

A BRIEF INTRODUCTION TO UNIX

by James Beauchamp

8/87

rev: 9/95

To use the commands described in this tutorial you will need to open a **terminal** window (also called the "shell"). On the NeXT, this is a matter of double-clicking on the Terminal icon. Using "Preferences" under this application, you can stylize the Terminal presentation according to your own tastes.

Normally we use the C Shell since it has many convenient features. Therefore, your prompt will be '%' unless you change it. In order to work with Unix, you need to learn a small subset of commands in response to '%'. Note that all commands are terminated with RETURN, and when a command is complete, the % prompt should return. Other important keys are BACKSPACE (delete previous character), ^C (CNTRL c) (kill the current command), ^D (CNTRL d) (used to terminate certain commands), and ^Z (CNTRL z) (used to stop commands).

Logging On

You can either log onto a workstation directly or log onto a computer remotely. For logging on remotely you have this sequence:

```
Login: <your login name>
Password: <your password> (does not show on screen)
<Message of the Day>
%
```

Logging directly onto a workstation will be very similar, but differs from one type of workstation to the next. It is usually very obvious.

Note that in our notation above the <> brackets are meant to mean that the enclosed text describes what you type (or is typed), but not the actual characters. <your password> is the password that has been assigned to you. If you wish to change it, use the 'passwd' command as follows:

```
% passwd → yppasswd
Changing password for <your login name>
Old password: <your current password> (does not show on screen)
New Password: <your new password> ( " " " " " )
Re-enter new Password: <your new password> ( " " " " " )
%
```

Logging Out

```
% logout
```

Creating a File with cat >

The easiest way to create a file (a text file) is to use the command 'cat >'. Here's how it works:

Rename or Move a File

```
% mv file1 file2
```

This can be used to rename a file from file1 to file2. If there is already a file named file2, Unix will ask you if you want it removed, unless the "-i" option is not set. mv can also be used to move a file or group of files from one directory (or subdirectory) to another. The 'mv' command is like the 'cp' command except that the original copy is destroyed. In addition, mv can be used to move a directory from one location to another, as long as you stay in the same file system.

Remove a File

```
% rm file
```

After typing this command, Unix responds with

```
rm: remove file?
```

giving you a chance to change your mind by responding with 'n'. Actually, only if you respond 'y' will it remove the file (I think; better to be affirmative and say 'n' if you mean it.) (Also, this assumes you have not changed the alias for rm; it should be set to 'rm -i'.)

Text File Typing to the Screen

To view a file use

```
% cat filename          (use ^C to abort. although this doesn't always work immediately)
```

```
% more filename        { view one page at a time. SPACEBAR is used to get  
                        to the next page. 'q' will let you exit. }
```

```
% head filename        {look at first few lines of file}
```

```
% tail filename        {look at last few lines of file}
```

To print a file

```
% lpr filename
```

Other Uses of Cat Command

cat is a versatile command and can be used to concatenate text files as well as list them. In its simplest form it can be used to make a file:

```
% cat > filename
```

There are two main modes in vi : the Command Mode and the Insert Mode. Most commands are accomplished with one or two characters. For example, cursor movement is accomplished with the keys

h	j	k	l
<backspace>	<down one line>	<up one line>	<forward space>

The backspace key and spacebar may be used alternatively to h and l.

For new users, the following vi commands are also very useful:

		CAUSES INSERT MODE?
x	delete character	
dd	delete line	
#dd	delete # lines (to register)	
#yy	save # lines (to register)	
p	print lines (from register)	
i	insert before present position	yes
o	enter insert mode on line below	yes
O	enter insert mode on line above	yes
a	insert after present position (append)	yes
R	type over characters	yes
cw	change word	yes
r	replace character	
dw	delete word	
w	jump to beginning of next word (to the right)	
e	jump to end of next word	
b	jump backwards one word	
#G	position at line #	
G	position at last line of file	
^G	give current line number	
H	jump to top of screen	
M	jump to middle of screen	
L	jump to bottom of screen	
/string	jump (forward) to string	
?string	jump (backward) to string	
n	jump to next occurrence of string	
^F	jump forward one screen	
^B	jump backward one screen	
^L	reprint the screen	(only needed if there is a problem with the TERMCAP)

Editing consists mainly of cursor positioning, deleting, and inserting. Six of the vi commands given above cause you to enter the insert mode. Upon entering this mode you continue to type in characters until you wish to stop, whereupon you terminate the insert mode by typing the ESCAPE key.

Use of vi requires a terminal with lower case and cursor control AND a "termcap" (a coded definition of

~v throws you into the vi editor. The message you have typed up to this point (if any) will be displayed on the screen and you can continue editing to finish the message. Use 'ZZ' to exit, and this will throw you back to the normal message input level. Unfortunately, your message will not show at this level, so just remember that your last look at the entire message is when you are in vi. It is possible to add more text after leaving vi, however. Also, you can reenter vi using ~v as many times as you want. Finally, when you are at the lower (normal) level for the last time, hit ^D, respond to the cc: prompt, RETURN, and you are out.

Working with Subdirectories and Paths

Rather than crowding all files into one directory it is generally more convenient to create several subdirectories, each of which contains files of a certain type. For example, documentation can be stored in a subdirectory called 'doc', and sound files could be stored in a subdirectory called 'sounds'. The sounds subdirectory could be further subdivided into 'pitched_sounds', 'scrape_sounds', and 'noise_sounds'.

To create a subdirectory beneath your current directory type

```
% mkdir subdir
```

Then, to enter (get inside) this subdirectory type

```
% cd subdir
```

where cd means "change directory". You can now create files in this subdirectory.

In the Unix operating system all directories are subdirectories of another directory, except the one at the top, which is called "root". Root is indicated by the '/' symbol. This is where system files are kept. Your home directory is the one you arrive at each time you log in and will contain all of your subdirectories. However, your home directory is below one or more levels of other directories. If you are in your home directory, you can change to the subdirectory of other users (up one level) by typing

```
% cd ..
```

'..' always refers to the subdirectory immediately above the current directory. You can always verify your current directory by typing

```
% pwd
```

which means 'print working directory', tells you the entire path of your current directory.

Directory paths can be used to jump from one directory to another or to refer to files in various subdirectories. Absolute paths always start with root and work down. For example, if your home directory is under /u2 and you wish to copy a file called 'examples' from the home directory of 'ken',

`%fg`

more resumes

This is very handy when working with vi or mail. It allows you to do something else and then immediately return to where you were in vi or mail.