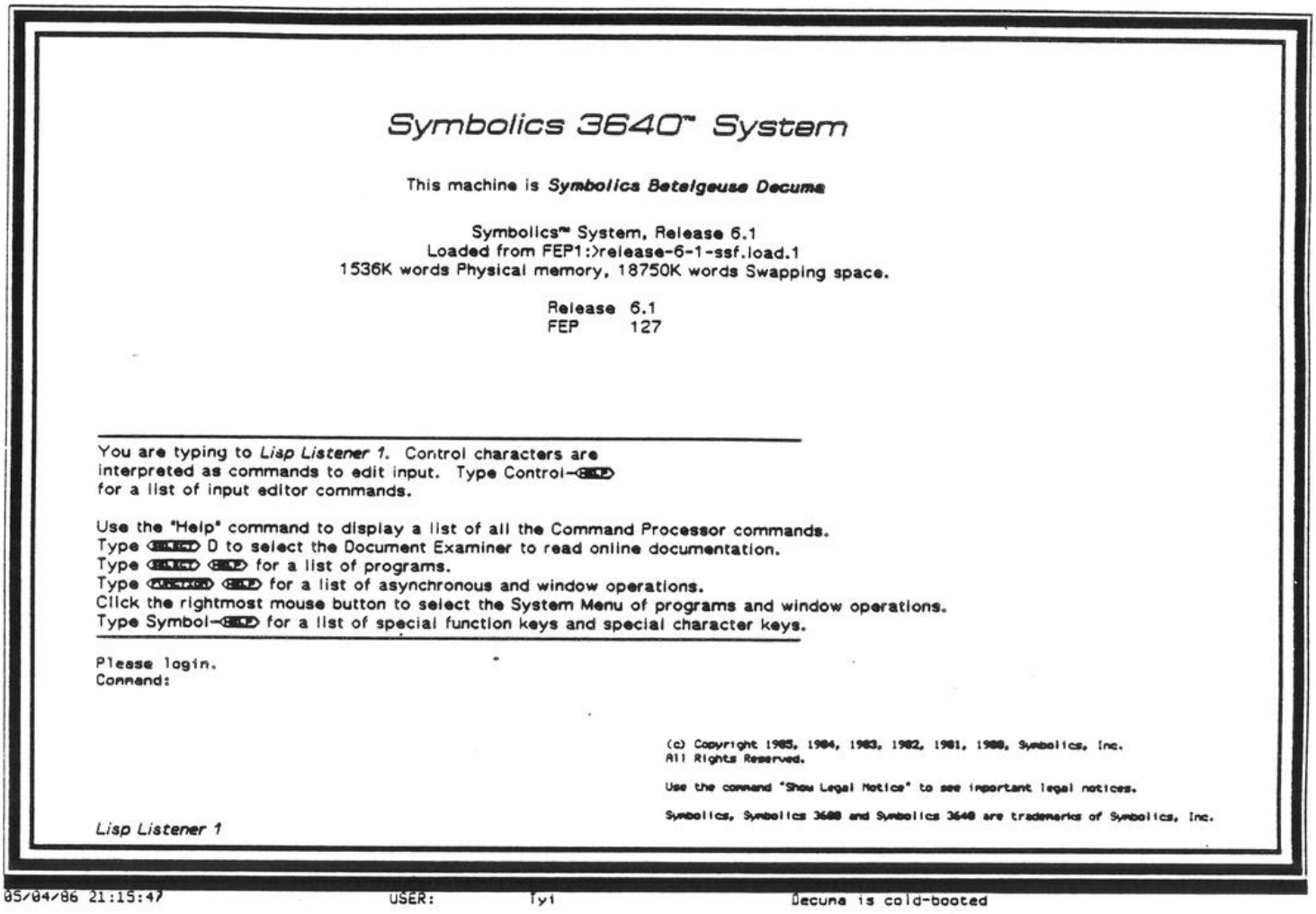


Figure 2.7 The screen of a **cold-booted machine**. *A cold-booted machine has a Lisp world in a "pure" state. As soon as you touch the keyboard, the Lisp world changes. It is a courtesy to others to leave your machine cold-booted when you don't need it.*

there is a login-name in the status line, because that person might need to save his or her work.

2.8 Walk-through for Cold booting

1. Press SELECT L. This puts you into a Lisp Listener window, which is described in more detail later. If you are already in a Lisp Listener, this might put you into another Lisp Listener, or the screen might flash.

2. On the screen you see the following prompt (unless someone using your machine has changed the prompt), with a black, blinking cursor:

   ```
   Command:
   ```

   This is the prompt for the Command Processor. Look at the status line. If there is someone currently logged in to the machine, type

   ```
   Logout<RETURN>
   ```

   to log them out. (The notation <RETURN> used this way in a walk-through means that you should press the RETURN key after typing in a command.) If you see **MORE** at the bottom left of the screen, press the SPACE bar. (**MORE** means that the machine wants to give you more information, but it is waiting so that you can read what it has already displayed. Pressing SPACE makes the machine display another screenful of information.)

3. If you log someone out, you might be asked about saving buffers. If so, you should ask the previous user of the machine if any of their work should be saved, and then enter Y or N. After the Logout command completes, you see the message:

   ```
   Logged out.
   ```

4. When the Command: prompt appears again, type:

   ```
   Halt Machine<RETURN>
   ```

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

   ```
   Lisp stopped itself          or          Lisp stopped itself
   Fep>                                     FEP command:
   ```

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

   ```
   boot<RETURN>
   ```

6. 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 2.8). After a few minutes, the screen displays *Lisp Listener 1* in the lower

left corner, and (your machine's name) is cold-booted in the lower right corner (Figure 2.7).

7. If your machine does not look like Figure 2.7, find your site administrator or someone else who can help you.

8. You are now in the *Lisp Listener 1* window. It has a distinctive multiple border. You are being prompted:

```
Please login.
Command:
```

   Look at the notation in the lower right corner: (your machine's 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?

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

2.9 Logging In

Logging in records your name for file operations and loads a file (your *lispm-init* file) from your *home directory*, which is the directory where you keep your files. This lispm-init file can contain Lisp code that allows you to customize your Lisp environment. If you do not yet have a lispm-init file, you can log in, but the machine notifies you that the file cannot be found.

On a Symbolics computer, your login-name can be any *string* (a collection of characters, like "abc" or "d2r2"), as long as the string does not conflict with any login-name that is already present. Popular login-names are first or last names, initials or nicknames. Note that the machine expects your login-name and the name of your home directory to be the same.

If you have any problems logging in see your site administrator (system manager).

2.10 Walk-through for Logging In

1. Select the Lisp Listener by pressing SELECT L. If you are already in a Lisp Listener, the screen flashes.

2. If the Lisp Listener is not prompting you with:

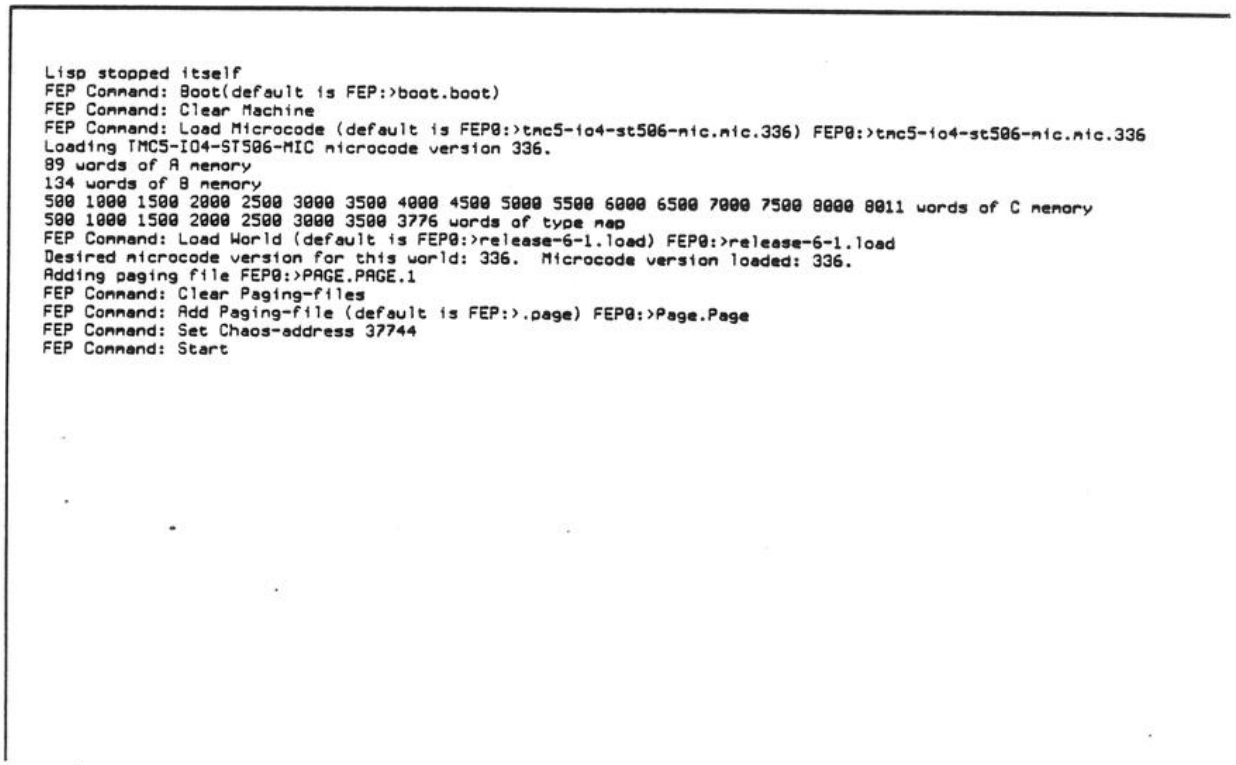


Figure 2.8 The screen of a machine in the middle of **booting**. *The user typed a* Boot *command. The other commands come from a boot file called boot.boot. During booting, the machine reports on its memory and performs other setup operations. The* Start *command is about to execute. The next thing you'll see is a cold-booted system, as in Figure 2.7*

```
Command:
```

press the CLEAR INPUT key to cancel whatever you might have typed. You then get the Command: prompt again unless someone at your site has changed the prompt. If you see the words *Lisp Listener 1* at the lower left of the screen you know 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 2.9).

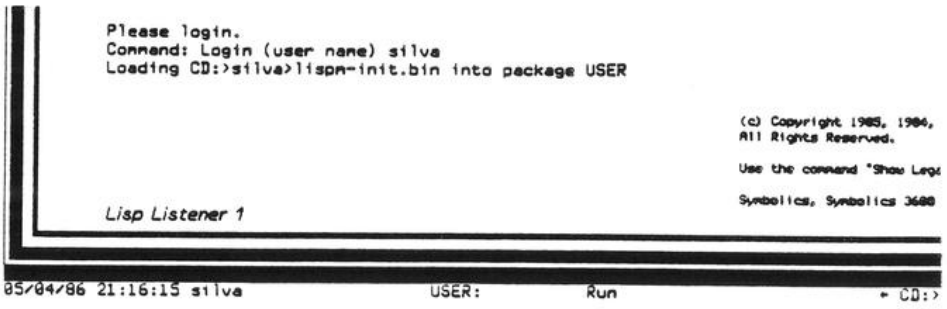


Figure 2.9 **Logging in** *is necessary for getting mail and writing files out to the disk. You have the option of loading an file called a* lispm-init *file (for Lisp-machine initialization) as part of logging in.*

2.11 Logging Out

When you are finished with the machine, you should save, in files, any work you want to keep, and then log out. Logging out informs the machine that you're leaving; it queries you about any mail or editor buffers that have *not* been saved.

2.12 Walk-through for Logging Out

1. Select the Lisp Listener by entering SELECT L. If you are already in a Lisp Listener, the screen flashes.

2. Type

   ```
   Logout<RETURN>
   ```

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

If you share this machine, it is good practice 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.

2.13 Documentation References

- **The Mouse**
  Vol. 1 pages 7-8, 149-150
- **The Screen**
  Vol. 1 pages 5-6
- **The Keyboard**
  Vol. 1 pages 7, 144
- **The Front End Processor and the Lisp Processor**
  Vol. 1 pages 197-199
- **Cold booting**
  Vol. 1 pages 1-3
- **Logging In**
  Vol. 1 pages 1-3
- **Logging out**
  Vol. 1 pages 1-3

3. Getting Around in the System

3.1 Purpose

This chapter introduces you to the Lisp Listener, a window that is a primary means of communicating with your Symbolics computer. The Lisp Listener accepts commands and evaluates Lisp forms. The chapter also introduces the HELP key, which provides different kinds of help depending on the part of the system you are using, and the SELECT key, which allows you to choose different activities, such as editing, mail, and so forth.

3.2 Contents

- **The Lisp Listener** comes in two varieties. The default Lisp Listener is selected with SELECT L and assumes Zetalisp syntax. The Common Lisp Listener is selected with SELECT sy-sh-L (SELECT-λ) and assumes Common Lisp syntax.
- **The HELP Key** provides online help in all sorts of situations. Try it whenever you are in doubt. Right now sy-HELP, SELECT HELP and c-HELP are good combinations to try. Try just plain HELP too.
- **The Command Processor** allows you to issue commands to the machine without having to run Lisp functions. Type various parts of commands and pathnames and press HELP or COMPLETE in the middle just to see what happens.
- **Walk-through for Command Arguments**
- **Walk-through for Getting Help in the Command Processor**
- **Selecting a New Activity** is generally done by pressing the SELECT key in combination with some other key or by using the System menu, which is reached by holding down the SHIFT key and clicking the right button on the mouse.
- **Walk-through for Selecting a New Activity**
- In **Using the Window System**, remember that all activities on the Symbolics machine are executing concurrently, but you are generally only looking at one window at a time.
- **Documentation References**

3.3 The Lisp Listener

When you cold boot your machine, the first screen you see has a *Lisp Listener window* nearly filling it. A window is an area of the screen that is used by programs for input and output. A Lisp Listener is a window that expects you to input *Command Processor commands* or *Lisp forms*. The Command Processor is explained in detail later in this chapter. A Lisp form is a piece of Lisp code. When you input a command or a Lisp form, the Lisp Listener executes it at once.

There are two kinds of Lisp Listeners, the regular Lisp Listener, which accepts Zetalisp code, and the Common Lisp Listener, which accepts Common Lisp code. Both kinds accept identical commands.

3.4 The HELP Key

When you are in need of help in any context, try the HELP key.

Find the HELP key (below the ABORT key at the right edge of the keyboard). Usually, you can press the HELP key to get online documentation designed to help you with what you're doing in any context.

The following is a partial listing of situations in which pressing the HELP key provides context-related assistance. These situations are explained more fully in later sections.

- Pressing either the FUNCTION key or the SELECT key (see 3.2 followed by the HELP key provides on-screen documentation of commands that you execute by using that key.

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

- CONTROL-HELP (c-HELP) provides a listing of Input Editor commands.

- Typing the word Help to the Command Processor prompt gives you a listing of Command Processor commands.

- In the editor (which is discussed in the next chapter), 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, this message is explained.

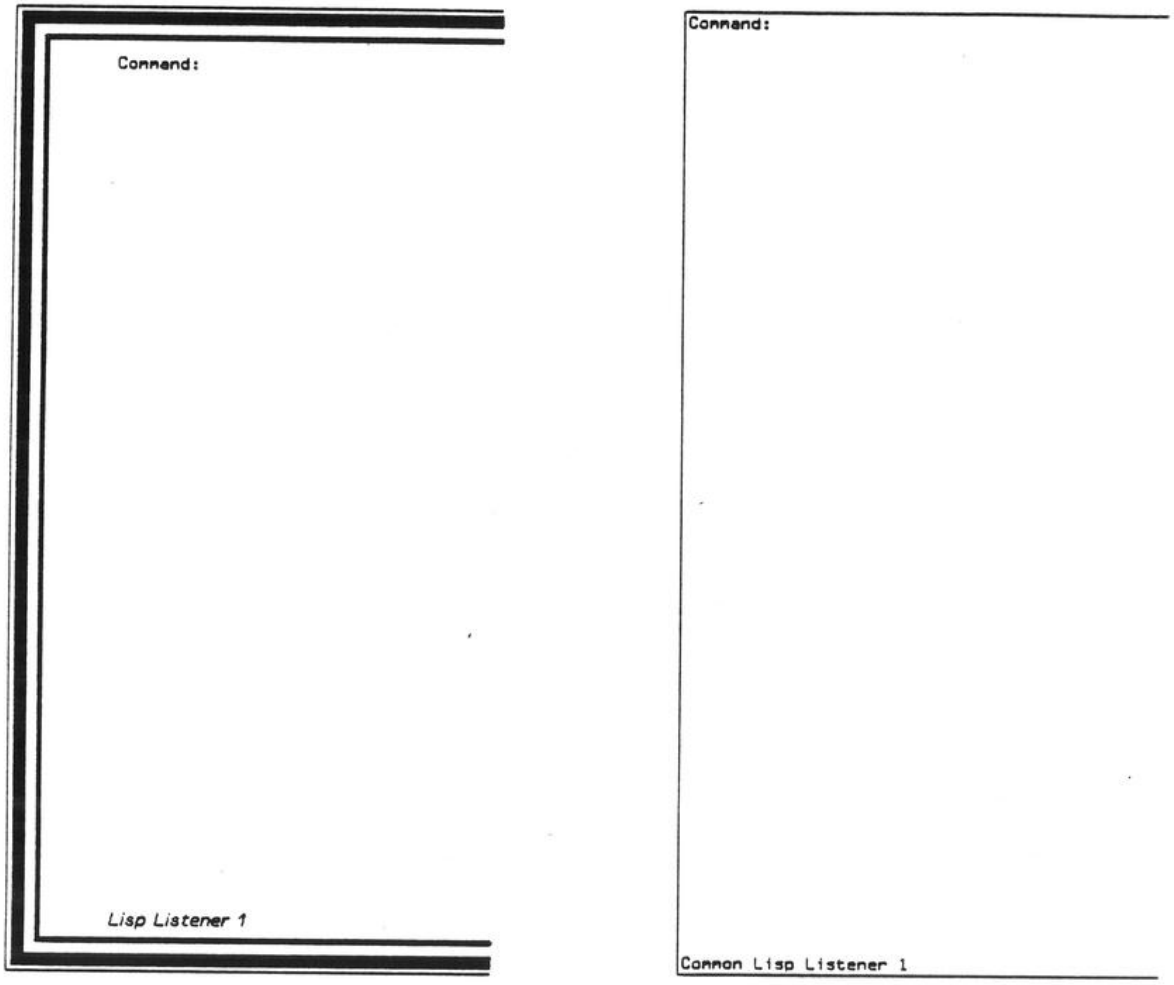


Figure 3.1 The default **Lisp Listener** *on the left assumes Zetalisp syntax; choose it by pressing* SELECT L. *The Common Lisp Listener on the right assumes Common Lisp syntax; choose it by pressing* SELECT SYMBOL-SHIFT-L *(*SELECT λ*).*

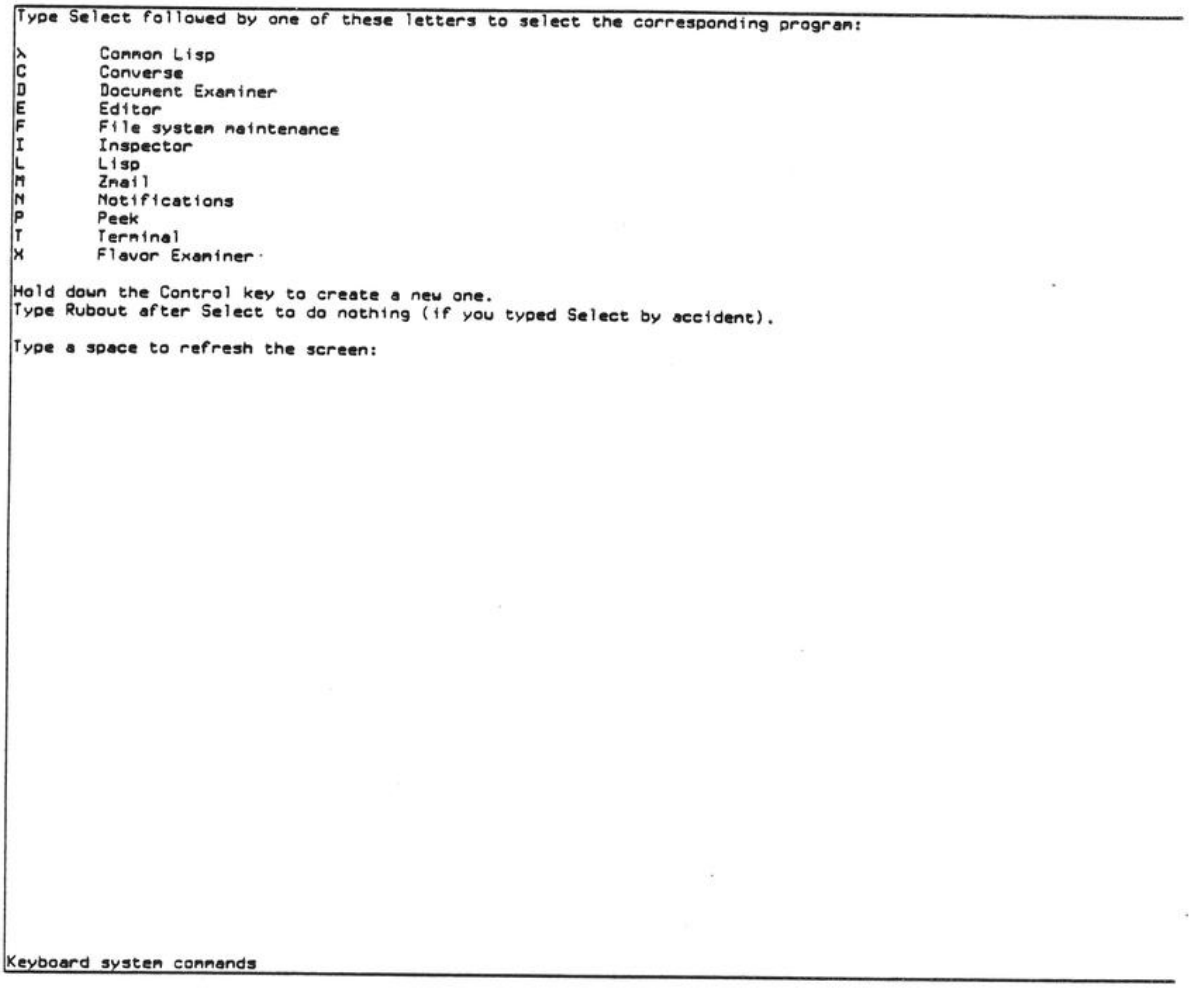


Figure 3.2 Press SELECT HELP *to find out what the* SELECT *key does for you.*

3.5 The Command Processor

The *Command Processor* or *CP* helps you run many functions that are part of the Symbolics operating system. You have already used a few Command Processor commands: Login, Logout, and Halt Machine. Over sixty commands are available.

Parsing commands

A command has three logical parts: A *name*, zero or more *positional arguments*, followed by zero or more *keyword arguments*.

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

```
Show Command Processor Status

Show Legal Notice

Copy File

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 either:

- providing a *default* (which you may or may not elect to use).
- providing a *list of possibilities* that you must choose from.

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 doesn't matter.

A keyword argument always begins with a colon, and it might 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 CP is prompting you with (keywords) by pressing the HELP key.

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

- You can press CLEAR INPUT or ABORT at any time to abort a command and start again.
- You must press RETURN or END to make the CP execute a command.
- Pressing SPACE gets you the default value for an argument.

3.6 Walk-through for Command Arguments

Note that none of these commands is to be executed; they are all for illustration. Do not press RETURN when you type these examples.

1. Press SELECT L to get to the Lisp Listener.
2. Log in (if you haven't already done so).
3. For an example of a command that provides a default, type:

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

4. Press the CLEAR INPUT key to erase what you typed in the previous step.
5. For an example of a command that provides a list of possibilities, type:

   ```
   Edit<SPACE>Namespace<SPACE>Object<SPACE>
   ```

6. Press the CLEAR INPUT key again.
7. Type:

   ```
   Logout<SPACE>
   ```

   Notice that the text right before the cursor says (keywords).
8. Press the HELP key for a description of the optional keyword arguments to the Logout command.
9. 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.
10. Now the text before the cursor looks like:

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

    Press SPACE again to get the default argument for :Save Buffers.
11. 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

Typing:

```
Help<RETURN>
```

to the CP prompt gives you a list of all the possible CP commands.

Pressing the HELP key before you have typed any part of a command gives you useful information about getting help for various things.

You can also press the HELP key to get useful information at various points during the command-issuing process. Refer to the three parts of Figure 3.3 as you read the following three steps.

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

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.

3.7 Walk-through for Getting Help in the Command Processor

1. Type:

   ```
   Help<RETURN>
   ```

   if you haven't done so already, and look at the list of CP commands. Eventually, you will use many of these, but for now you only need to know that they are there.

2. Press the HELP key and read the help information.

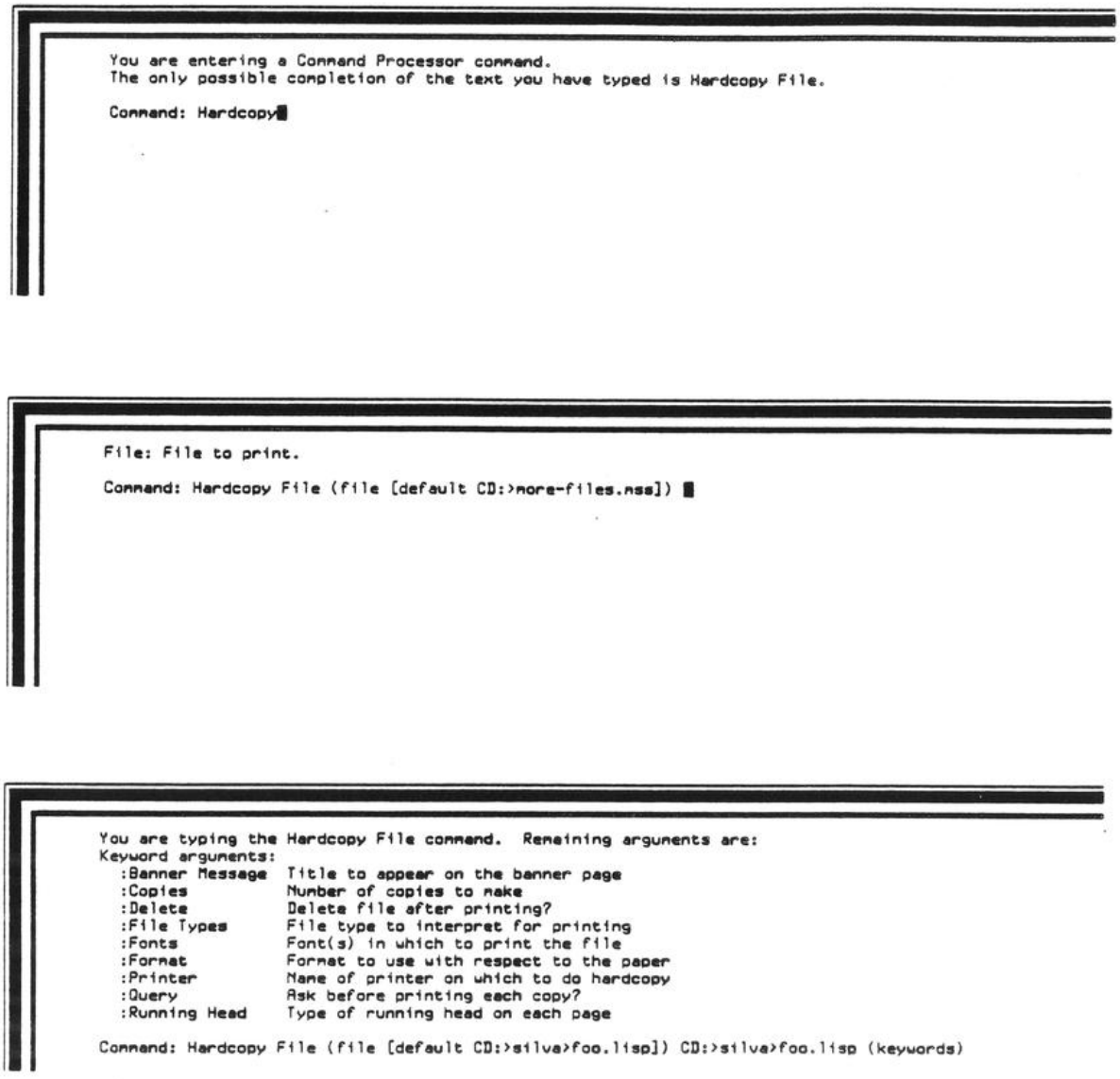


Figure 3.3 The action of the HELP *key varies according to the context. At the top, it tells you how to finish the command. In the middle, it tells you what arguments to give to the command. At the bottom, it tells you what keyword arguments you can give the command. In other contexts, it tells you other things. Try* HELP *whenever you wonder what to do.*

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 HELP key and read the help information.

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

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.

3.8 Selecting a New Activity

System programs on the Symbolics computer 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 look just at using the SELECT key here.

The SELECT key is in the middle of the left-hand side of the keyboard. Here are all the activities available through the SELECT key.

λ	Common Lisp. Common Lisp Listeners use Common Lisp syntax. You obtain the letter λ by typing SYMBOL-SHIFT-L (abbreviated sy-sh-L).
C	Converse. An interactive activity for sending messages to users on other machines.
D	Document Examiner. Lets you read the Symbolics documentation online.
E	Editor. The editor is called Zmacs. Edits text (and program) files and directories.
F	File system maintenance. Lets you do local Lisp Machine File

	System 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	Zetalisp. Lisp Listeners use Zetalisp syntax.
M	Mail. Zmail is a mail reading and sending activity.
N	Notifications. Allows you to see any notifications you have received from the system or from other users.
P	Peek. Lets you monitor and change some of the active parts of the system.
T	Terminal. Allows you to log in to any other machine which has a data connection to your machine.
X	Flavor Examiner. Allows you to browse through flavor information. This is very useful when you are writing programs.

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 3.4).

2. Press one of the above letters.

Figure 3.4 The status line changes from Tyi *to* Select: *after you've pressed the* SELECT *key. If you don't want to select anything, press* SELECT *again.*

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 go back to the Lisp Listener, type SELECT L.

Notice 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 can sit down at your machine, log in, select and use the editor, select the Lisp Listener again, reselect the editor, and the editor is in the same state as when you left it. You can then select Zmail to read your mail, reselect the Lisp Listener, and so on. None of these activities ever "finish" and stop running, they just sit in the background waiting to be selected.

3.9 Walk-through for Selecting a New Activity

1. Start by cold booting your machine (which you learned to do in the previous chapter) to ensure that your machine starts this walk-through in the right condition.

2. Log in. You should always log in when you start using a machine and after you cold boot.

3. Now you can use the Lisp Listener activity to do something. The Lisp Listener should be prompting you with: Command:. Press the REFRESH key. This key is on the top row, left side. The Lisp Listener window is cleared, and you get the Command: prompt again.

4. Type

   Show<SPACE>Font<SPACE>BIGFNT<RETURN>

   If you did this correctly you should see all the characters in the font customarily used (CPTFONT), with the corresponding characters in BIGFNT displayed underneath them. (See Figure 3.5.)

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

6. Press the E key. It takes a few seconds this first time, because the window must be created, and some odd things happen 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 that is not under the editor window is still visible, but it is greyed-over.

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

   ```
   Old MacDonald had a farm
   ```

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

8. Press SELECT L to get back 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>43VXMS<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've progressed this far into this document, you can't 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 SPACE to get rid of the help window. Make sure you get in the habit of doing this. Don't just use the SELECT key to switch to something else when the help display is showing without pressing SPACE first.

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

14. Now, 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.

3.10 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 sending output to their windows even when you cannot see the window on your screen anymore. Then, when you select that activity

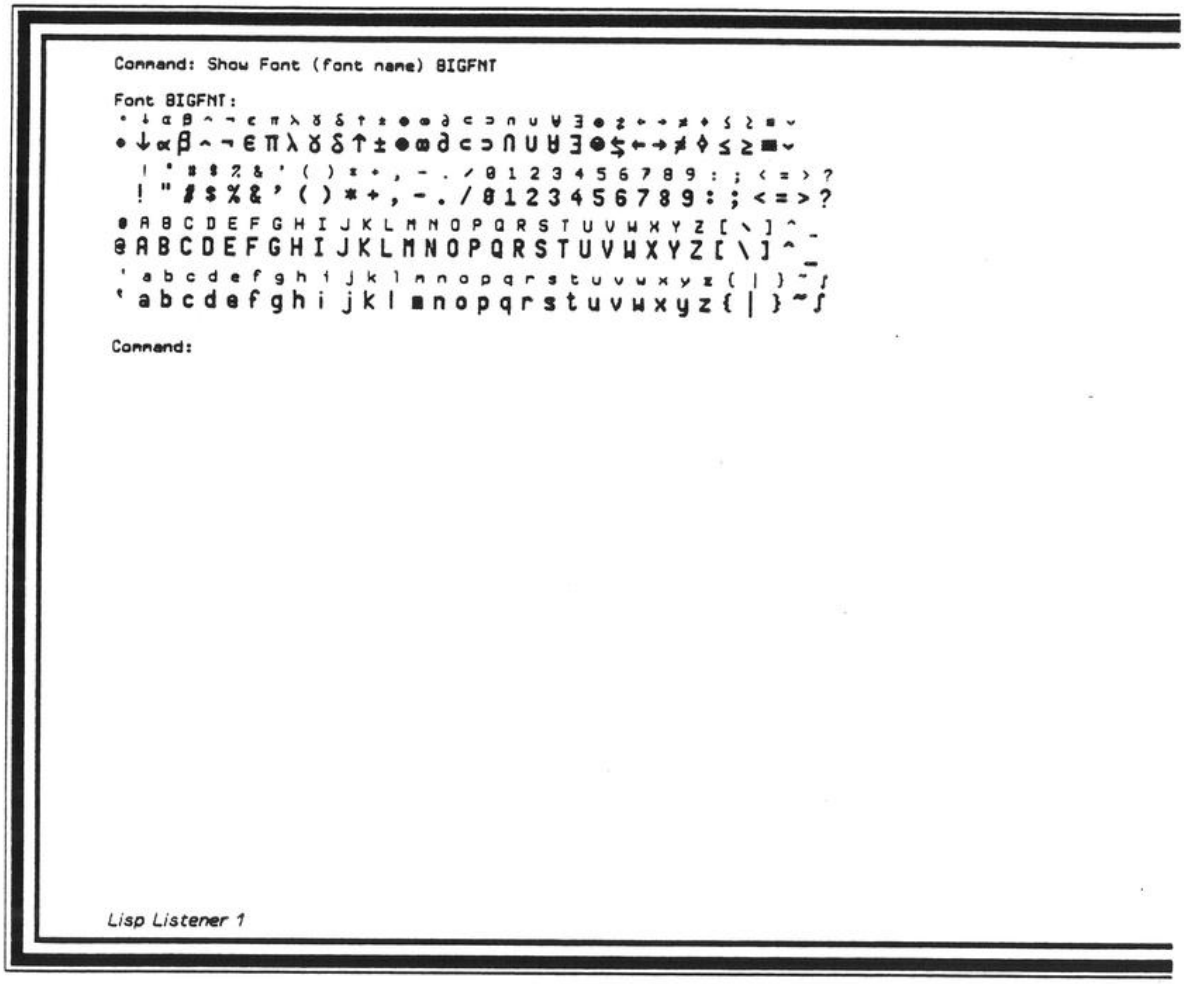


Figure 3.5 The **Show Font** *command displays all the characters in a font. This particular font is called BIGFNT. If you don't know what fonts are available, type Show Font and then press* HELP *for a list.*

again, there is new output on the screen. Some activities give you a notification when they want input or output.

3.11 Documentation References

- **The** HELP **Key**
  Vol. 1 page 83
- **The Command Processor**
  Vol. 1 pages 9-42, 164-167

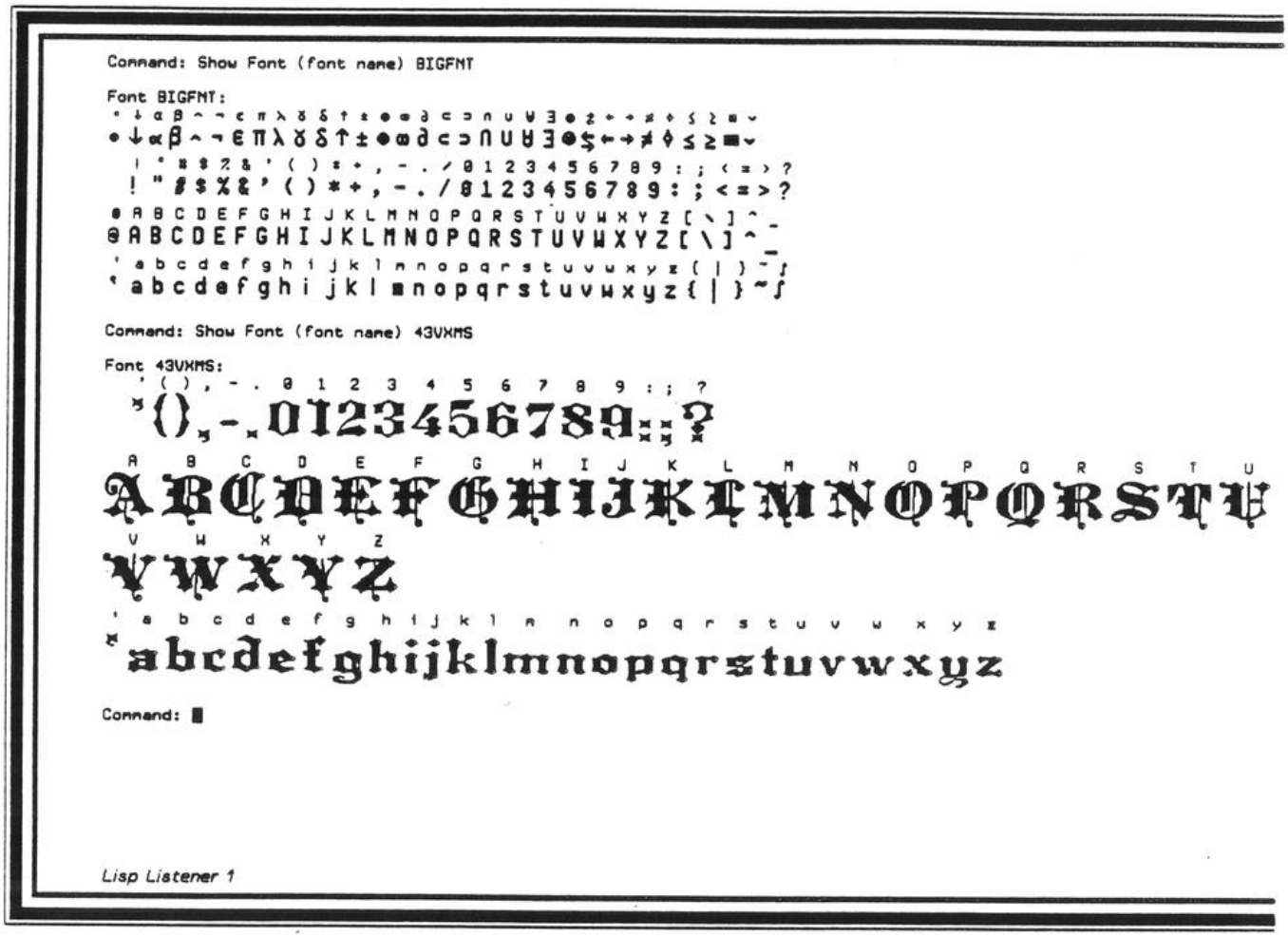


Figure 3.6 Here are two **Show Font** *commands, one after the other, showing BIGFNT and 43VXMS.*

- **Selecting a New Activity**
  Vol. 1 page 145

4. Zmacs

4.1 Purpose

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

4.2 Contents

- **The Zmacs Window** gives you a good deal of information about what you are doing, or can do, in the Zmacs editor. Depending on the file type of the file you are editing, the window lets you do different things. Lisp files and text files have different actions available, for instance.
- **Files and Buffers** are the major units you work with in Zmacs. You can have many buffers open at once and you can access files on your machine or any other machine connected to it.
- **Walk-through for Finding a File**
- **Inserting Text** is the normal action in Zmacs – what you type is inserted into a buffer.
- **Basic Cursor Movement** is done through keyboard commands and the mouse.
- **Walk-through for Inserting Text and Basic Cursor Movement**
- **Deleting and Modifying Text** is less hazardous in Zmacs than with other editors because everything you delete is saved and much of what you modify can be returned to what it was before you modified it.
- **Walk-through for Deleting and Modifying Text**
- **Documentation References**

4.3 The Zmacs Window

Zmacs is a screen-oriented text editor. You can access Zmacs by pressing SELECT E. When you do this, you see a window that has a large area at the top and a small area at the bottom, separated by a single horizontal line (Figure 4.1). 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 CP command positional arguments.) The part of the echo area in which you are prompted for arguments is called the *minibuffer*.

Remember, we use the following abbreviations for combined keystrokes:

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

4.4 Files and Buffers

When you read a file into Zmacs, the text goes into a buffer. A *buffer* is a work area. Buffers are temporary; they go away when the machine 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, many users have a dozen or more buffers available at any one time. 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; you can retrieve those contents at any time.

File Pathnames

To perform any file operation, you must specify the file you want. Each file is identified by a unique *pathname*. A pathname has several *components*. All these components are used to specify a file, but due to pathname *defaulting* (which are discussed in another section), you rarely have to type all of them.

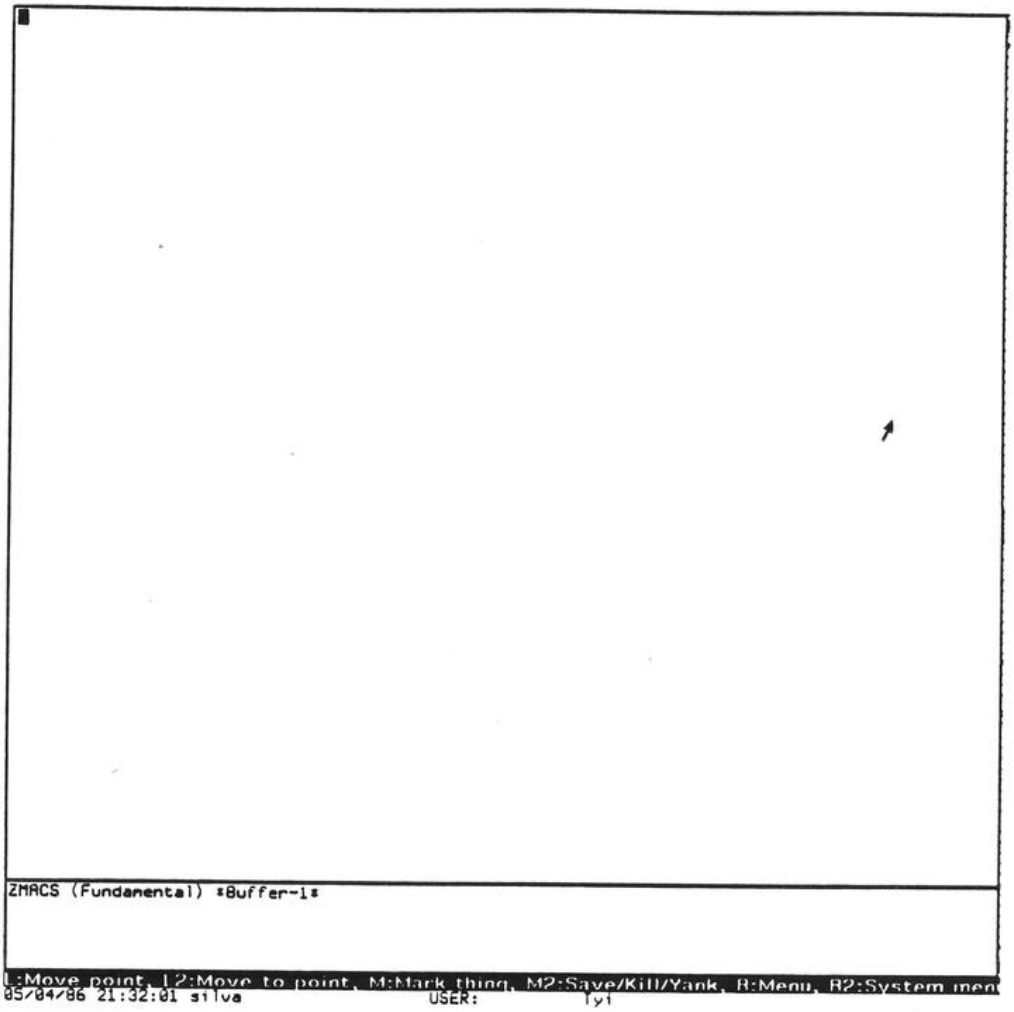


Figure 4.1 The initial **Zmacs window**. *Zmacs windows change depending on the type of file you are working on, but this is the first window Zmacs shows. Press* SELECT E *on a cold-booted machine, and this is what you see.*

The pathname components you need to recognize are:

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). The 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 Symbolics system, these components are grouped together into a pathname according to the following pattern:

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

Finding and Saving Files

All the prompts and defaults appear in the Zmacs minibuffer at the bottom of the window (Figure 4.2).

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.

```
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 no file is 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.

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

4.5 Walk-through for Finding a File

Before you begin this walk-through, find out the name of your file server if you don't already know it. You can then type in this name when you see *your-file-server* in the examples in this and other walk-throughs.

1. Press SELECT E to select the editor. You should be in *Buffer-1*, with the line of text:

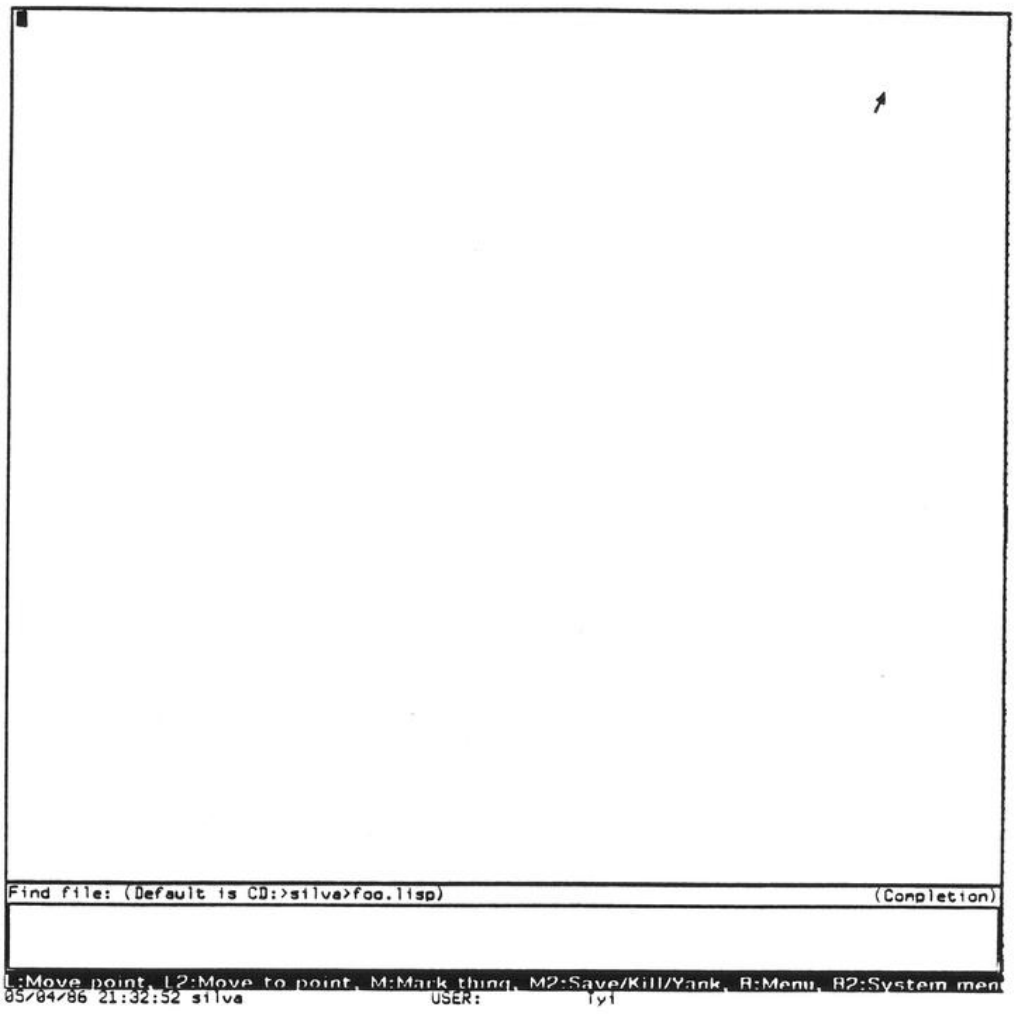


Figure 4.2 When you give a command to Zmacs, prompts, defaults, and the commands you type all appear in the **minibuffer** *at the bottom of the screen. The minibuffer is a separate window.*

```
Old MacDonald had a farm
```

still there from the last walk-through.

2. Press `c-X c-F`.

3. In the echo area, you should see the words `Find File` and a default pathname. 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>

This file does not exist, so Zmacs should create an empty buffer with this pathname as its name. Note that *Buffer-1* has not disappeared – it is no longer visible to you, but you could reselect it. Zmacs accumulates buffers, unless you explicitly delete one. Now you are ready to begin editing text. The remaining sections of this chapter tell you how to do so.

4.6 Inserting Text

Whenever 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.

4.7 Basic Cursor Movement

Cursor movement commands move the cursor from its current position to a place that is some logical unit of text away. Some common units are characters, words, lines, sentences, and Lisp forms. You cannot move around in an empty buffer, as there are no units of text to operate on.

Figure 4.3 gives a summary of simple cursor movement commands.

The Mouse

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

Do not click on the echo area with the mouse, however. The only time the cursor should be in the echo area is when you are typing in a command argument or an extended command. If the cursor does end up in the echo area at the wrong time, press c-m-ABORT, and the cursor returns to its proper place.

Movement

Forward

c-F	Moves cursor forward one character.

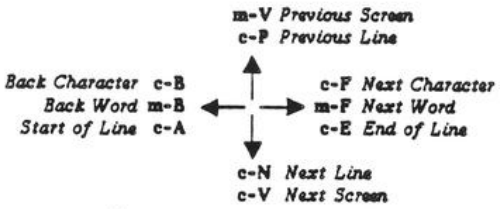


Figure 4.3 Here is a summary of some important **Zmacs movement commands**. *You can move by character, line, word, or screen. Copy this and keep it next to you while you edit. See Figure 4.4 for more movement commands.*

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

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

Note that some of 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 string of characters you are looking for (you are prompted for this string in the minibuffer), the cursor moves further along in the buffer. You can search both forward and backward.

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 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. However, you might find the before you enter the entire string.

If the occurrence of the string that you find is not the one you want, just type the search command again, and the cursor moves to the next (or previous) occurrence of the string. 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. (To return to where you started, press c-G.)

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.

4.8 Walk-through for Inserting Text and Basic Cursor Movement

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, wo'n't you, will you, wo'n'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), press a.

14. Now press 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 which 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.

4.9 Deleting and Modifying Text

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

A useful part of the window system is the *global kill ring*, where text you delete (words, lines, paragraphs, and so on – everything except characters) is stored. 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 something in one buffer and now you want to save it in a file associated with another buffer, you can delete it in the first buffer and yank it back in the second.

Figure 4.4 summarizes movement and deletion commands by the units on which they operate.

	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 4.4 Here is a summary of the major **Zmacs movement, deletion and transposition commands.** *The commands for Lisp forms work in all buffers, not just Lisp buffers. Copy this and keep it next to you while you edit. See Figure 4.3 for a different summary of movement commands.*

Deleting commands

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. *Query replace* waits at each occurrence of the string being replaced for you to tell it whether or not to replace the string.

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 do the replace or the RUBOUT key to

skip to the next candidate. (Other actions are possible; press HELP to see them.)

m-sh-Z<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 (or containing) the cursor with the word preceding the cursor.
c-X c-T	Transposes the line the cursor is over with the preceding line.

4.10 Walk-through for Deleting and Modifying Text

1. Go to the top of the buffer with m-sh-<.

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

   will<RETURN>

   When you are prompted for the string to replace will with, type:

   would<RETURN>

   Notice that all occurrences of will in the buffer are changed to would. 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.)

3. Now press m-sh-Z. Type:

   would<RETURN>will<RETURN>

   for the command arguments.

4. 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 text you killed in large chunks appears.

13. Press c-N.

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

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

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

17. Press m-B.

18. Press c-T.

19. Press c-T again.

20. Press c-A.

21. Press m-T.

22. Press m-T again.

23. Make any changes you care to make in the text.

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

25. 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 will tell you (No changes need to be written.)

26. 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.

27. 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; it simply puts you into the buffer you already have associated with that file.

4.11 Documentation References

- **The Zmacs Window**
  Vol. 3 pages 17-21
- **Files and Buffers**
  Vol. 3 pages 30-32, 114-128
  Vol. 1 pages 47-55
- **Inserting Text**
  Vol. 3 pages 22-23
- **Basic Cursor Movement**
  Vol. 3 pages 26-27, 57-59, 60-64, 98-112
  Vol. 1 pages 149-150
- **Deleting and Modifying Text**
  Vol. 3 pages 28-29, 72-83

5. Getting Familiar with Files

5.1 Purpose

This chapter introduces files and file handling. It covers file pathnames and the system of defaults used on the Symbolics machine. The chapter also explains commands for common file operations, such as Edit File, Delete File, Undelete File, Show File, Create Directory, Show Directory, and so forth.

5.2 Contents

- **The File System** is in two parts. The FEP file system always resides on your machine's disk and controls all your access to the machine. One file in particular, called `lmfs.file`, controls the organization of all your files into directories, subdirectories, and files. FEP files and files in directories are quite different.
- **Pathnames** tell the system how and where to find a file. Pathname defaults and wildcards give you considerable ease and flexibility in using the file system.
- **File and Directory Operations in the Command Processor** as listed in the **Purpose** section above give you the means of storing files, editing them, reading them, copying them, deleting and undeleting them, and expunging them. Notice that deleting a file on a Symbolics file server does not eliminate it entirely. You must expunge a file to get rid of it entirely.
- The **Walk-through for File and Directory Operations** shows only one possible method – the Command Processor way – of dealing with files and directories. Two other methods are introduced in Chapter 15.
- **Documentation References**

5.3 The File System

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 friends, to name a few.

On the Symbolics computer, there are two kinds of files, *FEP files* and *LMFS files*. One of the FEP files is called *lmfs.file*. *All* the LMFS files are stored in this file. LMFS files are the user and the Lisp system source files. The **Overview of the**

Machine chapter discusses FEP files; this chapter shows you how to use Lisp Machine File System (LMFS) files.

Within lmfs.file, your files are organized into *directories*. Within a directory, files can again be organized into *subdirectories* (Figure 5.1). 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).

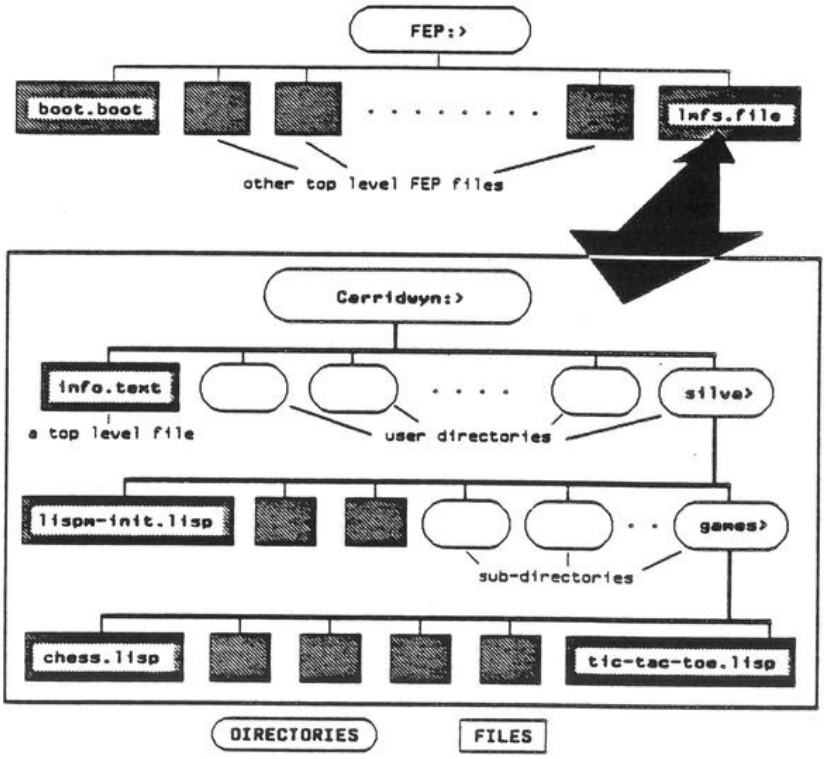


Figure 5.1 Here is a layout of the **file system structure**. *Notice that the conventional file system structure of directories, subdirectories, and files is all part of another file system structure called FEP files and that all your files are controlled by a FEP file called the* `lmfs.file`. *The acronym* `LMFS` *means Lisp Machine File System. Don't make the common rookie mistake of deleting your LMFS (pronounced "lim-fuss") file.*

5.4 Pathnames

In the Zmacs chapter, you learned about the components of a pathname. 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 your pathname that match those of the default pathname. See Figure 5.2. If the default pathname is:

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

and you want the file:

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

all you have to type for the pathname is:

```
new-hack
```

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

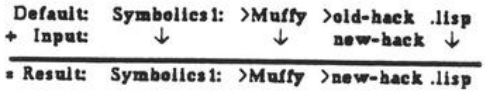


Figure 5.2 Here is a simple illustration of **pathname defaulting**. *The user specified only a new file name –* new-hack. *All the rest of the pathname was provided by the system, using defaults.*

See Figure 5.3. If, instead, you want the file:

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

you have to type:

```
>Frank>interesting-program
```

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

See Figure 5.4. If you want a file from a subdirectory, such as:

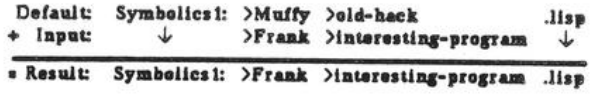


Figure 5.3 In this example, the user has changed several parts of the pathname, and retained **defaults** *for the rest.*

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

given the default:

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

you type:

```
star>resume.text
```

Notice that there is no > before the directory *star*. This says that the directory *star* is a subdirectory of *Muffy*, rather than being a directory at the same level, the way *Frank* was (see Figure 5.4. 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.

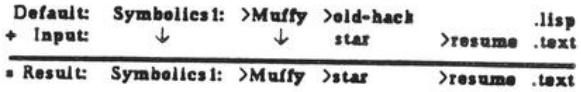


Figure 5.4 In this example, the user has changed only a single part of the pathname – the subdirectory – and used **defaults** *for the rest.*

Pathnames can include file version numbers. If no version number is specified, the version of a file defaults to the most recent version (however, some file systems do not have version numbers at all). In general, you do not want to specify a version of a file. However, you can specify a version number if you do not want the most recent version of a file. It is not a good idea to specify a version number when you are editing a file. If you do so, then try to save the file, the editor will try to save the file with that specific version number, thus trying to write over an existing file and causing an error.

Wildcards

A pathname can contain asterisks (*), which 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 SEATTLE, you type:

```
SEATTLE:>**>*.*.*
```

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

```
SEATTLE:>**>*.text.*
```

5.5 File and Directory Operations in the Command Processor

You can use several programs to perform different operations on files and directories. We're only going to talk about the basic Command Processor commands for handling files and directories. Later chapters discuss other ways of operating on files and directories.

As you know, CP commands are typed wherever you see the `Command:` prompt, usually at a Lisp Listener window. They generally take keywords; 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 defaults to the last pathname you specified in the Command Processor.

File operations

`Edit File` *<pathname>*

Automatically selects the Zmacs window, creates a new buffer for the specified file, and brings that file into the buffer (Figure 5.5). 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.

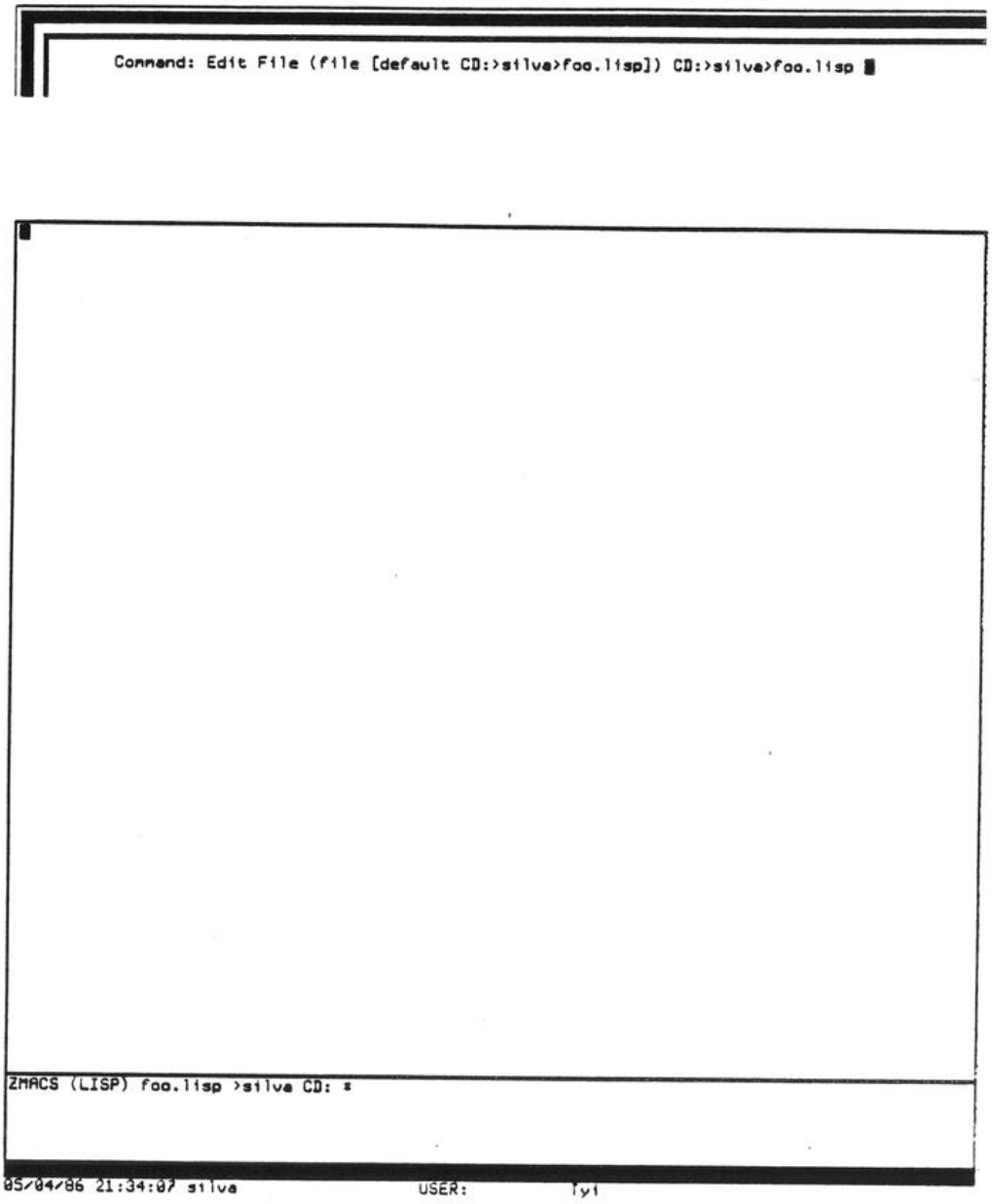


Figure 5.5 The user issued an **Edit File** *command to the Lisp Listener as shown at the top. This invoked the Zmacs editor and caused the editor to find the file and read it in, as shown at the bottom.*

`Delete File` *<pathname>*

`Undelete File` *<pathname>*

If your file server is a Symbolics machine, a file that 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 Delete File and Undelete File offer a default pathname and take a 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 and just press RETURN, the default is *Yes*, meaning "ask before deleting or undeleting each file". You are then 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, if you want to delete the directory SYMBOLICS1:>useless>, you type:

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

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

Show File *<pathname>*

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

Directory operations

Create Directory *<pathname>*

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

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.

Expunge Directory *<pathname>*

Expunges the directory specified, causing all the files marked for deletion in that directory to disappear permanently.

Show Directory *<pathname>*

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

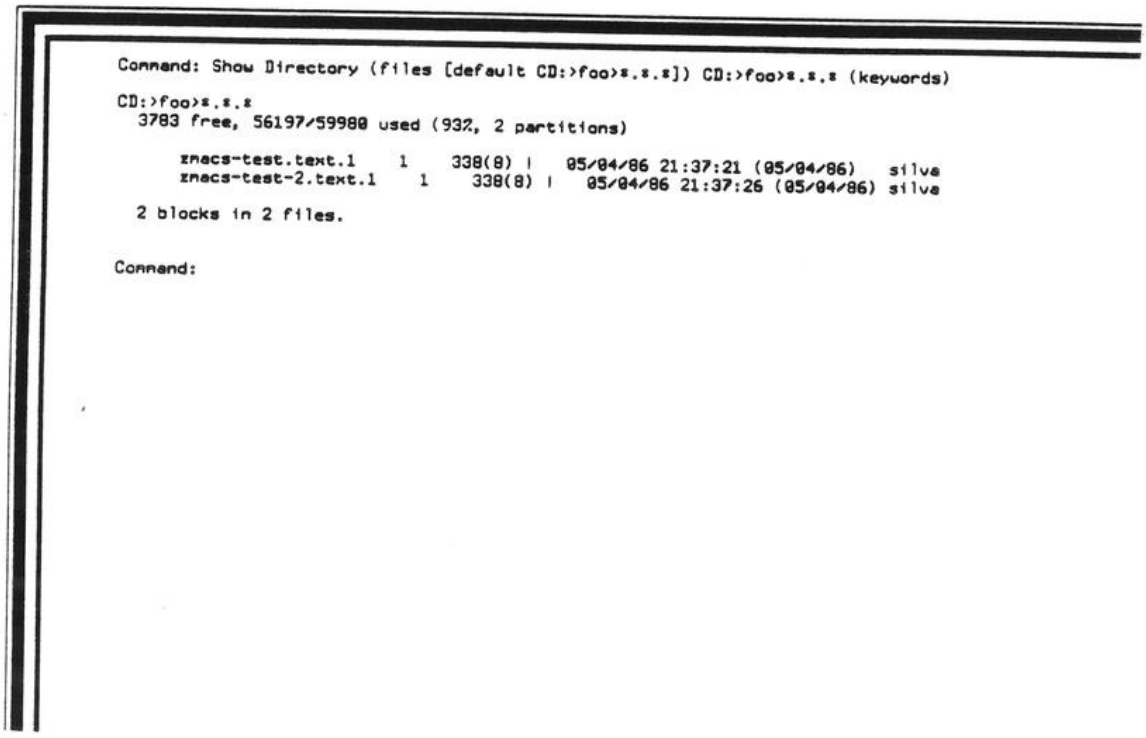


Figure 5.6 The **Show Directory** *command is issued at the Lisp Listener. The command uses * wildcards as used on many computer operating systems.*

5.6 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 not, type in that pathname; otherwise, press SPACE.

4. Press <RETURN>.

5. A listing of the files in your directory appears (Figure 5.6). Notice the files zmacs-test.text and zmacs-test-2.text, which 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, as above. Notice that the listing for the file zmacs-test-2.text now has a D before the file name. This indicates that the file has been deleted.

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.)

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 must 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.

5.7 Documentation References

- **Pathnames**
  Vol. 5 pages 127-129
- **File Operations**
  Vol. 1 pages 16-41
- **Directory Operations**
  Vol. 1 pages 16-41

6. The Document Examiner

6.1 Purpose

Symbolics documentation is all available online through the Document Examiner. This chapter introduces the Document Examiner and teaches you some easy methods of using the Document Examiner to search through the documentation database for information. You can read documentation, print it, jump from topic to topic, and survey all the documentation for information.

6.2 Contents

- **Overview**
- **The Document Examiner Frame** is reached by pressing SELECT D. The initial frame shows a list of the books in the Symbolics document set and several commands. Move the mouse around and see what happens. Press HELP or click on the [Help] item in the command menu. Much of what is displayed in the Document Examiner, including parts of the text, is mouse-sensitive.
- **Basic Document Examiner Commands** include help *via* the HELP key or [Help] menu item and the [Find] menu item, which allows you to search for topics in the database by supplying all or part of a topic name. [Find] provides a new list of possibilities in the Candidates pane in the upper right corner of the frame.
- **Walk-through for Basic Document Examiner Commands**
- **More Document Examiner Commands** include the [Show] menu item, which displays either a particular piece of documentation, an overview of that piece, or a table of contents to show where it fits in; and the [Select] menu item, which allows you to go back to previous sets of selections.
- **Walk-through for More Document Examiner Commands**
- **Bookmarks** are inserted in the Bookmarks pane in the lower right corner of the frame each time you look at a topic. You can also add to the bookmarks list by clicking on the middle mouse button while holding down the SHIFT key. Once a topic has a bookmark, you can go back to it easily by clicking.
- **Documentation References**

6.3 Overview

The 11-volume Symbolics documentation set also resides online, where it can be easily accessed via a program called the *Document Examiner*. The Document Examiner allows you to:

- Search for specific topics or associated ones.
- Learn where in the bound documentation a topic is discussed.
- 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 return to the original topic via an automatic bookmark.

Note: If you cannot find information about a given topic or if you get an error message while trying to read a topic into the Document Examiner, it might mean that not all the Document Examiner files have been installed at your site. Check with your site administrator.

6.4 The Document Examiner Frame

Select the Document Examiner by typing `SELECT D` (if you haven't done so already). You see the *Document Examiner frame*, which is divided into four *panes* (Figure 6.1):

- **Default Viewer** - When you request documentation on a particular topic, the text is displayed in the *Default Viewer* pane. It is the largest pane of the Document Examiner and is initially blank, except for a suggestion to use the `<HELP>` key.
- **Current Candidates** - When you ask the 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 have the text brought into the viewer. When you first enter the Document Examiner, the Current Candidates pane contains the title of each volume in the Symbolics documentation set.
- **Bookmarks** - Every time the text of a topic is brought into a viewer, the section name is listed in the *Bookmarks* pane. You can always click left on one of these *bookmarks* to have a topic redisplayed.

- **Commands** - When you want the Document Examiner to perform some action, type a command in the *Commands* pane (all 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 very useful partner to the Document Examiner program. Be certain to always look at the mouse documentation line before clicking, in order to learn what effect a left, middle, or right click will have.

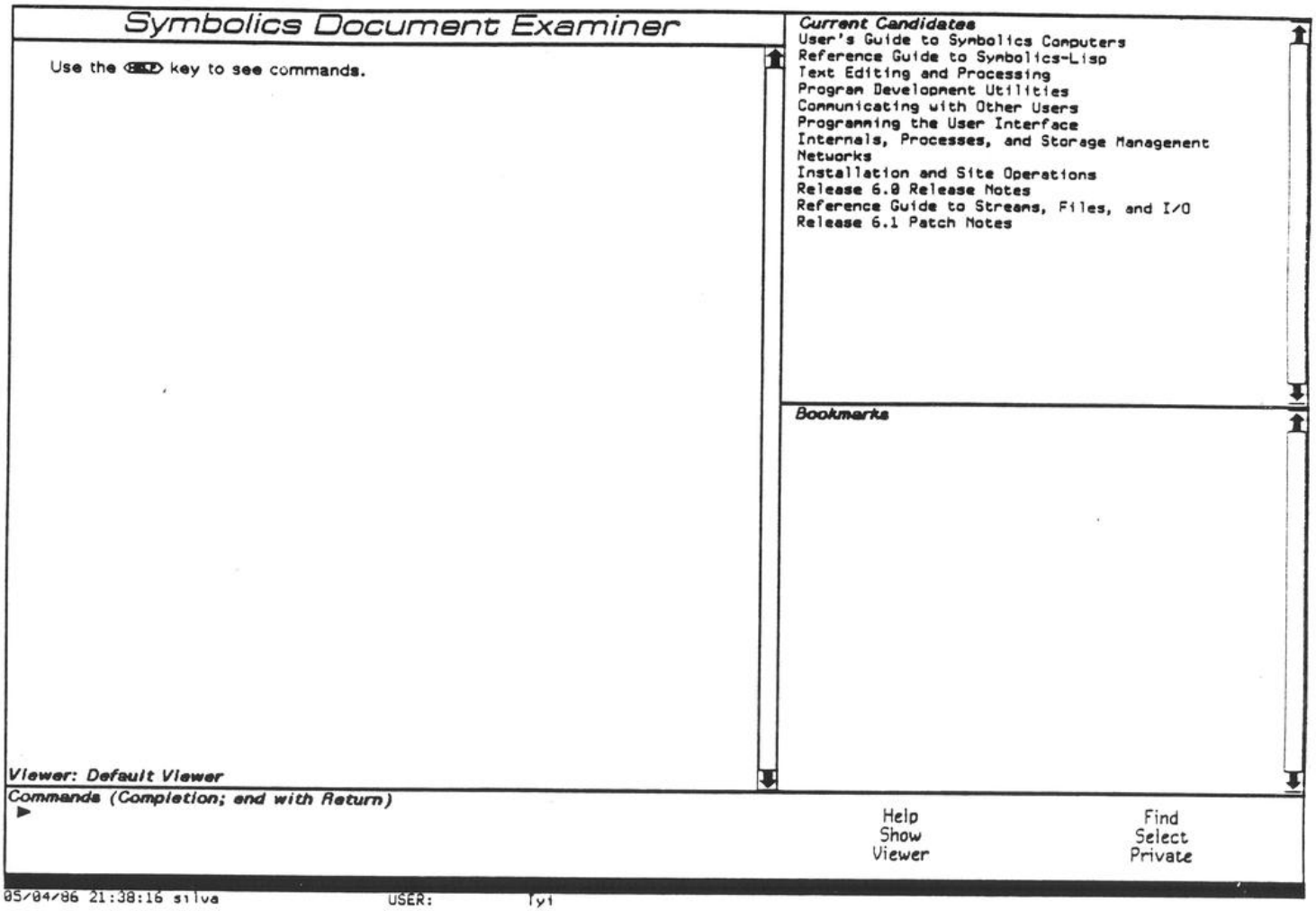


Figure 6.1 The first **Document Examiner** *frame. You get this by pressing* SELECT D. *Do it now and move the mouse around the screen. When you see a box, look at the mouse documentation line and click to do something. Or press* HELP.

6.5 Basic Document Examiner Commands

- [Help] - You can get help either by clicking left on [Help] in the Commands pane or, as the note in the viewer says, by pressing the HELP key. It takes some time to generate the help display. These two ways of getting help list the various Document Examiner commands, including ways of scrolling text via the keyboard. If you click middle on the [Help] option in the Commands pane, a section of text explaining the Document Examiner in detail appears in the Viewer.

- [Find] - You can locate all of the sections that discuss a particular topic by searching with [Find]. [Find] generates a list of *candidates* (sections of documentation with that topic in the title or that topic in their discussion) that appears in the Current Candidates pane. The walk-through exercise describes the effects of various mouse-clicks on how [Find] searches for documentation regarding a particular topic.

  If you move the mouse over one of the choices in the Current Candidates pane and read the mouse documentation line, you see that not only can you read that topic into the viewer, but you can also show an overview of the topic or put in a bookmark (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). You can perform operations on these mouse-sensitive topics exactly as though they were candidate choices.

6.6 Walk-through for Basic Document Examiner Commands

1. Enter the Document Examiner (SELECT D).

2. Press the HELP key.

3. Look at the commands that don't initially fit in the viewer by using c-SCROLL to move downward a few lines at a time. Watch the viewer's *scrolling bar*, adjacent to its right margin, move too. The scrolling bar is discussed in the chapter *More Document Examiner*.

4. Try c-m-SCROLL to move backward a few lines at a time.

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 [Find] command in the Commands Pane and look at the mouse documentation line.

   The descriptions you see define string patterns that [Find] searches for to generate a list of candidates. Clicking left ("XXXXX") causes [Find] to prompt you for the exact topic-string to look for. Clicking middle ("XXX.....") causes [Find] to search for candidates that contain topic-strings *beginning* with the string you type. Clicking right ("..XXX..") causes [Find] to list candidates with *any* topic-string containing the string you type.

   For example, typing `fix` after clicking left would list candidates referring to "fix". A middle click would include any references to "fixed", "fixes", "fixing", or "fixnum". A right click would add candidates referring to "prefix" or "prefixing".

7. Click left on [Find] and enter the topic:

   ```
   yank<RETURN>
   ```

8. Note that there are fourteen possible candidates, four of which don't contain the word `yank` in their names. `Yank` has been designated as a *keyword* (discussed later in this chapter) to these topics, because it is relevant to them even though it is not contained within their names.

9. Click right on [Find] and take the default by pressing `RETURN`. Notice that additional candidates, most having `yanking` in their name, are found; [Find] is matching *any* string containing the topic you typed.

10. Click middle on [Find] and take the default by pressing `RETURN`. Notice that the same candidates are found; no candidates from the previous list had a prefix before the `yank` in their names.

11. 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 line: you can read a topic into the viewer, show an overview, or put in a bookmark.

12. Move the mouse over `Introduction: New Yank System`. Click right to see a menu of the operations you can perform on this candidate. Click left on [Put this Topic in Viewer]. At this point, reading about *Yanking* would be a good idea; it's easy to use and saves you from having to retype text.

6.7 More Document Examiner Commands

- [Select] - Clicking left on [Select] allows you to choose, via a menu, any list of candidates. The reactivated list of candidates you choose appears in the Current Candidates pane.

- [Show] - Clicking left on [Show] brings a topic directly into the Viewer pane without first listing candidates in the Current Candidates pane. To use this command, you should know the exact topic you are looking for.

  Clicking middle on [Show] allows you to see the overview of a topic that you type in (Figure 6.2). The overview pops up in the Viewer pane, describing the topic's place in the documentation. The section, parent topic, and document volume in which it appears are given, as well as a list of *keywords* (other topics that, when searched for, will include this topic in their list of candidates). Below the description is a map of where this topic appears in relation to other topics in the same chapter. These other topics are mouse-sensitive as well. You press the SPACE key 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.

  Clicking right on [Show] displays a table of contents for a topic (Figure 6.3. Titles of documentation volumes, as well as subheadings within each volume, can produce a table of contents.

6.8 Walk-through for More Document Examiner Commands

1. Let's look at the first fourteen candidates again. (If you don't still have the candidates lists from the last walk-through, make some [Find] queries using yank again.) Click left on [Select]. Notice that all queries you entered are available for viewing. You can also select `Table of contents for the document set`, the original candidate list. Click left on `Whole word match using "yank"`.

2. Move the mouse up to the candidates list and click middle on `Introduction to Yanking` to show an overview of the topic. Move the mouse to the default viewer; each subtopic of the overview is mouse-sensitive. The overview also lists references to the location of the current topic in the documentation set.

3. Move the mouse over the subtopic `Introduction to Yanking` in the overview and click left to put this topic in the viewer. Now, information about *Yanking Commands* is listed.

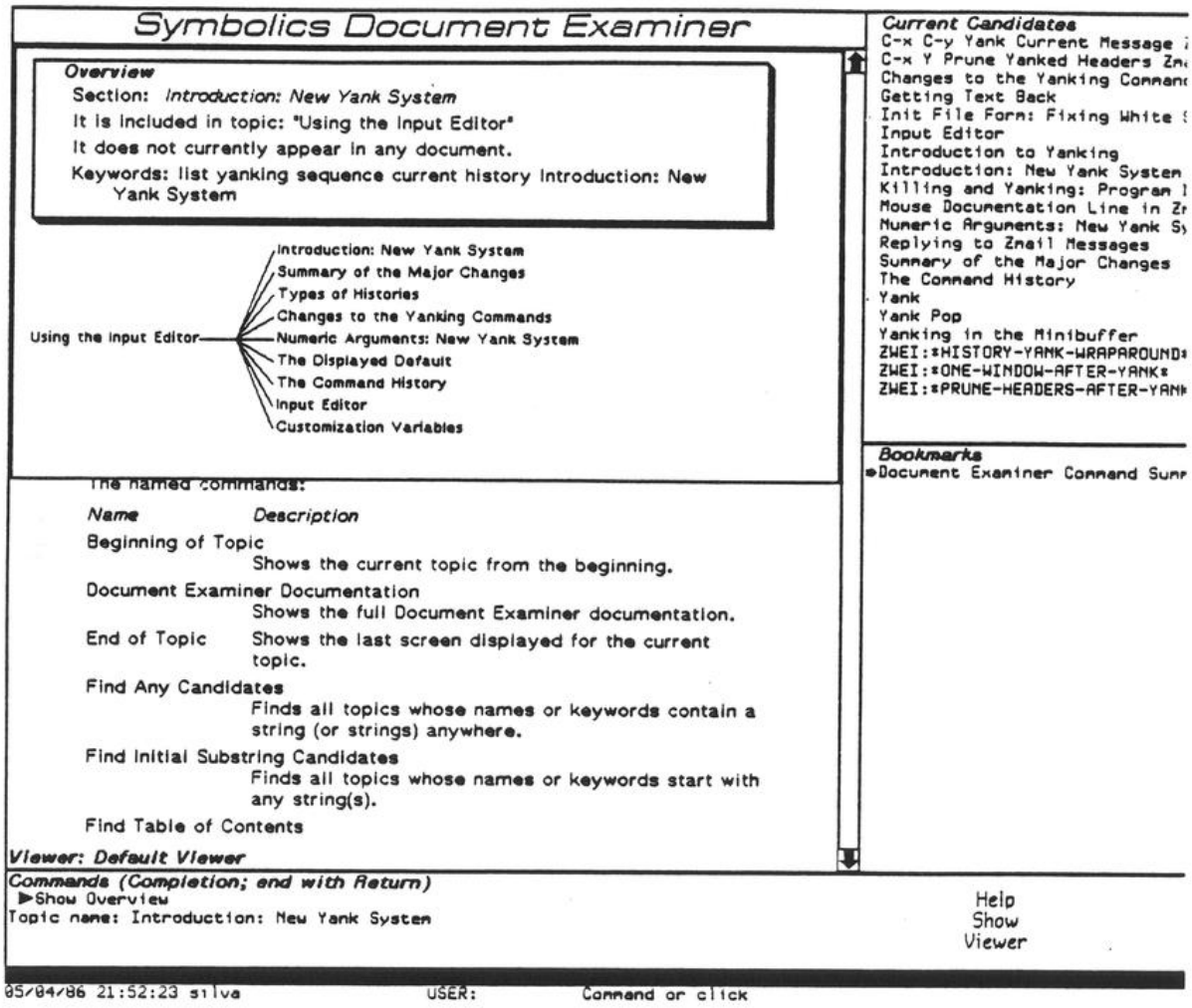


Figure 6.2 The **Overview** *in the Document Examiner gives a summary of a topic, including keywords and a tree diagram of where the topic is placed in the system documentation. The command Show Overview produces an overview, as does a middle mouse click on a topic name.*

4. At the bottom of the text, "Deleting and Transposing Text in Zmacs" is also mouse-sensitive. Click right on it to learn what operations can be performed. Move the mouse off the menu. Many items in a documentation text can be mouse-sensitive in this way; they are usually in bold print or in quotations.

5. If you know exactly what topic you would like to see documented, use the [Show] command. Click left on [Show], then type:

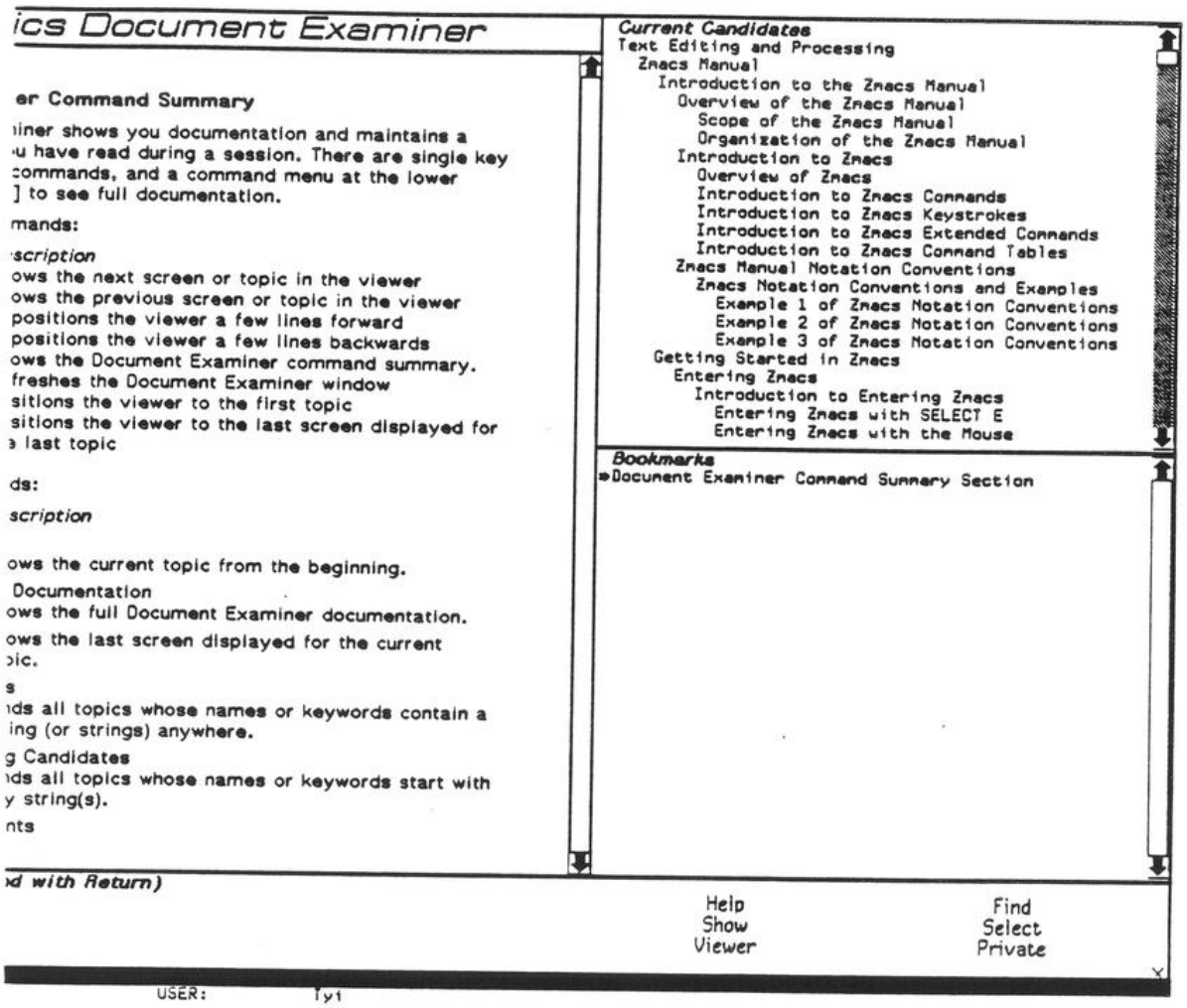


Figure 6.3 The figure shows the **Table of Contents** *for the topic "Text Editing and Processing". The Find Table of Contents command produces this display. You can also get it by clicking right on a topic and selecting Find Table of Contents from the menu. Another way is to click right on the Show command at the lower right.*

```
what is a world load<RETURN>
```

The topic goes directly into the default viewer, a bookmark is inserted, and no candidates are listed.

6.9 Bookmarks

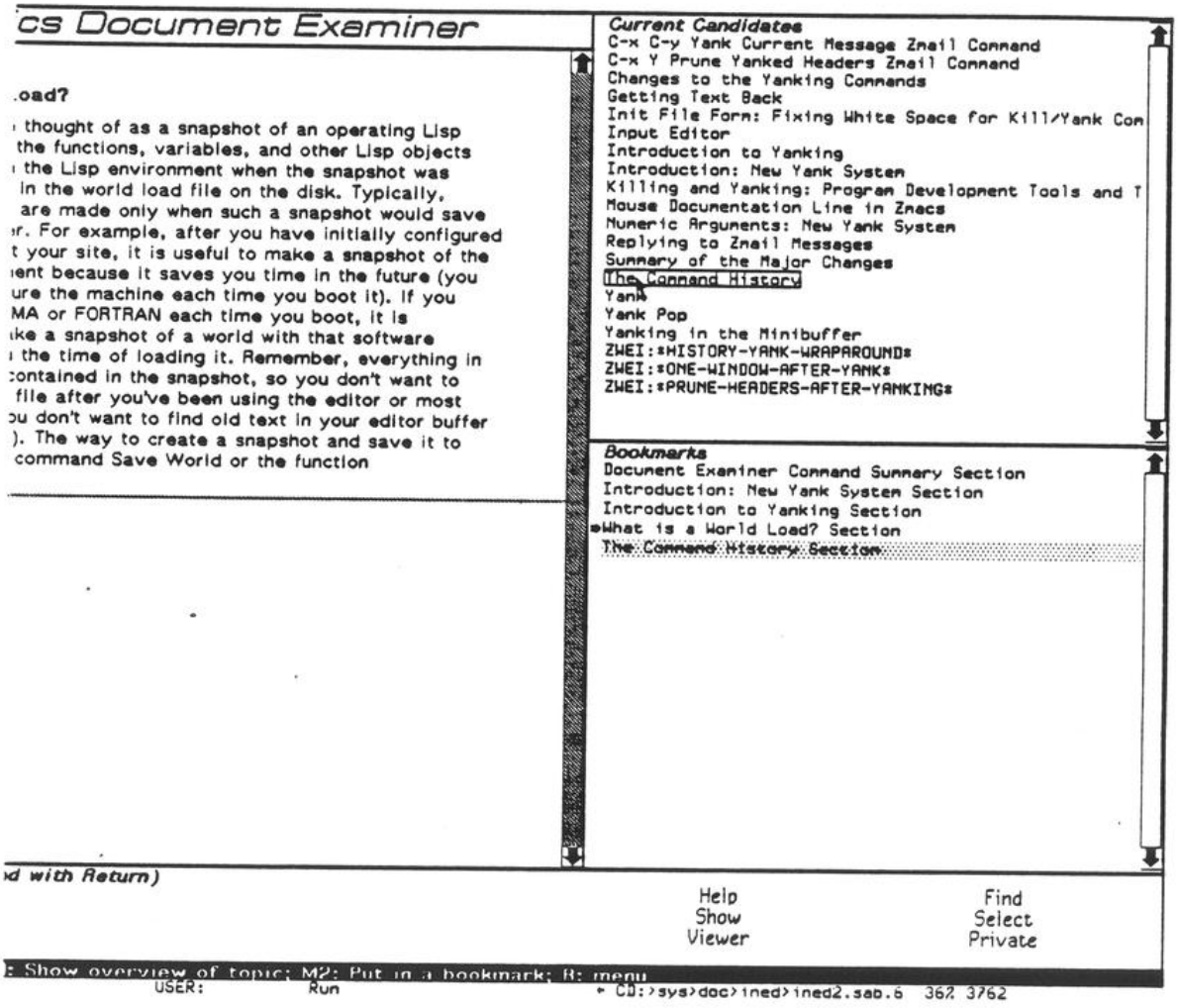


Figure 6.4 **Bookmarks** *allow you to go back to a topic quickly. They're created automatically for every topic you read, but you can also add them by pressing the* SHIFT *key and clicking the middle mouse key in a topic.*

Every time you read a topic into the viewer, the section name is placed into the Bookmarks pane. An arrow at the left margin of the Bookmarks pane points to the section that is currently in the viewer. Move the mouse over to one of the bookmarks and look at the mouse documentation line. 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 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 that one choice is to put in a bookmark. You can do the same with any of the words in the Viewer pane that are mouse-sensitive.

When you read a section into the Bookmarks pane, the section name is covered by a gray strip (Figure 6.4). 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.

6.10 Documentation References

Once you're more familiar with the system it is well worth your while to read the Document Examiner documentation (using the 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**
  Vol. 1 pages 119-122
- **The Document Examiner Frame**
  Vol. 1 page 129
- **Basic Document Examiner Commands**
  Vol. 1 pages 122-129
- **Bookmarks**
  Vol. 1 pages 129-134
- **Other Commands for Documentation**
  Vol. 1 pages 128-129

Section 2

7. Introduction

Section 2 expands on many of the topics covered in Section 1. This section does not teach essential material that you must know before coming to class. Rather, it covers material that will make your work on a Symbolics machine much easier.

Section 2 has five chapters explaining more interesting and useful commands to build on the skills which you developed in Section 1:

- The first chapter, **Making Things Easier**, details additional features available in the Lisp Listener and other windows. We provide more information on the Command Processor as well as introduce the Input Editor and the Command History features.

- The second chapter, **More Getting Around**, introduces the System menu and how to use it to select activities. The material in this chapter and in the previous section give you four different ways to select an activity.

- The third chapter, **More Zmacs**, provides additional information on Zmacs features like help, scrolling, and completion. Using regions to move text and extended commands are also discussed.

- The fourth chapter, **More Files**, builds on the information about files and directories you learned in the previous section. Directory listing format and pathnames for other machines are also described.

- The final chapter, **More Document Examiner**, expands on the material in the previous section. It provides information on using private documents and additional viewers to increase the ease with which you use the Document Examiner.

8. Making Things Easier

8.1 Purpose

The Symbolics system has a number of features that make it easier to use the Command Processor. The Command Processor has completion, meaning that once you've typed the significant part of a command name, the Command Processor fills in the rest for you, and also the Input Editor, meaning you can edit what you type in. A history of all commands you've entered is also kept so you can figure out what you did, or even repeat a previous command.

8.2 Contents

- **Introduction**
- **Completion in the Command Processor** is activated by pressing the SPACE bar after you've typed enough characters to uniquely identify a command. If you haven't yet typed enough, the Command Processor completes as much as it can. The COMPLETE key also fills out a command or keyword argument name.
- **Walk-through for Completion**
- **The Input Editor** uses many of the same commands used in the Zmacs editor.
- **Walk-through for the Input Editor**
- **The Command History** is kept from the time you cold boot. c-m-Y calls back the last command entered. Successively pressing m-Y works back through the command history one command at a time. You can see the recent history by pressing ESCAPE, or the entire history by pressing c-0 ESCAPE. You can repeat a command from the history by pressing c-N c-m-Y with the N being the number of the command you want back. You can use the Input Editor to edit commands you've brought back in this way.
- **Walk-through for the Command History**
- **Useful Keys** described include ABORT, which halts the current program and c-ABORT, which halts it immediately; CLEAR INPUT, which erases what you've

typed on a line; and REFRESH, which refreshes the screen. Other useful keys are also described in this section.

- **Creating More Than One of an Activity** is sometimes handy. You select the Lisp Listener by pressing SELECT L. If you want another Lisp Listener, press c-SELECT L. To go from one to the other, just press SELECT L. You can make as many Lisp Listeners as you want. This works for other activities, such as Zmacs and the Document Examiner.

- **Walk-through for Creating More Than One of an Activity**

- **Documentation References**

8.3 Introduction

In general, it is not necessary to type all of a command. It is also possible to edit what you are typing and to bring back things you have already typed. These features, which will be described in this chapter, make it much easier to use the Symbolics system. Most of these features are available everywhere on the machine, but here we describe only their use in the Command Processor.

8.4 Completion in the Command Processor

Completion is a feature that lets you type only as much of a command as is necessary to uniquely specify that command. Very often, it is necessary 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 CP 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 you are typing as is unique.

8.5 Walk-through for Completion

1. Type:

   L<SPACE>

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

2. Now type:

   F<SPACE>

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

3. Press CLEAR INPUT to erase this command.

4. Type:

   Se<SPACE>0

5. Press the COMPLETE key. Notice that it fills in:

   Set Output Base (number base)

6. Press CLEAR INPUT to erase this command.

7. Type:

   Se<SPACE>0

   again. Press SPACE. Notice that the CP fills in only:

   Set Output

   even though you have seen that Se 0 is enough to uniquely specify a command.

8. Press SPACE again. Notice that the command is now completed.

9. Press CLEAR INPUT to erase this command.

8.6 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 the same ones you would use in Zmacs to do the same operations. You can get a complete list of these commands by typing c-HELP (Figure 8.1).

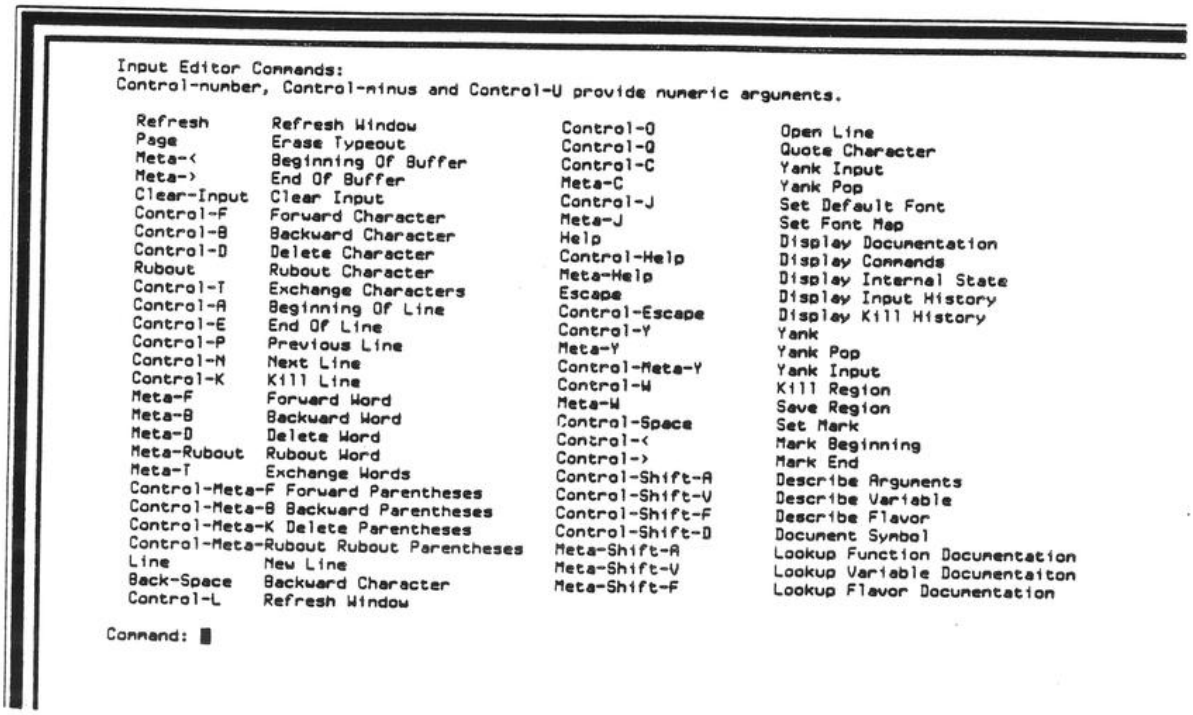


Figure 8.1 For a full list of **Input Editor commands**, *press* c-HELP. *The Input Editor commands are invoked by pressing keys, not by typing commands. Try some of them out.*

8.7 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 clear the screen, press REFRESH.

1. Type (just as you see it):

```
lgoin<SPACE>
```

2. You should see the message:

```
"lgoin" does not complete to a valid command.
Press RUBOUT to correct your input.
```

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

3. Press c-T. Notice that the g and o switch places.

4. Press END. Now the command completes to:

```
Login
```

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

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

6. 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, which could be the completions of what you typed.

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

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

9. Type:

```
e<RETURN>
```

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

8.8 The Command History

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 the machine was cold-booted. You can see this history by pressing the ESCAPE key (Figure 8.2). The most recent commands appear first. If you have typed many things to your Lisp Listener, not all of them will appear, just the twenty most recent ones. To see the entire history, press CONTROL-0 ESCAPE. If several consecutive commands were the same, only the most recent one will be shown in the history list, to save space.

You can bring back (*yank*) the most recent command from the history by typing 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 can 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.

8.9 Walk-through for the Command History

1. Log in, if you haven't already done so.

2. Press the ESCAPE key. Notice how many commands are listed. There might be only one, the Login command which 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.)

3. Type the command:

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

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

5. Press the REFRESH key to clear your screen.

6. Press the ESCAPE key again. Notice that the top item in the history is now:

```
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 listed only once.

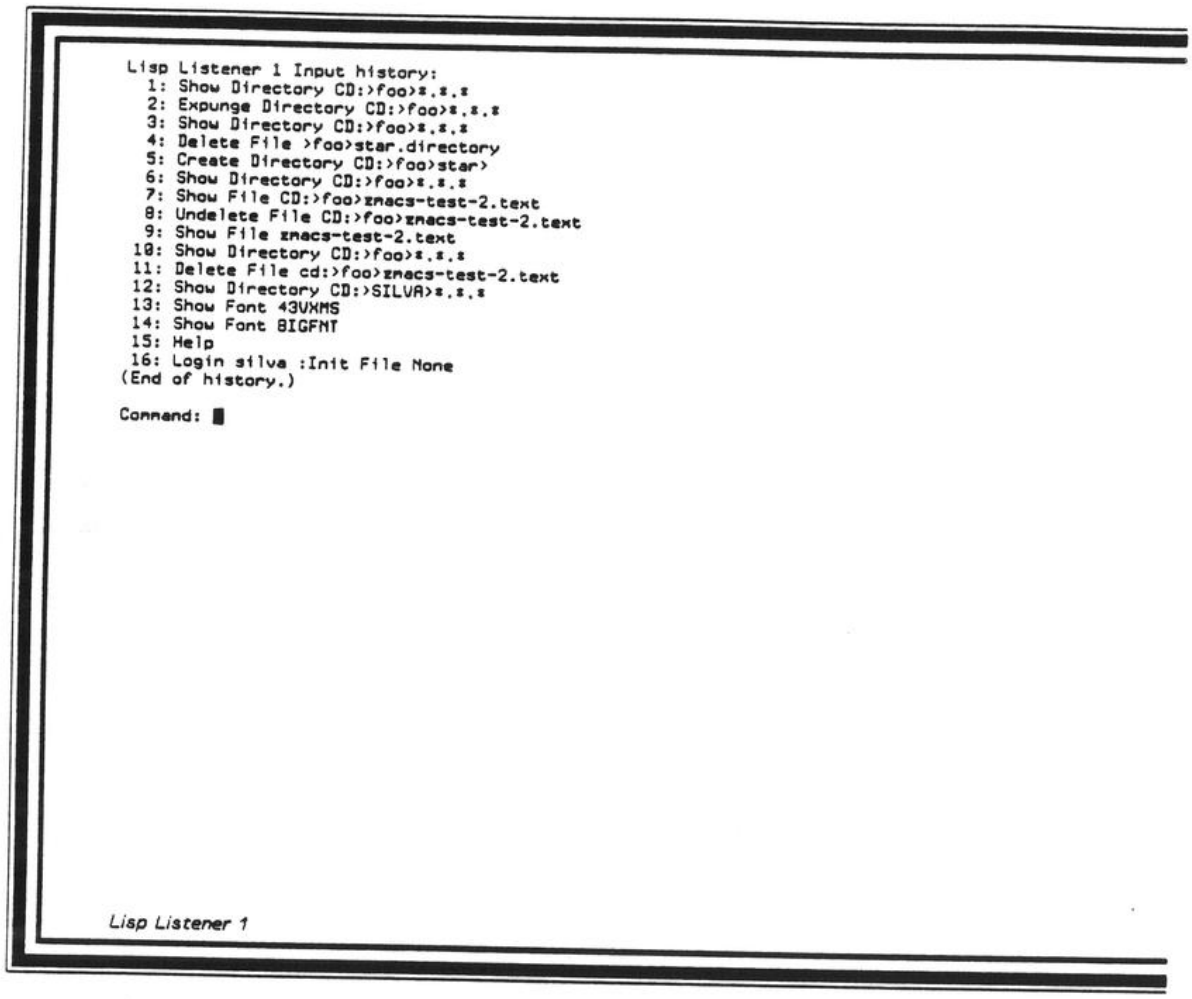


Figure 8.2 To look at your recent **command history**, *press* ESCAPE. *To see it all, press* CONTROL-0 ESCAPE. *You can also yank back commands from your history.* c-m-Y *gets the most recent command. To get* Show Font BIGFNT *in this example, you would type* c-14 c-m-Y.

8.10 Useful Keys

ABORT

Aborts the operation currently in progress. It returns to the command level in such programs as the Lisp Listener, Editor, File System Editor, and so forth. It takes effect when the program reads it, rather than immediately.

- c-ABORT is like ABORT, except it takes effect immediately.
- m-ABORT causes the process to return to 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 it takes effect immediately.

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

COMPLETE — Completes (as far as possible) the characters entered. 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 line. This is particularly useful when yanking a previously typed Lisp form. If a 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 a line.

FUNCTION — Allows you to do interesting things with windows and processes. Press FUNCTION HELP to see a listing of all the actions that can be taken by pressing FUNCTION key combinations.

REFRESH — Clears the window, sometimes redrawing the contents. For instance, in a Lisp Listener, pressing REFRESH clears the window, 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 the machine. 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 CP executes it. If, on the other hand, 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, the machine waits for you to add enough to specify a unique command.

SYMBOL

The character set contains more characters than there are keyboard keys, even when they are combined with the modifier keys. SYMBOL allows you to access special characters. Press SYMBOL-HELP. 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, most often, the fastest way to enter commands. For example, typing s<space>ca<space><space> in the Command Processor causes:

```
Set Calendar Clock (time)
```

to be displayed.

8.11 Creating More Than One of An Activity

No doubt you noticed a paragraph in the SELECT HELP display that discussed creating new activities. This allows you to 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.

8.12 Walk-through for Creating More Than One of an Activity

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

1. Select *Lisp Listener 1.* (You should know how to do this by now.)
2. Now, *with Lisp Listener 1 selected*, press SELECT L.

   It should beep at you (or at least flash the screen). It did this because you already have the Lisp Listener selected. What we're going to do now is to create *another* 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 Lisp Listener 2. This activity can do anything *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 use SELECT c-M to create another Zmail window. You also *shouldn't* use SELECT c-E to make extra Zmacs windows. It will work most of the time, but you should get into the habit of using one Zmacs window, since having more than one can cause a lot of problems. There are multiple buffers and ways of splitting the one window into several, so it's not a serious drawback.

   Having two Lisp Listeners now, you might ask, "How do we get to the other one?"

6. Press SELECT L.

   This brings you back to *Lisp Listener 1.*

7. Press SELECT L again.

   This brings you back to 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.

8.13 Documentation References

- **Completion in the Command Processor**
  Vol. 1 pages 9-42, 164-167
- **The Input Editor**
  Vol. 1 pages 101-103
  Vol. 5 pages 53-73
- **The Command History**
  Vol. 1 page 15
- **Useful Keyboard Keys**
  Vol. 1 pages 140, 144-146
- **Creating More Than One of an Activity**
  Vol. 1 page 145

9. More Getting Around

9.1 Purpose

The SELECT key is not your only means of selecting activities. You can also use the System menu, which is reached by pressing the SHIFT key while you click the right mouse button.

9.2 Contents

- **The System Menu** allows you to select all the activities you can select with the SELECT key and also allows you to change and manipulate windows as well.
- **A Summary of Activities on the System Menu** includes everything you can reach through the SELECT key, plus the Font Editor, the Namespace Editor, the Hardcopy system, and the **trace** facility.
- **Walk-through for Selecting Activities Using the System Menu**
- **The Select Submenu of the System Menu** is reached by clicking on the Select item on the System menu. If you create windows of your own that are not otherwise available, you'll find it handy.
- **The Mouse** allows you to select any window you can see by simply clicking left on it.
- **Walk-through for Selecting Activities Using the Mouse**

9.3 The System Menu

The System menu is a facility for doing three kinds of operations:

1. Selecting an activity.
2. Changing a window.
3. Dealing with window activities in general.

We're only going to examine "Selecting an activity" here.

The right hand column of the System menu is a list of activities that can be selected. Some of these are available via the SELECT key, and some are not.

To bring up the System menu, you hold the SHIFT key down and click the right mouse button once (Figure 9.1). (You can also get the System menu by clicking the right button twice, but many people find clicking twice to be difficult.) The System menu appears where the mouse was. To make it disappear (and not do anything), just move the mouse off the menu.

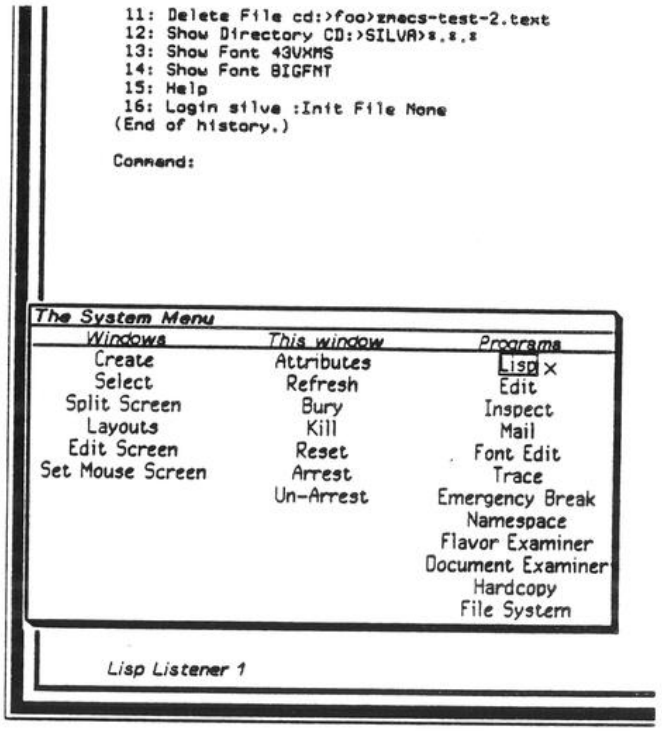


Figure 9.1 The **System menu** *appears when you click right while holding down the* SHIFT *key. Many of the items on the right are the same as you can get through the* SELECT *key. The items in the left and middle affect the window you were on when you called for the System menu.*

To select the activity specified by one of the menu items, move the mouse over 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.

Note: You can bring up the System menu by holding down the SHIFT key and clicking right anywhere *except* while the mouse is on a menu.

9.4 A 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.

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

[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 without using the window system. Rarely used, it can be useful for programmers.

[Namespace] 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.

[Flavor Examiner] The Flavor Examiner. Also available on SELECT X.

[Document Examiner] The Document Examiner. Also available on SELECT D.

[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.

9.5 Walk-through for Selecting Activities Using the System Menu

1. Press SELECT L. You should be in *Lisp Listener 1* 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 line 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 the SHIFT key and 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 *Lisp Listener 1.*

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. Do not use the other columns of the System menu right now.

9.6 The Select Submenu of the System Menu

Another way to select an activity is via the [Select] item on the System Menu. If you click on this item, the System menu disappears and another menu pops up (Figure 9.2. 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.

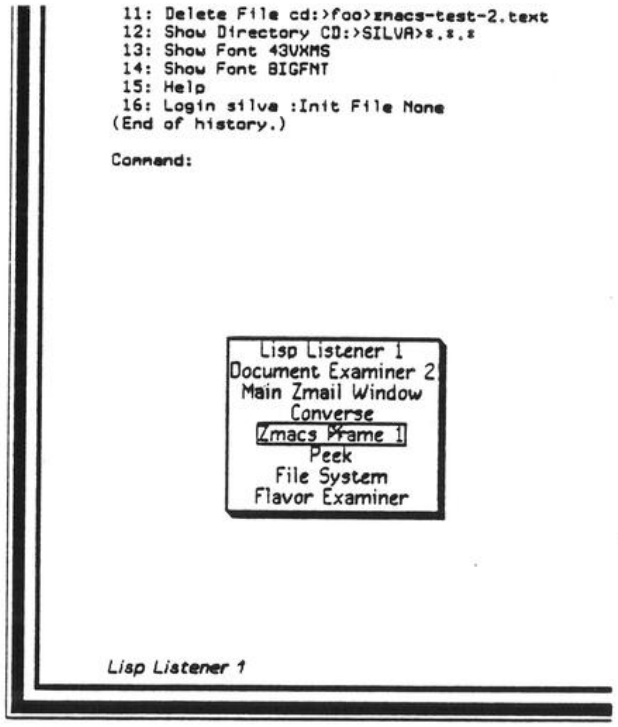


Figure 9.2 The **Select Submenu** *is reached by clicking on the* `Select` *item on the System menu. It is superfluous as it is shown here, but if you create windows of your own that are not otherwise available, you'll find it handy.*

9.7 The Mouse

You can use the last way to select an activity only 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 obscured, then it comes up to the top of the pile (becomes fully visible) and becomes the selected activity. This technique also works for selecting one of two completely visible windows.

9.8 Walk-through for Selecting Activities Using the Mouse

1. Select the editor. Use either SELECT E or the System menu. The far right-hand part of the Lisp Listener should be visible.

2. Click left on the partially visible portion of the Lisp Listener. This will select the Lisp Listener.

Notice, now, that you can't use this technique to get back to the editor. This is a technique that you can use much more frequently when you have changed the size or position of your windows such that parts of them stick out from under the others.

10. More Zmacs

10.1 Purpose

This section explains more about Zmacs. It teaches you how the HELP key works in Zmacs, more about how to move around in your buffer, how to use regions and how to yank back previously deleted material, and how to use extended Zmacs commands and a number of other useful Zmacs commands.

10.2 Contents

- **Help** from Zmacs can show you all commands that contain a certain word, all documentation for commands or keys, and also the last 60 Zmacs commands you've entered. In addition, it can undo a previous major operation and report on the state of Zmacs itself. Press HELP twice in Zmacs for a summary.
- **Scrolling** in Zmacs is accomplished through the SCROLL key or c-V (to go down) or the m-SCROLL key or m-V (to go up). To center the cursor, use c-L. You can also use the scroll bars on the left side of the buffer or pane for scrolling.
- **Regions** are marked by marking the beginning with c-SPACE and then moving to the end of the desired region. Many Zmacs commands take advantage of regions. Kill a region with c-G.
- **Walk-through for Scrolling and Regions**
- **Modes** include Lisp mode, Text mode, and Auto Fill mode.
- **Extended Commands** number well over a hundred and allow you to do all manner of things to text and source code.
- **Completion** works in Zmacs much like it does in the Command Processor.
- **Other Useful Commands** allow you to capitalize a word, lowercase it, center a line, move the cursor from sentence to sentence, or list and select buffers.
- **Walk-through for Other Useful Commands**
- **Documentation References**

10.3 Help

To get help in Zmacs, press the HELP key (Figure 10.1). 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.

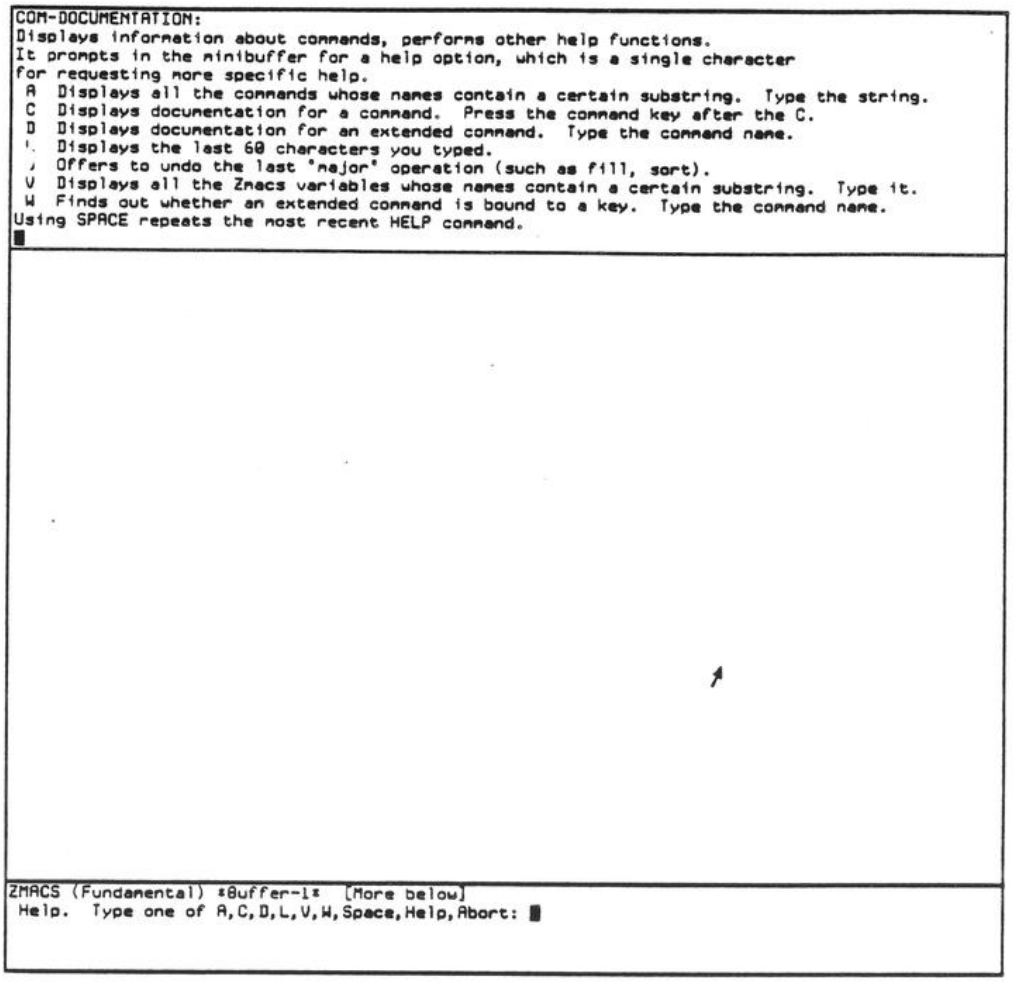


Figure 10.1 **Help in Zmacs** *is quite extensive. This is what you see when you press the* HELP *key. Press the other characters after you press* HELP *to get the actions shown. Lots of rookies overlook* HELP U *(for Undo), but it can be very handy.*

Some of the more useful help options are:

- A – Apropos. This 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 – This displays documentation for a command. Just type the command (c-A, for instance) and you are told what it does.
- D – This displays documentation for an extended command.
- U – Undo. This allows you to undo the last major change to the buffer. This is very helpful if you accidentally make a major change, such as a global replace string.

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 240-odd possible m-X commands. c-m-X HELP gives you a listing of the 690-odd possible c-m-X commands.

10.4 Scrolling

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

Keyboard

SCROLL, c-V	Moves to the next screenful of text.
m-SCROLL, m-V	Moves to 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

Some windows, such as the editor and file system editor, implement scroll bar scrolling. To use it, bump the mouse against the left border until the cursor changes shape to become a thick arrow with heads at top and bottom. At the same time, a *scroll bar* appears adjacent to and parallel to the left border. The length of the scroll bar relative to the height of the window or window-pane in which it appears represents the length of the visible portion of the contents of the buffer or pane as compared to the entire contents of the buffer or pane. The position of the scroll bar along the left edge indicates from what part of the entire contents the visible portion comes. (See Figure 10.2.) For example, if the scroll bar were at the top of the left edge, the beginning of the contents would be visible in the window.

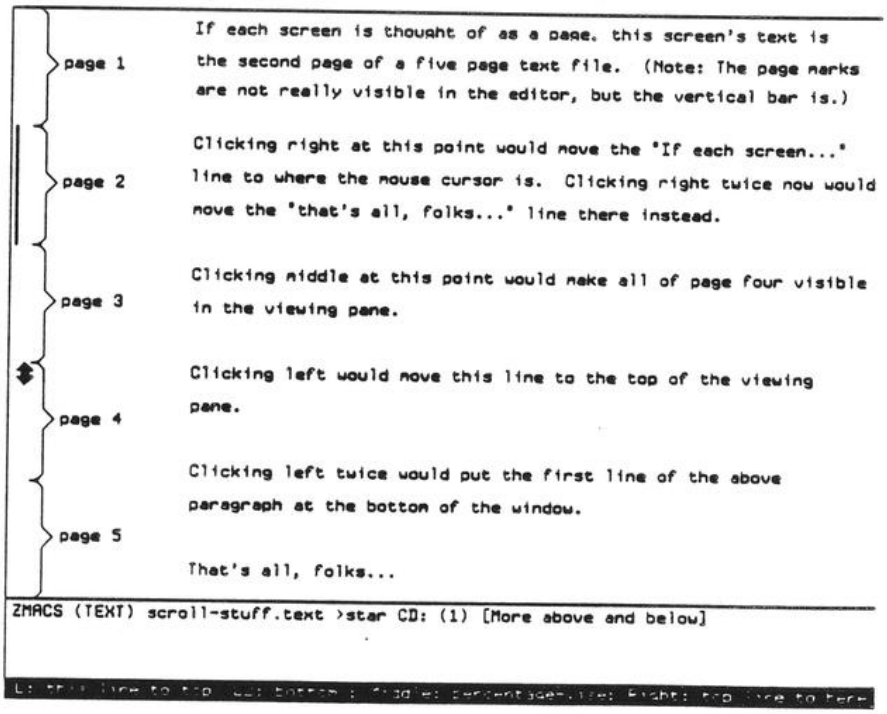


Figure 10.2 The vertical line on the left is a **scroll bar**. *The text shown in the figure tells you how to use the scroll bar and what happens when you do. You'll need to try this out to understand it fully.*

When the mouse cursor is double-headed, the mouse line reads:

```
Left: this line to top (L2:bottom); Middle: percentage-wise; Right: top line to here.
```

`This line to top` means redisplay so that the line indicated by the mouse cursor is at the top of the window or pane.

`Percentage-wise` means display the portion of the contents such that the top of the scroll bar appears at the present mouse cursor location. For example, if you click left when the double-headed mouse cursor is 60 percent of the way from the top of the window to the bottom, the line that is 60 percent of the way from the top of the file to the bottom of the file becomes the first line displayed in the window.

`Top line to here` means redisplay so that the line now at the top of the window display is at the position indicated by the mouse cursor. Although it doesn't say so, clicking right twice moves the bottom line to the position indicated by the mouse cursor.

10.5 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 of the windows in the system look into the same global kill history, so you can kill anything bigger than a character and put it into the global kill history. You can pull anything in 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. Holding down the left button, move 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 previously marked region.
m-W	Copies the previously 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.

10.6 Walk-through for Scrolling and Regions

1. Press SELECT E to select the 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:

```
Speak the speech, I pray you, as I pronounced it to you, trippingly on
the tongue.  But if you mouth it, as many of our players do, I had as
lief the town crier spoke my lines.  Nor do not slay the air too much
with your hand, thus, but use all gently; for in the very torrent,
tempest and (as I may say) whirlwind of your passion, you must acquire
and beget a temperance that may give it smoothness.
```

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 Nor 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 Nor, still 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 it turns into a fat, double-headed arrow. Notice the scroll bar, the thin line between the mouse cursor and the left edge of the buffer.

29. Watch the scroll bar and press m-V.

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

31. Notice the line of text nearest the mouse cursor. Watch the scroll bar and click left. Notice that the line of text that was near the mouse cursor has moved to the top of the buffer. Notice also that the scroll bar has moved lower, as you moved lower in the text.

10.7 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 exactly one major mode. Minor modes are options to make the editing environment more precisely what you would like. You can have more than one minor mode. 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.

10.8 Extended Commands

There are more Zmacs commands than can be represented with `CONTROL-` and `meta`-keystrokes. You can get an entire set of commands by typing `m-X` and then 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.

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 Already Existing Text

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

m-X Insert File
: Inserts the 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 Hardcopy Status
: Prompts you for the name of a printer (hardcopy device) and displays a list of the jobs that are currently in the printer queue.

10.9 Completion

Extended commands support completion, just like CP 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 out, 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.

10.10 Other Useful Commands

c-O	Inserts a carriage return after the cursor.
c-m-L	Selects the previous buffer. This is useful for switching between two buffers.
m-A	Backward sentence. Moves the cursor to the beginning of the sentence.
m-E	Forward sentence. 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 the current paragraph by adjusting the 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.

10.11 Walk-through for Other Useful Commands

1. Make sure you are still in an editor buffer. (If not, type 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 only need to type Tex M and press RETURN. Answer Y to the question. You are now in Text Mode in the editor. Move to the top of the buffer, and notice that a line has appeared that says -*- Mode: Text -*- at the top of your buffer.

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 should select 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 uppercased only 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 made changes to this buffer without saving the changes, 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.

10.12 Documentation References

- **Help**
  Vol. 1 pages 86-101
  Vol. 3 pages 14-16, 41-52
- **Regions**
  Vol. 3 pages 85-95
- **Modes**
  Vol. 3 pages 142-143, 155-157, 195-198
- **Extended Commands**
  Vol. 3 pages 114-128
- **Completion**
  Vol. 1 pages 84-85
  Vol. 3 page 14
- **Other Useful Commands**
  Vol. 3 pages 159-167

11. More Files

11.1 Purpose

This chapter shows you more Command Processor commands, including Load File, Rename File, Copy File, Hardcopy File, and Save File Buffers. It also explains the contents of the directory listings and how to use VMS and UNIX pathnames.

11.2 Contents

- **More Command Processor File Commands**
- **Walk-through for More Command Processor Commands**
- **Directory Listings Explained**
- **Pathnames On Other Machines**
- **Documentation References**

11.3 More CP File Commands

`Load File` *<pathname>*

Cause the compiled version of the specified file to be *loaded*, which means that you can execute any of the functions and variables defined in the file. If you wish to edit or view this file, you must use the Edit File or View File command.

This command takes a *:Query* keyword, which can have a value of *(Yes, No, or Ask)* (Figure 11.1). If you don't type in 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.

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

Changes the name of the 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>*. This command does not copy the file; to do that, use the Copy File command.

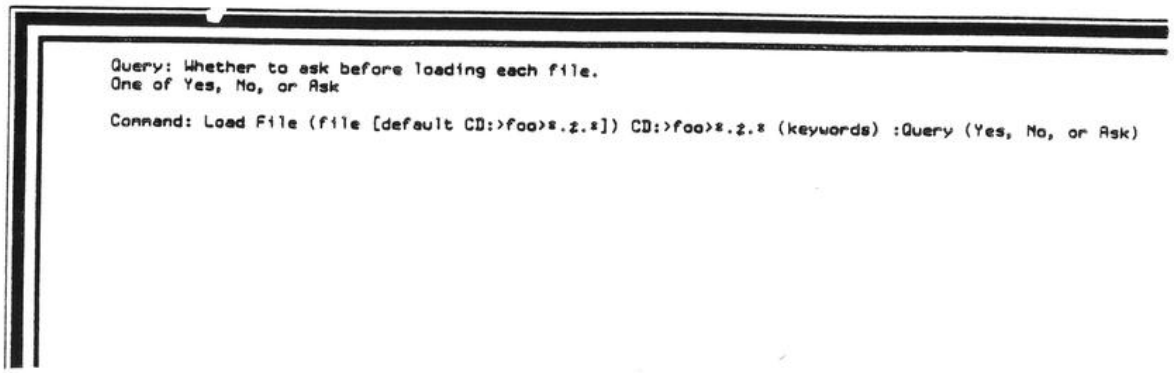


Figure 11.1 Some commands include the **:Query keyword**. *You can find out which by typing a colon (:) after the command and then pressing* HELP. *Use the :Query keyword in conjunction with wildcards to process a large group of files with a single command.*

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

This command takes a *:Query* keyword, which can have a value of *(Yes, No, or Ask)*. The value defaults to *No* if you don't 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>`

Hardcopying a file means printing a file. To do this, you *must* have a printer hooked up to a machine on your network.

This command takes keywords of *:Copies* and *:Query*. *:Copies* allows you to specify how many copies of the file you would like to print. The *:Query* keyword can have values of *(Yes, No, or Ask)*. The value defaults to *No* if you don't type in the keyword, meaning don't ask about hardcopying each file. If you give the keyword, the value defaults to *Yes*. The *<pathname>* can contain wildcards. This command causes the file to be printed on your printer. If you do not have a printer, do not try to hardcopy files.

`Save File Buffers<RETURN>`

Saves all the buffers in Zmacs that are associated with files and need to be saved (because they have been modified).

This command takes a *:Query* keyword, which can have a value of *(Yes, No, or Ask)*. The value defaults to *No* if you don't 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.

11.4 Walk-through for More Command Processor File Commands

1. Select Zmacs and type:

   `c-X c-F`*<your-file-server>*`:>`*<your-login-ID>*`best-seller.text`

2. To right-justify the text you are about to type, type:

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

3. Type in:

```
I was walking down Van Ness Avenue 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.

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 yourself 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>assignment--foobar<RETURN>
```

That's more like it.

11.5 Directory Listings Explained

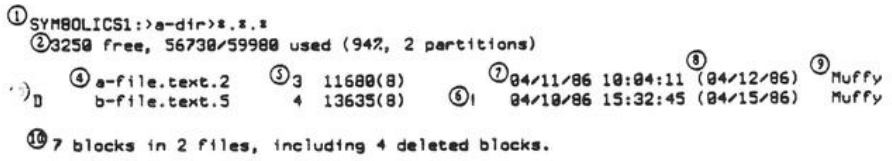


Figure 11.2 The **Directory Listing** *gives the name of the directory and name, type, and version number of the file as well as the size of the file, creation date, and the name of the person who created the file.*

The following items are keyed to the numbers in Figure 11.2.

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 blocks, followed by bytes.
6. This column contains an exclamation point if the file has not been *backed up* on tape. 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.
7. The date and time the file was created.
8. The date the file was last accessed.
9. The user who created the file.
10. The total number of blocks and files in the directory.

11.6 Pathnames on Other Machines

Files do not have to reside on a Symbolics machine 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 machine, the pathname will have essentially the same components (although many machines do not have automatic version numbers), but they will be put together in different forms. Below are a few of the possible host/pathname combinations.

- LMFS

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

- UNIX

```
SYMBOLICS-VAX:/Muffy/cards/solitaire.lisp
```

- VMS

```
SYMBOLICS-VAX-VMS:[Muffy.cards]solitaire.lisp;10
```

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

11.7 Documentation References

- **More Command Processor File Commands**
  Vol. 1 pages 18-23, 26, 29-30
- **Directory Listings Explained**
  Vol. 5 pages 236-237
- **Pathnames on Other Machines**
  Vol. 1 page 228

12. More Document Examiner

12.1 Purpose

This chapter explains how to create private documents in the Document Examiner, how to create additional viewers, how to scroll in the Document Examiner, and how to get hardcopy versions of Document Examiner topics.

12.2 Contents

- **Private Documents** are parts of the documentation that you use frequently and want always to have loaded. The [Private] item on the Document Examiner menu allows you to save and recover private documents. Click right on [Private] to save as a private document everything listed in the Bookmarks pane.

- **Additional Viewers** are handy for setting up the Bookmarks pane prior to creating a private document. Additional viewers also a good place to read your private documents.

- **Mouse Scrolling** is accomplished by the mouse in conjunction with the scroll bar, which appears as a vertical line on the left side of the pane.

- **Walk-through for Additional Document Examiner Features**

- **Other Available Commands/Hardcopying** explains the Beginning of Topic and End of Topic commands and how to hardcopy a private document, a viewer, or any documentation topic.

- **Documentation References**

12.3 Private Documents

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

- **[Private]** - When you click right on [Private] you are prompted for a

pathname of a file to store the topics that currently appear in the Bookmarks pane. You can recover these topics later by clicking left on [Private] to read your private document, or by clicking middle to load it.

If you read in the private document, the actual text for all of the topics is read in. This takes a noticeable amount of time, especially if you read in several large topics.

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 this.

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 the name of a viewer that does not exist, a viewer with that name is created.

12.4 Additional Viewers

Up to now, you've only used the default viewer to display documentation. You can, however, create multiple viewers in order to better keep track of the text you're interested in.

- **[Viewer]** - To make a new viewer (or choose a viewer that already exists) you click left on [Viewer], then enter the viewer's name. It's a good idea to give short and 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.

  Clicking middle on [Viewer] prompts you to type the name of a viewer you wish to discard.

  Clicking right on [Viewer] hardcopies a viewer. This is discussed at the end of the chapter.

12.5 Mouse Scrolling

You can move full screens of text by using the mouse. Each pane (except the Commands pane) has a *scrolling bar*, a tall rectangle, along the right edge of the pane. The walk-through exercise below illustrates scrolling and the other features previously discussed.

Some mouse scrolling options in the Document Examiner are identical to those

found in any Zmacs pane; they are not discussed in this chapter. See the chapter *Zmacs* for basic scrolling operations.

12.6 Walk-through for Additional Document Examiner Features

1. Select the Document Examiner if you haven't already. Refer to the earlier chapter on the Document Examiner if necessary.

2. Read a topic (any topic) into the Default Viewer by clicking left on a candidate. (If you already have text in the viewer, that's sufficient.)

3. Create a new viewer. Click left on [Viewer] and type:

   ```
   new-viewer<RETURN>
   ```

   Notice that all panes have been 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.

4. Begin creating a private document by reading a topic into the Viewer. Click left on Show and type:

   ```
   Customizing the E<COMPLETE><RETURN>
   ```

   The topic `Customizing the Editor and the Environment: Lispm-init.lisp` appears in the Bookmarks pane, along with the appropriate text in the Viewer pane.

5. Include another bookmark, this time regarding *booting*. Click left on [Find] and type:

   ```
   booting
   ```

6. When the topics appear in the Current Candidates pane, put the `Cold Booting` candidate in the Bookmarks pane; hold down the `SHIFT` key while clicking middle on `Cold Booting`.

7. Save the two bookmarks as your private document. Click right on [Private] and take the default (if you don't want to specify a pathname of your own) by pressing `RETURN`. You now have a private document file in your directory.

8. Go back to the Default Viewer. Click left on [Viewer] and press `c-sh-?`. 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 [Private] and press RETURN to take the default (or type the name you previously entered followed by RETURN) to read in your private document.

10. You must also specify which viewer the text will be read into. Type:

    ```
    new-viewer<RETURN>
    ```

    The text should soon appear in the Viewer pane that you created. Anytime you want to read documentation on these topics, the *lispm-init-file* and *cold-booting*, you can simply look in this viewer.

11. Once a topic is read into the Viewer, the scrolling bar at the right margin of the ` iewer pane allows you to reposition or scroll through text. Move the mouse into the gray part of the scrolling bar; it turns into a thick arrow. Position the mouse at the middle of the bar and click left. The white box you see represents the text visible in the pane. The scrolling area from top to bottom represents all the text that has been read into the viewer.

12. Move the mouse below the white box and look at the mouse documentation line. Click left; this last operation is equivalent to pressing SCROLL.

13. Move the mouse above the white box and read the documentation line. Clicking left above the box is equivalent to pressing m-SCROLL.

14. You can also reposition text by dragging the box in the scrolling bar. Move the mouse into the white box; the cursor turns into a diamond shape. Hold down the left mouse button while moving the mouse vertically. The box moves with the mouse. When you release the mouse button, the box is moved and the appropriate text is read into the viewer.

15. Scrolling is best learned by practice. Beginners are occasionally surprised by the behavior of the scrolling system.

12.7 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. Some of the more useful commands include:

- Beginning of Topic
- End of Topic

- Hardcopy Private Document
- Hardcopy Documentation
- Hardcopy Viewer

Beginning of Topic and **End of Topic** 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).

Hardcopy commands print, respectively, 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:

- Check the length of the document you wish to print (avoid hardcopying long documents – use the printed documentation volumes instead).
- Make *sure* a printer is connected to your Symbolics machine.

12.8 Documentation References

- **Private Documents**
  Vol. 1 pages 136-137
- **Adding Viewers**
  Vol. 1 pages 129-131
- **Mouse Scrolling**
  Vol. 1 pages 134-136
- **Other Available Commands/Hardcopying**
  Vol. 1 pages 122-129

Section 3

13. Introduction

Section 3 discusses topics that not every user of Symbolics computers needs to know. For instance, a portion of the chapter on Zmacs describes formatting text, a capability that is not needed by everyone. The way to read through this section is to choose from the various topics those that interest you.

- The first chapter, **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.

- The second chapter, **Additional Files**, details two more ways to manipulate files and directories. *Dired*, short for the Directory Editor, is a program available 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.

- The third chapter, **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.

- The fourth chapter, **The Namespace**, explains how to add a user to the namespace database. This is the briefest of introductions to this program.

- The final chapter, **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.

14. Additional Zmacs

14.1 Purpose

This section introduces some additional optional features of Zmacs, including numeric arguments, which allow you to repeat Zmacs commands, and which also change the meaning of some commands; saving, killing, renaming, comparing and merging, and otherwise manipulating files and buffers; splitting the screen between two buffers, and some commands for simple text formatting.

14.2 Contents

- **Numeric Arguments** commonly allow multiple uses of the same command. Just as c-P takes you up one previous line, c-4 c-2 c-P takes you up 42 previous lines. Similarly, c-4 c-2 M enters the character M in your buffer 42 times.

- **Additional File and Buffer Operations** use the following extended, or m-X commands: Rename Buffer, Save All Files, Kill Some Buffers, Kill All Buffers, Kill or Save Buffers, Copy File, Source Compare, Source Compare Merge, and others.

- **Splitting the Screen** is done with the m-X Split Screen command. Move from buffer to buffer with the mouse.

- **Formatting Text** allows for *italics*, **boldface** and a number of other improvements in the appearance of text files. Use the m-X Format File, Format Buffer, or Format Region command to see the results on the screen. Press c-U before entering these commands to see the results on a printer.

- **Documentation References**

14.3 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. 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. Pressing c-U is like typing a numeric argument of 4. If you press c-U more than once, your numeric argument is 4 to the *n*th power, where *n* is the number of times you press 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.

Numeric arguments are generally used to 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 60 asterisks, you could type c-6 c-0 *, and 60 asterisks would be inserted at the current cursor position, just as if you had typed all 60 of them, one by one.

Note, however, that the behavior of some commands is changed by any numeric argument, regardless of what number you specify. For example, c-X B, Select Buffer, creates a new buffer when you precede it with any numeric argument; c-U c-X B, m-5 c-X B, and c-m-9 c-X B all produce the same result.

You should type the numeric argument with the same modifier key as the command. This saves keystrokes, as you can hold down the modifier key while you type both the numeric argument and the command. However, there is nothing wrong with using some other modifier key to specify the numeric argument.

14.4 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 All Files

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 are not 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 current region to the end of that buffer.

m-X Append To File

Prompts for a file name and adds the current region to the end of that file.

m-X Prepend To File

Prompts for a file name and adds the current region to the beginning of that file.

Copying

m-X Copy File — Prompts for the name of a file to copy and the name of a file to which to copy.

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. This 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

By adding a new buffer name and a c-RETURN you can create a new buffer.

```
c-X B<name>c-<RETURN>
```

14.5 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. However, several commands are 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, plus options for creating new buffers and reading in files. There are also three menu 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 cancelled the split-screen or split-screen element.

14.6 Formatting Text

This section is for people who want to write "pretty" manuscripts. Unless you want to write large segments of text that others have to read, you can skip this section. This section covers only a very small portion of the formatting commands to give you an idea of how they work.

How to do it

Two steps are required for the production of such text.

- You write text in an editor buffer and embed formatting instructions in the text.
- You use the appropriate extended command to display the text in formatted style.

Formatting Instructions

Formatting instructions all begin with an @. The next part of the instruction can take one of two forms.

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)

This 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 face type.
- The parentheses () enclose a small **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 am of those who like to stay late at the cafe," the older
waiter said. "With all those who do not want to go to bed.
With all those who need a light for the night."
@end(b)
```

The (b) causes all of the text between @begin(b) and @end(b) to be displayed in bold face.

How to Display Your Formatted Text

Use the extended command `m-X Format Buffer` to see the formatted text in the editor. The above text would look like this:

"I am of those who like to stay late at the cafe," the older waiter said. "With all those who do not want to go to bed. With all those who need a light for the night."

If you have an LGP1 or LGP2 printer, you can get a hardcopy by using:

```
m-0 m-X Format Buffer
```

and specifying the appropriate hardcopy device.

14.7 Documentation References

- **Numeric Arguments**
  Vol. 3 pages 24-25
- **Additional File and Buffer Operations**
  Vol. 3 pages 119-128
- **Splitting the Screen**
  Vol. 3 page 130
- **Formatting Text**
  Vol. 3 pages 33-39

15. Additional Files

15.1 Purpose

In addition to the Command Processor commands, you can handle files through the m-X Dired command in Zmacs, or through the File System Editor, reached *via* SELECT F. This chapter also describes properties of files and directories.

15.2 Contents

- **The Directory Editor - Dired** is part of Zmacs and is reached by the Command Processor Edit Directory command, the m-X Dired command, or c-X D. Once in Dired, press HELP to see what it can tell you and do for you.
- **Walk-through for Dired**
- **The File System Editor - FSEdit** is reached by SELECT F and allows you to handle your files through a menu interface. Click on [Tree Edit home dir] to take a look at your directory.
- **Walk-through for FSEdit**
- **File Properties** can be edited to state how many copies of a particular file you want to keep and to state that a certain file should not be deleted as well to state other characteristics of files. You can do this editing through Zmacs, Dired, or FSEdit.
- **Directory Properties** can be edited to set how often you want to get rid of excess files in your directory, how many copies of each file you want to keep, and whether you want the whole directory protected against deletion. You can do this editing through Zmacs, Dired, or FSEdit.
- **Documentation References**

15.3 The Directory Editor - Dired

Zmacs includes 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 on 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 15.1).

You can also press the `HELP` key when you are prompted for an argument in the minibuffer. A summary of the operation in progress appears on the screen that tells you what you are being prompted for and what command you are executing.

Some commands take effect only when you press `Q` to exit Dired. When you press `Q`, 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 undeletes the files and then expunges the directory and prints the files.
- `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.

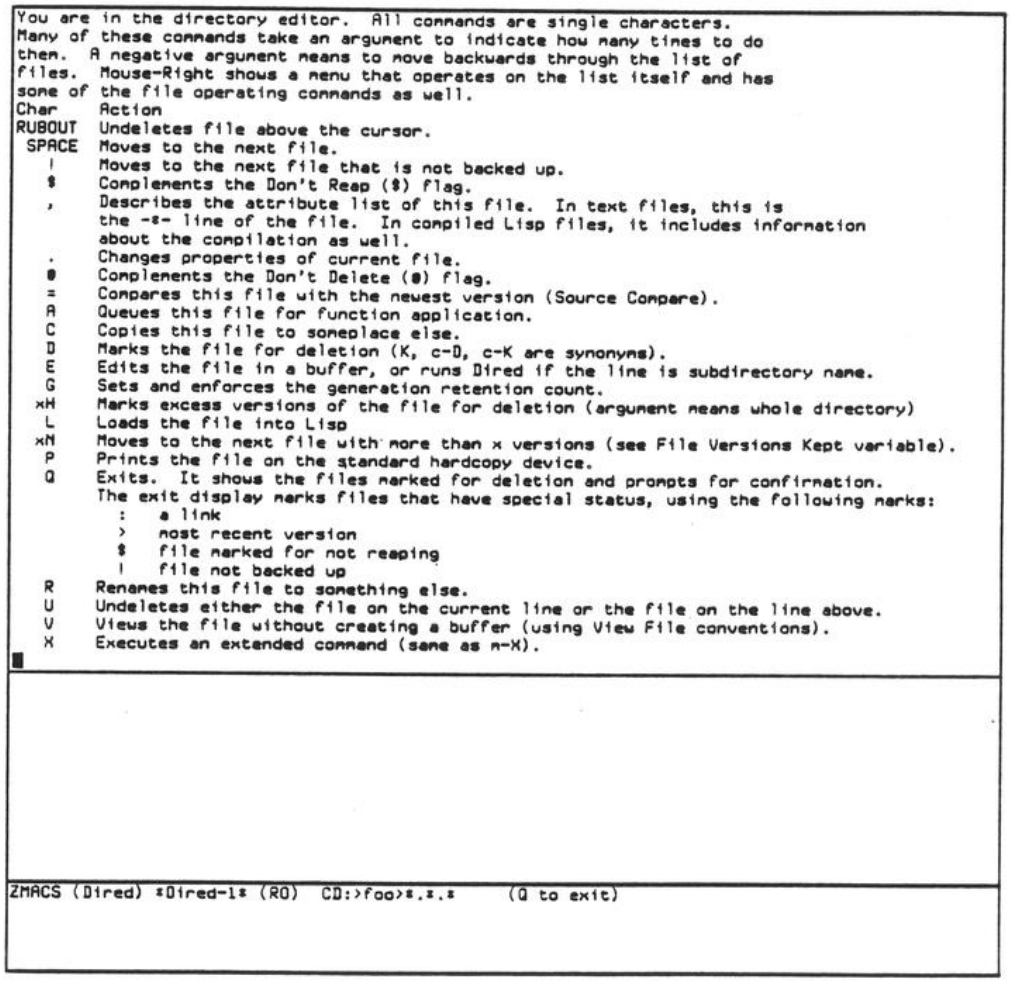


Figure 15.1 Type m-X Dired *to see the directory editor in Zmacs. Then press* HELP *to see what Dired can do.*

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 (Figure 15.2) 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.

```
SYMBOLICS1:>Muffy>*.*.*
  745 free, 59235/59980 used (98%, 2 partitions)
  2 blocks in the files listed
    1 editing.directory.1     1   DIRECTORY !   04/29/86 16:18:53 R=04/29/86
    2 notes.text.1            1     31(8)         12/24/85 13:57:22 (12/24/85)
```

Figure 15.2 The **Dired listing** *gives you the name and size of a file, what type it is, its creation date, whether or not it has been backed up, and if you can delete it. See the text for more information.*

If the cursor were on line 1, all Dired commands would refer to the directory `editing>`.
If the cursor were on line 2, all Dired commands would refer to the file `notes.text.1`

`D` On a Symbolics computer, 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, 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>*.

`P` Marks the file for printing. When you exit Dired, you are asked

to confirm that the marked files should be printed. If you do not have a printer, do not try to hardcopy files.

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 — Upon exiting Dired with *Q*, typing an *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><RETURN>
```

15.4 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-id.directory

   (The line might 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. Press `SELECT L`.

11. Type

    `Show Directory` *your-file-server*`:>`*your-login-id*`>*.*.*`

    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. Press `E` again.

16. Once again, move the cursor to the line that says:

    ```
    zmacs-text-2.text
    ```

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.

15.5 The File System Editor - FSEdit

FSEdit is a program that lets you examine directories and the files contained in them.

You can access FSEdit by pressing `SELECT F` or clicking on [File System] in the System menu. See Figure 15.3. A menu appears at the top of the screen, above a window with a `Command:` prompt. Although you can type Command Processor commands to an FSEdit window, you generally do that from a Lisp Listener.

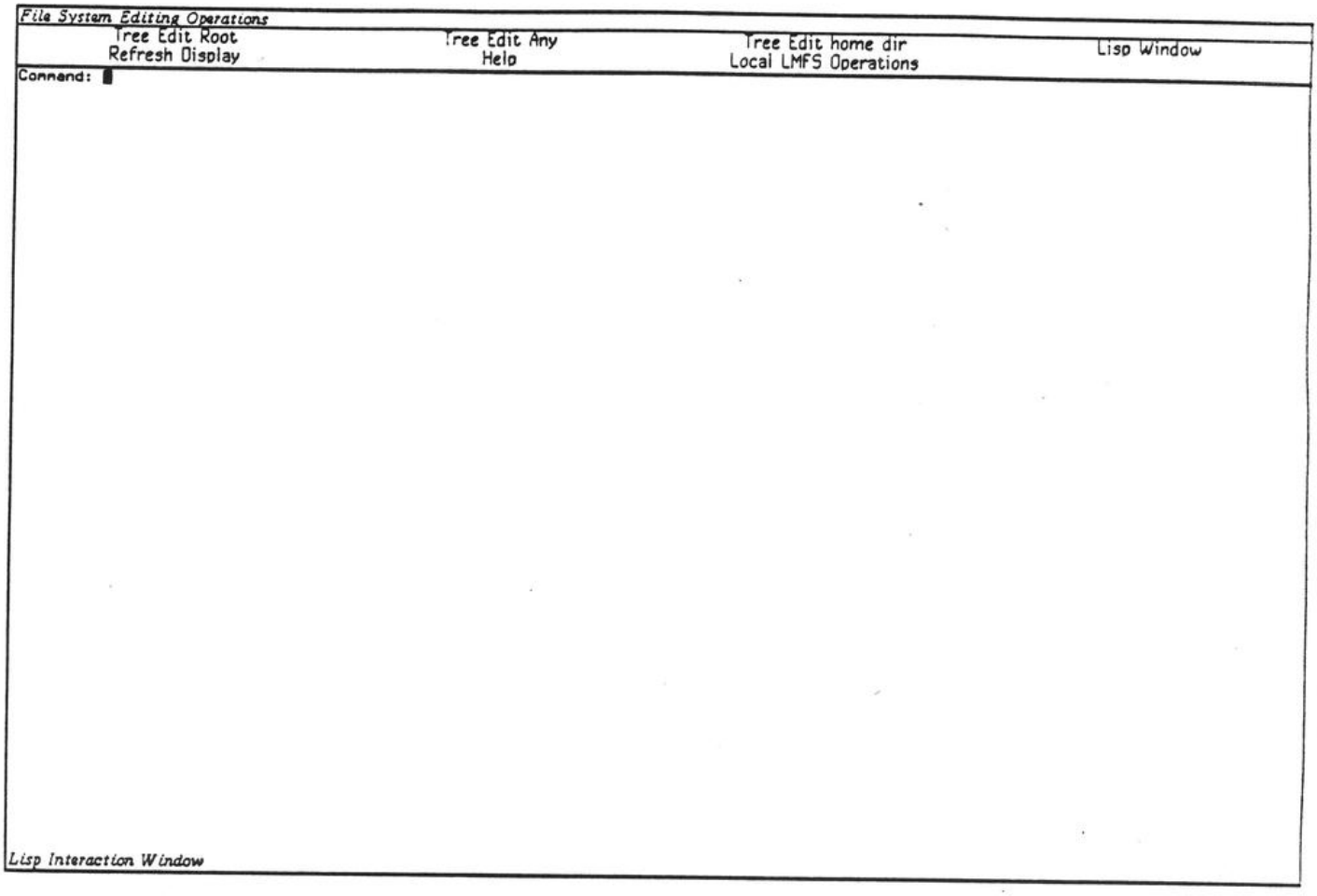


Figure 15.3 The **File System Editor** *is an alternative to Dired. Press* SELECT F *to select it and use the mouse to issue commands.* Tree Edit home dir *edits your home directory.*

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; 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 looks like this:

```
letter.text.2   2   5252(8)  !   02/14/86 10:55:18 (02/23/86)
```

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 Object* has no meaning when you move the mouse over a file name. Click left and see that nothing happens.
- Middle - *Close Containing Object* means close this file's directory. If you click middle, you will notice that all of the file names disappear. Don't despair – they're easily retrieved! All you have to do is move the mouse cursor over the name of your directory, which should be on the top line (below the menu), and click left. This reopens your directory and reads in the files. You can also click on [Tree Edit home dir] again.
- 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 will 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, and your deleted files will disappear. Therefore, do not mark a file for deletion unless you're certain that you no longer need it.

[View] 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, first prompting 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.

[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 for now practice using these commands only on your own directory. Click left on *Tree Edit home dir*, put the mouse on your directory so that you can see the box around your directory name, and read the mouse documentation line.

- Left - *Open Object* opens the directory and lists the files that are contained in it.

- Middle - *Close Containing Object* means close this directory's directory. If you click middle, you will notice that all of the file names disappear. All you have to do is move the mouse cursor over the name of your directory, which should be on the top line (below the menu), and click left. This reopens your directory and reads the information about the files. Or you can click on [Tree Edit home dir] again.

- 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 that are contained in it. 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 and prompts you for a new directory name.

15.6 Walk-through for FSEdit

1. Press SELECT F to select the File System Editor.

2. Move the mouse over [Tree Edit Root] and click middle. A display of the directories on your file server will appear.

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 [View].

9. Press SPACE to make the contents of the file disappear from the screen.

10. Click middle 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 can be seen or changed by users. 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, and the rest are marked for deletion. This can be useful when you've saved out many versions of a file, and don't need early versions that would just be taking up space.

The *Don't Delete* protection flag is generally set to *No*, but you can change it to *Yes* if you wish to protect an important file from being accidentally deleted. Later, you can set this flag back to *No* when you wish to dispose of the file.

- **Editing properties**

  All these commands cause a pop-up 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 you want 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 `Do It` to update the file's properties or `Abort` to abort.

  Zmacs:

  `m-X Change File Properties<RETURN>`*<pathname>*`<RETURN>`

  Dired:

  .

  (This is the character "dot" or "period".)

  FSEdit:

```
[Edit Properties]
```

- **Viewing properties**

  These commands only display a list of properties and their values.

  Zmacs:

  ```
  m-X View File Properties<RETURN><pathname><RETURN>
  ```

  FSEdit:

  ```
  [View Properties]
  ```

Directory Properties

Directories also have several properties that can be seen or changed by users. Some of these are: Auto expunge interval, Default Generation (version) Retention Count, Author, Creation, Modification, Reference Dates, and Protection Flags. 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, you can set it 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, or one week, and so on.

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, and the rest are marked for deletion. This can be useful when you've saved out many versions of a file, and don't need early versions that 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. Later, you can set this flag back to *No* when you wish to get rid of the directory.

- **Editing properties**

  All of these commands cause a pop-up 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 `Do It` to update the directory's properties or `Abort` to abort.

Zmacs:

`m-X Change File Properties<RETURN><pathname><RETURN>`

Dired:

.

(This is the character "dot" or "period".)

FSEdit:

`Edit Properties`

- **Viewing properties**

These commands just display a list of properties and their values. Specify the *.directory* extension in pathnames to view the properties of a directory.

Zmacs:

`m-X View File Properties<RETURN><pathname><RETURN>`

FSEdit:

`[View Properties]`

15.7 Documentation References

- **The Directory Editor - Dired**
  Vol. 3 pages 131-153
- **The File System Editor - FSEdit**
  Vol. 5 pages 231-237
- **File Properties**
  Vol. 5 pages 204-208
- **Directory Properties**
  Vol. 5 pages 117-120

16. Additional Window Features

16.1 Purpose

This chapter introduces you to some of the basic concepts of the window system and demonstrates more window features.

16.2 Contents

- **Introduction**
- The **Basic Concepts** of the window system include the fact that windows represent the primary means for programs to accept input and present output. Most of your interaction with the Symbolics computer is through the window system.
- **Programs**
- **Customizing the CP** covers the ability to set the CP to accept commands only, Lisp forms only, commands and then forms, or forms and then commands.
- **Walk-through for Customizing the CP**
- **Flashy Scrolling** is indicated when you see *More above* or *More below* in a window. When you see such indications, move the mouse cursor near the words and bump the arrow against the top or bottom of the screen to scroll up or down.
- **Making Your Own Windows** can be done through the System menu.
- **Walk-through for Making Your Own Windows**
- **Getting Out of Trouble** introduces a number of problems you might run into and ways to solve them.
- **Documentation References**

16.3 Introduction

You now know the things necessary to use the window system. In this chapter, we introduce the basic concepts of the window system as well as some additional features that are part of the window system and some ways to get out of trouble if something goes wrong.

16.4 Basic Concepts

What is the Window System?

The window system is a collection of software that provides an interface between humans and the Symbolics software. All input and output goes through the window system. The window system software controls both keyboard and mouse input and both graphic and textual output to the screen.

Why is there a Window System?

Symbolics provides the window system so that users 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 piece of screen real estate 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 might, 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.

16.5 Programs

In this document, 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 The Symbolics machine provides 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 machine, 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* 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 isn't still there. 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, like the Document Examiner.

Activities The combination of a process and a window makes up an *activity*. An activity is a program that you can choose to use for a while. After a time, you can choose to use a different activity.

16.6 Customizing the CP

You can change the way Command Processor behaves if you want to. The `Set Command Processor` command allows you to change the CP *mode* and the CP *prompt*.

The CP mode

The CP has four *modes* that determine how it interprets what you type. The modes are:

Command-Only The CP 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 CP assumes you are typing a command. Otherwise, it assumes you are typing a Lisp form.

  If you want the CP to treat a word that begins with an alphabetic character as a Lisp form, 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 CP 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 CP 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 CP Prompt

By default, the CP prompt is the string `Command:`. 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 CP mode to Form-Preferred and the prompt to a right angle-bracket (>)

16.7 Walk-through for Customizing the CP

1. Type:

```
Se C P<SPACE>
```

2. You are now prompted for a mode. Type:

   ```
   C<SPACE>
   ```

   The CP informs you that `"C" is ambiguous.`

3. Press `RUBOUT` 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, 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.

16.8 Flashy Scrolling

Windows 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 is changed to a thick single-headed arrow pointing up. Each time the thick cursor is bumped against the sensitive part of the top edge, there is one line of downward scrolling. 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. In an editor window, it is necessary to bump against the top or bottom at the right; other windows have regions in the center of the top and bottom that are sensitized for flashy scrolling.

16.9 Making Your Own Windows

It is possible (in fact, easy) on a Symbolics machine 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.

16.10 Walk-Through for Making Your Own Windows

The leftmost 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 line.

1. Cold boot your machine before starting this exercise.

2. Press SELECT E to select the editor window.

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 will be located at the cursor location. If you click right, the upper left corner will be 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 on [Edit Screen] on the System menu.

5. Click on [Reshape], or [Move Multiple], or [Move Single], or [Move Window] 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 that you wish to move or alter and click appropriately.

7. Follow instructions in the mouse documentation line. Continue editing the screen until you wish to exit (in which case click on [Exit] in the Edit Screen menu).

16.11 Getting Out of Trouble

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, because it contains useful information. If you do not understand the message, you can usually press the ABORT key to fix the problem. If ABORT doesn't work, try the solutions listed below.

Things That Can Get You Stuck

- Killing windows.
- Overlapping windows.
- 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 wants 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 wants 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 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, Unlock all window system locks? is usually one of your options. You should answer Yes.

16.12 Documentation References

- **Basic Concepts**
  Vol.7 pages 73-78
- **Programs**
  Vol.7 pages 73-78
- **Customizing the CP**
  Vol. 1 page 32
- **Flashy Scrolling**
  Vol. 1 pages 49-50
- **Making Your Own Windows**
  Vol. 7 pages 75-76
- **Getting Out of Trouble**
  Vol. 1 pages 43-45, 143

17. The Namespace

17.1 Purpose

The namespace database allows computers in a network to share information. This information includes the names and network addresses of other computers, the names of available printers, and the names of individual users and their mail addresses. This section explains a few things about the namespace database and shows you how to add yourself to the database.

17.2 Contents

- **The Namespace and Logging In** describes how the association of a user name with a file server works when you log in.
- **Adding Yourself to the Namespace** is a walk-through of the process.
- **Optional User Attributes** include birthdays, supervisors, personal mottoes, and so forth.
- **Documentation References**

17.3 The Namespace and Logging In

The association of your name with a particular *file-server* is made by the database known as the namespace. The namespace contains information about users, hosts, printers, and so on. In particular, the information about users includes the *home host* that is associated with a *user ID*. Your user ID is the name by which the machine knows you. Your home host is the machine on which you keep your files. The machine looks for your lispm-init file in the directory `homehost:>userID>`. This directory is your home directory. If you do not appear in the database, you are invited to add yourself when you first log in. You must do this only once.

17.4 Adding Yourself to the Namespace

To add yourself to the namespace, you use the *Namespace Editor*. If you have difficulty logging in because the machine does not recognize your user ID, you are

asked if you wish to add yourself to the namespace database. If you say yes, you are automatically put into the Namespace Editor, where you can edit your user object.

The pieces of information about a user are known as *attributes*. You must to enter four of these attributes to add yourself to the namespace. The four required attributes are marked with asterisks (*) after their names.

You can also access the Namespace Editor from the System menu. If you select it this way, you do not initially edit an object. You can use the menu options in the Namespace Editor to read in, modify, and save out objects. In this chapter, we will talk only 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.

How to add yourself to the namespace when you have a problem logging in

1. The first thing to do is make sure that you have typed your user ID 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 need to do this only once.

2. When you try to log in, if you are not part of the user database or if you have mistyped your user ID, you see the following:

```
The user named "silva" was not found.:
Do you want to log in as silva on some specific host? (Y, N, or R)
```

   You should type the letter Y.

3. You then see:

```
Host to log in to:
```

   You should type the name of your *file-server*, the machine on which your files are to be stored. All the hosts that are known by the namespace database have names. This user typed

```
Cerridwyn<RETURN>
```

4. You then see:

```
No init file: The directory >SILVA does not exist.
For CD:>silva>lispm-init.bin
Do you wish to add silva to the user database? (Y or N)
```

You should type the letter Y.

5. You are now presented with a new, smaller window (Figure 17.1). This is the Namespace Editor. Here you are told by the namespace database exactly how to add a *user object* for yourself to the namespace database. Remember, follow the steps outlined here; do not experiment with the Namespace Editor.

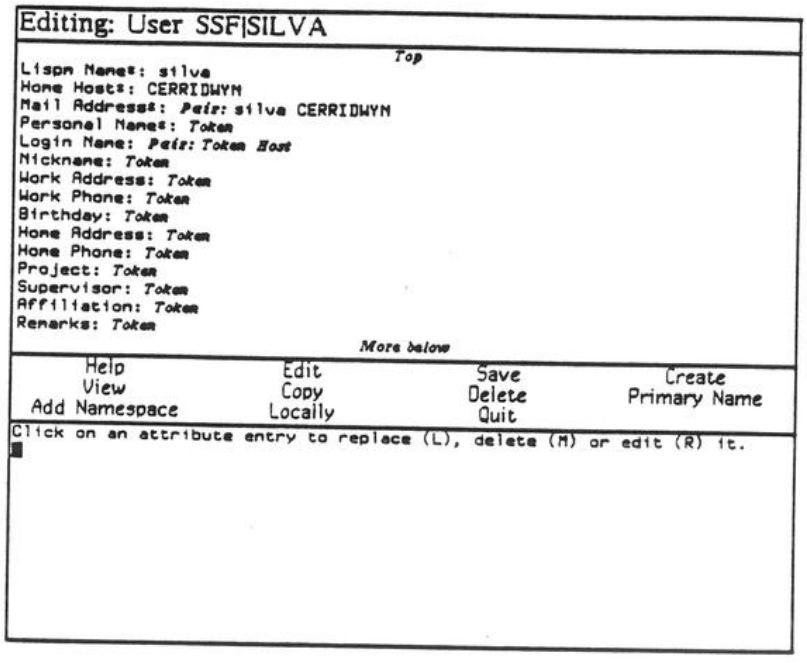


Figure 17.1 **The Namespace Editor** *is selected from the System menu. This shows the namespace entry for a particular user. It includes required and optional information about objects in the namespace, including users, printers, hosts, networks, sites, and namespaces.*

The cursor is in the lower part of the window. The upper part of the window contains attributes for users. The first four attributes in the list are required, and are marked by an asterisk (*). Note that the first three already have values:

```
Lispm Name*: silva
Home Host*: CERRIDWYN
Mail Address*: Pair: silva CERRIDWYN
Personal Name*: Token
```

The *Personal Name* attribute must be added. To do this, move the mouse until the mouse cursor is close to the word *Token* following `Personal Name:` . The cursor changes into a hollow rectangle surrounding *Token*. Carefully click left. The machine prompts you, in the lower part of the menu, for the personal name of the user. Type your name, followed by `RETURN`. Your name then appears in the upper part of the menu, next to `Personal Name:` .

6. Now move the mouse cursor to the [Save] option in the center of the menu. Click left. This saves the user object in the database.

7. After the message "the user named *Site|Login-name* has been saved" appears, move the mouse to the [Quit] option. Click left. The small window disappears, leaving you in Lisp Listener 1. You have now added yourself to the namespace, and you can log in normally.

17.5 Optional User Attributes

In the Namespace Editor, you should have seen 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.

17.6 Documentation References

- **The Namespace and Logging In**
  Vol. 1 pages 111-113
- **Adding Yourself to the Namespace**
  Vol. 1 pages 116-118
- **Optional User Attributes**
  Vol. 1 pages 113-115

18. Overview of the Machine

18.1 Purpose

The purpose of this chapter is to familiarize you with the "basic anatomy" of a Symbolics computer. We will discuss the parts of the machine and the way files are distributed on the disks. This chapter does not tell you how to do anything.

18.2 Contents

- **Basic Parts** describes the Symbolics computer. Remember that you are not working at a time-sharing terminal, you are controlling a complete computer through its console.
- **Parts of the Processor** are for the most part the usual parts – CPU, physical memory, hard disk, an so on – and one unusual part – the Front End Processor, or FEP.
- **About the FEP** tells you what the FEP does and warns you not to feel free to play around with FEP commands.
- **FEP Files in Contrast With LMFS Files** explains the difference and warns you once again **never delete your lmfs.file.**
- **Hello Files** are a form of boot files that load FEP load (.flod) files.
- **The FEP and Files** explains the different kinds of FEP files, which include boot files, world loads, paging files, and FEP load files.
- **A Closer Look at Booting** gives you a rundown of all the FEP commands necessary to boot a machine. You can put all these commands in a boot file.
- **Documentation References**

18.3 Basic Parts

Symbolics machines are single-user computers; there is one complete computer for each user. This means that Symbolics machines are not time-sharing. If more than one person is going to use the Symbolics computer you will be using, then the *people* are going to have to do the time-sharing.

There are many personal 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*, the big gray

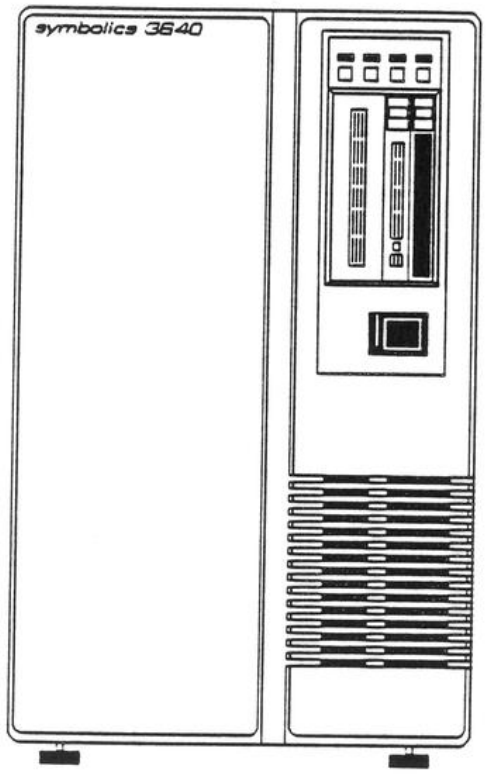


Figure 18.1 Front view of a **Symbolics 3640** *with a cartridge tape drive.*

box (Figure 18.1). 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*.

18.4 Parts of the Processor

The processor unit contains the following major parts:

1. Lisp processor (CPU)
2. Front End Processor (FEP)
3. Physical Memory (RAM)
4. Hard disk
5. Connections and other things

A brief description of each of these hardware components follows.

Lisp Processor The Lisp Processor is the CPU for the Symbolics computer. It consists of three major boards, the *datapath* (DP) board, the *sequencer* (SQ) board, and the *memory controller* (MC) board. Some models have an *Instruction Fetch Unit* (IFU) in place of the memory controller and an *extended sequencer* (XSQ) in place of the normal sequencer. These three boards are the heart of the Symbolics computer.

Front End Processor The Front End Processor (FEP) is a separate 32-bit computer (M68000). 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 only used 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 (Figure 18.2), so that you can back up your file system and 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

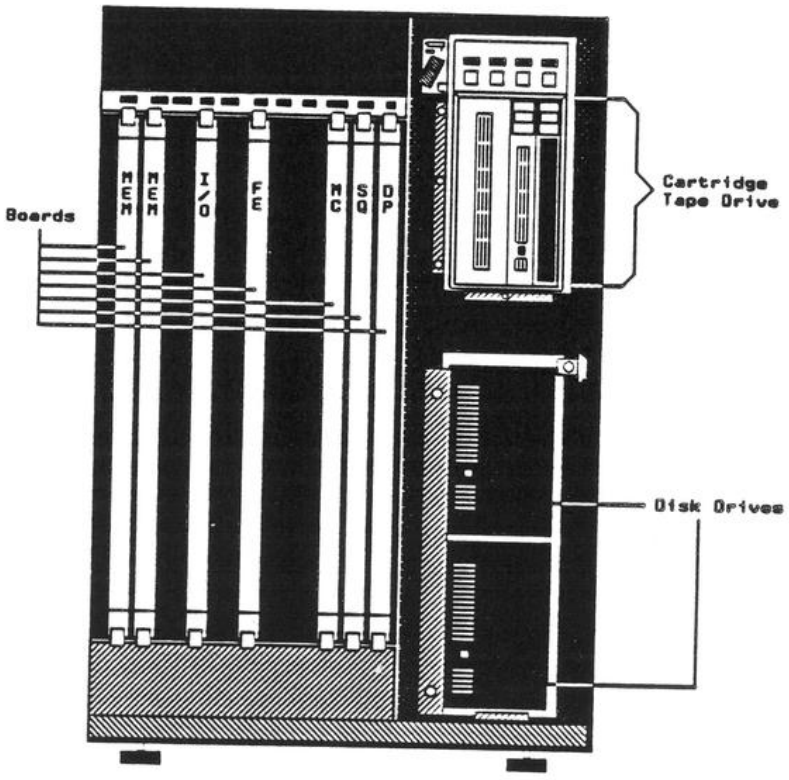


Figure 18.2 Open-door view of a Symbolics 3640, showing the **basic boards**, *the cartridge tape, and the disk drives. The disk drives serve as the system disk, including provision of virtual memory. If you are in a site with more than one computer, your actual file storage might be elsewhere.*

cable. Several machines can be connected together by network cables.

18.5 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 command you need right now is the *Boot* command, which executes 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.
- A separate processor gives you access to the FEP file system and other parts of your Symbolics computer, even when the main Lisp Processor is not running.

18.6 FEP Files in Contrast With LMFS Files

Each disk on the Symbolics computer has a separate file system called the FEP file system. The disk is divided up into *FEP files*. One of these files is the `boot.boot` file that 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 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 machine 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 local machine. All of the partitions together make up a single LMFS.

At many sites, there is one computer on the network with a lot of disk space (the *file server*), and that machine 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 can 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.

18.7 Hello files

If your FEP prompt is `FEP Command:` (as opposed to `Fep>`), whenever your machine is powered up you must give the FEP's Hello command. 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.

18.8 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. FEP files should only be altered/deleted by the site administrator.

The FEP file system contains several kinds of files:

1. *.load* - world load files.

2. *.mic* - microcode files.

3. *.page* - paging files.

4. *.boot* - boot files.

5. *.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.

6. *.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.

18.9 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.

Booting involves several FEP commands:

1. *Clear Machine.* This erases anything from the previous time the Lisp Processor was run. You must clear the machine before loading the microcode.

2. *Mount 1.* 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.

3. *Load Microcode.* Microcode is software that is loaded into the Lisp Processor to customize internally the way that the Lisp Processor works. A different microcode exists for each of the different hardware configurations.

4. *Load World.* At least one FEP file on your disk is a *world load* file.

   - A world load file is a snapshot of a running Lisp Environment from some time in the past. Symbolics distributes a basic world load file with each Symbolics computer.

   - 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.

5. *Clear Paging.* We recommend that you clear paging before you add paging even if you have only one paging file. However, it is required that you clear paging files if you have more than one paging file.

6. *Add Paging.* 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 machine.

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 in the context of the particular world load file that was loaded. Note well: you must have a world load file in order to start the Lisp Processor. **Never delete all your world load files.**

After you boot, the Lisp Processor is running. All normal operations, such as editing files, and compiling and running programs, are done using the Lisp Processor. Besides booting, only a few irregular operations are executed by the FEP.

18.10 Documentation References

- **About the FEP**
  Vol. 1 pages 197-210
- **FEP Files in contrast with LMFS Files**
  Vol. 5 pages 203-219
- **The FEP and Files**
  Vol. 5 pages 221-229
- **A Closer Look at Booting**
  Vol. 1 pages 198-199

Pocket Guides

19. Pocket Guide for Logging In

19.1 To Log In

Type:

Login *user-name*<RETURN>

to the Command Processor.

19.2 To Log Out

Type:

Logout<RETURN>

to the Command Processor.

19.3 To Cold Boot

1. Log out
2. Type:

   Halt Machine<RETURN>

3. Type either:

   Boot<RETURN> or Boot >*boot-file-name*.boot<RETURN>

20. Pocket Guide for Selecting Activities

20.1 SELECT Key Options

λ	Common Lisp Listener. The letter λ is obtained by pressing SY-SH-L.
C	Converse.
D	Document Examiner.
E	Editor (Zmacs).
F	File system maintenance (FSMaint or FSEdit).
I	Inspector.
L	Lisp Listener.
M	Zmail.
N	Notifications.
P	Peek.
T	Terminal.
X	Flavor Examiner.

20.2 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.
[Font Edit]	The Font Editor.
[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.
[Namespace]	The Namespace Editor.
[Flavor Examiner]	The Flavor Examiner. Also available on SELECT X.
[Document Examiner]	The Document Examiner. Also available on SELECT D.
[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.

21. Pocket Guide for Input Editor Commands

21.1 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.

22. Pocket Guide for Keyboard Keys

22.1 Keyboard Keys

ESCAPE	Show recent command history.
c-0 ESCAPE	Show entire command history.
END	Signal that you have finished typing a command.
RETURN	Signal that you have finished typing a command.
SPACE	Ask the Command Processor to complete the word you just typed as well as the previous words, as far as possible.
COMPLETE	Ask the Command Processor to complete the entire command you are typing, as far as possible.
CLEAR INPUT	Throw away all characters you have typed since the last prompt.

22.2 Some Useful Keystrokes

FUNCTION	With another key, lets you do useful things with different parts of the system.
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	Like m-ABORT, but 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, lets you get 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.

23. Pocket Guide for Zmacs

	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 23.1 Chart of Zmacs Commands

23.1 File Operations

c-X c-F	Find or Create Buffer
c-X c-S	Save Buffer
c-X c-W	Write Buffer

23.2 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

23.3 Buffer Operations

c-X B	Select Buffer
c-X c-B	List Buffers
c-X K	Kill Buffer

m-X Hardcopy Buffer
m-X Kill or Save Buffers
m-X Insert Buffer

23.4 Region Operations

Mark region	Hold left mouse button and drag to mark region
c-W	Kill Region
m-W	Copy Region into Kill History
c-Y	Yank Region

24. Pocket Guide for File Operations

24.1 File Operations

	Dired	FSEdit	CP
Create	--	--	Edit File <*pathname*>
Delete	D	[Delete]	Delete File <*pathname*>
Undelete	U	[Undelete]	Undelete File <*pathname*>
Edit	E	[Edit File]	Edit File <*pathname*>
Load	L	[Load]	Load File <*pathname*>
Rename	R<*pathname*>	[Rename]	Rename File <*pathname*>
Copy	C<*pathname*>	--	Copy File <*pathname*>
Hardcopy	P	[Hardcopy]	Hardcopy File <*pathname*>
View	V	[View]	Show File <*pathname*>

24.2 Directory Operations

	Dired	FSEdit	CP
Create	--	[Create Inferior Directory]	Create Directory <*pathname*>
Delete	D	[Delete]	Delete File <*pathname*>
Undelete	U	[Undelete]	Undelete File <*pathname*>
Edit	E	--	Edit Directory <*pathname*>
Expunge	Q E	[Expunge]	Expunge Directory <*pathname*>

View	V	--	Show Directory *<pathname>*

24.3 Zmacs

All of the commands above are also available in Zmacs. Press `m-X` and then enter the name of the CP command you want. `Edit File` is also available as `c-X c-F`. `Edit Directory` is also available as `m-X Dired`.

25. Pocket Guide for the Document Examiner

25.1 Menu Mouse Clicks

[Help]	Left: Brief Command Summary Middle: Show Full Document Examiner Documentation
[Find]	Left: Find Whole Word Candidates Middle: Find Initial Substring Candidates Right: Find Any Candidates
[Show]	Left: Show Documentation Middle: Show Overview Right: Find Table of Contents
[Select]	Left: Select Candidate List
[Viewer]	Left: Select Viewer Middle: Remove Viewer Right: Hardcopy Viewer
[Private]	Left: Read Private Document Middle: Load Private Document Right: Save Private Document

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, until there is more than one possible command that 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; relevant only to Lisp programmers.
- *Default* - A value for an argument that is supplied by the system, often the last value that argument had. You can choose to take or override a default value.
- *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 pressing 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.

- *Hierarchical system* - One in which everything can be said to be either below, above, or on the same level as every 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.

- *Host* - Any computer, not necessarily a Symbolics machine.

- *Incremental searching* - Finding a string of characters in a Zmacs buffer, 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 generally, but not always, removes the text from the buffer.

- *Lisp form* - A piece of Lisp code.

- *Lisp Listener* - A window that accepts Command Processor commands and Lisp forms (either Zetalisp or Common Lisp).

- *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 customizations, written in Lisp. The file is stored in your home directory and is run when you log in.

- *LMFS file* - A file maintained by 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* - 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 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-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.

- *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 active only 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 user ID.

- *Subdirectory* - A directory contained in another directory.

- *System menu* - The main menu for the system. This menu is available anywhere in the window system by pressing the SHIFT key while clicking the right mouse button.

- *The local machine* - The Symbolics machine you are using, as opposed to any other host that you can access over the network.

- *The machine* - Either the entire Symbolics machine, the Lisp Processor, or the basic software included in the world load file.

- *The system* - Either the entire Lisp Machine, the Lisp Processor, or the basic 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 with the machine yet.

- *Window system* - A collection of software that provides an interface between the user and the lower-level Symbolics 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* - The screen-oriented text editor on the Symbolics computer.