UNIX Documentation Road Map

G. A. Snyder
J. R. Mashey

Bell Laboratories
Murray Hill, New Jersey 07974

1. INTRODUCTION

A great deal of documentation exists for the UNIX† time-sharing system. New users are often overcome by the volume and distributed nature of the documentation. This "road map" attempts to be a terse, up-to-date outline of important documents and information sources.

☞ *The information in this document applies only to UNIX Release 4.0.*

1.1 Things to Do

See a local UNIX "system administrator" to obtain a "login name" and get other appropriate system information. See also Section 12 below.

1.2 Notation Used in This Road Map (B.1.1) ●●

{*N*} → Section N in this road map.
●● → Item required for everyone.
● → Item recommended for most users.

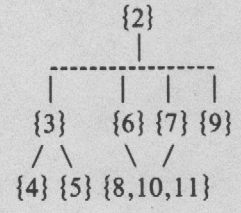
All other items are optional and depend on specific interests. If the name of a document mentioned here is followed by a number such as "(A.1.1)," then that document can be found in *Documents for UNIX*. Examine Section G of the *Annotated Table of Contents* in that volume for additional sources of information.

Entries in Section *n* of the *UNIX User's Manual* are referred to by *name*(n).

1.3 List of Following Sections ●●

{2} BASIC INFORMATION
{3} BASIC TEXT PROCESSING AND DOCUMENT PREPARATION
{4} SPECIALIZED TEXT PROCESSING
{5} ADVANCED TEXT PROCESSING
{6} COMMAND LANGUAGE (SHELL) PROGRAMMING
{7} FILE MANIPULATION
{8} C PROGRAMMING
{9} NUMERICAL COMPUTATION
{10} SOURCE CODE CONTROL SYSTEM
{11} INTER-SYSTEM COMMUNICATION
{12} LOCAL INFORMATION

1.4 Prerequisite Structure of Following Sections ●●



† UNIX is a trademark of Bell Laboratories.

Each section contains a list of relevant documents and a list of pertinent manual entries; some sections also contain a list of suggested things to do.

Only the manual entries for the most frequently used commands are listed here; other relevant entries may be found by consulting the *Table of Contents* and the *Permuted Index* of the *UNIX User's Manual* {2.1}; it is also wise to periodically scan Section 1 of that manual—you will often discover new uses for commands.

2. BASIC INFORMATION

You won't be able to do much until you have learned most of the material in {2.1}, {2.2}, and {2.3}. You must know how to log into the system, make your terminal work correctly, enter and edit files, and perform basic operations on directories and files. Get the *UNIX Programming Starter Package* from your local Computer Information Service Library.

2.1 UNIX User's Manual ●●

— Read *Introduction* and *How to Get Started*.
— Read the *intro* entry in each section.
— Look through Section 1 to become familiar with command names.
— Get into the habit of using the *Table of Contents* and the *Permuted Index*.

Section 1 will be especially needed for reference use.

2.2 UNIX for Beginners (Second Edition) (B.3.1) ●●

2.3 A Tutorial Introduction to the UNIX Text Editor (B.2.1) ●●

2.4 Advanced Editing on UNIX (B.2.2) ●

2.5 The Bell System Technical Journal, Vol. 57, No. 6, Part 2 ●

Contains several articles on UNIX. In particular, the first paper gives a good overview of UNIX.

2.6 Things to Do

— Do all the exercises found in {2.2} and {2.3}, and maybe {2.4}.
— If you want some sequence of commands to be executed each time you log in, create a file named **.profile** in your login directory.[1] A sample **.profile** can be found in *profile*(5).
— Files in directory **/usr/news** contain recent information on various topics. To print all the news items that have been added since you last looked, type:

```
news
```

2.7 Manual Entries

The following commands are described in Section 1 of the *UNIX User's Manual* and are used for creating, editing, moving (i.e., renaming), and removing files:

cat(1)	concatenate and print files (no pagination).
cd(1)	change working (current) directory.
chmod(1)	change the mode of a file.
cp(1)	copy (*cp*), move (*mv*) or link (*ln*) files.
ed(1)	edit a file.
ls(1)	list a directory; file names beginning with . are not listed unless the **−a** flag is used.
mkdir(1)	make a (new) directory.

1. The directory you are in when you log into the system.

pr(1)	print files (paginated listings).
pwd(1)	print working directory.
rm(1)	remove (delete) file(s); *rmdir* removes the named directories, which must be empty.

The following help you communicate with other users, make proper use of different kinds of terminals, and print manual entries on-line:

login(1)	sign on.
mail(1)	send mail to other users or inspect mail from them.
man(1)	print entries of *UNIX User's Manual*.
mesg(1)	permit or deny messages to your terminal.
news(1)	print news items: *news* $-n$ prints a list of recent items.
passwd(1)	change your login password.
stty(1)	set terminal options; i.e., inform the system about the hardware characteristics of your terminal.
tabs(1)	set tab stops on your terminal.
term(7)	a list of commonly-used terminals.
who(1)	print list of currently logged-in users.
write(1)	communicate with another (logged-in) user.

Several useful status commands also exist:

date(1)	print time and date.
du(1)	summarize disk usage.
ps(1)	report active process status.

3. BASIC TEXT PROCESSING AND DOCUMENT PREPARATION

You should read this section if you want to *use* existing text processing tools to write letters, memoranda, manuals, etc. Get the *UNIX Text Editing and Phototypesetting Starter Package* from your local Computer Information Service Library.

3.1 MM—Memorandum Macros (C.2.1) ●●

This is a reference manualthat can be moderately heavy going for a beginner. Try out some of the examples and stick close to the default options.

3.2 Typing Documents with MM (C.2.2) ●●

A handy fold-out.

3.3 A TROFF Tutorial (C.1.1) ●

An introduction to formatting text with the phototypesetter.

3.4 NROFF/TROFF User's Manual (C.1.2) ●

Describes the text formatting language in great detail; look at the *SUMMARY AND INDEX*, but don't try to digest the whole manual on first reading.

3.5 Manual Entries

mm(1)	print a document using the memorandum macros.
troff(1)	typeset or format (*nroff*) text files; read this to become familiar with options.
spell(1)	identify possible spelling errors.

To obtain some special functions (e.g., reverse paper motion, subscripts, superscripts), you must either indicate the terminal type to *nroff* or post-process *nroff* output through one of the following:

col(1)	process text for terminals lacking physical reverse vertical motion, such as the Texas Instruments 700 series, Model 43 *TELETYPE®*, etc.
greek(1)	handle special functions for many terminals, such as DASI 300, Tektronix 4014, Diablo 1620, Hewlett-Packard 2645, etc.
tc(1)	simulate phototypesetter output on a Tektronix 4014 terminal.

4. SPECIALIZED TEXT PROCESSING

The tools listed here are of a more specialized nature than those in {3}.

4.1 TBL—A Program to Format Tables (C.3.1) •

Great help in formatting tabular data (see also *tbl*(1)).

4.2 Typesetting Mathematics—User's Guide (Second Edition) (C.3.2) •

Read this if you need to produce mathematical equations. It describes the use of the equation-setting command *eqn*(1).

4.3 A Macro Package for View Graphs and Slides (C.2.3)

Tells how to prepare typeset visuals.

4.4 UNIX Graphics Overview (E.6.1)

Describes the Graphics sub-system of UNIX.

4.5 Manual Entries

cw(1)	use a special constant-width "example" font.
diffmk(1)	mark changes between versions of a file, using output of *diff*(1) to produce "revision bars" in the right margin.
eqn(1)	preprocessor for mathematical equations.
eqnchar(7)	special character definitions for *eqn*(1).
graphics(1G)	get into the graphics sub-system.
mmt(1)	typeset documents, view graphs, and slides.
tbl(1)	preprocessor for tabular data.

5. ADVANCED TEXT PROCESSING

You should read this section if you need to *design* your own package of formatting macros or perform other actions beyond the capabilities of existing tools; {3} is a prerequisite, and familiarity with {4} is very helpful, as is an experienced advisor.

5.1 NROFF/TROFF User's Manual (C.1.2) ••

Look at this in detail and try modifying the examples. Read *A TROFF Tutorial* {3.3}.

5.2 Things to Do

It is fairly easy to use the text formatters for simple purposes. A typical application is that of writing simple macros that print standard headings in order to eliminate repetitive keying of such headings. It is extremely difficult to set up general-purpose macro packages for use by large numbers of people. Don't re-invent what you can borrow from an existing package (such as MM—see {3.1} and {3.2}).

5.3 Manual Entries

All entries mentioned in {3.5} and {4.5}.

6. COMMAND LANGUAGE (SHELL) PROGRAMMING

The shell provides a powerful programming language for combining existing commands. This section should be especially useful to those who want to automate manual procedures and build data bases.

6.1 The UNIX Time-Sharing System (A.1.2) ●●

6.2 UNIX Shell Tutorial (B.4.1) ●●

6.3 An Introduction to the UNIX Shell (B.4.2)

6.4 Things to Do

If you want to create your own library of commands, for example **/usr/gas/bin**, set the **PATH** parameter in your **.profile** so that your own library is searched when a command is invoked. For example:

```
PATH=:$HOME/bin:/bin:/usr/bin
```

The **HOME** parameter is described in *sh*(1).

6.5 Manual Entries

Read *sh*(1) first; the following entries give further details on commands that are most frequently used within command language programs:

echo(1)	echo arguments (typically to terminal).
env(1)	set environment for command execution.
expr(1)	evaluate an algebraic expression; includes some string operations.
line(1)	read a line from the standard input.
nohup(1)	run a command immune to communications line hang-up.
sh(1)	shell (command interpreter and programming language).
test(1)	evaluate a logical expression.

7. FILE MANIPULATION

In addition to the basic commands of {2}, many UNIX commands exist to perform various kinds of file manipulation. Small data bases can often be managed quite simply by combining text processing {5}, shell programming {6}, and the commands listed below in {7.3}.

7.1 SED—A Non-Interactive Text Editor (B.2.3)

7.2 AWK—A Pattern Scanning and Processing Language (E.3.1)

7.3 Manual Entries

The starred (*) items below are especially useful for dealing with "fielded data," i.e., data where each line is a sequence of delimited fields. The following are used to search or edit files in a single pass:

awk(1)*	perform actions on lines matching specified patterns.
grep(1)	search a file for a pattern; more powerful and specialized versions include *egrep* and *fgrep*.
sed(1)*	stream editor.
tr(1)	transliterate (substitute or delete specified characters).

The following compare files in different ways:

cmp(1)	compare files (byte by byte).
comm(1)	print lines common to and/or different in two files.
diff(1)	differential file comparator (minimal editing for conversion).

The following combine files and/or split them apart:

ar(1)	archiver and library maintainer.
cpio(1)	general file copying and archiving.
cut(1)*	cut out selected fields of each line of a file.
join(1)	join two relations specified by the lines of two files.
paste(1)*	merge lines from several files.
split(1)	split file into chunks of specified size.

The following interrogate files and print information about them:

file(1)	determine file type (best guess).
od(1)	octal dump (and other kinds also).
sum(1)	sum and count blocks in a file.
wc(1)	word (and line and character) count.

Miscellaneous commands:

find(1)	search directory structure for specified kinds of files.
sort(1)*	sort or merge files.
tail(1)	print the last part of a file.
tee(1)	copy single input to several output files.
uniq(1)*	report repeated lines in a file, or obtain unique ones.

8. C PROGRAMMING

Try to use existing tools first, before writing C programs at all.

8.1 The C Programming Language

A book written by B. W. Kernighan and D. M. Ritchie; published by Prentice Hall (1978). It contains comprehensive text and includes a tutorial and a reference manual. Read the tutorial; try the examples. Check for updates to the reference manual {8.2} from time to time.

8.2 The C Programming Language—Reference Manual (D.1.1) ●●

8.3 UNIX Programming (D.3.1) ●

8.4 A Guide to the C Library for UNIX Users (D.1.2) ●

8.5 SDB—A Symbolic Debugger (D.5.1)

8.6 YACC—Yet Another Compiler-Compiler (E.1.2)

8.7 LEX—A Lexical Analyzer Generator (E.1.1)

8.8 LINT, a C Program Checker (D.1.3)

8.9 MAKE—A Program for Maintaining Computer Programs (D.4.1)

8.10 An Augmented Version of MAKE (D.4.2)

8.11 Things to Do

Read {8.1} and do some of the exercises. A good way to become familiar with C is to look at the source code of existing programs, especially ones whose functions are well known to you. Much code can be found in directory **/usr/src**. In particular, the directory **cmd** contains the source for most of the commands. Also, investigate directory **/usr/include**.

8.12 Manual Entries

ar(1)	archive and library maintainer.
cc(1)	compile C programs.
ld(1)	link edit object files; you must know about some of its flags.

lex(1)	generate lexical analyzers.
lint(1)	verify C programs.
lorder(1)	find ordering relation for an object library.
make(1)	automate program (re)generation procedures.
nm(1)	print name (i.e., symbol) list.
prof(1)	display profile data; used for program optimization.
ps(1)	report active process status.
sdb(1)	debug C and F77 programs symbolically on the VAX 11/780.
strip(1)	remove symbols and relocation bits from executable files.
time(1)	time a command.
yacc(1)	parser generator.

9. NUMERICAL COMPUTATION

9.1 DC—An Interactive Desk Calculator (E.5.2)

9.2 BC—An Arbitrary Precision Desk-Calculator Language (E.5.1)

9.3 AWK—A Pattern Scanning and Processing Language (E.3.1)

9.4 A Portable FORTRAN 77 Compiler (D.2.1)

9.5 RATFOR—A Preprocessor for a Rational FORTRAN (D.2.2)

9.6 SDB—A Symbolic Debugger (D.5.1)

9.7 Manual Entries

awk(1)	perform actions on lines matching specified patterns.
bc(1)	an interactive language, acts as front end for *dc*(1).
bs(1)	a compiler/interpreter for modest-sized programs.
dc(1)	a desk calculator.
f77(1)	a FORTRAN compiler.
ratfor(1)	a rational FORTRAN dialect.
sdb(1)	debug C and F77 programs symbolically on the VAX 11/780.

10. SOURCE CODE CONTROL SYSTEM

10.1 Source Code Control System User's Guide (E.4.1) •

10.2 Manual Entries

admin(1)	create and administer SCCS files.
cdc(1)	change the delta commentary of an SCCS file.
comb(1)	combine deltas of an SCCS file.
delta(1)	create a new version or delta of a file under SCCS control.
get(1)	get a particular version of an SCCS file, usually for editing.
help(1)	print helpful error messages and information about a command.
prs(1)	print delta information of an SCCS file in a specified format.
rmdel(1)	remove a delta.
sact(1)	print current SCCS file editing activity.
sccsdiff(1)	print the different lines between two deltas of an SCCS file.
unget(1)	undo the version control mechanism created by a *get* for editing.
val(1)	validate an SCCS file.
what(1)	print out embedded information lines placed in a file by SCCS.

11. INTER-SYSTEM COMMUNICATION

11.1 A Dial-up Network of UNIX Systems (E.8.1) ●

11.2 UNIX Remote Job Entry User's Guide (E.7.1) ●

11.3 Manual Entries

The following commands (most of which are site-dependent) are useful in communicating with other systems:

cu(1C)	call another system.
dpr(1C)	print files off-line at a specified destination.
fget(1C)	retrieve files from the HONEYWELL 6000.
fsend(1C)	send files to the HONEYWELL 6000.
gcat(1C)	send phototypesetter output to the HONEYWELL 6000.
send(1C)	send files to an IBM host for execution using Remote Job Entry.
uucp(1C)	copy files from one UNIX system to another.
uux(1C)	execute command(s) on another UNIX system.

12. LOCAL INFORMATION

☞ This section should be provided by each individual UNIX installation.

January 1981