UNIX™ System V

DOCUMENTER'S WORKBENCH™ Software
Introduction and Reference Manual

Western Electric

December 1983
307-150 Issue 1

UNIX™ System V Release 2.0

DOCUMENTER'S WORKBENCH™ Software Introduction and Reference Manual

Western Electric

CONTENTS

Chapter 1	DOCUMENTER'S WORKBENCH SOFTWARE
Chapter 2	DOCUMENT PREPARATION
APPENDIX	USER REFERENCE MANUAL

Chapter 1

DOCUMENTER'S WORKBENCH SOFTWARE

PAGE

INTRODUCTION 1-1

HOW TO USE THIS BOOK 1-2

Chapter 1

DOCUMENTER'S WORKBENCH SOFTWARE

INTRODUCTION

This book is the introductory volume of a set of documents that provide information about the DOCUMENTER'S WORKBENCH software on the UNIX* system. Other books in this series are:

307-151	*Text Formatters Reference*—provides a reference covering the text formatters **nroff**, **troff** (device independent), **otroff** (old troff), and **sroff**.
307-152	*Macro Packages Reference*—provides a reference for the macro packages **mm** (memorandum macros), **sroff/mm** (mm macros for the sroff formatter), and **mv** (viewgraph macros).
307-153	*Preprocessors Reference*—provides a reference for the preprocessors **tbl**, **pic**, and **eqn/neqn**.

This book is both an introduction to the DOCUMENTER'S WORKBENCH software and a DOCUMENTER'S WORKBENCH software command reference manual. Chapter 2 contains introductory material on some of the software, and the appendix provides pages describing the various commands that are part of the DOCUMENTER'S WORKBENCH software.

* UNIX is a trademark of Bell Laboratories.

HOW TO USE THIS BOOK

If you are a beginning user, read Chapter 2, *Document Preparation,* first. That chapter contains general information about getting started and covers most of the basic topics needed to produce a document. If you are a more experienced user, you may only want to skim Chapter 2 , then refer to the appropriate manual (listed above) for more specific information. The appendix can be used as a reference manual about how to use the various commands at the UNIX system level.

Chapter 2

DOCUMENT PREPARATION

PAGE

INTRODUCTION 2-1

GETTING STARTED 2-3

INPUTTING YOUR DOCUMENT 2-5

1. Text 2-5
2. Requests 2-5
3. Macros 2-7
4. Macro Packages 2-9
5. Strings 2-11
6. Registers 2-13

FORMATTING YOUR DOCUMENT 2-16

1. Text Processors 2-16
2. Using Text Processors 2-18
3. Preprocessors 2-19
4. Using Preprocessors 2-20
5. Postprocessors 2-20

RECOMMENDATIONS AND SUMMARY 2-21

1. Inputting 2-21
2. Formatting 2-22
3. Printing 2-22

Chapter 2

DOCUMENT PREPARATION

INTRODUCTION

This chapter introduces you to the process of producing a document using the DOCUMENTER'S WORKBENCH software on the UNIX system. The three major tasks involved in document preparation are:

- Inputting
- Formatting
- Printing or typesetting.

Entering data into a computer system is known as inputting. Inputting is normally done by using a keyboard and an editing program. The data stored in a file for document preparation consists of text and text formatting commands that the text formatter understands. Before the data can be printed, the formatting commands must be converted by a text formatter to information a printer or phototypesetter understands. This process is called formatting. There are several formatting programs that prepare the data to be printed.

Everything from how to get started to producing a final copy is covered in this chapter. Only the basic concepts of the facilities available and how they fit together are described. This chapter does not cover enough of the details for you to be able to write a document after reading it. The details are covered in the following DOCUMENTER'S WORKBENCH software documents:

307-150	*Introduction and Reference Manual*
307-151	*Text Formatters Reference*
307-152	*Macro Packages Reference*

307-153	*Preprocessors Reference*
307-155	*Memorandum Macros (MM) Quick Reference*
307-156	*Text Processing Quick Reference*

This chapter provides a foundation so that you can understand these details later.

Why use the DOCUMENTER'S WORKBENCH software to produce a document? Simply because documents can be produced faster using it. Assume that you are going to produce a large document using tools found in the old office environment. The old way is to spend much time planning the appearance of the following items:

- Page headers
- Titles
- Headings
- Margins
- Lists
- Displays
- Footnotes
- Page Numbering
- Table of Contents
- Glossary
- Index.

For your job, you may need more or fewer items than listed.

You are concerned with how much white space is involved. How much white space goes above, below, to the right, and to the left of the text that makes up each item?

If you are lucky, the type of document you are trying to produce already has a set of standards associated with it. If not, make your own standards as you go. The DOCUMENTER'S WORKBENCH software contains some built-in standards that are implemented by default. These built-in default values can be tailored to meet your specific needs.

Assume that you have produced your document on a typewriter. To correct a single error, add text, delete text, or change text could mean you must retype a page or part of the entire document. Most documents go through several versions before they are finally finished. Therefore, much of your time would be spent in planning white space and physically manipulating the words of your document. This process takes away valuable time that could be spent on the content of your text.

The DOCUMENTER'S WORKBENCH software does not eliminate white space planning or manipulating words. It does speed up these processes significantly. Writing and editing becomes faster, more systematic, and more enjoyable. This software enables you to create, change, display, and print text by a method more efficient than found in the old office environment. It also allows you to concentrate more on document content and less on document appearance by automatically controlling appearance.

GETTING STARTED

The first thing a beginning user should remember is don't get discouraged if your document doesn't look as you planned. Usually, this occurs when you don't fully understand the tools that are available to you. It takes time to develop your skills. Experiment with the software. Try different things just to see what the result will be. Then refer to the manual again. Usually this will give you a better understanding of the particular software tool you are using.

Some of the things you experience during the learning period are discussed in this chapter. Your goal is to learn how to produce letters, reports, memorandums, manuals, books, or some type of document. You need to grow daily in editing, document processing, and even some simple programming. You will evolve from experimenting with the software to producing documents from initial start through final copy. As you grow in your knowledge, you will learn how the tools provided assist you in your day-to-day job.

INPUTTING YOUR DOCUMENT

1. Text

The DOCUMENTER'S WORKBENCH software requires input text to contain instructions for the appearance you want. This text is prepared by you using a text editor such as **ed**(1) or **vi**(1) (see *UNIX System User Reference Manual*).

The input is composed of the text you want printed as well as formatter requests which tell the program how you want your text to look on the page.

Total input therefore consists of text lines that are printed, mixed with control lines that control format. These control lines are interpreted by a text processor and do not appear in the formatted output. Text is controlled in document preparation by the following features:

- Requests
- Macros
- Strings
- Registers.

2. Requests

A request is a control line embedded in your text. A text processing request has the following characteristics:

- The request must be entered on a line by itself.
- The request must start with a period (.) or an acute accent (´) at the beginning of the line, followed by two lowercase characters such as (.xx).
- The request can be located anywhere in the document but normally affects only the text that follows it.

- The request can have optional arguments located on the same line.

Your entire document consists of text interspersed with requests. Writing your document with requests is a lot like writing a program in a low level language.

Your document could contain thousands of paragraphs. Assume you need every new paragraph to start with some vertical space and be indented. The following **nroff/troff** requests could be written to achieve this:

```
.sp 1          \"  space one line
.ti +5  \"  temporarily indent 5 spaces
```

To show you how complex a set of requests can be, the following list is included. This list of requests also produces a paragraph.

```
.if!(((\\n(!D=\\n(nl):(\\n(!D=(\\n(nl-.5v)))&\
(\\n(!Z=\\n(.k)&(\\n(Np=0)) \{\
.br
.nr;1 \\n(:J
.nr;2 \\n(nl
.sp \\n(Psu*.5v
.if!\\n(:D .ne 1v+.5p
.ie!\\n(;1-\\n(:J .nr ;2 \\n(;2-\\n(:J
.el.nr ;2 \\n(nl-\\n(:J
.nr:J \\n(;2
.if\\n(.$>0&(0\\$1) .ti+\\n(Pin
.if\\n(.$=0 \{\
.if\\n(Pt=1 .ti+\\n(Pin
.if\\n(Pt>1&(\\n(:I) .ti+\\n(Pin
.if\\n(Pt>1&(\\n(:I=0)&(\\n(:J>0).ti+\\n(Pin \}
.if\\n(Np \{\
\\n(H1.\\n+(!4\ \ \c
'br\}
.nr:I 1 \}
.nr:u  0
```

As you can see, this is very complicated. These requests are used by the memorandum macro package to define a paragraph. The reason for such complexity is that several decisions are made concerning spacing, indent, and numbering, based on the values of formatter variables. These variables are in the form of number registers (Pi, Pt, :J, Ps, Np) and character strings ($1). These will be discussed in more detail later.

How would you like to type in a long collection of requests just to begin a paragraph? Is there some way to avoid this? Yes, a macro can condense a collection of requests into one command line.

3. Macros

A macro is a 1- or 2-character abbreviation (name) that is replaced by a sequence of formatting requests when processed. For example, the long collection of requests to produce spacing and indentation for a paragraph can be replaced with

```
.P
```

on a line by itself. This is done by defining the **P** macro as follows:

```
.de P
.if!(((\\n(!D=\\n(nl):(\\n(!D=(\\n(nl-.5v)))&\
(\\n(!Z=\\n(.k)&(\\n(Np=0)) \{\
.br
.nr;1 \\n(:J
.nr;2 \\n(nl
.sp \\n(Psu*.5v
.if!\\n(:D .ne 1v+.5p
.ie!\\n(;1-\\n(:J .nr ;2 \\n(;2-\\n(:J
.el.nr ;2 \\n(nl-\\n(:J
.nr:J \\n(;2
.if\\n(.$>0&(0\\$1) .ti+\\n(Pin
.if\\n(.$=0 \{\
.if\\n(Pt=1 .ti+\\n(Pin
.if\\n(Pt>1&(\\n(:I) .ti+\\n(Pin
.if\\n(Pt>1&(\\n(:I=0)&(\\n(:J>0).ti+\\n(Pin \}
.if\\n(Np \{\
\\n(H1.\\n+(!4\ \ \c
'br\}
.nr:I 1 \}
.nr:u  0
..
```

This is the paragraph macro from the memorandum macro package that was shown earlier. Notice that the definition of the macro begins with ".de" and ends in "..". Thus by issuing one simple macro request, several formatting requests can be invoked. Macros and formatting requests can be interspersed in the same document.

A macro can be predefined by some existing macro package or you can define your own macros. When preparing documents that repeat the same text processing primitive requests over and over, it becomes desirable to define your own macro. Some rules for defining and using macros follow:

1. Macro names are one or two characters that should not be the same as any text processing requests. User-defined macro names are normally made uppercase.

2. The **.de** request begins a macro definition. This request is on a line by itself and has the following form:

```
.de XX
```

where XX is the macro name.

3. The list of text processing requests to be repeated follows the start of the macro definition. These requests make up the macro function.

4. The macro definition is ended with a double period (..) on a line by itself.

5. Up to nine arguments can be passed to a macro. Each argument is separated from other arguments by a space. When you invoke the macro in your input, the form is as follows:

   ```
   .XX arg1 arg2 arg3 ... arg9
   ```

   where XX is the macro name and " arg" are possible arguments. A single argument can contain a space only if it is surrounded by double quotes.

6. The macro definition must appear before its invocation. It is a good practice to place macro definitions at the beginning of your text file or in a separate file that is always called before your text files.

7. Macros may be nested. This means that other macros may be defined or invoked within a macro.

4. Macro Packages

Text processing requests are hard to use effectively. It is very difficult for most beginners to define their own macros. Therefore, several " packages" of predefined formatting macros are provided. These macro packages can be used to create displays, headers, headings, paragraphs, titles, footnotes, listings, multicolumn output, and most of the other items needed to produce a document. You can learn how to use these macro packages with little effort compared to learning how to use formatting requests or defining your own macros. These packages take some effort to learn, but the rewards for using them are so great that it is time well spent.

Macro packages supported are as follows:

- Memorandum macros (mm)
- Sroff/mm
- Macro package for viewgraphs and slides (mv)

These macro packages are more fully explained in the *Macro Packages Reference*—select code 307-152.

The memorandum macro package is the standard general purpose package of text formatting macros used with the **nroff**, **otroff**, and **troff** text formatters to produce many common types of documents. A brief description of the " memorandum macros" package known as **mm** follows. Formatting macro requests typically consist of a period and two uppercase letters, such as

```
.PH
```

that is used to define a page header or

```
.P
```

to begin a new paragraph.

The text of a typical document is entered so that it looks something like this:

```
.TL
title
.AU "author information"
.MT "memorandum type"
.P
Enter text ---
---
.P
More text ---
---
.SG "signature"
```

The lines that begin with a period are the formatting macro requests. For example, **.P** calls for starting a new paragraph. The precise meaning of **.P** depends on the output device being used (typesetter or terminal, for instance) and the publication the document will appear in. For example, the memorandum macro package normally assumes that a paragraph is preceded by a space—one line in **nroff** and one-half vertical space in **otroff** or **troff**, and that the first line is left justified. These rules can be changed if desired, but they are changed by changing the interpretation of **.P**, not by retyping the document.

The **sroff/mm** package is a smaller subset of the nroff/mm package plus a few additional macros. The **sroff/mm** package is used with the **sroff** text processor to produce documents.

The **mv** macro package is designed to format your text into viewgraphs and slides. The **mv** package is used in conjunction with **otroff** or **troff** to produce phototypeset output. This output can be photographed and made into slides or copied for making transparencies.

5. Strings

A string is a 1- or 2-character variable that is embedded in text or in a macro. It is actually a text register that can be defined to contain a string of characters that can be printed simply by calling the string name. A string may be predefined by a macro package or you may define your own strings. A text string is a group of characters that may be more than one word. Sequences of words or names that occur repeatedly in a document can be replaced by a string. The contents of a string is printed by entering ***x** for a single character string name (**x**), or ***(xx** for a two character string name (**xx**). For example, assume that you want to print the current date in your document. If the string **DT** contains the current date, then rather than entering:

```
The date is December 23, 1923
```

and updating your file when the date changes, it is simpler to enter:

```
The date is \*(DT
```

Unless redefined, the predefined string **DT** in the memorandum macro package contains the current date. Strings are often used in page headers, page footers, and lists.

The rules for defining strings are as follows:

1. A string name consists of one or two uppercase characters that should not be the same as a text processing request or macro.

2. The **.ds** request defines a string. A string definition has the following form:

   ```
   .ds XX text string
   ```

   where XX is the string name.

3. The " text string" can be of any length and can include concealed new lines. [To “conceal” a “newline”, precede it with a backslash (\)].

An example of some string definitions follow:

```
.ds UG \fIUNIX System User Reference Manual\fR
.ds U UNIX operating system
```

Now that you know how to define a string, how do you use it? The rules for using strings follow.

1. To use a string in regular text, precede the name by "*" (for a 1-character name) or "*(" (for a 2-character name).

2. To use a string in a macro definition, precede the name with "*" (for a 1-character name) or "\\*(" (for a 2-character name).

3. Strings may be used anywhere in the text or in a macro.

4. The string must be defined before it can be used.

5. Strings may be nested.

An example of using strings in text follows. The input

```
The \*U
user commands are described in the
\*(UG.
```

results in the following output when formatted with the previous string definitions of **U** and **UG**:

> The UNIX operating system
> user commands are described in the
> *UNIX System User Reference Manual.*

6. Registers

Text processors provide three different kinds of registers:

- Predefined general number registers
- Predefined read-only number registers
- User-defined number registers.

The predefined registers have default values. These registers are maintained by the text formatters. They are used to define part of the overall appearance of your document. A general number register can be read, written, automatically incremented or decremented, and interpolated into the input. Number registers may also be used in numerical expressions (arithmetic), for flags, and for automatic numbering.

In the memorandum macro package, the appearance of a paragraph is controlled by the following registers:

- Pi
- Pt
- Ps

A register can be given a value using the **.nr** text processing request. For example, to indent paragraphs by three spaces, the paragraph indent register (**Pi**) is set to 3 and the paragraph type register (**Pt**) is set to 1 (for all paragraphs indented) at the beginning of your document:

```
.nr Pi 3
.nr Pt 1
```

The initial values of **Pt** and **Pt** are 0, which causes paragraphs to be left justified.

By default, the amount of space between paragraphs is one blank line. The **Ps** number register controls spacing between paragraphs. To force two blank lines (double spacing) between paragraphs, the following would be used:

```
.nr Ps 2
```

These and other registers used to control format by the memorandum macro package are more fully explained in the *Macro Packages Reference*—select code 307-152.

To interpolate the value of a number register into the input, simply prepend a backslash **n** to the register name. For instance, the input:

```
Paragraphs are separated by \n(Ps
blank lines in this chapter.
```

would result in the following output when formatted:

```
Paragraphs are separated by 3
blank lines in this chapter.
```

The rules for defining number registers follow:

1. The name is one or two characters and is not the same as any other number register name. An easy way to avoid conflict with names already used by the formatters and **mm** is to use two letters, the first being lowercase and the second being uppercase.

2. The **.nr** request defines number registers. A number register definition has the following form:

   ```
   .nr R N M
   ```

   where R is the register name, N is the initial value of the register, and M is the amount to automatically increment or decrement the register.

3. The **.af** request sets the format for output of the number register and has the following form:

   ```
   .af R c
   ```

   where R is register name and c defines output format to be arabic, arabic with leading zeros, lowercase roman, uppercase roman, lowercase alphabetic, or uppercase alphabetic as shown for example,

c	Numbering Format
1	0,1,2,3, ...
001	000,001,002,003, ...
i	0,i,ii,iii, ...
I	0,I,II,III, ...
a	0,a,b,...,z,aa,ab,...,zz,aaa, ...
A	0,A,B,...,Z,AA,AB,...,ZZ,AAA, ...

4. When defining many registers, it is necessary to remove registers after they are used. This will recapture internal storage space for newer registers. Registers are removed by entering the **.rr** request. The **.rr** request has the following form:

```
.rr R
```

   where R is the number register name.

   Number register usage is explained in more detail in the *Text Formatters Reference*—select code 307-151.

FORMATTING YOUR DOCUMENT

1. Text Processors

Once you have created a file of text, you are ready to format it. Text processors prepare your files of text for printout on printers and phototypesetters. The DOCUMENTER'S WORKBENCH software provides the following text processors:

- **nroff** formats files for printing on typewriter-like devices (low-speed letter quality printers) and line printers.

- **otroff** formats files for a Wang Laboratories, Inc., C/A/T phototypesetter.

- **sroff** formats files for printing on typewriter-like devices and line printers.

- **troff** formats files for printing on a typesetter.

These text formatters are explained in detail in the *Text Formatters Reference*—select code 307-151.

The basic idea of formatting programs is that the text to be formatted contains within it "formatting commands" that determine in detail how the formatted text is to look. For example, there may be commands that specify how long lines are, whether to use single or double spacing, and the running titles to use on each page.

Formatting programs produce text with justified right margins, automatic page numbering and titling, automatic hyphenation, etc. The **nroff** (pronounced "en-roff") program is designed to produce output on terminals and line printers. The **nroff** program formats the text into a printable paginated document. The **troff** (pronounced "tee-roff") program is designed to drive a typesetter that produces high quality output on photographic paper. (This document was formatted with **troff**.) The **troff** output is a device-independent ASCII language that describes where characters are placed on a page by the typesetter. The output has been tailored to the resolution and font descriptions of a particular typesetter, but otherwise is independent of any particular device. The device-independent ASCII language is translated into the machine codes needed to run the particular typesetter by a program called a postprocessor.

2. Using Text Processors

The input form for invoking formatting programs is

> **nroff** options files
>
> or
>
> **otroff** options files
>
> or
>
> **sroff** options files
>
> or
>
> **troff** options files

where options are optional arguments and files are the names of files containing the document to be formatted. For example, to produce a document in standard format using the memorandum macros, use the option "**−mm**" as follows:

> **troff −mm** files ...

for the typesetter and

> **nroff −mm** files ...

for a terminal.

The **−mm** argument tells **troff** and **nroff** to use the memorandum macro package of formatting requests. There are several similar packages. Check with a local expert to determine what is in common use on your machine.

3. Preprocessors

The text processing programs are powerful and flexible. In many ways these text processing programs resemble assembly programs. Many operations must be specified at a level of detail and in a form that is too difficult for most people to use effectively. This is why macro packages were created. Macro packages are easier to use than the detailed requests of the text processing programs. Preprocessors were created for the same reason. You would have difficulty producing text containing mathematics, tables, or simple pictures using text processing requests. The preprocessors will aid you in producing these special applications. The DOCUMENTER'S WORKBENCH software provides the following preprocessors:

- **eqn** converts files containing mathematical equations and expressions for **troff** or **otroff** output.
- **neqn** converts files containing mathematical equations and expressions for **nroff** output.
- **ocw** converts files of constant width text for **otroff** output.
- **pic** converts files containing simple pictures for **troff** output.
- **tbl** converts files containing tables for **nroff**, **otroff**, or **troff** output.

These preprocessors are explained in detail in the *Preprocessors Reference*—select code 307-153.

The mathematics, pictures, or tables can be interspersed in your text. Each of the preprocessors has special names to define the beginning and end of input for it as follows:

Preprocessor	**Start Macro**	**End Macro**
eqn	.EQ	.EN
neqn	.EQ	.EN
ocw	.CW	.CN
pic	.PS	.PE
tbl	.TS	.TE

Each preprocessor simply copies the input files to the standard output except for the lines between its start and end macros. These lines are assumed to describe a mathematical equation or expression, picture, or table. The preprocessor converts the lines between the start and end macros into text processing (**nroff**, **otroff**, and **troff**) requests.

4. Using Preprocessors

The general form for using a preprocessor is

preprocessor options files ¦ textprocessor options

Some more specific examples are:

eqn options files ¦ **troff** options

eqn −**T**cat options files ¦ **otroff** options

neqn options files ¦ **nroff** options

ocw options files ¦ **otroff** options

pic options files ¦ **troff** options

tbl options files ¦ **nroff** options

See the manual entries in the Appendix to this book for more details.

5. Postprocessors

The DOCUMENTER'S WORKBENCH software provides the following post processors:

- **dx9700** prepares **troff** documents for the Xerox 9700 laser printer.
- **daps** prepares **troff** output for the Autologic APS-5 typesetter.

- **di10** prepares **troff** output for the Imagen Imprint-10 laser printer.
- **otc** prepares **otroff** output for a Tektronix 4014.
- **tc** prepares **troff** output for a Tektronix 4014.
- **x9700** prepares **nroff** documents for the Xerox 9700 laser printer.

The general form for using a postprocessor follows:

```
preprocessor opt files | textprocessor opt | postprocessor opt
```

where " opt" is options.

Some examples of using a postprocessor follow:

tbl options files | **nroff** options | **x9700** options

troff −Taps options files | **daps** options

otroff options files | **otc** options

RECOMMENDATIONS AND SUMMARY

1. Inputting

Most documents go through several versions (always more than expected) before they are finally finished. Accordingly, you should do whatever possible to make revisions easy.

First, when you do the purely mechanical operation of typing, type so that later editing will be easy. Start each sentence on a new line. Make lines short, and break lines at natural places, such as after commas and semicolons, rather than randomly. Since most people change documents by rewriting phrases and adding, deleting, and rearranging sentences, these precautions simplify any editing needed later.

Keep the individual files of a document down to modest size, perhaps less than 20,000 charcters. Larger files edit more slowly. If a mistake is made, it is better to clobber a small file than a big one. Split the files at natural boundaries in the document for the same reasons that you start each sentence on a new line.

2. Formatting

The second aspect of allowing documents to be easily changed is to not specify the formatting details too early. One advantage of formatting packages is permitting format decisions to be delayed until the last possible moment. Indeed, until a document is printed, it is not even decided whether it will be typeset or printed out on a line printer.

As a rule of thumb, a document should be produced by a set of requests or commands (macros) for all but the most trivial jobs. The macros used are defined either by using an existing macro package (the recommended way) or by defining your own **nroff** and **troff** macros. As long as the text is entered in some systematic way, it can always be cleaned up and formatted by a judicious combination of editing commands and macro definitions.

3. Printing

If you are a beginning user, obtain a hard copy of documents more than a few pages long for making corrections. Beginning users tend to make more mistakes in large documents than they can remember. Making major corrections at your desk prevents terminal tie-up. Mark the copy with corrections. Refer to the set of guides on the DOCUMENTER'S WORKBENCH software and the *UNIX System Editing Guide* when necessary. Then enter the corrections at the terminal in the unformatted raw text files, not the formatted files. The next step is to determine if the corrections worked as follows:

- If there were many corrections, reformat and print the entire document. Check the output to ensure all changes worked.

- For minor changes, reformat the corrected pages only. This can save time. For example, assume your document is 60 pages and corrections were made on pages 16, 23, 47, and 58. These pages can be formatted and stored in a file as follows:

```
nroff -o16,23,47,58 files>FILE
```

The same example that uses a preprocessor and postprocessor would be:

```
tbl files | nroff -o16,23,47,58 >FILE
```

The formatted file " FILE" can then be edited or printed to determine if the changes worked. Assume your printer is connected to your terminal and prints one page in two minutes. Instead of waiting two hours for a 60-page document to print out, the wait is about eight minutes. This saves paper and time, and allows others to use the terminal.

If you are an experienced user, view your formatted text on the screen. Text files formatted by **nroff** and **sroff** are printed on your terminal screen by default. Remember the mistakes and correct the input using the text editor of your choice. Follow the same advice given to beginners for determining if the corrections worked.

APPENDIX

USER REFERENCE MANUAL

PAGE

cat - phototypesetter interface A-2
daps, di10 .... A-3
dx9700 .... A-4
eqn, neqn, checkeq A-5
eqnchar A-7
font A-8
man, manprog A-12
man macros A-15
mm, osdd, checkmm A-18
mm macros A-20
mmlint A-21
mmt, mvt A-22
mosd adapter macros A-23
mptx macros A-25
mv - viewgraph macros A-26
non-btl A-28
nroff, otroff A-29
ocw, checkcw A-31
pic A-34
sroff A-35
tbl A-36
tc, otc A-38
troff A-39
troff output language A-40
x9700 .... A-43

APPENDIX

USER REFERENCE MANUAL

The following pages contain descriptions of the commands that are part of the DOCUMENTER'S WORKBENCH software. It is intended that these pages serve as a memory jogger for more experienced users and a source of information for less experienced users.

NAME

cat − phototypesetter interface

DESCRIPTION

Cat provides the interface to a Wang Laboratories, Inc. C/A/T phototypesetter. Bytes written on the file specify font, size, and other control information as well as the characters to be flashed. The coding will not be described here.

Only one process may have this file open at a time. It is write-only.

FILES

/dev/cat

SEE ALSO

nroff(1).

Wang Laboratories, Inc. specification (available on request).

NAME

daps, di10 − Postprocessors for the Autologic APS-5 phototypesetter and the Imagen Imprint-10 laser printer

SYNOPSIS

daps [option] ... [file] ...
di10 [option] ... [file] ...

DESCRIPTION

Daps and *di10* (formerly known as *dcan*) print *files* created by *troff*(1) on an Autologic APS-5 phototypesetter or on an Imagen Imprint-10 laser printer. If no *file* is mentioned, the standard input is printed. The following options are understood.

−b Report whether the typesetter is busy; do not print.

−h*string*
Print *string* in this job's header. A header will only be generated if either this option or the -H option is used. (*daps* only)

−H*file* Print the first line from *file* in this job's header. (*daps* only)

−o*list* Print pages whose numbers are given in the comma-separated *list*. The list contains single numbers *N* and ranges *N1* −*N2*. A missing *N1* means the lowest-numbered page, a missing *N2* means the highest.

−r Report the number of 11-inch pages generated by this job. (*daps* only)

−s*n* Stop after every *n* pages of output. Continue when the PROCEED button is pushed on the typesetter.

−t Direct output to the standard output instead of the typesetter.

−w Wait for typesetter to become free, then print.

The *files* submitted to *daps* should be prepared under the **−Taps** option of *troff*. *Di10* is a phototypesetter simulator and can handle *troff* output prepared for any supported typesetter. However, files sent to *di10* will look best when prepared with the **-Ti10** option of *troff*.

FILES

/dev/aps	APS-5 phototypesetter device
/usr/lib/font/devaps/*	description files for APS-5
/usr/lib/font/devi10/*	description files for Imagen Imprint-10
/usr/lib/font/devi10/rasti10/*	raster files for Imprint-10
/tmp/dcan*	output of *di10* ready for Imagen

SEE ALSO

tc(1), troff(1), troff(5).

BUGS

Installations with an Autologic APS-5 phototypesetter should be aware that getting a good match to their Autologic fonts will almost certainly require hand-tuning of the distributed font description files.

NAME

dx9700 — prepare troff documents for the Xerox 9700 printer

SYNOPSIS

dx9700 *name*

DESCRIPTION

The *dx9700* filter is a post-processor for device independent *troff* output, and produces codes suitable for being sent to a Xerox 9700 laser printer.

The single argument to *dx9700* should be the *name* part of the **−T***name* argument given to *troff*.

The output of the *dx9700* filter should be directed to the input of a Xerox 9700 printer.

Note that the Xerox 9700 treats different point sizes as different fonts. Hence, the font tables specified to *troff*(1) and *dx9700* actually specify a family of typefaces and point sizes. The font families that are supported for the Xerox 9700 and that can be specified to *troff* using the **-T** option follow:

name	**contains**
X97.tim10p	Times, 7, 10, and 15 point
X97.tim12p	Times, 9, 12, and 17 point

SEE ALSO

troff(1), troff(5).

BUGS

Special fonts for the Xerox 9700 printer are needed to use with this post-processor.

NAME

eqn, neqn, checkeq — format mathematical text for nroff or troff

SYNOPSIS

eqn [**−d**xy] [**−p**n] [**−s**n] [**−f**n] [**−T**dest] [files]

neqn [**−d**xy] [**−p**n] [**−s**n] [**−f**n] [files]

checkeq [files]

DESCRIPTION

Eqn is a *troff*(1) preprocessor for typesetting mathematical text on a phototypesetter, while *neqn* is used for the same purpose with *nroff* on typewriter-like terminals. Usage is almost always:

```
eqn files | troff
neqn files | nroff
```

or equivalent. If no files are specified (or if − is specified as the last argument), these programs read the standard input. *Eqn* prepares output for the typesetter named in the −T option. Currently supported devices are −Taps (Autologic APS-5), **-TX97** (Xerox 9700), **-Ti10** (Imagen Imprint-10), and −Tcat (Wang CAT). Default is −Taps.

A line beginning with **.EQ** marks the start of an equation; the end of an equation is marked by a line beginning with **.EN**. Neither of these lines is altered, so they may be defined in macro packages to get centering, numbering, etc. It is also possible to designate two characters as *delimiters*; subsequent text between delimiters is then treated as *eqn* input. Delimiters may be set to characters *x* and *y* with the command-line argument **−d***xy* or (more commonly) with **delim** *xy* between **.EQ** and **.EN**. The left and right delimiters may be the same character; the dollar sign is often used as such a delimiter. Delimiters are turned off by **delim off**. All text that is neither between delimiters nor between .EQ and .EN is passed through untouched.

The program *checkeq* reports missing or unbalanced delimiters and **.EQ/.EN** pairs.

Tokens within *eqn* are separated by spaces, tabs, new-lines, braces, double quotes, tildes, and circumflexes. Braces {} are used for grouping; generally speaking, anywhere a single character such as *x* could appear, a complicated construction enclosed in braces may be used instead. Tilde (~) represents a full space in the output, circumflex (^) half as much.

Subscripts and superscripts are produced with the keywords **sub** and **sup**. Thus *x sub j* makes x_j, *a sub k sup 2* produces a_k^2, while $e^{x^2+y^2}$ is made with *e sup {x sup 2 + y sup 2}*. Fractions are made with **over**: *a over b* yields $\frac{a}{b}$; **sqrt** makes square roots: *1 over sqrt {ax sup 2+bx+c}* results in $\frac{1}{\sqrt{ax^2+bx+c}}$.

The keywords **from** and **to** introduce lower and upper limits: $\lim\limits_{n \to \infty} \sum_0^n x_i$ is made with *lim from {n −> inf } sum from 0 to n x sub i*. Left and right brackets, braces, etc., of the right height are made with **left** and **right**: *left [x sup 2 + y sup 2 over alpha right] ~=~ 1* produces $\left[x^2 + \frac{y^2}{\alpha} \right] = 1$.

Legal characters after **left** and **right** are braces, brackets, bars, **c** and **f** for ceiling and floor, and "" for nothing at all (useful for a right-side-only bracket). A **left** *thing* need not have a matching **right** *thing*.

Vertical piles of things are made with **pile**, **lpile**, **cpile**, and **rpile**: *pile {a above b above c}* produces $\begin{matrix}a\\b\\c\end{matrix}$. Piles may have arbitrary numbers of elements; **lpile** left-justifies, **pile** and **cpile** center (but with different vertical spacing), and **rpile** right justifies. Matrices are made with **matrix**: *matrix { lcol { x sub i above y sub 2 } ccol { 1 above 2 } }* produces $\begin{matrix}x_i & 1\\ y_2 & 2\end{matrix}$. In addition, there is **rcol** for a right-justified column.

Diacritical marks are made with **dot**, **dotdot**, **hat**, **tilde**, **bar**, **vec**, **dyad**, and **under**: *x dot = f(t) bar* is $\dot{x}=\overline{f(t)}$, *y dotdot bar ~=~ n under* is $\overline{\ddot{y}} = \underline{n}$, and *x vec ~=~ y dyad* is $\vec{x} = \overleftrightarrow{y}$.

Point sizes and fonts can be changed with **size** *n* or **size** ±*n*, **roman**, **italic**, **bold**, and **font** *n*. Point sizes and fonts can be changed globally in a document by **gsize** *n* and **gfont** *n*, or by the command-line arguments **−s***n* and **−f***n*.

Normally, subscripts and superscripts are reduced by 3 points from the previous size; this may be changed by the command-line argument **−p***n*.

Successive display arguments can be lined up. Place **mark** before the desired lineup point in the first equation; place **lineup** at the place that is to line up vertically in subsequent equations.

Shorthands may be defined or existing keywords redefined with **define**:

    define thing % replacement %

defines a new token called *thing* that will be replaced by *replacement* whenever it appears thereafter. The % may be any character that does not occur in *replacement*.

Keywords such as **sum** ($\sum$), **int** ($\int$), **inf** (∞), and shorthands such as >= ($\geqslant$), != ($\neq$), and −> ($\rightarrow$) are recognized. Greek letters are spelled out in the desired case, as in **alpha** (α), or **GAMMA** (Γ). Mathematical words such as **sin**, **cos**, and **log** are made Roman automatically. *Troff*(1) four-character escapes such as **\(dd** (‡) and **\(bs** (🔔) may be used anywhere. Strings enclosed in double quotes ("...") are passed through untouched; this permits keywords to be entered as text, and can be used to communicate with *troff*(1) when all else fails. Full details are given in the manual cited below.

SEE ALSO

mm(1), mmt(1), nroff(1), tbl(1), troff(1), eqnchar(5), mm(5), mv(5).

DOCUMENTER'S WORKBENCH Software Preprocessor Reference.

BUGS

To embolden digits, parentheses, etc., it is necessary to quote them, as in **bold "12.3"**.
See also *BUGS* under *troff*(1).

NAME

eqnchar − special character definitions for eqn and neqn

SYNOPSIS

eqn /usr/pub/eqnchar [files] | **troff** [options]

neqn /usr/pub/eqnchar [files] | **nroff** [options]

eqn -Taps /usr/pub/apseqnchar [files] | **troff** [options]

eqn -Tcat /usr/pub/cateqnchar [files] | **otroff** [options]

DESCRIPTION

Eqnchar contains *troff*(1) and *nroff*(1) character definitions for constructing characters that are not available on a phototypesetter. These definitions are primarily intended for use with *eqn*(1) and *neqn*; *eqnchar* contains definitions for the following characters:

ciplus	*ciplus*	| |	||	*square*	*square*
citimes	*citimes*	*langle*	*langle*	*circle*	*circle*
wig	*wig*	*rangle*	*rangle*	*blot*	*blot*
−*wig*	−*wig*	*hbar*	*hbar*	*bullet*	*bullet*
>*wig*	>*wig*	*ppd*	*ppd*	*prop*	*prop*
<*wig*	<*wig*	< −>	<→	*empty*	*empty*
=*wig*	=*wig*	< =>	⩽>	*member*	*member*
star	*star*	|<	|<	*nomem*	*nomem*
bigstar	*bigstar*	|>	|>	*cup*	*cup*
=*dot*	=*dot*	*ang*	*ang*	*cap*	*cap*
orsign	*orsign*	*rang*	*rang*	*incl*	*incl*
andsign	*andsign*	*3dot*	3*dot*	*subset*	*subset*
=*del*	=*del*	*thf*	*thf*	*supset*	*supset*
oppA	*oppA*	*quarter*	*quarter*	*!subset*	!*subset*
oppE	*oppE*	*3quarter*	3*quarter*	*!supset*	!*supset*
angstrom	*angstrom*	*degree*	*degree*	*scrL*	*scrL*
= =<	==<	= =>	==>		

Apseqnchar is a version of *eqnchar* tailored for the Autologic APS-5 phototypesetter. This will not look optimal on other phototypesetters. Similarly, *cateqnchar* is the old *eqnchar* tailored for the Wang CAT and the old *otroff*. Until a phototypesetter-independent version of *eqnchar* is available, *eqnchar* should be a link to the default version on each system. The standard default is *apseqnchar*.

FILES

/usr/pub/eqnchar
/usr/pub/apseqnchar
/usr/pub/cateqnchar

SEE ALSO

eqn(1), nroff(1), troff(1).

NAME

font − description files for device-independent troff

SYNOPSIS

troff −Tptty ...

DESCRIPTION

For each phototypesetter supported by *troff*(1) and available on this system, there is a directory containing files describing the device and its fonts. This directory is named **/usr/lib/font/dev***ptty* where *ptty* is the name of the phototypesetter. Currently the only *ptty* supported is **aps** for the Autologic APS−5.

For a particular phototypesetter, *ptty*, the ASCII file *DESC* in the directory **/usr/lib/font/dev***ptty* describes its characteristics. Each line starts with a word identifying the characteristic and followed by appropriate specifiers. Blank lines and lines beginning with a # are ignored.

The legal lines for *DESC* are:

res *num*	resolution of device in basic increments per inch
hor *num*	smallest unit of horizontal motion
vert *num*	smallest unit of vertical motion
unitwidth *num*	pointsize in which widths are specified
sizescale *num*	scaling for fractional pointsizes
paperwidth *num*	width of paper in basic increments
paperlength *num*	length of paper in basic increments
spare1 *num*	available for use
spare2 *num*	available for use
sizes *num num ...*	list of pointsizes available on typesetter
fonts *num name ...*	number of initial fonts followed by the names of the fonts. For example: fonts 4 R I B S
charset	this always comes last in the file and is on a line by itself. Following it is the list of special character names for this device. Names are separated by a space or a newline. The list can be as long as necessary. Names not in this list are not allowed in the font description files.

Res is the basic resolution of the device in increments per inch. **Hor** and **vert** describe the relationships between motions in the horizontal and vertical directions. If the device is capable of moving in single basic increments in both directions, both **hor** and **vert** would have values of 1. If the vertical motions only take place in multiples of two basic units while the horizontal motions take place in the basic increments, then **hor** would be 1, while **vert** would be 2. **Unitwidth** is the pointsize in which all width tables in the font description files are given. *Troff* automatically scales the widths from the **unitwidth** size to the pointsize it is working with. **Sizescale** is not currently used and is 1. **Paperwidth** is the width of the paper in basic increments. The APS-5 is 6120 increments wide. **Paperlength** is the length of a sheet of paper in the basic increments.

For each font supported by the phototypesetter, there is also an ASCII file with the same name as the font (e.g., **R**, **I**, **CW**). The format for a font description file is:

name *name*	name of the font, such as **R** or **CW**
internalname *name*	internal name of font
special	sets flag indicating that the font is special
ligatures *name* ... **0**	Sets flag indicating font has ligatures. The list of ligatures follows and is terminated by a zero. Accepted ligatures are: **ff fi fl ffi ffl**.
spare1	available for use
spacewidth *num*	width of space if something other than 1/3 of \(em is desired as a space.
charset	The charset must come at the end. Each line following the word *charset* describes one character in the font. Each line has one of two formats: *name* *width* *kerning code* *name* "

where *name* is either a single ASCII character or a special character name from the list found in *DESC*. The width is in basic increments. The kerning information is 1 if the character descends below the line, 2 if it rises above the letter 'a', and 3 if it both rises and descends. The kerning information for special characters is not used and so may be 0. The code is the number sent to the typesetter to produce the character. The second format is used to indicate that the character has more than one name. The double quote indicates that this name has the same values as the preceding line. The kerning and code fields are not used if the width field is a double quote character.

Troff and its postprocessors read this information from binary files produced from the ASCII files by a program distributed with *troff* called *makedev*. For those with a need to know, a description of the format of these files follows:

The file *DESC.out* starts with the *dev* structure, defined by *dev.h*:

```
 /*
dev.h: characteristics of a typesetter
* /

struct dev {
short    filesize;     /* number of bytes in file, */
                /* excluding dev part */
short    res;          /* basic resolution in goobies/inch */
short    hor;          /* goobies horizontally */
short    vert;
short    unitwidth;    /* size at which widths are given*/
short    nfonts;       /* number fonts physically available */
short    nsizes;       /* number of pointsizes */
short    sizescale;    /* scaling for fractional pointsizes */
short    paperwidth;   /* max line length in units */
short    paperlength;   /* max paper length in units */
short    nchtab;       /* number of funny names in chtab */
short    lchname;      /* length of chname table */
short    spare1;       /* in case of expansion */
short    spare2;
};
```

Filesize is just the size of everything in *DESC.out* excluding the *dev* structure. *Nfonts* is the number of different font positions available. *Nsizes* is the number of different pointsizes supported by this typesetter. *Nchtab* is the number of special character names. *Lchname* is the total number of characters, including nulls, needed to list all the special character names. At the end of the structure are two spares for later expansions.

Immediately following the *dev* structure are a number of tables. First is the *sizes* table, which contains *nsizes* + 1 shorts(a null at the end), describing the pointsizes of text available on this device. The second table is the *funny_char_index_table*. It contains indices into the table which follows it, the *funny_char_strings*. The indices point to the beginning of each special character name which is stored in the *funny_char_strings* table. The *funny_char_strings* table is *lchname* characters long, while the *funny_char_index_table* is *nchtab* shorts long.

Following the *dev* structure will occur *nfonts* {*font*}*.out* files, which are used to initialize the font positions. These {*font*}*.out* files, which also exist as separate files, begin with a *font* structure and then are followed by four character arrays:

```
struct font {        /* characteristics of a font */
char nwfont;        /* number of width entries */
char specfont;      /* 1 == special font */
char  ligfont;      /* 1 == ligatures exist on this font */
char spare1;        /* unused for now */
char namefont[10]; /* name of this font, e.g., R */
char intname[10];  /* internal name of font, in ASCII */
};
```

The *font* structure tells how many defined characters there are in the font, whether the font is a "special" font and if it contains ligatures. It also has the ASCII name of the font, which should match the name of the file it appears in, and the internal name of the font on the typesetting device (*intname*). The internal name is independent of the font position and name that *troff* knows about. For example, you might say mount R in position 4, but when asking the typesetter to actually produce a character from the R font, the postprocessor which instructs the typesetter would use *intname*.

The first three character arrays are specific for the font and run in parallel. The first array, *widths*, contains the width of each character relative to *unitwidth*. *Unitwidth* is defined in *DESC*. The second array, *kerning*, contains kerning information. If a character rises above the letter 'a', 02 is set. If it descends below the line, 01 is set. The third array, *codes*, contains the code that is sent to the typesetter to produce the character.

The fourth array is defined by the device description in *DESC*. It is the *font_index_table*. This table contains indices into the *width*, *kerning*, and *code* tables for each character. The order that characters appear in these three tables is arbitrary and changes from one font to the next. In order for *troff* to be able to translate from ASCII and the special character names to these arbitrary tables, the *font_index_table* is created with an order which is constant for each device. The number of entries in this table is 96 plus the number of special character names for this device. The value 96 is 128 - 32, the number of printable characters in the ASCII alphabet. To determine whether a normal ASCII character exists, *troff* takes the ASCII value of the character, subtracts 32, and looks in the *font_index_table*. If it finds a 0, the character is not defined in this font. If it finds anything else, that is the index into *widths*, *kerning*, and *codes* that describe that character.

To look up a special character name, for example **\(pl**, the mathematical plus sign, and determine whether it appears in a particular font or not, the following procedure is followed. A *counter* is set to 0 and an index to a special character name is picked out of the *counter'th* position in the *funny_char_index_table*. A string comparision is performed between *funny_char_strings [funny_char_index_table [counter]]* and the special character name, in our example **pl**, and if it matches, then *troff* refers to this character as (96 + *counter*). When it wants to determine whether a specific font supports this character, it looks in *font_index_table[(96+counter)]*, (see below), to see whether there is a 0, meaning the character does not appear in this font, or number, which is the index into the *widths*, *kerning*, and *codes* tables.

Notice that since a value of 0 in the *font_index_table* indicates that a character does not exist, the 0th element of the *width*, *kerning*, and *codes* arrays are not used. For this reason the 0th element of the *width* array can be used for a special purpose, defining the width of a space for a font. Normally a space is defined by *troff* to be 1/3 of the width of the \(em character, but if the 0th element of the *width* array is non-zero, then that value is used for the width of a space.

SEE ALSO

troff(1), troff(5).

FILES

/usr/lib/font/dev{X}/DESC.out	description file for phototypesetter X
/usr/lib/font/dev{X}/{font}.out	font description files for phototypesetter X

NAME

macref – produce cross-reference listing of macro files

SYNOPSIS

macref [−t] [−s] [−n] file ...

DESCRIPTION

The *macref* program reads the named files (which are assumed to be *nroff*(1)/*troff*(1) input) and produces a cross-reference listing of the symbols in the input.

A **−t** in the command line causes a macro table of contents to be printed. A **−s** causes symbol use statistics to be output.

The default output is a list of the symbols found in the input, each accompanied by a list of all references to that symbol. (This output may be defeated by using a **−n** in the command line). The symbols are listed alphabetically in the leftmost column, with the references following to the right. Each reference is given in the form:

[[(*NMname*)] *Mname*—] *type lnum* [#]

where the fields have the following meanings:

Mname the name of the macro within which the reference occurs. This field is missing if the reference occurs at the text level. Any names listed in the *NMname* part are macros within which *Mname* is defined.

type the type associated, by context, with this occurrence of the symbol. The types may be:

r	request
m	macro
d	diversion
s	string
n	number register
p	parameter (e.g.,\$x is a parameter reference to x. Note that parameters are never modified, and that the only valid parameter symbol names are 1, 2, ... 9).

lnum the line number on which the reference occurred.

# this reference modifies the value of the symbol.

Generated names are listed under the artificial symbol name "~sym".

SEE ALSO

nroff(1), troff(1).

NAME

man, manprog − print entries in this manual

SYNOPSIS

man [options] [section] titles

/usr/lib/manprog file

DESCRIPTION

Man locates and prints the entry of this manual named *title* in the specified *section*. (For historical reasons, the word "page" is often used as a synonym for "entry" in this context.) The *title* is entered in lower case. The *section* number may not have a letter suffix. If no *section* is specified, the whole manual is searched for *title* and all occurrences of it are printed. *Options* and their meanings are:

−t	Typeset the entry in the default format (8.5″×11″).
−s	Typeset the entry in the small format (6″×9″).
−Tcat	Use *otroff*(1) to generate output for an on-line Wang CAT phototypesetter.
−D4014	Display the typeset output on a TEKTRONIX 4014 terminal using *tc*(1).
−Dtek	Same as **−D4014**.
−Di10	Send typeset output to the local Imagen Imprint-10 laser printer.
−T*term*	If *term* is one of the recognized *troff* devices (see *troff*(1)), format the entry for that device. Otherwise format the entry using *nroff* and print it on the standard output (usually, the terminal); *term* is the terminal type (see *term*(5) and the explanation below); for a list of recognized values of *term*, type **help term2**. The default value of *term* is **450**.
−w	Print on the standard output only the *pathnames* of the entries, relative to **/usr/man**, or to the current directory for **−d** option.
−d	Search the current directory rather than **/usr/man**; requires the full file name (e.g., **cu.1c**, rather than just **cu**).
−12	Indicates that the manual entry is to be produced in 12-pitch. May be used when **$TERM** (see below) is set to one of **300**, **300s**, **450**, and **1620**. (The pitch switch on the DASI 300 and 300s terminals must be manually set to **12** if this option is used.)
−c	Causes *man* to invoke *col*(1); note that *col*(1) is invoked automatically by *man* unless *term* is one of **300**, **300s**, **450**, **37**, **4000a**, **382**, **4014**, **tek**, **1620**, and **X**.
−y	Causes *man* to use the non-compacted version of the macros.
−z	Invokes no output filter to process or redirect the output of *troff*(1).

The above *options* other than **−d**, **−c**, and **−y** are mutually exclusive, except that the **−s** and **−z** options may be used in conjunction with any typesetter option (6″×9″ pages may be produced with *nroff* by including the **−rs1** option). Any other *options* are passed to *troff*, *nroff*, or the *man*(5) macro package.

When using *nroff*, *man* examines the environment variable **$TERM** (see *environ*(5)) and attempts to select options to *nroff*, as well as filters, that adapt the output to the terminal being used. The −T*term* option overrides the value of **$TERM**; in particular, one should use **−Tlp** when sending the output of *man* to a line printer.

Section may be changed before each *title*.

As an example:

```
man man
```

would reproduce on the terminal this entry, as well as any other entries named *man* that may exist in other sections of the manual, e.g., *man*(5).

If the first line of the input for an entry consists solely of the string:

```
'\" x
```

where *x* is any combination of the two characters **e**, and **t**, and where there is exactly one blank between the double quote (") and *x*, then *man* will preprocess its input through the appropriate combination of *eqn*(1) (*neqn* for *nroff*) and *tbl*(1), respectively. If *eqn* or *neqn* are invoked, they will automatically read the file **/usr/pub/eqnchar** (see *eqnchar*(5)).

The *man* command executes *manprog* that takes a file name as its argument. *Manprog* calculates and returns a string of three register definitions used by the formatters identifying the date the file was last modified. The returned string has the form:

−rd*day* **−rm***month* **−ry***year*

and is passed to *nroff* which sets this string as variables for the *man* macro package. Months are given from 0 to 11, therefore month is always 1 less than the actual month. The *man* macros calculate the correct month. If the *man* macro package is invoked as an option to *nroff*/*troff* (i.e., *nroff −man file*), then the current day/month/year is used as the printed date.

FILES

/usr/man/u_man/man[1,6]/*	the *UNIX System User Reference Manual*
/usr/man/a_man/man[1,7,8]/*	the *UNIX System Administrator Reference Manual*
/usr/man/p_man/man[2-5]/*	the *UNIX System Programmer Reference Manual*
/usr/man/local/man[1-8]/*	local additions
/usr/man/*/man[1-8]/*	any other additions
/usr/lib/manprog	calculates modification dates of entries

SEE ALSO

daps(1), eqn(1), nroff(1), tbl(1), tc(1), troff(1), environ(5), man(5), term(5).

BUGS

All entries are supposed to be reproducible either on a typesetter or on a terminal. However, on a terminal some information is necessarily lost.

Pages bearing the same name in all three manuals will result in the *UNIX System Administrator Reference Manual* entry being printed first, if no *section* argument is supplied.

6"×9" manual entries formatted by *nroff* (with the **−rs1** option) are not guaranteed to look as good as regular-sized entries.

NAME

man − macros for formatting entries in this manual

SYNOPSIS

nroff −man files

troff −man [**−rs1**] files

DESCRIPTION

These *troff*(1) macros are used to lay out the format of the entries of this manual. A skeleton entry may be found in the file **/usr/man/u_man/man0/skeleton**. These macros are used by the *man*(1) command.

The default page size is 8.5″×11″, with a 6.5″×10″ text area; the **−rs1** option reduces these dimensions to 6″×9″ and 4.75″×8.375″, respectively; this option (which is *not* effective in *nroff*(1)) also reduces the default type size from 10-point to 9-point, and the vertical line spacing from 12-point to 10-point. The **−rV2** option may be used to set certain parameters to values appropriate for certain Versatec printers: it sets the line length to 82 characters, the page length to 84 lines, and it inhibits underlining; this option should not be confused with the **−Tvp** option of the *man*(1) command, which is available at some UNIX system sites.

Any *text* argument below may be one to six "words". Double quotes (**""**) may be used to include blanks in a "word". If *text* is empty, the special treatment is applied to the next line that contains text to be printed. For example, **.I** may be used to italicize a whole line, or **.SM** followed by **.B** to make small bold text. By default, hyphenation is turned off for *nroff*(1), but remains on for *troff*(1).

Type font and size are reset to default values before each paragraph and after processing font- and size-setting macros, e.g., **.I**, **.RB**, **.SM**. Tab stops are neither used nor set by any macro except **.DT** and **.TH**.

Default units for indents *in* are ens. When *in* is omitted, the previous indent is used. This remembered indent is set to its default value (7.2 ens in *troff*(1), 5 ens in *nroff*this corresponds to 0.5″ in the default page size) by **.TH**, **.P**, and **.RS**, and restored by **.RE**.

.TH *t s c n*	Set the title and entry heading; *t* is the title, *s* is the section number, *c* is extra commentary, e.g., "local", *n* is new manual name. Invokes **.DT** (see below).
.SH *text*	Place subhead *text*, e.g., **SYNOPSIS**, here.
.SS *text*	Place sub-subhead *text*, e.g., **Options**, here.
.B *text*	Make *text* bold.
.I *text*	Make *text* italic.
.SM *text*	Make *text* 1 point smaller than default point size.
.RI *a b*	Concatenate roman *a* with italic *b*, and alternate these two fonts for up to six arguments. Similar macros alternate between any two of roman, italic, and bold: **.IR .RB .BR .IB .BI**
.P	Begin a paragraph with normal font, point size, and indent. **.PP** is a synonym for **.P**.
.HP *in*	Begin paragraph with hanging indent.
.TP *in*	Begin indented paragraph with hanging tag. The next line that contains text to be printed is taken as the tag. If the tag does not fit, it is printed on a separate line.
.IP *t in*	Same as **.TP** *in* with tag *t*; often used to get an indented paragraph without a tag.
.RS *in*	Increase relative indent (initially zero). Indent all output an extra *in* units from the current left margin.

.RE *k*	Return to the *k*th relative indent level (initially, *k*=1; *k*=0 is equivalent to *k*=1); if *k* is omitted, return to the most recent lower indent level.
.PM *m*	Produces proprietary markings; where *m* may be **P** for **PRIVATE**, **N** for **NOTICE**, **BP** for **BELL LABORATORIES PROPRIETARY**, or **BR** for **BELL LABORATORIES RESTRICTED**.
.DT	Restore default tab settings (every 7.2 ens in *troff*(1), 5 ens in *nroff*(1)).
.PD *v*	Set the interparagraph distance to *v* vertical spaces. If *v* is omitted, set the interparagraph distance to the default value (0.4v in *troff*(1), 1v in *nroff*(1)).

The following *strings* are defined:

***R**	® in *troff*(1), **(Reg.)** in *nroff*.
***S**	Change to default type size.
***(Tm**	Trademark indicator.

The following *number registers* are given default values by .TH:

IN	Left margin indent relative to subheads (default is 7.2 ens in *troff*(1), 5 ens in *nroff*(1)).
LL	Line length including **IN**.
PD	Current interparagraph distance.

CAVEATS

In addition to the macros, strings, and number registers mentioned above, there are defined a number of *internal* macros, strings, and number registers. Except for names predefined by *troff (1)* and number registers **d**, **m**, and **y**, all such internal names are of the form *XA*, where *X* is one of),], and }, and *A* stands for any alphanumeric character.

If a manual entry needs to be preprocessed by *eqn*(1) (or *neqn*), and/or *tbl*(1), it must begin with a special line (described in *man*(1)), causing the *man* command to invoke the appropriate preprocessor(s).

The programs that prepare the Table of Contents and the Permuted Index for this Manual assume the *NAME* section of each entry consists of a single line of input that has the following format:

```
name[, name, name ...] \— explanatory text
```

The macro package increases the inter-word spaces (to eliminate ambiguity) in the *SYNOPSIS* section of each entry.

The macro package itself uses only the roman font (so that one can replace, for example, the bold font by the constant-width font (**CW**). Of course, if the input text of an entry contains requests for other fonts (e.g., **.I**, **.RB**, **\fI**), the corresponding fonts must be mounted.

FILES

/usr/lib/tmac/tmac.an
/usr/lib/macros/cmp.n.[dt].an
/usr/lib/macros/ucmp.n.an
/usr/man/[uap]_man/man0/skeleton

SEE ALSO

ocw(1), eqn(1), man(1), nroff(1), tbl(1), tc(1), troff(1).

BUGS

If the argument to **.TH** contains *any* blanks and is *not* enclosed by double quotes (**""**), there will be strange irregular dots on the output.

NAME

mm, osdd, checkmm − print/check documents formatted with the MM macros

SYNOPSIS

mm [options] [files]

osdd [options] [files]

checkmm [files]

DESCRIPTION

Mm can be used to type out documents using *nroff* and the MM text-formatting macro package. It has options to specify preprocessing by *tbl*(1) and/or *neqn* (see *eqn*(1)) and postprocessing by various terminal-oriented output filters. The proper pipelines and the required arguments and flags for *nroff* and MM are generated, depending on the options selected.

Osdd is equivalent to the command **mm −mosd**. For more information about the OSDD adapter macro package, see *mosd*(5).

Options for *mm* are given below. Any other arguments or flags (e.g., **−rC3**) are passed to *nroff* or to MM, as appropriate. Such options can occur in any order, but they must appear before the *files* arguments. If no arguments are given, *mm* prints a list of its options.

−T*term*	Specifies the type of output terminal; for a list of recognized values for *term*, type **help term2**. If this option is *not* used, *mm* will use the value of the shell variable **$TERM** from the environment (see *profile*(4) and *environ*(5)) as the value of *term*, if **$TERM** is set; otherwise, *mm* will use **450** as the value of *term*. If several terminal types are specified, the last one takes precedence.
−12	Indicates that the document is to be produced in 12-pitch. May be used when **$TERM** is set to one of **300**, **300s**, **450**, and **1620**. (The pitch switch on the DASI 300 and 300s terminals must be manually set to **12** if this option is used.)
−c	Causes *mm* to invoke *col*(1); note that *col*(1) is invoked automatically by *mm* unless *term* is one of **300**, **300s**, **450**, **37**, **4000a**, **382**, **4014**, **tek**, **1620**, and **X**.
−e	Causes *mm* to invoke *neqn*; also causes *neqn* to read the **/usr/pub/eqnchar** file (see *eqnchar*(5)).
−t	Causes *mm* to invoke *tbl*(1).
−E	Invokes the **−e** option of *nroff*.
−y	Causes *mm* to use the non-compacted version of the macros (see *mm*(5)).

As an example (assuming that the shell variable **$TERM** is set in the environment to **450**), the two command lines below are equivalent:

```
mm −t −rC3 −12 ghh*
tbl ghh* | nroff −cm −T450−12 −h −rC3
```

Mm reads the standard input when − is specified instead of any file names. (Mentioning other files together with − leads to disaster.) This option allows *mm* to be used as a filter, e.g.:

```
cat dws | mm −
```

Checkmm is a program for checking the contents of the named *files* for errors in the use of the Memorandum Macros, missing or unbalanced *neqn* delimiters, and **.EQ/.EN** pairs. Note: The user need not use the *checkeq* program (see *eqn*(1)). Appropriate messages are produced. The program skips all directories, and if no file name is given, standard input is read.

HINTS

1. *Mm* invokes *nroff* with the **−h** flag. With this flag, *nroff* assumes that the terminal has tabs set every 8 character positions.
2. Use the **−o***list* option of *nroff* to specify ranges of pages to be output. Note, however, that *mm*, if invoked with one or more of the **−e**, **−t**, and **−** options, *together* with the **−o***list* option of *nroff* may cause a harmless "broken pipe" diagnostic if the last page of the document is not specified in *list*.
3. If you use the **−s** option of *nroff* (to stop between pages of output), use line-feed (rather than return or new-line) to restart the output. The **−s** option of *nroff* does not work with the **−c** option of *mm*, or if *mm* automatically invokes *col*(1) (see **−c** option above).
4. If you lie to *mm* about the kind of terminal its output will be printed on, you will get (often subtle) garbage; however, if you are redirecting output into a file, use the **−T37** option, and then use the appropriate terminal filter when you actually print that file.

SEE ALSO

col(1), env(1), eqn(1), greek(1), mmt(1), nroff(1), tbl(1), profile(4), mm(5), mosd(5), term(5).

DIAGNOSTICS

mm	"mm: no input file" if none of the arguments is a readable file and *mm* is not used as a filter.
checkmm	"Cannot open *filename*" if file(s) is unreadable. The remaining output of the program is diagnostic of the source file.

NAME

mm − the MM macro package for formatting documents

SYNOPSIS

mm [options] [files]

nroff −mm [options] [files]

nroff −cm [options] [files]

mmt [options] [files]

troff −mm [options] [files]

DESCRIPTION

This package provides a formatting capability for a very wide variety of documents. It is the standard package used by the BTL typing pools and documentation centers. The manner in which a document is typed in and edited is essentially independent of whether the document is to be eventually formatted at a terminal or is to be phototypeset. See the references below for further details.

The **−mm** option causes *nroff*(1) and *troff*(1) to use the non-compacted version of the macro package, while the **−cm** option results in the use of the compacted version, thus speeding up the process of loading the macro package.

FILES

/usr/lib/tmac/tmac.m	pointer to the non-compacted version of the package
/usr/lib/macros/mm[nt]	non-compacted version of the package
/usr/lib/macros/cmp.n.[dt].m	compacted version of the package
/usr/lib/macros/ucmp.n.m	initializers for the compacted version of the package

SEE ALSO

mm(1), mmt(1), nroff(1), troff(1).

NAME

mmlint − sroff/MM nroff/MM document compatibility checker

SYNOPSIS

mmlint −s file
mmlint −n file

DESCRIPTION

Mmlint reads *file* (an input document) and reports the document changes needed to convert the document to be runnable by the text formatter specified by the option.

−s *mmlint* will flag nroff/MM constructs that are illegal in sroff/MM.

−n *mmlint* will flag sroff/MM constructs that are illegal in nroff/MM.

Constructs are commands, embedded commands, or register references.

There are three types of messages:

Equivalent messages,
which give the equivalent construct in the target formatter.

Non-equivalent messages,
which indicate that there is no equivalent construct in the target formatter.

Warning messages,
which describe the different meanings of a command or argument in each formatter.

Messages are output on standard output.

CAVEATS

With the **-s** option, *mmlint* assumes the input file is in *nroff/MM* format. However, if the file is in *sroff/MM* format, some erroneous messages may appear. For example,

> **\(ad\(asr)): no special chars in sroff**

although this is a legal register construct in *sroff*.

The same characteristic is true for the **−n** option, with the following messages:

> **\(sl): use \n(sl) in nroff**

although in *nroff*, this is the character sequence "/)".

> **\t: use \nt in nroff**

although in *nroff*, **\t** is the tab escape sequence.

> **\(:Mu): register names can only be two characters long in nroff**

although **:M** is a legal register name in *nroff*.

.so and **.nx** requests are ignored by **mmlint**.

NAME

mmt, mvt – typeset documents, viewgraphs, and slides

SYNOPSIS

mmt [options] [files]

mvt [options] [files]

DESCRIPTION

These two commands are very similar to *mm*(1), except that they both typeset their input via *troff*(1), as opposed to formatting it via *nroff*(1); *mmt* uses the MM macro package, while *mvt* uses the Macro Package for View Graphs and Slides. These two commands have options to specify preprocessing by *tbl*(1) and/or *pic*(1) and/or *eqn*(1). The proper pipelines and the required arguments and flags for *troff*(1) and for the macro packages are generated, depending on the options selected.

Options are given below. Any other arguments or flags (e.g., **−rC3**) are passed to *troff*(1) or to the macro package, as appropriate. Such options can occur in any order, but they must appear before the *files* arguments. If no arguments are given, these commands print a list of their options.

−e	Causes these commands to invoke *eqn*(1); also causes *eqn* to read the **/usr/pub/eqnchar** file (see *eqnchar*(5)).
−t	Causes these commands to invoke *tbl*(1).
−p	Invokes *pic*(1).
−Taps	Creates output for an Autologic APS-5 phototypesetter, and sends it to the default destination at this installation.
−T*dest*	Creates output for *troff* device *dest* (see *troff*(1)). The output is sent through the appropriate postprocessor (see *daps*(1)).
−Tcat	Uses *otroff*(1) to generate output for an on-line Wang CAT phototypesetter.
−D4014	Directs the output to a TEKTRONIX 4014 terminal via the *tc*(1) filter.
−Dtek	Same as **−D4014**.
−Di10	Directs the output to the local Imagen Imprint-10 laser printer.
−a	Invokes the **−a** option of *troff*(1).
−y	Causes *mmt* to use the non-compacted version of the macros. This is the default except when using **−Tcat**.
−z	Invokes no output filter to process or redirect the output of *troff*(1).

These commands read the standard input when − is specified instead of any file names.

Mvt is just a link to *mmt*.

HINT

Use the **−o***list* option of *troff*(1) to specify ranges of pages to be output. Note, however, that these commands, if invoked with one or more of the **−e**, **−t**, and − options, *together* with the **−o***list* option of *troff*(1) may cause a harmless "broken pipe" diagnostic if the last page of the document is not specified in *list*.

SEE ALSO

daps(1), env(1), eqn(1), mm(1), nroff(1), pic(1), tbl(1), tc(1), troff(1), profile(4), environ(5), mm(5), mv(5).

DIAGNOSTICS

"m[mv]t: no input file" if none of the arguments is a readable file and the command is not used as a filter.

NAME

mosd − the OSDD adapter macro package for formatting documents

SYNOPSIS

osdd [options] [files]

mm −mosd [options] [files]

nroff −mm −mosd [options] [files]

nroff −cm −mosd [options] [files]

mmt −mosd [options] [files]

troff −mm −mosd [options] [files]

DESCRIPTION

The OSDD adapter macro package is a tool used in conjunction with the MM macro package to prepare Operations Systems Deliverable Documentation. Many of the OSDD Standards are different from the default format provided by MM. The OSDD adapter package sets the appropriate MM options for automatic production of the OSDD Standards. The OSDD adapter package also generates the correct OSDD page headers and footers, heading styles, Table of Contents format, etc.

OSDD document (input) files are prepared with the MM macros. Additional information which must be given at the beginning of the document file is specified by the following string definitions:

```
.ds H1 document-number
.ds H2 section-number
.ds H3 issue-number
.ds H4 date
.ds H5 rating
```

The *document-number* should be of the standard 10-character format. The words "Section" and "Issue" should not be included in the string definitions; they will be supplied automatically when the document is printed. For example:

```
.ds H1 OPA−1P135−01
.ds H2 4
.ds H3 2
```

automatically produces

```
OPA-1P135-01
Section 4
Issue 2
```

as the document page header. Quotation marks are not used in string definitions.

If certain information is not to be included in a page header, then the string is defined as null; e.g.,

```
.ds H2
```

means that there is no *section-number*.

The OSDD Standards require that the *Table of Contents* be numbered beginning with *Page 1*. By default, the first page of text will be numbered *Page 2*. If the *Table of Contents* has more than one page, for example *n*, then either **−rP***n+1* must be included as a command line option or **.nr P n** must be included in the document file. For example, if the *Table of Contents* is four pages then use **−rP5** on the command line or **.nr P 4** in the document file.

The OSDD Standards require that certain information such as the document *rating* appear on the *Document Index* or on the *Table of Contents* page if there is no index. By default, it is assumed that an index has been prepared separately. If there is no index, the following must be included in the document file:

```
        .nr Di 0
```

This will ensure that the necessary information is included on the *Table of Contents* page.

The OSDD Standards require that all numbered figures be placed at the end of the document. The **.Fg** macro is used to produce full page figures. This macro produces a blank page with the appropriate header, footer, and figure caption. Insertion of the actual figure on the page is a manual operation. The macro usage is

```
        .Fg page-count "figure caption"
```

where *page-count* is the number of pages required for a multi-page figure (default 1 page).

The **.Fg** macro cannot be used within the document unless the final **.Fg** in a series of figures is followed by a **.SK** macro to force out the last figure page.

The *Table of Contents* for OSDD documents (see Figure 4 in Section 4.1 of the OSDD Standards) is produced with:

```
        .Tc
        System Type
        System Name
        Document Type
        .Td
```

The **.Tc/.Td** macros are used instead of the **.TC** macro from MM.

The **.PM** macro may be used to generate proprietary markings — see the MM document for legal styles.

The **.P** macro is used for paragraphs. The **Np** register is set automatically to indicate the paragraph numbering style. It is very important that the **.P** macro be used correctly. All paragraphs (including those immediately following a **.H** macro) must use a **.P** macro. Unless there is a **.P** macro, there will not be a number generated for the paragraph. Similarly, the **.P** macro should not be used for text which is not a paragraph. The **.SP** macro may be appropriate for these cases, e.g., for "paragraphs" within a list item.

The page header format is produced automatically in accordance with the OSDD Standards. The OSDD Adapter macro package uses the **.TP** macro for this purpose. Therefore the **.TP** macro normally available in MM is not available for users.

FILES

/usr/lib/tmac/tmac.osd

SEE ALSO

mm(1), mmt(1), nroff(1), troff(1), mm(5).

NAME

mptx – the macro package for formatting a permuted index

SYNOPSIS

nroff –mptx [options] [files]

troff –mptx [options] [files]

DESCRIPTION

This package provides a definition for the **.xx** macro used for formatting a permuted index as produced by *ptx*(1). This package does not provide any other formatting capabilities such as headers and footers. If these or other capabilities are required, the *mptx* macro package may be used in conjuction with the *MM* macro package. In this case, the **–mptx** option must be invoked *after* the **–mm** call. For example:

```
nroff –cm –mptx file
```

or

```
mm –mptx file
```

FILES

/usr/lib/tmac/tmac.ptx	pointer to the non-compacted version of the package
/usr/lib/macros/ptx	non-compacted version of the package

SEE ALSO

mm(1), nroff(1), ptx(1), troff(1), mm(5).

NAME

mv – a troff macro package for typesetting viewgraphs and slides

SYNOPSIS

mvt [**−a**] [options] [files]

troff [**−a**] [**−rX1**] **−mv** [options] [files]

DESCRIPTION

This package makes it easy to typeset viewgraphs and projection slides in a variety of sizes. A few macros (briefly described below) accomplish most of the formatting tasks needed in making transparencies. All of the facilities of *troff*(1), *eqn*(1), and *tbl*(1) are available for more difficult tasks.

The output can be previewed on most terminals, and, in particular, on the TEKTRONIX 4014. For this device, specify the **−rX1** option (this option is automatically specified by the *mvt* command—q.v.—when that command is invoked with the **−T4014** option). To preview output on other terminals, specify the **−a** option.

The available macros are:

.VS [*n*] [*i*] [*d*]	Foil-start macro; foil size is to be 7″×7″; *n* is the foil number, *i* is the foil identification, *d* is the date; the foil-start macro resets all parameters (indent, point size, etc.) to initial default values, except for the values of *i* and *d* arguments inherited from a previous foil-start macro; it also invokes the .A macro (see below). The naming convention for this and the following eight macros is that the first character of the name (V or S) distinguishes between viewgraphs and slides, respectively, while the second character indicates whether the foil is square (S), small wide (**w**), small high (**h**), big wide (W), or big high (H). Slides are "skinnier" than the corresponding viewgraphs: the ratio of the longer dimension to the shorter one is larger for slides than for viewgraphs. As a result, slide foils can be used for viewgraphs, but not vice versa; on the other hand, viewgraphs can accommodate a bit more text.
.Vw [*n*] [*i*] [*d*]	Same as .VS, except that foil size is 7″ wide × 5″ high.
.Vh [*n*] [*i*] [*d*]	Same as .VS, except that foil size is 5″×7″.
.VW [*n*] [*i*] [*d*]	Same as .VS, except that foil size is 7″×5.4″.
.VH [*n*] [*i*] [*d*]	Same as .VS, except that foil size is 7″×9″.
.Sw [*n*] [*i*] [*d*]	Same as .VS, except that foil size is 7″×5″.
.Sh [*n*] [*i*] [*d*]	Same as .VS, except that foil size is 5″×7″.
.SW [*n*] [*i*] [*d*]	Same as .VS, except that foil size is 7″×5.4″.
.SH [*n*] [*i*] [*d*]	Same as .VS, except that foil size is 7″×9″.
.A [*x*]	Place text that follows at the first indentation level (left margin); the presence of *x* suppresses the ½ line spacing from the preceding text.
.B [*m* [*s*]]	Place text that follows at the second indentation level; text is preceded by a mark; *m* is the mark (default is a large bullet); *s* is the increment or decrement to the point size of the mark with respect to the *prevailing* point size (default is 0); if *s* is 100, it causes the point size of the mark to be the same as that of the *default* mark.
.C [*m* [*s*]]	Same as **.B**, but for the third indentation level; default mark is a dash.

.D	[*m* [*s*]]	Same as **.B**, but for the fourth indentation level; default mark is a small bullet.
.T	*string*	*String* is printed as an over-size, centered title.
.I	[*in*] [*a* [*x*]]	Change the current text indent (does not affect titles); *in* is the indent (in inches unless dimensioned, default is 0); if *in* is signed, it is an increment or decrement; the presence of *a* invokes the **.A** macro (see below) and passes *x* (if any) to it.
.S	[*p*] [*l*]	Set the point size and line length; *p* is the point size (default is "previous"); if *p* is 100, the point size reverts to the *initial* default for the current foil-start macro; if *p* is signed, it is an increment or decrement (default is 18 for **.VS**, **.VH**, and **.SH**, and 14 for the other foil-start macros); *l* is the line length (in inches unless dimensioned; default is 4.2" for **.Vh**, 3.8" for **.Sh**, 5" for **.SH**, and 6" for the other foil-start macros).
.DF	*n f* [*n f* ...]	Define font positions; may not appear within a foil's input text (i.e., it may only appear after all the input text for a foil, but before the next foil-start macro); *n* is the position of font *f*; up to four "*n f*" pairs may be specified; the first font named becomes the *prevailing* font; the initial setting is (**H** is a synonym for **G**): .DF 1 H 2 I 3 B 4 S
.DV	[*a*] [*b*] [*c*] [*d*]	Alter the vertical spacing between indentation levels; *a* is the spacing for **.A**, *b* is for **.B**, *c* is for **.C**, and *d* is for **.D**; all non-null arguments must be dimensioned; null arguments leave the corresponding spacing unaffected; initial setting is: .DV .5v .5v .5v 0v
.U	*str1* [*str2*]	Underline *str1* and concatenate *str2* (if any) to it.

The last four macros in the above list do not cause a break; the **.I** macro causes a break only if it is invoked with more than one argument; all the other macros cause a break.

The macro package also recognizes the following upper-case synonyms for the corresponding lower-case *troff* requests:

.AD .BR .CE .FI .HY .NA .NF .NH .NX .SO .SP .TA .TI

The **Tm** string produces the trademark symbol.

The input tilde (~) character is translated into a blank on output.

See the user's manual cited below for further details.

FILES

/usr/lib/tmac/tmac.v
/usr/lib/macros/vmca

SEE ALSO

eqn(1), mmt(1), tbl(1), troff(1).

BUGS

The **.VW** and **.SW** foils are meant to be 9" wide by 7" high, but because the typesetter paper is generally only 8" wide, they are printed 7" wide by 5.4" high and have to be enlarged by a factor of 9/7 before use as viewgraphs; this makes them less than totally useful.

NAME

non-btl — reinstall MM macros without Bell Laboratories specific features

SYNOPSIS

sh non-btl.sh

DESCRIPTION

The *non-btl.sh* command will modify and re-install the source for the Memorandum Macros (used with *nroff* and *troff*) when Bell Labs specific macros are not desired.

Specifically, use of the *non-btl.sh* command will remove the **.TM**, **.PM**, **.CS** macros, and the **}2** string (which normally contains the name "Bell Laboratories") from the macro package. After running *non-btl.sh*, use of these features will have no effect.

This command does not remove the source for these features from the macro file, but does erase their definition. Those users who wish to tailor the macro package to their own environment may choose not to run *non-btl.sh*, but to modify the definition of the affected macros and string to their own specifications. Remember to re-install the macros after they are modified.

IMPORTANT

The *non-btl.sh* command is found in the **/usr/src/cmd/text/macros.d** directory, and may only be run by the super-user.

NAME

nroff, otroff − format or typeset text

SYNOPSIS

nroff [options] [files]

otroff [options] [files]

DESCRIPTION

Nroff formats text contained in *files* (standard input by default) for printing on typewriter-like devices and line printers; similarly, *otroff* formats text for a Wang Laboratories, Inc., C/A/T phototypesetter. Their capabilities are described in the *DOCUMENTERS WORKBENCH Software Text Formatters Reference* cited below.

An argument consisting of a minus (−) is taken to be a file name corresponding to the standard input. The *options*, which may appear in any order, but must appear before the *files*, are:

−o*list*	Print only pages whose page numbers appear in the *list* of numbers and ranges, separated by commas. A range *N*−*M* means pages *N* through *M*; an initial −*N* means from the beginning to page *N*; and a final *N*− means from *N* to the end. (See *BUGS* below.)
−n*N*	Number first generated page *N*.
−s*N*	Stop every *N* pages. *Nroff* will halt *after* every *N* pages (default *N*=1) to allow paper loading or changing, and will resume upon receipt of a line-feed or new-line (new-lines do not work in pipelines, e.g., with *mm*(1)). This option does not work if the output of *nroff* is piped through *col*(1). *Otroff* will stop the phototypesetter every *N* pages, produce a trailer to allow changing cassettes, and resume when the typesetter's start button is pressed. When *nroff* (*otroff*) halts between pages, an ASCII **BEL** (in *otroff*, the message **page stop**) is sent to the terminal.
−r*aN*	Set register *a* (which must have a one-character name) to *N*.
−i	Read standard input after *files* are exhausted.
−q	Invoke the simultaneous input-output mode of the **.rd** request.
−z	Print only messages generated by **.tm** (terminal message) requests.
−m*name*	Prepend to the input *files* the non-compacted (ASCII text) macro file **/usr/lib/tmac/tmac.***name*.
−c*name*	Prepend to the input *files* the compacted macro files **/usr/lib/macros/cmp.[nt].[dt].***name* and **/usr/lib/macros/ucmp.[nt].***name*.
−k*name*	Compact the macros used in this invocation of *nroff/otroff*, placing the output in files **[dt].***name* in the current directory (see the *DOCUMENTERS WORKBENCH Software Text Formatters Reference* for details of compacting macro files).

Nroff only:

−T*name*	Prepare output for specified terminal. Known *names* are **37** for the (default) TELETYPE® Model 37 terminal, **tn300** for the GE TermiNet 300 (or any terminal without half-line capability), **300s** for the DASI 300s, **300** for the DASI 300, **450** for the DASI 450, **lp** for a (generic) ASCII line printer, **382** for the DTC-382, **4000A** for the Trendata 4000A, **832** for the Anderson Jacobson 832, **X** for a (generic) EBCDIC printer, and **2631** for the Hewlett Packard 2631 line printer.
−e	Produce equally-spaced words in adjusted lines, using the full resolution of the particular terminal.
−h	Use output tabs during horizontal spacing to speed output and reduce output character count. Tab settings are assumed to be

every 8 nominal character widths.

−u*n* Set the emboldening factor (number of character overstrikes) for the third font position (bold) to *n*, or to zero if *n* is missing.

Otroff only:

−t Direct output to the standard output instead of the phototypesetter.

−f Refrain from feeding out paper and stopping phototypesetter at the end of the run.

−w Wait until phototypesetter is available, if it is currently busy.

−b Report whether the phototypesetter is busy or available. No text processing is done.

−a Send a printable ASCII approximation of the results to the standard output.

−p*N* Print all characters in point size *N* while retaining all prescribed spacings and motions, to reduce phototypesetter elapsed time.

−Tcat Use font-width tables for Wang CAT phototypesetter. This device is both the default and the only choice.

FILES

/usr/lib/suftab	suffix hyphenation tables
/tmp/ta$#	temporary file
/usr/lib/tmac/tmac.*	standard macro files and pointers
/usr/lib/macros/*	standard macro files
/usr/lib/term/*	terminal driving tables for *nroff*
/usr/lib/font/*	font width tables for *otroff*

SEE ALSO

eqn(1), ocw(1), tbl(1), mm(5).

480.sp0u

nroff only-

col(1), greek(1), mm(1).

otroff only-

mmt(1), mv(5).

BUGS

Nroff/otroff believes in Eastern Standard Time; as a result, depending on the time of the year and on your local time zone, the date that *nroff/otroff* generates may be off by one day from your idea of what the date is.

When *nroff/otroff* is used with the **−o***list* option inside a pipeline (e.g., with one or more of *ocw*(1), *eqn*(1), and *tbl*(1)), it may cause a harmless "broken pipe" diagnostic if the last page of the document is not specified in *list*.

NAME

ocw, checkcw — prepare constant-width text for otroff

SYNOPSIS

ocw [**-l**xx] [**-r**xx] [**-f**n] [**-t**] [**+t**] [**-d**] [files]

checkcw [**-l**xx] [**-r**xx] files

DESCRIPTION

Ocw is a preprocessor for *otroff* (see *nroff*(1)) input files that contain text to be typeset in the constant-width (CW) font on the Wang CAT phototypesetter. This preprocessor is not necessary for users of the new device-independent *troff*(1), nor is it compatible with it. Refer to the Addendum to the NROFF/TROFF User Manual for details on how to eliminate the use of this command.

Text typeset with the CW font resembles the output of terminals and of line printers. This font is used to typeset examples of programs and of computer output in user manuals, programming texts, etc. (An earlier version of this font was used in typesetting *The C Programming Language* by B. W. Kernighan and D. M. Ritchie.) It has been designed to be quite distinctive (but not overly obtrusive) when used together with the Times Roman font.

Because the CW font on the Wang CAT contains a "non-standard" set of characters and because text typeset with it requires different character and inter-word spacing than is used for "standard" fonts, documents that use the CW font must be preprocessed by *ocw*.

The CW font contains the 94 printing ASCII characters:

```
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789
!$%&()`'*+@.,/:;=?[]¦-_^~"<>{}#\
```

plus eight non-ASCII characters represented by four-character *otroff* names (in some cases attaching these names to "non-standard" graphics):

Character	*Symbol*	*Troff Name*
"Cents" sign	¢	`\(ct`
EBCDIC "not" sign	¬	`\(no`
Left arrow	←	`\(<-`
Right arrow	→	`\(->`
Down arrow	↓	`\(da`
Vertical single quote	'	`\(fm`
Control-shift indicator	⌒	`\(dg`
Visible space indicator	⊓	`\(sq`
Hyphen	-	`\(hy`

The hyphen is a synonym for the unadorned minus sign (-). Certain versions of *ocw* recognize two additional names: `\(ua` for an up arrow and `\(lh` for a diagonal left-up (home) arrow.

Ocw recognizes five request lines, as well as user-defined delimiters. The request lines look like *otroff* macro requests, and are copied in their entirety by *ocw* onto its output; thus, they can be defined *by the user* as *otroff* macros; in fact, the **.CW** and **.CN** macros *should* be so defined (see *HINTS* below). The five requests are:

.CW Start of text to be set in the CW font; **.CW** causes a break; it can take precisely the same options, in precisely the same format, as are available on the *ocw* command line.

.CN End of text to be set in the CW font; **.CN** causes a break; it can take the same options as are available on the *ocw* command line.

.CD Change delimiters and/or settings of other options; takes the same options as are available on the *ocw* command line.

.CP *arg1 arg2 arg3 ... argn*
All the arguments (which are delimited like *otroff* macro arguments) are concatenated, with the odd-numbered arguments set in the CW font and the even-numbered ones in the prevailing font.

.PC *arg1 arg2 arg3 ... argn*
Same as **.CP**, except that the even-numbered arguments are set in the CW font and the odd-numbered ones in the prevailing font.

The **.CW** and **.CN** requests are meant to bracket text (e.g., a program fragment) that is to be typeset in the CW font "as is." Normally, *ocw* operates in the *transparent* mode. In that mode, except for the **.CD** request and the nine special four-character names listed in the table above, every character between **.CW** and **.CN** request lines stands for itself. In particular, *ocw* arranges for periods (.) and apostrophes (') at the beginning of lines, and backslashes (\) everywhere to be "hidden" from *otroff*. The transparent mode can be turned off (see below), in which case normal *otroff* rules apply; in particular, lines that begin with . and ' are passed through untouched (except if they contain delimiters—see below). In either case, *ocw* hides the effect of the font changes generated by the **.CW** and **.CN** requests; *ocw* also defeats all ligatures (**fi**, **ff**, etc.) in the CW font.

The only purpose of the **.CD** request is to allow the changing of various options other than just at the beginning of a document.

The user can also define *delimiters*. The left and right delimiters perform the same function as the **.CW**/**.CN** requests; they are meant, however, to enclose CW "words" or "phrases" in running text (see example under *BUGS* below). *Ocw* treats text between delimiters in the same manner as text enclosed by **.CW**/**.CN** pairs, except that, for aesthetic reasons, spaces and backspaces inside **.CW**/**.CN** pairs have the same width as other CW characters. While spaces and backspaces between delimiters are half as wide, so they have the same width as spaces in the prevailing text (but are *not* adjustable). Font changes due to delimiters are *not* hidden.

Delimiters have no special meaning inside **.CW**/**.CN** pairs.

The options are:

-l*xx* The one- or two-character string *xx* becomes the left delimiter; if *xx* is omitted, the left delimiter becomes undefined, which it is initially.

-r*xx* Same for the right delimiter. The left and right delimiters may (but need not) be different.

-f*n* The CW font is mounted in font position *n*; acceptable values for *n* are 1, 2, and 3 (default is 3, replacing the bold font). This option is only useful at the beginning of a document.

-t Turn transparent mode *off*.

+t Turn transparent mode *on* (this is the initial default).

-d Print current option settings on file descriptor 2 in the form of *otroff* comment lines. This option is meant for debugging.

Ocw reads the standard input when no *files* are specified (or when **-** is specified as the last argument), so it can be used as a filter. Typical usage is: ocw *files* | otroff ... *Checkcw* checks that left and right delimiters, as well as the **.CW**/**.CN** pairs, are properly balanced. It prints out all offending lines.

HINTS

Typical definitions of the **.CW** and **.CN** macros meant to be used with the *mm*(5) macro package:

```
.de CW
.DS I
.ps 9
.vs 10.5p
.ta 16m/3u 32m/3u 48m/3u 64m/3u 80m/3u 96m/3u ...
..
.de CN
.ta 0.5i 1i 1.5i 2i 2.5i 3i 3.5i 4i 4.5i 5i 5.5i 6i
.vs
.ps
.DE
..
```

At the very least, the **.CW** macro should invoke the *otroff* no-fill (**.nf**) mode.

When set in running text, the CW font is meant to be set in the same point size as the rest of the text. In displayed matter, on the other hand, it can often be profitably set one point *smaller* than the prevailing point size (the displayed definitions of **.CW** and **.CN** above are one point smaller than the running text on this page). The CW font is sized so that, when it is set in 9-point, there are 12 characters per inch.

Documents that contain CW text may also contain tables and/or equations. If this is the case, the order of preprocessing should be: *ocw*, *tbl*, and *eqn*. Usually, the tables contained in such documents will not contain any CW text, although it is entirely possible to have *elements* of the table set in the CW font; of course, care must be taken that *tbl*(1) format information not be modified by *ocw*. Attempts to set equations in the CW font are not likely to be either pleasing or successful.

In the CW font, overstriking is most easily accomplished with backspaces: letting ← represent a backspace, `d←←\(dg` yields $\widehat{\texttt{d}}$. (Because backspaces are half as wide between delimiters as inside **.CW**/**.CN** pairs—see above—two backspaces are required for each overstrike between delimiters.)

FILES

/usr/lib/font/ftCW CW font-width table

SEE ALSO

eqn(1), nroff(1), tbl(1), mm(5), mv(5).

WARNINGS

If text preprocessed by *ocw* is to make any sense, it must be set on a typesetter equipped with the CW font or on a STARE facility; on the latter, the CW font appears as bold, but with the proper CW spacing.

BUGS

Only a masochist would use periods (.), backslashes (\), or double quotes (") as delimiters, or as arguments to **.CP** and **.PC**.
Certain CW characters do not concatenate gracefully with certain Times Roman characters, e.g., a CW ampersand (**&**) followed by a Times Roman comma(,). In such cases, judicious use of *otroff* half- and quarter-spaces (\| and \^) is most salutary, e.g., one should use _**&**_\^, (rather than just plain _**&**_,) to obtain **&**, (assuming that _ is used for both delimiters).
Using *ocw* with *nroff* is silly.

NAME

pic — troff preprocessor for drawing simple pictures

SYNOPSIS

pic [**−T***t*] [files]

DESCRIPTION

Pic is a *troff*(1) preprocessor for drawing simple figures on a typesetter. The basic objects are *box*, *line*, *arrow*, *circle*, *ellipse*, *arc* and text.

The optional argument **−T***t* specifies device *t*; currently supported devices are **aps** (Autologic APS-5), **X97** (Xerox 9700), and **i10** (Imagen Imprint-10). Default is **−Taps**.

SEE ALSO

troff(1).

PIC — A Graphics Language for Typesetting.

NAME

sroff − format text

SYNOPSIS

sroff [options] [files]

DESCRIPTION

Sroff formats text contained in *files* (standard input by default) for printing on typewriter-like devices and line printers, including the XEROX 9700 printer.

An argument consisting of a minus (−) is taken to be a file name corresponding to the standard input. The **options**, which may appear in any order, but must appear before the *files*, are:

−**o***list*	Print only pages whose page numbers appear in the *list* of numbers and ranges, separated by commas. A range *N*−*M* means pages *N* through *M*; an initial −*N* means from the beginning to page *N*; and a final *N*− means from *N* to the end.
−**s***N*	Stop every *N* pages. *Sroff* will halt *after* every *N* pages (default *N*=1) to allow paper loading or changing, and will resume upon receipt of a line-feed or new-line.
−**m***name*	Prepend to the input *files* the macro file **/usr/lib/smac/m***name*. (None available so far. Development of an MM-like macro package for *sroff* is in progress.)
−**x***file*	Write any index information onto *file*.

SEE ALSO

col(1), pg(1).

BUGS

%# is the name of a register that contains the number of lines used on a page in single-column mode, or the number of lines in a diversion. %# should work in multi-column mode, but what should it count?

NAME

tbl − format tables for nroff or troff

SYNOPSIS

tbl [−**TX**] [files]

DESCRIPTION

Tbl is a preprocessor that formats tables for *nroff* or *troff*(1). The input files are copied to the standard output, except for lines between .TS and .TE command lines, which are assumed to describe tables and are re-formatted by *tbl*. (The .TS and .TE command lines are not altered by *tbl*).

.TS is followed by global options. The available global options are:

center	center the table (default is left-adjust);
expand	make the table as wide as the current line length;
box	enclose the table in a box;
doublebox	enclose the table in a double box;
allbox	enclose each item of the table in a box;
tab (*x*)	use the character *x* instead of a tab to separate items in a line of input data.

The global options, if any, are terminated with a semi-colon (;).

Next come lines describing the format of each line of the table. Each such format line describes one line of the actual table, except that the last format line (which must end with a period) describes *all* remaining lines of the table. Each column of each line of the table is described by a single key-letter, optionally followed by specifiers that determine the font and point size of the corresponding item, that indicate where vertical bars are to appear between columns, that determine column width, inter-column spacing, etc. The available key-letters are:

c	center item within the column;
r	right-adjust item within the column;
l	left-adjust item within the column;
n	numerically adjust item in the column: units positions of numbers are aligned vertically;
s	span previous item on the left into this column;
a	center longest line in this column and then left-adjust all other lines in this column with respect to that centered line;
^	span down previous entry in this column;
_	replace this entry with a horizontal line;
=	replace this entry with a double horizontal line.

The characters **B** and **I** stand for the bold and italic fonts, respectively; the character | indicates a vertical line between columns.

The format lines are followed by lines containing the actual data for the table, followed finally by .TE. Within such data lines, data items are normally separated by tab characters.

If a data line consists of only _ or =, a single or double line, respectively, is drawn across the table at that point; if a *single item* in a data line consists of only _ or =, then that item is replaced by a single or double line.

Full details of all these and other features of *tbl* are given in the reference manual cited below.

The −**TX** option forces *tbl* to use only full vertical line motions, making the output more suitable for devices that cannot generate partial vertical line motions (e.g., line printers).

If no file names are given as arguments (or if – is specified as the last argument), *tbl* reads the standard input, so it may be used as a filter. When it is used with *eqn*(1) or *neqn*, *tbl* should come first to minimize the volume of data passed through pipes.

EXAMPLE

If we let → represent a tab (which should be typed as a genuine tab), then the input:

```
.TS
center box ;
cB s s
cI | cI s
^ | c c
l | n n .
Household Population
_
Town→Households
→Number→Size
=
Bedminster→789→3.26
Bernards Twp.→3087→3.74
Bernardsville→2018→3.30
Bound Brook→3425→3.04
Bridgewater→7897→3.81
Far Hills→240→3.19
.TE
```

yields:

Household Population		
Town	*Households*	
	Number	Size
Bedminster	789	3.26
Bernards Twp.	3087	3.74
Bernardsville	2018	3.30
Bound Brook	3425	3.04
Bridgewater	7897	3.81
Far Hills	240	3.19

SEE ALSO

ocw(1), eqn(1), mm(1), mmt(1), nroff(1), troff(1), mm(5), mv(5).

BUGS

See *BUGS* under *nroff*(1).

NAME

tc, otc − troff output interpreter

SYNOPSIS

tc [−t] [−olist] [−an] [−e] [file]

otc [−t] [−sn] [−pl] [file]

DESCRIPTION

Tc interprets its input (standard input default) as output from *troff*(1). The standard output of *tc* is intended for a TEKTRONIX 4015 (a 4014 terminal with ASCII and APL character sets). The various typesetter sizes are mapped into the 4014's four sizes; the entire TROFF character set is drawn using the 4014's character generator, using overstruck combinations where necessary, producing an altogether displeasing effect. *Otc* performs a similar function for the old TROFF, *otroff* (see *nroff*(1)). Typical usage:

```
troff file | tc
otroff -t file | otc
```

At the end of each page *tc* waits for a new-line (empty line) from the keyboard before continuing on to the next page. In this wait state, the following commands are recognized:

!*cmd*	Send *cmd* to the shell.
e	Invert state of the screen erase (*tc*); do not erase screen before next page (*otc*).
−*n*	Skip backward *n* pages. (*tc* only).
n	Print page *n*. (*tc* only).
s*n*	Skip forward *n* pages. (*otc* only).
a*n*	Set the aspect ratio to *n*. (*tc* only).
?	Print list of available options. (*tc* only).

The command line options are:

−t	Do not wait between pages (for directing output into a file).
−o*list*	Prints only the pages enumerated in *list*. The list consists of pages and page ranges (e.g., 5-17) separated by commas. The range *n*− goes from *n* to the end; the range −*n* goes from the beginning to and including page *n*. (*tc* only).
−a*n*	Set the aspect ratio to *n*; default is 1.5. (*tc* only).
−e	Do not erase before each page. (*tc* only).
−s*n*	Skip the first *n* pages. (*otc* only).
−p*l*	Set page length to *l*; *l* may include the scale factors **p** (points), **i** (inches), **c** (centimeters), and **P** (picas); Default is picas. (*otc* only).

SEE ALSO

4014(1), nroff(1), tplot(1G), troff(1).

BUGS

Font distinctions are lost.
It needs a **−w** option to wait for input to arrive.

NAME

troff − text formatting and typesetting

SYNOPSIS

troff [option] ... [file] ...

DESCRIPTION

Troff formats text in the named *files* for printing on a phototypesetter. It is the new "device-independent" version of the old *otroff* (see *nroff*(1)). Its capabilities are described in the *DOCUMENTER'S WORKBENCH Software Text Formatters Reference* plus the Addendum.

If no *file* argument is present, the standard input is read. An argument consisting of a single minus (−) is taken to be a file name corresponding to the standard input. The options, which may appear in any order so long as they appear before the files, are:

−o*list*	Print only pages whose page numbers appear in the comma-separated *list* of numbers and ranges. A range *N*−*M* means pages *N* through *M*; an initial −*N* means from the beginning to page *N*; and a final *N*− means from *N* to the end. (See BUGS below.)
−n*N*	Number first generated page *N*.
−s*N*	Generate output to encourage typesetter to stop every *N* pages.
−m*name*	Prepend the macro file **/usr/lib/tmac/tmac.***name* to the input *files*.
−r*aN*	Set register *a* (one character name) to *N*.
−i	Read standard input after the input files are exhausted.
−q	Invoke the simultaneous input-output mode of the **.rd** request.
−z	Print only messages generated by **.tm** requests.
−a	Send a printable ASCII approximation of the results to the standard output.
−T*dest*	Prepare output for typesetter *dest*. Currently the only supported typesetter is the Autologic APS-5, (**−Taps**). Users of the Wang CAT should use *otroff* (see *nroff*(1)). Supported laser printers are the Imagen Imprint -10 (**−T***i10*) and the Xerox 9700 (see *dx9700*(1)).

FILES

/tmp/trtmp*	temporary file
/usr/lib/tmac/tmac.*	standard macro files
/usr/lib/macros/*	standard macro files
/usr/lib/font/dev*/*	font width tables

SEE ALSO

daps(1), dx9700(1), eqn(1), mmt(1), nroff(1), pic(1), tbl(1), tc(1).

DOCUMENTER'S WORKBENCH Software Text Formatters Reference.

BUGS

The **.tl** request may not be used before the first break-producing request in the input to *troff*.

Troff believes in Eastern Standard Time; as a result, depending on the time of the year and on your local time zone, the date that *troff* generates may be off by one day from your idea of what the date is.

When *troff* is used with the **−o***list* option inside a pipeline (e.g., with one or more of *pic*(1), *eqn*(1), and *tbl*(1)), it may cause a harmless "broken pipe" diagnostic if the last page of the document is not specified in *list*.

NAME

troff — description of output language

DESCRIPTION

The device-independent *troff* outputs a pure ASCII description of a typeset document. The description specifies the typesetting device, the fonts, and the point sizes of characters to be used as well as the position of each character on the page. A list of all the legal commands follows. Most numbers are denoted as *n* and are ASCII strings. Strings inside of **[ ]** are optional. *Troff* may produce them, but they are not required for the specification of the language. The character **\n** has the standard meaning of "newline" character. Between commands white space has no meaning. White space characters are spaces and newlines. All commands which have an arbitary length numerical parameter or word must be followed by white space. For example, the command to specify point size, **s###**, must be followed by a space or newline.

s*n*	The point size of the characters to be generated.
f*n*	The font mounted in the specified position is to be used. The number ranges from 0 to the highest font presently mounted. 0 is a special position, invoked by *troff*, but not directly accessible to the troff user. Normally fonts are mounted starting at position 1.
c*x*	Generate the character *x* at the current location on the page; *x* is a single ASCII character.
C*xyz*	Generate the special character *xyz*. The name of the character is delimited by white space. The name will be one of the special characters legal for the typesetting device as specified by the device specification found in the file *DESC*. This file resides in a directory specific for the typesetting device. (See *font*(5) and **/usr/lib/font/dev*.**)
H*n*	Change the horizonal position on the page to the number specified. The number is in basic units of motions as specified by *DESC*. This is an absolute "goto".
h*n*	Add the number specified to the current horizontal position. This is a relative "goto".
V*n*	Change the vertical position on the page to the number specified (down is positive).
v*n*	Add the number specified to the current vertical position.
nnx	This is a two-digit number followed by an ASCII character. The meaning is a combination of **h***n* followed by **c***x*. The two digits *nn* are added to the current horizontal position and then the ASCII character, *x*, is produced. This is the most common form of character specification.
n*b a*	This command indicates that the end of a line has been reached. No action is required, though by convention the horizontal position is set to 0. *Troff* will specify a resetting of the *x,y* coordinates on the page before requesting that more characters be printed. The first number, *b*, is the amount of space before the line and the second number, *a*, the amount of space after the

	line. The second number is delimited by white space.
w	A **w** appears between words of the input document. No action is required. It is included so that one device can be emulated more easily on another device.
p*n*	Begin a new page. The new page number is included in this command. The vertical position on the page should be set to 0.
{	Push the current environment, which means saving the current point size, font, and location on the page.
}	Pop a saved environment.
t*xxxxx*	Print the string of characters, *xxxxx*, using the natural width of each character to determine the next x coordinate. *Troff* does not currently produce this form of command. It is not recommended. The characters will probably be too close together.
# .... **\n**	A line beginning with a pound sign is a comment.
Dl *x y***\n**	Draw a line from the current location to *x,y*. At the end of the drawing operation the current location will be *x,y*.
Dc *d***\n**	Draw a circle of diameter *d* with the leftmost edge being at the current location (x, y). The current location after drawing the circle will be x+*d*,y, the rightmost edge of the circle.
De *dx dy***\n**	Draw an ellipse with the specified axes; *dx* is the axis in the x direction and *dy* is the axis in the y direction. The leftmost edge of the ellipse will be at the current location. After drawing the ellipse the current location will be x+*dx*,y.
Da *x y r***\n**	Draw a counterclockwise arc from the current location to *x,y* using a circle of radius *r* . The current location after drawing the arc will be *x,y*.
D~ *x y x y*...**\n**	Draw a spline curve (wiggly line) between each of the *x,y* coordinate pairs starting at the current location. The final location will be the final *x,y* pair of the list. Currently there may be no more than 36 *x,y* pairs to this command.
x i[nit]\n	Initialize the typesetting device. The actions required are dependent on the device. An **init** command will always occur before any output generation is attempted.
x T *device***\n**	The name of the typesetter is *device*. This is the same as the argument to the **−T** option. The information about the typesetter will be found in the directory **/usr/lib/font/dev**{*device*}**.**
x r[es] *n h v***\n**	The resolution of the typesetting device in increments per inch is *n*. Motion in the horizontal direction can take place in units of *h* basic increments. Motion in the vertical direction can take place in units of *v* basic increments. For example, the APS-5 typesetter has a basic resolution of 723 increments per inch and can move in either direction in 723rds of an inch. Its specification is: **x res 723 1 1**

x p[ause]\n	Pause. Cause the current page to finish but do not relinquish the typesetter.
x s[top]\n	Stop. Cause the current page to finish and then relinquish the typesetter. Perform any shutdown and bookkeeping procedures required.
x t[railer]\n	Generate a trailer. On some devices no operation is performed.
x f[ont] *n name***\n**	Load the font *name* into position *n*.
x H[eight] *n***\n**	Set the character height to *n* points. This causes the letters to be elongated or shortened. It does not affect the width of a letter.
x S[lant] *n***\n**	Set the slant to *n* degrees. Only some typesetters can do this and not all angles are supported.

NAME

x9700 - prepare nroff documents for the Xerox 9700 printer

SYNOPSIS

x9700 [**-1**|**-2**] [[**-f**] file] [**-h** indent] [**-v** indent] [**-l** leng]
[-[**p**|**l**]**k** mask [n]] [[**-o** orient] [**-s** style] [**-T** c] [files]

DESCRIPTION

The *x9700* command reads the named *files* and writes standard output which is suitable to be sent to the Xerox 9700 printer. The special name - means standard input. Each file will begin on a new page. If no files are specified, then *x9700* reads from standard input. Options and their meanings:

−1	print output on one side of the page
−2	print output on both sides of the page
−f *file*	Take input from *file.* This option is necessary to process file names which begin with a hyphen.
−h *indent*	horizontal indent: offset output *indent* units to the right. A *c* appended to *indent* sets the unit of offset to centimeters; an *i*, sets the unit to inches; neither, sets the unit to character positions. The default indent is zero. Fractional character positions are ignored.
−v *indent*	Vertical indent: offset output *indent* lines from top of page. Default is zero.
−l *length*	Print *length* lines per page. Defaults for the fonts are given below. A *length* of zero obtains the default.
−lk *mask n*	
−pk *mask n*	
−k *mask n*	Overlay output with preprinted *mask*. The *lk* overlays the mask in landscape orientation; the *pk*, in portrait orientation. The *k* alone uses the current orientation. The default mask is *none*. A number following the mask name specifies the page on which to overlay the mask. If no number follows the mask name, then all pages not specifically named are overlaid with the mask. Available masks are installation-dependant.
−o *orient*	Page orientation, either **portrait** or **landscape**, with **port** and **land** respectively, acceptable abbreviations. Each font style has a default, given below. Specifying an empty orientation obtains the default.
−s *style*	Select font style. Current possibilities and default values:

style	**abbr**	**default**	**portrait**		**landscape**	
		orient	**length**	**width**	**length**	**width**
elite	elit	port	71	102	51	131
gothic	goth	port	66	85	51	110
goth24		port	33	42	25	55
mini		port	137	131	106	131
pica		port	66	85	51	110
times14		port	46	˜90	36	˜118
times28		port	23	˜45	18	˜59
vintage	vint	port	71	102	55	131
vint20		port	35	51	27	66
xerox	xrox	land	99	116	77	131
xerox18		land	44	58	34	75

Note that the lengths and widths are maximum values for a page and make no provision for margins. The ˜ indicates approximate widths for proportionally spaced fonts. The default style is *vintage*. Both the style names and their abbreviations are accepted. Not all styles have all fonts, and not all fonts have a full character set (including the full TX train). A summary of available combinations appears below. **Note: these fonts are under development and subject to change without notice.**

−T *c* If and only if *c* is **X**, then *x9700* expects input from *nroff* with the **-TX** option.

Options may be repeated and may appear in any order. The space between an option and its argument may be omitted. The options are cumulative and apply only to succeeding file names. Thus

```
x9700 -o port -h 10 file1 -o land file2
```

prints *file1* in portrait orientation and *file2* in landscape but indents both files by 10 characters.

ESCAPES

The command *X9700* recognizes four control characters (backspace, formfeed, horizontal tab, and carriage return) and the following set of escapes:

escape sequence	**meaning**	**from NROFF**
esc X *c*	hyperascii *c* ('*c*'|0200)	
esc esc *c*	hyperascii *c* ('*c*'|0200)	
esc B	bold font	\fB
esc R	Roman font	\fR
esc I	Italic font	\fI
esc L	logo font	
esc D	reverse half-line feed	\u
esc U	half-line feed	\d
esc \n	reverse line feed	
esc si	intensify shading	
esc so	lessen shading	

The half-line motions effect superscripts and subscripts, but the TX train contains only a limited number of these. There are three levels of shading available: dark (character e9), darker (e8), and darkest (c4).

input this column	to get
none	none
\33\17level 1 (dark)	level 1 (dark)
\33\17level 2 (darker)	level 2 (darker)
\33\17level 3 (darkest)	level 3 (darkest)
\33\16back to level 2	back to level 2
\33\16back to level 1	back to level 1
\33\16back to none	back to none

SEE ALSO

nroff(1).

EXCEPTIONS

Lines that exceed the page width are truncated. Page breaks occur not only at the logical end of page (controlled by the **-l** option), but also at the physical end of page (controlled by the machine). Lines which exceed the latter limit are usually forced to an extra, overflow page. The number of lines on a page includes the indent of the **-v** option.

It is difficult to get to all of the *TX* train.

FONT SUMMARY

style	bold-italic		graph	
	port	land	port	land
elite	y	y	n	n
Gothic	y	y	n	n
goth24	n	n	n	n
mini	n	n	n	n
pica	y	y	n	n
times14	n	n	n	n
times28	n	n	n	n
vintage	y	y	y	y
vint20	n	n	y	y
xerox	y	y	n	n
xerox18	n	n	n	n

DIAGNOSTICS

"*missing parameter to -option*"
"*can't open file*"
"*unsupported style/orientation combination*"
"*bad mask name*"
"*bad horizontal indent specification*"
"*bad page length specification*"
"*bad vertical indent specification*"

Check parameter list.

"*page length larger than max*"

X9700 has been directed to place more than 140 lines on a page.

"*attempt to back off page*"

An attempt to field a reverse line feed would cause a return to a previous page.

"*file too wide*"

X9700 has encountered a line with more than 132 characters on it. This usually happens when input *not* produced with *nroff -TX* is given to *x9700* with the *-TX* option.

"*unknown escape sequence*"

X9700 has been given an escape sequence which does not correspond to a reverse line feed, a font change, a shade change, or a hyperascii character. Escape sequences are introduced with an ascii *esc* character (octal 33). This usually happens when *-TX* is not supplied to *nroff.*

"*too many masks*"

X9700 allows a total of only ten separate mask specifications.

"*page too dense*"

X9700 has encountered a page with too much overprinting. The cause may be too much backspacing or too many font changes. It may be small comfort that even if the *x9700* program could format the page, the Xerox printer would probably fail to print it.

"*internal error*"
"*machine seized*"

Get help.

EXAMPLES

The following examples do not include the final pipeline to direct the output to the Xerox 9700 printer, because that is an installation-dependent procedure.

To obtain standard memo format:

```
nroff -rA3 -rE1 -rU1 -rL71 -TX -cm file |
x9700 -h10 -TX -k prin1
```

To obtain manual page:

```
nroff -TX -man file |
x9700 -l66 -v3 -h10 -TX
```

To obtain this manual page:

```
nroff -man -TX file |
x9700 -h12 -v2 -l66 -TX -k prin1 1 -k prin2 2 -k prin3 3 \
-lk prin1 4 -lk prin2 5 -k vgraf 6 -k sdisc 7
```

To obtain viewgraphs:

```
nroff -TX - file <<eof |
.pl 35
.ll 45
eof
x9700 -s vint20 -TX
```

Your comments and suggestions are appreciated and will help us to provide the best documentation for your use.

1. How would you rate this document for COMPLETENESS? (Please Circle)

   Excellent Adequate Poor
   4 ----------------------3 ----------------------2 ----------------------1 ----------------------0

2. Identify any information that you feel should be included or removed.

   ____

   ____

3. How would you rate this document for ACCURACY of information? (Please Circle)

   Excellent Adequate Poor
   4 ----------------------3 ----------------------2 ----------------------1 ----------------------0

4. Specify page and nature of any error(s) found in this document.

   ____

   ____

5. How would you rate this document for ORGANIZATION of information? (Please Circle)

   Excellent Adequate Poor
   4 ----------------------3 ----------------------2 ----------------------1 ----------------------0

6. Describe any format or packaging problems you have experienced with this document.

   ____

   ____

7. Do you have any general comments or suggestions regarding this document?

   ____

   ____

8. We would like to know a little about your background as a user of this document:

A. Your job function ____________________

B. Number of years experience with computer hardware: operation _______, maintenance _______.

C. Number of years experience with computer software: user _______, programmer _______.

Your Name ____________________ Phone No. ____________
Company ______________________________________
Address ______________________________________
City & State ____________________ Zip Code _______

Western Electric

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 1999 GREENSBORO, N.C.

POSTAGE WILL BE PAID BY ADDRESSEE

DOCUMENTATION SERVICES
2400 Reynolda Road
Winston-Salem, N.C. 27106-9989

Do Not Tear—Fold Here and Tape