sun®
microsystems

Setting Up Your SunOS™ Environment: Beginner's Guide

Part Number: 800-1704-10
Revision A, of 9 May 1988

sun®
microsystems

Setting Up Your SunOS™ Environment: Beginner's Guide

Part Number: 800-1704-10
Revision A, of 9 May 1988

Contents

Chapter 1 Overview **3**

1.1. The SunOS Environment 3

1.2. Interactive Programs and Setup Files 3

1.3. Installing the Setup Files In Your Home Directory 4

1.4. Setting Up Your Terminal 5

1.5. The File System Hierarchy 7

Chapter 2 The C Shell and the `.cshrc` File **11**

2.1. Selecting C Shell Features 11

2.2. A Sample `.cshrc` File 12

Explanation of Command Lines 16

Chapter 3 The C Shell and the `.login` File **27**

3.1. A Sample `.login` File 28

Explanation of Command Lines 30

Chapter 4 The C Shell and the `.logout` File **35**

4.1. A Sample `.logout` File 35

Explanation of Command Lines 37

Chapter 5 `vi` and the `.exrc` File **41**

5.1. Setting Options While in `vi` 41

5.2. A Sample `.exrc` File 43

Explanation of Command Lines 45

Index 47

Figures

Figure 1-1 Check for These Files 4

Figure 1-2 Installing Set-Up Files 4

Figure 1-3 Viewing All Terminal Characteristics 5

Figure 1-4 Selected Terminal Characteristics 5

Figure 1-5 Using `stty` 6

Figure 2-1 The `set` Command 11

Figure 2-2 The `unset` Command 11

Figure 2-3 The `history` Command 12

Figure 2-4 Changing Your Prompt 12

Figure 3-1 The `setenv` Command 27

Figure 3-2 The `printenv` Command 28

Figure 3-3 `unsetenv` 28

Figure 5-1 Using `set` to See Current Options 41

Figure 5-2 Using `set all` 42

Figure 5-3 Viewing Options 42

Figure 5-4 Turning an Option Off 42

Preface

This manual describes the setup files for the C shell command interpreter, and the interactive editor `vi`. Each of these files is read in automatically by the appropriate program, and contains commands and instructions to set up (or disable) various features of that program.

In this manual, the term *environment* is loosely defined as the various options and features that affect how the system or interactive program responds to you.

There is a sample of each setup file and a line-by-line explanation of its contents. Culled from a variety of expert users, these files contain some very convenient combinations of features and commands. Most importantly, these samples provide a starting point from which you can begin tailoring the system to your specific needs and style.

Chapter 1 is an overview of various setup files and a description of how they are used by the interactive programs.

Chapter 2 describes the `.cshrc` file for the C shell.

Chapter 3 describes the `.login` file for the C shell.

Chapter 4 describes the `.logout` file for the C shell.

Chapter 5 describes the `.exrc` file for `vi` (and the line editor `ex`).

Online copies of the sample files are located in:

`.cshrc`	`/usr/lib/Cshrc`
`.login`	`/usr/lib/Login`
`.logout`	`/usr/lib/Logout`
`.exrc`	`/usr/lib/Exrc`

Prerequisite Documents

You should be familiar with the contents of the following manuals before you read this one (this is a suggested reading order):

Getting Started with SunOS: Beginner's Guide

SunView 1 Beginner's Guide

Mail and Messages: Beginner's Guide

Setting Up Your SunOS Environment: Beginner's Guide

Companion Documents

You should also read the following manuals:

Doing More with SunOS: Beginner's Guide

Self-Help with Problems: Beginner's Guide

SunOS Reference Manual

1

Overview

Overview 3

1.1. The SunOS Environment 3

1.2. Interactive Programs and Setup Files 3

1.3. Installing the Setup Files In Your Home Directory 4

1.4. Setting Up Your Terminal 5

1.5. The File System Hierarchy 7

1

Overview

If you have been reading through the other Beginner's Guides, and have been using the system for a while, you have probably discovered features that you like and features you *would* like. The interactive programs that you have used so far have many optional features that you may not know about. This manual describes a number of these features, and how set things up so that you get the features you want automatically.

1.1. The SunOS Environment

When working with the system, the interactive program that is currently running on your terminal provides a context in which you accomplish your work. When you first log in, you are said to be "in" the command interpreter or *shell.* When you change directories, you are said to be "in" a new one. When using the text editor, you are said to be "in" `vi`.

While in the shell, you typically run the commands described in the *SunOS Reference Manual.* When in `vi`, you typically use text editing commands to read and modify files, as described in *Getting Started with SunOS: Beginner's Guide* While in `mail` or Mail Tool, you typically use commands to read and dispose of messages, or to compose and post messages.

Actually *you* are probably in your office while all this computing activity is going on, but the metaphor is helpful.

In keeping with this analogy you can think of the environment as characteristics of the system and the current interactive program that affect the way you work.

The technical meaning with respect to SunOS is more restricted: the *environment* is a body of information that is inherited from the "parent" of a process (program currently running). For example, the name of the current directory is passed along when you start *vi*, so it appears that you "stay in the same directory." See *Doing More with SunOS: Beginner's Guide* for more information about processes.

When you change interactive programs (by entering `mail` for instance), some characteristics, such as the commands that are accepted, also change. Others, such as the current working directory, may not. But most importantly, you usually wish to perform different sorts of tasks, so your expectations about what is a proper response from the system also change.

1.2. Interactive Programs and Setup Files

The interactive programs that you use most often, such as the C shell command interpreter, `mail`, and `vi`, each have a variety of optional settings that affect the way they respond to your commands. Unlike options that you type in on the command line (such as `ls -t`), you typically select interactive features by typing in commands while you are using that program.[1]

To save you time, most interactive programs allow you to put a list of commands (to select features that you normally want in effect) in a *setup* file in your home directory. Each program reads its setup file(s) automatically and performs the commands it contains.

Subsequent chapters present samples of the various setup files for `vi` and the C shell command interpreter. Each file is described command by command.

The `.mailrc` File

There is another file, called `.mailrc`, which modifies the behavior of `mail` and Mail Tool. Although it is similar to the other setup files discussed here, it is covered in the manual *Mail and Messages: Beginner's Guide*.

1.3. Installing the Setup Files In Your Home Directory

The sample files described in this manual are located in the directory `/usr/lib`, and you can copy them to your home directory. But before you do, check to see if there are setup files already present:

Figure 1-1 *Check for These Files*

```
venus% cd
venus% ls .cshrc .exrc .login .logout
.cshrc not found
.exrc not found
.login not found
.logout not found
venus%
```

CAUTION **If one or more of these files *are* present in your home directory, check with your System Administrator before you copy the samples.**

To copy the setup files, type in these commands:

Figure 1-2 *Installing Set-Up Files*

```
cd
cp /usr/lib/Cshrc   .cshrc
cp /usr/lib/Exrc    .exrc
cp /usr/lib/Login   .login
cp /usr/lib/Logout .logout
```

Once copied, you can modify them as you like using `vi`, or any other text editor.

These samples have been culled from the setup files of a variety of expert users. They contain many useful features and ideas. Even so, you will want to edit them to suit your own personal needs and tastes, and to remove references to features that you don't want. Generally speaking, the smaller the setup file is, the

[1] Most interactive commands also have command-line options that you can specify. Refer to the entry for the command of interest in the *SunOS Reference Manual* for information about its command-line options.

faster the program will start up.

1.4. Setting Up Your Terminal

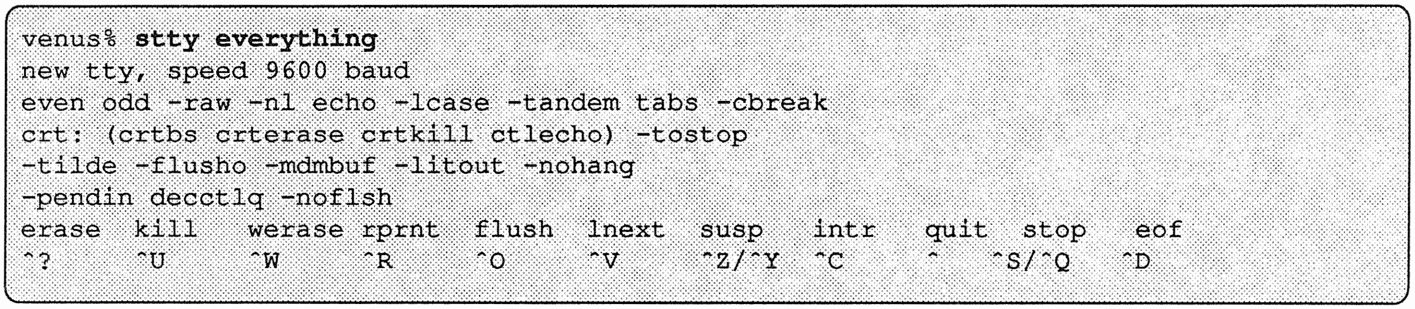
Underlying all of these considerations is the terminal you are using, and *its* characteristics. As indicated in *Getting Started with SunOS: Beginner's Guide* you can assign terminal functions such as `erase` and `kill` (erase the entire line typed in so far) to control keys such as [Del] and [Back Space].[2]

The command `stty` provides you with a means to set up these and other terminal characteristics. To find out what your current terminal characteristics are, type in the command:

```
stty everything
```

This gives you a list of all terminal characteristics currently in effect.

Figure 1-3 *Viewing All Terminal Characteristics*



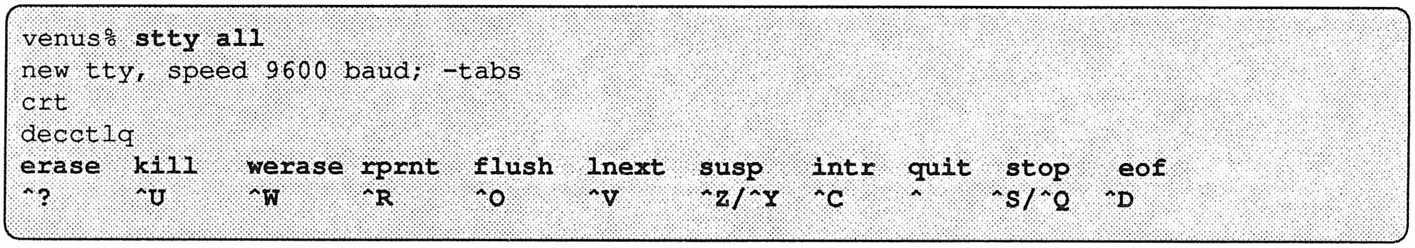
```
venus% stty everything
new tty, speed 9600 baud
even odd -raw -nl echo -lcase -tandem tabs -cbreak
crt: (crtbs crterase crtkill ctlecho) -tostop
-tilde -flusho -mdmbuf -litout -nohang
-pendin decctlq -noflsh
erase  kill   werase rprnt  flush  lnext  susp   intr   quit  stop   eof
^?     ^U     ^W     ^R     ^O     ^V     ^Z/^Y  ^C     ^   ^S/^Q   ^D
```

The command

```
stty all
```

displays a shorter list. For now, you can ignore all but the the last two lines,[3] which describe terminal-control functions and the keys they are set to. These are described below:

Figure 1-4 *Selected Terminal Characteristics*



```
venus% stty all
new tty, speed 9600 baud; -tabs
crt
decctlq
erase  kill   werase rprnt  flush  lnext  susp   intr   quit  stop   eof
^?     ^U     ^W     ^R     ^O     ^V     ^Z/^Y  ^C     ^   ^S/^Q   ^D
```

`erase`

Erase character. Backspace and erase one character. This is the [Del] key by default on some keyboards. On others, this function is assigned to the

[2] If you are using `SunView` on the Sun Workstation, you can assign commands and functions to the special function keys on the Sun Workstation keyboard. See *SunView 1 Beginner's Guide* for details.

[3] For more information about the remaining terminal characteristics, refer to `stty` in the *SunOS Reference Manual*.

Back Space key.

`kill`
Kill the whole line. Erase the entire command line typed in so far.

`werase`
Delete word. Erase the rightmost word typed in so far (back to a space or tab); usually assigned to Ctrl-W.

`rprnt`
Reprint. Reprint the line typed in so far. This is useful when you type ahead and the prompt gets displayed in the middle of your text.

`flush`
Wait for a keystroke. Stops terminal output until you press a key.

`lnext`
Literal next-character. Interprets the next (control) character as literal text.

`susp`
Suspend the program. Temporarily halts execution of the program currently running and puts it in the background. To resume execution of the command, type `%`.[4] When you type the suspend character, usually Ctrl-Z, in the middle of a C shell command-line, the shell ignores that line and issues a new prompt.

`intr`
Interrupt. Interrupt the program currently running.

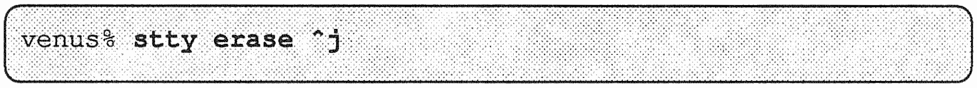
`quit`
Halt the current program and leave a binary image in a file called `core`.

`stop`
Stop the display. To resume, press Ctrl-Q.

`eof`
End-of-file. Send the program an end-of-file character.

To assign any of these functions to another control key, supply the function, a space and the circumflex character (`^`), followed by the new key, as an arguments to `stty`.

Figure 1-5 *Using* `stty`



```
venus% stty erase ^j
```

This avoids problems trying to type in a key that is already assigned.

To assign `erase` to the Back Space key, use the command:

```
stty erase ^h
```

[4] C shell only.

To assign werase to the Del key, use:

```
stty werase ^?
```

To assign kill to the Esc key, use:

```
stty kill ^[
```

Chapter 3 has more information about setting up terminal characteristics. For a description of other terminal characteristics that you can set up, refer to stty in the *SunOS Reference Manual.* To assign commands to the special function keys on the workstation keyboard, refer to *SunView 1 Beginner's Guide*

1.5. The File System Hierarchy

As mentioned above, the file system's directory hierarchy is a part of the "landscape" that you will want to become familiar with. You can see an example of a typical SunOS hierarchy by typing man hier or by looking under *hier* in Section 7 of the *SunOS Reference Manual.*

2

The C Shell and the .cshrc File

The C Shell and the .cshrc File **11**

2.1. Selecting C Shell Features 11

2.2. A Sample .cshrc File 12

Explanation of Command Lines 16

2

The C Shell and the `.cshrc` File

The C shell is one of the two command interpreters available on the Sun Workstation, and the one that we recommend for interactive use. Whenever you start running the C shell, such as when you log in or open a terminal (`shelltool`) window, the C shell looks for the `.cshrc` file in your home directory for its initial instructions. You can include in this file any command that you might ordinarily type on the command line.

The name is derived from `csh`, which is the program that uses it. The `rc` suffix is derived from the term "run command." Setup files ending in this suffix are read at the time you *run* the *command.* The "dot" at the beginning of the filename indicates that this file is to remain hidden from view when you do an `ls`. Setup files are rarely of interest unless you are editing them specifically, and in that case you already know the filename and directory location. (To list hidden files, use the `–a` option of `ls`.)

2.1. Selecting C Shell Features

While in the C shell, you can use the `set` command to select the options you would like. For instance, if you want the C shell to prevent you from accidentally logging out by typing a Ctrl-D, you can set the `ignoreeof` option (or, technically speaking, *variable*):

Figure 2-1 *The* `set` *Command*

```
venus% set ignoreeof
venus% ^D
Use "exit" to leave csh.
```

To turn off an option, use the `unset` command:

Figure 2-2 *The* `unset` *Command*

```
venus% unset ignoreeof
```

Some options allow you to supply a specific number or value. For instance, you can use the `history` variable to select the size of the history list; that is, the number of previous commands to remember:

Figure 2-3 *The* `history` *Command*

```
venus% set history = 40
```

Or, you can alter the prompt that the shell displays:

Figure 2-4 *Changing Your Prompt*

```
venus% set prompt = "THIS IS A VERY LONG PROMPT: "
THIS IS A VERY LONG PROMPT: ■
```

To see what options are currently in effect, and their values (if any) use the `set` command with no arguments:

```
venus% set
argv    ()
cdpath  (. .. /home/sam /home/sam/bin /usr/sam/src)
cwd     /home/sam/env
history 40
home    /home/sam
ignoreeof
noclobber
notify
path    (. /home/sam /home/sam/bin /usr/ucb /usr/bin /bin)
prompt  venus%
shell   /bin/csh
status  0
term    sun
user    sam
```

You can find descriptions of all C shell options (predefined variables) under `csh` in the *SunOS Reference Manual.* (Or type `man csh`.)

2.2. A Sample `.cshrc` File

The following pages contain an annotated listing of the sample `.cshrc` file located in `/usr/lib/Cshrc`. This is a very large sample file. Many of the commands and features it includes may not pertain to you, and we recommend that you delete those that you don't from your copy of the file.

A number of commands have been "commented out" by placing a pound-sign (#) to their left. The C shell will ignore these commands unless you remove the pound-sign character. Commands that are listed but commented out are felt to be interesting and educational, but not necessarily those that a beginner would use.

```
#################################################################
#
#          .cshrc file
#
#          initial setups for both interactive and noninteractive
#          C-Shells
#
#################################################################


#          set up search path

set lpath = ( )  #  add directories for local commands                 1
set path = (. ~ ~/bin /usr/local /usr/ucb /usr/bin /bin)               2

#          cd path

set lcd = ( )  #  add parents of frequently used directories           3
set cdpath = (.. ~ ~/bin ~/src $lcd)                                   4

#          set this for all shells

set noclobber                                                          5

#          aliases for all shells

alias cd              'cd \!*;echo $cwd'                               6
alias cp              'cp -i'                                          7
alias mv              'mv -i'                                          8
alias rm              'rm -i'                                          9
alias pwd             'echo $cwd'                                      10
#alias del            'rm -i'                                          11
#umask 002                                                             12

#          skip remaining setup if not an interactive shell

if ($?USER == 0 || $?prompt == 0) exit                                 13

#           settings  for interactive shells

set history=40                                                         14
set ignoreeof                                                          15
#set notify                                                            16
#set savehist=40                                                       17
#set prompt="% "                                                       18
#set prompt="`hostname`{`whoami`}\!: "                                 19
#set time=100                                                          20

#           commands for interactive shells
```

```
#date
#pwd

#          other aliases

#alias a              alias
#alias h              'history \!* | head -39 | more'
#alias u              unalias

#alias ^L             clear
#alias list           cat
#alias lock           lockscreen
#alias m              more
#alias mroe           more
#alias type           more

#alias .              'echo $cwd'
#alias ..             'set dot=$cwd;cd ..'
#alias ,              'cd $dot '

#alias dir            ls
#alias pdw            'echo $cwd'
#alias la             'ls -a'
#alias ll             'ls -la'
#alias ls             'ls -F'

#alias pd             dirs
#alias po             popd
#alias pp             pushd

#alias +w             'chmod go+w'
#alias -w             'chmod go-w'
#alias x              'chmod +x'

#alias j              'jobs -l'

#alias bye            logout
#alias ciao           logout
#alias adios          logout

#alias psg            'ps -ax | grep \!* | grep -v grep'
#alias punt           kill

#alias r              rlogin
#alias run            source

#alias nms 'tbl \!* | nroff -ms | more'                   # nroff -ms
#alias tms 'tbl \!* | troff -t -ms >! troff.output &'     # troff -ms
#alias tpr 'tbl \!* | troff -t -ms | lpr -t &'            # troff & print
```

```
#alias ppr 'lpr -t \!* &'                                # print troffed

#alias lp1            'lpr -P1'
#alias lq1            'lpq -P1'
#alias lr1            'lprm -P1'

#alias sd             'screendump | rastrepl | lpr -v &'

#alias edit           textedit

#alias help           man
#alias key            'man -k'

#alias mkae           make
```

Explanation of Command Lines

Line 1:

```
set lpath = ( )  #  add directories for local commands
```

Creates a variable in which to add the pathnames of directories containing local commands. Add the pathnames for any such directories between the parentheses (ask your System Administrator for the appropriate names). These directories are incorporated into the `path` variable in line *3*.

Line 2:

```
set path = (. ~ ~/bin /usr/local /usr/ucb /usr/bin /bin)
```

Sets the `path` variable to include the standard directories containing SunOS commands.

Line 3:

```
set lcd = ( )  #  add parents of frequently used directories
```

Creates a variable in which to add the pathnames of directories that are *parents* of those that you often `cd` to. For instance, if you often `cd` to `/usr/man/man1`, add put the pathname `/usr/man` between the parentheses. These directories are added to the `cdpath` variable in the next line.

Line 4:

```
set cdpath = (.. ~ ~/bin ~/src $lcd)
```

Sets the `cdpath` variable. You need not specify pathnames when you `cd` to directories that are contained in any of those listed. With `..` set in your `cdpath` (as above), if you were in `/usr/man/man1` and you wanted to `cd` to `/usr/man/cat1`, you could use the command `cd cat1` to do so.

Line 5:

```
set noclobber
```

Prevents unintentional overwrites of files when you use the > symbol. See *Doing More with SunOS: Beginner's Guide* for details.

Line 6:

```
alias cd       'cd \!*;echo $cwd'
```

Displays the new directory when you use `cd.` to change directories.

Line 7:

```
alias cp       'cp -i'
```

Asks for confirmation before overwriting existing files with `cp`.

Line 8:

```
alias mv        'mv -i'
```

Asks for confirmation before overwriting existing files with mv.

Line 9:

```
alias rm        'rm -i'
```

Asks for confirmation before removing files.

Line 10:

```
alias pwd       'echo $cwd'
```

A fast way of finding out which directory you're in; faster than the built-in pwd command.

Line 11:

```
#alias del       'rm -i'
```

A name for rm that is familiar to PC users.

Line 12:

```
#umask 002
```

Sets the default permissions mask for new files to allow read and write access to the owner's group as well as to the owner.

Line 13:

```
if ($?USER == 0 || $?prompt == 0) exit
```

Tests to see whether there is a variable called USER, or a variable called prompt currently set. If they're not set, then the shell is non-interactive, so the C shell doesn't bother to process the rest of the commands from this file. This saves time running sub-shells.

Line 14:

```
set history=40
```

The C shell records the last 40 commands typed in.

Line 15:

```
set ignoreeof
```

Prevents accidental logouts when you type Ctrl-D.

Line 16:

```
#set notify
```

Prevents waiting for display of C shell messages. Normally, the C shell waits until just before printing the prompt to print its messages. Commands, however, don't always wait to print their messages, so setting `notify` means that all messages will work the same way.

Line 17:

```
#set savehist=40
```

When you log out, the C shell saves the last 40 commands, and uses them as the starting history list for your next session.

Line 18:

```
#set prompt="% "
```

An alternate prompt favored by some SunOS wizards.

Line 19:

```
#set prompt="`hostname`{`whoami`}\!: "
```

An alternate prompt favored by some network wizards. This prompt gives the name of the machine, followed by the name of the user, followed by the history number of the command:

```
                    venus[medici}22:
```

Line 20:

```
#set time=100
```

Display time statistics for commands that take longer than 100 CPU seconds.

Line 21:

```
#date
```

Display the date and time when the C shell starts up.

Line 22:

```
#pwd
```

Display the working directory when the C shell starts up.

Line 23:

```
alias a         alias
```

Abbreviate the `alias` command.

Line 24:

```
#alias h         'history \!* | head -39 | more'
```

Abbreviate `history`, and delete the last line (containing `h`) from the display. (Assumes that you have the `history` variable set to 40, as above.)

Line 25:

```
#alias u        unalias
```

Abbreviate the `unalias` command.

Line 26:

```
#alias ^L       clear
```

The (Ctrl-L) character is the character often used to begin a new page or clear the current one. This alias mimics that behavior.

Line 27:

```
#alias list     cat
```

A name for `cat` that is familiar to PC users.

Line 28:

```
#alias lock     lockscreen
```

An abbreviation for `lockscreen`.

Line 29:

```
#alias m        more
```

An abbreviation for `more`.

Line 30:

```
#alias mroe     more
```

A remedy for "fat fingers" (ie., for mistyping the word "more").

Line 31:

```
#alias type     more
```

A name for `more` that is familiar to PC users.

Line 32:

```
#alias .        'echo $cwd'
```

An abbreviation for `pwd`.

Line 33:

```
#alias ..       'set dot=$cwd;cd ..'
```

A quick way to change from child directory to parent (and back again with alias on the next line).

Line 34:

```
#alias ,          'cd $dot '
```

A quick way to change back after using the `..` alias above.

Line 35:

```
#alias dir        ls
```

A name for `ls` that is familiar to PC users.

Line 36:

```
#alias pdw        'echo $cwd'
```

For those people who often mistype "pwd." Has the same effect as the `.` alias for `pwd`, above.

Line 37:

```
#alias la         'ls -a'
```

Abbreviation for command to list all filenames, including those that begin with a dot (`.`).

Line 38:

```
#alias ll         'ls -la'
```

Abbreviation for a command to give a long listing of filenames, including those that begin with a dot.

Line 39:

```
#alias ls        'ls -F'
```

This way, `ls` appends characters on the end of a filename to indicate that file's type. These characters are / (slash) for directories; * for executable files; @ for symbolic links; and = for AF_UNIX domain sockets.

Line 40:

```
#alias pd         dirs
```

Abbreviation to display the directory stack maintained by `pushd` and `popd`. See *Doing More with SunOS: Beginner's Guide* for details.

Line 41:

```
#alias po         popd
```

Change directories to the one on the top of the stack, and remove its name from the stack. See *Doing More with SunOS: Beginner's Guide*

Line 42:

```
#alias pp         pushd
```

Change directories, adding the current directory and the destination to the stack. See *Doing More with SunOS: Beginner's Guide*

Line 43:

```
#alias +w       'chmod go+w'
```

Make a file writeable to the group and public.

Line 44:

```
#alias -w       'chmod go-w'
```

Make a file unwriteable to all but you, the owner.

Line 45:

```
#alias x          'chmod +x'
```

Give a file execute permissions for all users.

Line 46:

```
#alias j          'jobs -l'
```

Display the list of background jobs. With `-l`, the `j` alias also gives the process id number of the stopped job.

Line 47:

```
#alias bye        logout
```

Another name for `logout`.

Line 48:

```
#alias ciao     logout
```

An international term for `logout`.

Line 49:

```
#alias adios      logout
```

The traditional South-of-the-Border way to log out.

Line 50:

```
#alias psg        'ps -ax | grep * | grep -v grep'
```

Check on the status of a command by its name. See *Doing More with SunOS: Beginner's Guide* for details.

Line 51:

```
#alias punt       kill
```

Another name for `kill`.

Line 52:

```
#alias r        rlogin
```

Log in to another host machine on the net. See *Using the Network: Beginner's Guide* for details.

Line 53:

```
#alias run       source
```

The `source` command instructs the C shell to take a file (such as the `.cshrc` file) as a list of commands to perform.

Line 54:

```
#alias nms 'tbl \!* | nroff -ms | more'
```

Format and display a document containing `tbl` instructions and `ms` macros on the terminal. For example, to format the file `wombat`, you would type `nms wombat`.

Line 55:

```
#alias tms 'tbl \!* | troff -t -ms >! troff.output &'
```

Format a document using `tbl` and `ms` macros, and place the output in a file for later printing.

Line 56:

```
#alias tpr 'tbl \!* | troff -t -ms | lpr -t &'
```

Format a document using `tbl` and `ms` macros and print it.

Line 57:

```
#alias ppr 'lpr -t \!* &'
```

Print a preformatted troff-output file.

Line 58:

```
#alias lp1       'lpr -P1'
```

Abbreviation to print on printer #1. See *Doing More with SunOS: Beginner's Guide* for details.

Line 59:

```
#alias lq1       'lpq -P1'
```

Abbreviation to check the queue for printer #1.

Line 60:

```
#alias lr1       'lprm -P1'
```

Abbreviation to remove a job or jobs from printer #1.

Line 61:

```
#alias sd         'screendump | rastrepl | lpr -v &'
```

Abbreviation to print the contents of the Workstation screen.

Line 62:

```
#alias edit     textedit
```

Abbreviation for the window-system text editor.

Line 63:

```
#alias help      man
```

Another name for the man command.

Line 64:

```
#alias key       'man -k'
```

Abbreviation for the man -k command (same as the whatis command described in *Doing More with SunOS: Beginner's Guide*).

Line 65:

```
#alias mkae      make
```

Useful if you tend to misspell the word "make."

3

The C Shell and the .login File

The C Shell and the .login File **27**

3.1. A Sample .login File 28

Explanation of Command Lines 30

3

The C Shell and the `.login` File

When you log in, after performing instructions in the `.cshrc` file, the C shell then performs instructions in the `.login` file. Subsequent C shells, such as those running within terminal (`shelltool`) windows, ignore the `.login` file.

Like the `.cshrc` file, you can include any command that you might type in on the command line. However, we recommend that you use the `.login` file for initializing remote terminals (for when you log in by phone), starting your window system (when you first log in to the workstation), and setting up special variables called *environment* variables. Unlike shell variables, environment variables are passed along to subsequent commands and programs automatically. You need not set them up again every time you start a new C shell or run a new program such as `vi`.

Environment variables are useful for storing information that all programs need to know about. For instance, many commands and programs need to know what type of terminal you are using. This information is stored in the `TERM` environment variable. Commands that send output to the printer need to know which printer to send their output to. You can use the `PRINTER` environment variable, to store the name of a printer to use by default.

To set an environment variable, use the `setenv` command. This command has two required arguments, the *name* of the variable, and its *value*.

`setenv` *name value*

For example

Figure 3-1 *The* `setenv` *Command*

```
venus% setenv PRINTER lw
```

Although not required, the convention is to use all capitals for names of environment variables (to distinguish them from ordinary shell variables). To see what environment variables are currently in effect, use the `printenv` command:

Figure 3-2 *The* `printenv` *Command*

```
venus% printenv
HOME=/home/sam
SHELL=/bin/csh
PATH=.:/home/sam:/home/sam/bin:/usr/local:/usr/ucb:/usr/bin:/bin
TERM=sun
USER=sam
EDITOR=/usr/ucb/vi
PRINTER=lw
WINDOW_PARENT=/dev/win0
WMGR_ENV_PLACEHOLDER=/dev/win1
WINDOW_ME=/dev/win9
WINDOW_GFX=/dev/win9
```

To remove an environment variable, use the `unsetenv` command:

Figure 3-3 `unsetenv`

```
venus% unsetenv PRINTER
venus% echo $PRINTER
PRINTER: Undefined variable.
```

3.1. A Sample `.login` File

The following pages contain an annotated listing of the sample `.login` file located in `/usr/lib/Login`. If you do not plan to log in from a remote terminal over the phone, you can delete the lines that pertain to remote terminals. Again, some commands are commented out. And again, we recommend that you delete commands that do not pertain to you.

```
##################################################################
#
#          .login file
#
#          Read in after the .cshrc file when you log in.
#          Not read in for subsequent shells.  For setting up
#          terminal and global environment characteristics.
#
##################################################################


#          terminal characteristics for remote terminals:

#          Leave lines for all but your remote terminal commented
#          out (or add a new line if your terminal does not appear).
if ($TERM != "sun") then
#eval `tset -sQ -m dialup:?925 -m switch:?925 -m dumb:?925 $TERM`
#eval `tset -sQ -m dialup:?h19 -m switch:?h19 -m dumb:?h19 $TERM`
#eval `tset -sQ -m dialup:?mac -m switch:?mac -m dumb:?mac $TERM`
#eval `tset -sQ -m dialup:?vt100 -m switch:?vt100 -m dumb:?vt100 $TERM`
#eval `tset -sQ -m dialup:?wyse-nk -m switch:?wyse-nk -m dumb:?wyse-nk $TERM`
#eval `tset -sQ -m dialup:?wyse-vp -m switch:?wyse-vp -m dumb:?wyse-vp $TERM`
endif

#          general terminal characteristics

#stty -crterase
#stty -tabs
#stty crt
#stty erase '^h'
#stty werase '^?'
#stty kill '^['
#stty new

#          environment variables

#setenv EXINIT 'set sh=/bin/csh sw=4 ai report=2'
setenv MORE '-c'
#setenv PRINTER lw

#          commands to perform at login

#w

if ("`tty`" != "/dev/console") exit
echo -n "SunView? (^C to interrupt) "
sleep 5
sunview
```

Explanation of Command Lines

Line 1:

```
if ($TERM != "sun") then
```

Perform the commands between this line and the `endif` line only when logging in on a terminal, rather than a Sun Workstation.

Line 2:

```
#eval `tset -sQ -m dialup:?925 -m switch:?925 -m dumb:?925 $TERM`
```

If logging in over a phone line, or some other remote means, set up terminal characteristics for a Televideo 925 terminal and place these characteristics in the environment for faster startup of interactive programs. Asks if terminal type 925 is correct by prompting you with

```
TERM 925
```

Pressing Return accepts the 925 as the terminal type; or you can enter another terminal type (from `/etc/termcap`). Refer to `tset` in the *SunOS Reference Manual* for more information.

All of the lines pertaining to specific types are commented out. To activate the line that pertains to your terminal, remove the pound-sign. If your terminal does not appear, duplicate this line, change the `925` to the name of your terminal (see your System Administrator for this information) and remove the pound-sign.

Line 3:

```
#eval `tset -sQ -m dialup:?h19 -m switch:?h19 -m dumb:?h19 $TERM`
```

Set up terminal characteristics for a Heathkit H19 terminal.

Line 4:

```
#eval `tset -sQ -m dialup:?mac -m switch:?mac -m dumb:?mac $TERM`
```

Set up terminal characteristics for a Macintosh running Macterminal.

Line 5:

```
#eval `tset -sQ -m dialup:?vt100 -m switch:?vt100 -m dumb:?vt100 $TERM`
```

Set up terminal characteristics for a VT100 terminal.

Line 6:

```
#eval `tset -sQ -m dialup:?wyse-nk -m switch:?wyse-nk -m dumb:?wyse-nk $TERM`
```

Set up terminal characteristics for a Wyse 50 terminal.

Line 7:

```
#eval `tset -sQ -m dialup:?wyse-vp -m switch:?wyse-vp -m dumb:?wyse-vp $TERM`
```

Set up terminal characteristics for a Wyse 50 in ADDS Viewpoint mode with "enhance" turned on.

Line 8:

```
endif
```

Marks last line to be skipped when an `if ... then` statement is found to be false; in this case, when logging in to a Sun Workstation directly (or from another Sun on the network).

Line 9:

```
#stty -crterase
```

Set up the erase function to backspace without blotting out erased characters. Erased characters remain visible on the screen until you overwrite them with new ones, but are not transmitted to the C shell when you press Return.

Line 10:

```
#stty -tabs
```

Convert tabs to spaces when displayed on the screen.

Line 11:

```
#stty crt
```

Set up standard CRT characteristics.

Line 12:

```
#stty erase '^h'
```

Set the erase character to Back Space. Note that with `stty`, control characters are indicated by the two-character symbol *circumflex-character*: *^c*.

Line 13:

```
#stty werase '^?'
```

Set the erase-word character to Del.

Line 14:

```
#stty kill '^['
```

Set line kill character to Esc.

Line 15:

```
#stty new
```

Use the new version of the terminal driver.

Line 16:

```
#setenv EXINIT 'set sh=/bin/csh sw=4 ai report=2'
```

Another way to set up options for `vi`. If you use the `EXINIT` environment variable, `vi` ignores your `.exrc` file.[5]

Line 17:

```
setenv MORE '-c'
```

Sets up `more` to overwrite the screen rather than scrolling. This makes reading `more` output much easier.

Line 18:

```
#setenv PRINTER lw
```

Indicate which printer is to receive jobs by default.

Line 19:

```
#w                  # see who is logged in
```

See who is logged in on your system.

Line 20:

```
if ("`tty`" != "/dev/console") exit
```

If the terminal is not your Workstation console (the Workstation when not running the Window-system), then stop further processing of this file.

Line 21:

```
echo -n "Sunview? (^C to interrupt) "
```

Warn you that SunView is about to start.

Line 22:

```
sleep 5
```

Wait 5 seconds before starting Sunview, to give you a chance to press Ctrl-C.

Line 23:

```
sunview
```

Start the window system.

[5] If `exrc` doesn't exist in the current directory.

4

The C Shell and the .logout File

The C Shell and the .logout File 35

4.1. A Sample .logout File 35

Explanation of Command Lines 37

4

The C Shell and the .logout File

When you log out completely (not just from a single window), the C shell performs instructions in the .logout file. This file is useful for running housekeeping type commands in the background while you are away.

Like .cshrc and .login you can include any command that you might type in on the C shell command line. We recommend that you use this file only for displaying information about the session just ending that you want to know about, and running background commands. You should *not* put commands that run interactively in this file, nor should you include commands that take any significant amount of time unless the command runs in the background. Otherwise someone may be able to interrupt the command and gain unauthorized access to your workstation or terminal.

4.1. A Sample .logout File

The following pages contain an annotated listing of the sample .logout file located in /usr/lib/Logout. Some commands are commented out, and we recommend that you delete commands that do not pertain to you.

```
############################################################
#
#          .logout file
#
#          Read in when you exit from the login shell.
#          For performing housekeeping while your are away.
#
############################################################


clear                                                                        1
#echo "`hostname`: `whoami` logged out at `date`"                            2
#echo "Goodbye\!"                                                            3

#if (-e /usr/games/fortune) /usr/games/fortune -a                            4
#if (-r /etc/motd) cat /etc/motd                                             5

#unalias rm                                                                  6
#nice find ~ '(' -name core -o -name '*.BAK' -o -name '*.CKP' \              7
#          -o -name '#*' -o -name junk ')' \
#          -atime +3 -mtime +3 -user $USER -type f -exec \rm '{}' \; &
```

Explanation of Command Lines

Line 1:

```
clear
```

Clears the terminal screen.

Line 2:

```
#echo "`hostname`: `whoami` logged out at `date`"
```

Displays the name of the host machine, your user name, and the date and time you logged out.

Line 3:

```
#echo "Goodbye\!"
```

A more traditional parting wish.

Line 4:

```
#if (-e /usr/games/fortune) /usr/games/fortune -a
```

If the fortune command is available, use it to display one of many humorous sayings.

Line 5:

```
#if (-r /etc/motd) cat /etc/motd
```

If the message of the day is readable, display it.

Line 6:

```
#unalias rm
```

If you used alias to change the rm command in some way, this resets it to its original, default value.

Line 7:

```
#nice find ~ '(' -name core -o -name '*.BAK' -o -name '*.CKP' \
#       -o -name '#*' -o -name junk ')' \
#       -atime +3 -mtime +3 -user $USER -type f -exec \rm '{}'
```

Run find at low priority in the background, starting with your home directory. Look for files named core, *.BAK, *.CKP, '#*' or junk. Of these, select only those that are at least 3 days old, haven't been modified for at least 3 days, belong to you, and are regular files (not directories). Remove each file selected, escaping any aliases that might be applied to rm.

To activate this command, you need to delete the first pound-sign in all three lines.

5

vi and the .exrc File

vi and the .exrc File **41**

5.1. Setting Options While in vi 41

5.2. A Sample .exrc File 43

Explanation of Command Lines 45

5

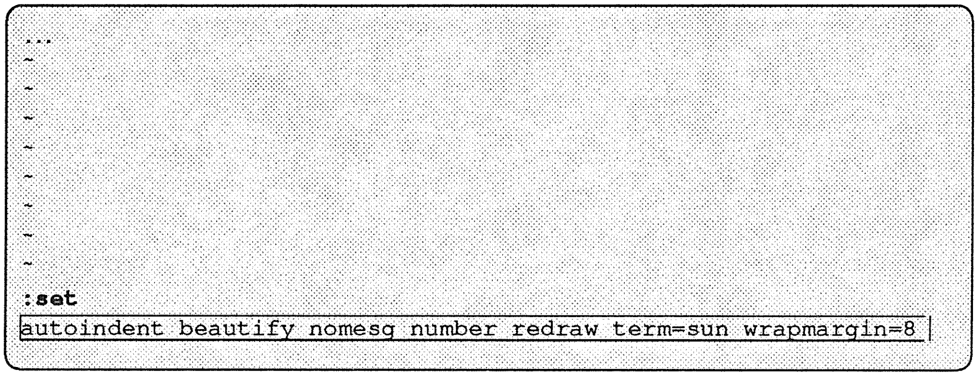
vi and the .exrc File

Whenever you run vi, the editor looks in the .exrc file for initial commands and option settings. The vi editor has a number of options that are described in detail in *Editing Text Files*. vi has a :set command with which you select the editing options that you want, but you cannot use it to create new variables, as you can with the C shell's set command.

5.1. Setting Options While in vi

To see the list of options that are currently in effect, type in :set with no arguments:

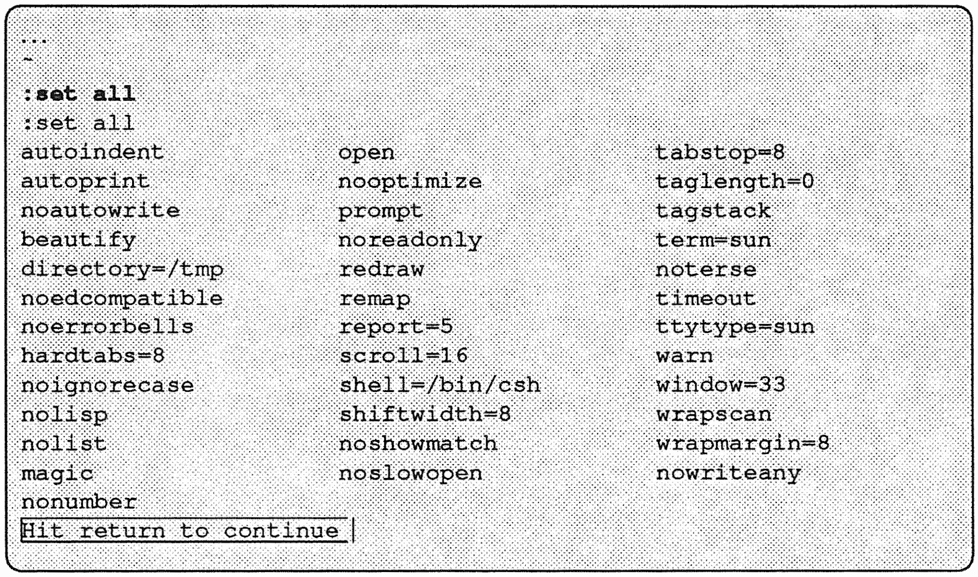
Figure 5-1 *Using* set *to See Current Options*



```
...
~
~
~
~
~
~
~
~
:set
autoindent beautify nomesg number redraw term=sun wrapmargin=8
```

To see the list of all possible settings, use the :set all command:

Figure 5-2 *Using* `set all`



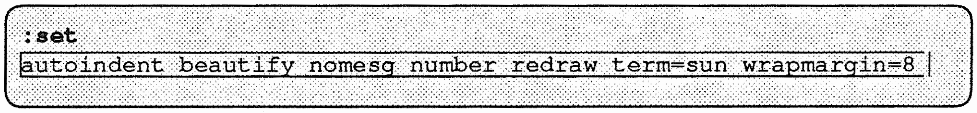
```
...
~
:set all
:set all
autoindent           open                 tabstop=8
autoprint            nooptimize           taglength=0
noautowrite          prompt               tagstack
beautify             noreadonly           term=sun
directory=/tmp       redraw               noterse
noedcompatible       remap                timeout
noerrorbells         report=5             ttytype=sun
hardtabs=8           scroll=16            warn
noignorecase         shell=/bin/csh       window=33
nolisp               shiftwidth=8         wrapscan
nolist               noshowmatch          wrapmargin=8
magic                noslowopen           nowriteany
nonumber
Hit return to continue
```

To select a specific option or options, include them as arguments to the `:set` command. Note that for options having values, there are *no* spaces between the name, the equal-sign, and the value for that option. When the value for an option includes spaces, such as that for `sections` above, the space is escaped with a backslash within the command:

```
    :set sections=NHSHH\ HUnhsh
```

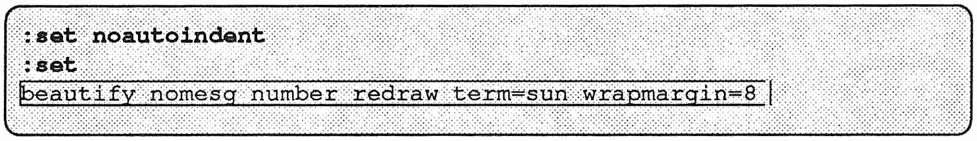
To turn off an option, add the prefix `no` to the name of that option as an argument to `:set`.

Figure 5-3 *Viewing Options*



```
:set
autoindent beautify nomesg number redraw term=sun wrapmargin=8
```

Figure 5-4 *Turning an Option Off*



```
:set noautoindent
:set
beautify nomesg number redraw term=sun wrapmargin=8
```

To change the value of a setting such as `wrapmargin`, use `set` to establish a new value:

```
    :set wrapmargin=0
```

(This has the effect of eliminating automatic wrapping at the end of the line).

5.2. A Sample .exrc File

The following page contains an annotated listing of the sample .exrc file located in /usr/lib/Exrc. Since vi does not accept comments as with the C shell, there are no lines commented out. So, you may wish to delete all lines starting with :map, and add them (or others like them) when you have read through *Editing Text Files*.

```
"
set autoindent                                                          1
set autoprint                                                           2
set noignorecase                                                        3
set nomesg                                                              4
set noslowopen                                                          5
set noterse                                                             6
set nonumber                                                            7
"
set report=2                                                            8
set tabstop=8                                                           9
set wrapmargin=8                                                        10
"
map ; :                                                                 11
map g :%                                                                12
map v ~                                                                 13
map m !} fmt -c                                                         14
map T !} sort                                                           15
"
```

Explanation of Command Lines

Line 1:

```
set autoindent
```

When adding new lines, maintain the same indentation as the line above.

Line 2:

```
set autoprint
```

Automatically print each line altered within ex, the line editor.

Line 3:

```
set noignorecase
```

The case (upper or lower) of a character is significant in searches and substitutions. Use set ignorecase to make searches and substitutions case insensitive. But be careful if you do!

Line 4:

```
set nomesg
```

Messages to the terminal do not interfere with the vi display.

Line 5:

```
set noslowopen
```

Sets up vi for operation with a fast terminal or window. For terminals on slow dialup lines, use set slowopen to suspend updates of the screen during insertions for smoother operation.

Line 6:

```
set noterse
```

vi gives more complete error messages for beginning users. For shorter messages, use set terse.

Line 7:

```
set nonumber
```

Inhibits display of line numbers in both ex and vi. For a display of line numbers, use set number.

Line 8:

```
set report=2
```

Report on all substitutions or deletions that affect more than two lines.

Line 9:

```
set tabstop=8
```

Set tab stops every 8 characters.

Line 10:

```
set wrapmargin=8
```

When a space is typed within 8 characters of the right screen edge, insert a carriage-return at the end of the previous word, starting a new line automatically.

Line 11:

```
map ; :
```

While in `vi` command (*visual*) mode, interpret a semicolon as if you had typed a colon. This allows you to use either a semicolon or a colon as the first character in a substitution command.

Line 12:

```
map g :%
```

While in visual mode, interpret a `g` as if you typed the characters `:%`. This allows you to start commands to substitute throughout the file with either a `g` or a `:%`.

Line 13:

```
map v ~
```

While in visual mode, interpret a `v` as if you typed a `~`, the command to invert the case of a character.

Line 14:

```
map m !} fmt -c
```

While in visual mode, interpret an `m` as if you typed in the command

```
!} fmt -c
```

to adjust line-breaks for the lines between the cursor and the end of the paragraph as close to column 80 as possible (without breaking across words). Refer to *Editing Text Files* and `fmt` in the *SunOS Reference Manual* for more information.

Line 15:

```
map T !} sort
```

While in visual mode, interpret a `T` as if you typed in the command

```
!} sort
```

a command to sort the remaining lines in the paragraph.

Index

Special Characters

# C shell comment symbol, 12
. prefix, explained, 11
^ and stty, 6

A

aliases, escaping with \command, 37

C

C shell setup file, .cshrc, 11
cd command
  and the cdpath variable, 16
cdpath variable, 16
circumflex character, to specify control keys for stty, 6
command
  find, 37
  man -k, 23
  :map (vi), 43
  printenv, 27
  set, 11
  :set (vi), 41
  setenv, 27
  stty, 5
  tset, 30
  unset, 11
comment symbol # in the C shell, 12
commented-out lines in startup files, 12
core file, 6
csh, name of C shell, 11
.cshrc C shell setup file., 11

D

delete word terminal function, 6
dot prefix, explained, 11

E

end-of-file terminal function, 6
endif statement, C shell, 31
environment defined, 3
environment variables, 27
erase terminal function, 5
EXINIT environment variable, 31
.exrc file, 41

F

find command, 37

H

history variable, 11

I

if ... then statement, C shell, 30
if statement, C shell, 17
ignoreeof C shell option, 11, 17
interactive options as variables, 11
interrupt terminal function, 6

L

line kill terminal function, 6
literal next-character terminal function, 6
.login file, 27
.logout file, 35

M

mail, 3
.mailrc file, 4
man command
  -k option, 23
:map command (vi), 43

P

printenv command, 27
PRINTER environment variable, 32

R

rc suffix, explained, 11
reprint terminal function, 6

S

set command, 11
:set command (vi), 41
setenv command, 27
setup file, 4
  command to list, 4
setup for remote terminals, 30
shell, name for a command interpreter, 3
stop display terminal function, 6
stty and the circumflex, 6
stty command, 5

suspend terminal function, 6

T

terminal functions, 5
`tset` command, 30

U

`unset` command, 11
unsetting a `vi` variable, 42

V

variables, environment, 27
`vi` editor, 3

W

wait-for-keystroke terminal function, 6

Corporate Headquarters
Sun Microsystems, Inc.
2550 Garcia Avenue
Mountain View, CA 94043
415 960-1300
TLX 37-29639

For U.S. Sales Office locations, call:
800 821-4643
In CA: 800 821-4642

European Headquarters
Sun Microsystems Europe, Inc.
Bagshot Manor
Green Lane
Bagshot
Surrey, GU19 5NL
England
0276 51440
TLX 859017

Australia: (02) 436 4699
Canada: 416 477-6745
France: (1) 46 30 23 24
Germany: (089) 95094-0
Japan: (03) 221-7021
Nordic Countries: (08) 764 78 10
Switzerland: (1) 82 89 555
The Netherlands: 02155 24888
UK: 0276 62111

Europe, Middle East, and Africa, call European Headquarters:
0276-51440

Elsewhere in the world, call Corporate Headquarters:
415 960-1300
Intercontinental Sales