The Software Toolworks
Walt Bilofsky, Prop.

14478 GLORIETTA DRIVE
SHERMAN OAKS, CALIFORNIA 91423

TELEPHONE
(213) 986-4885

TEXT (Version 3.3)
Text Formatter

Dr. James J. Gillogly
2520 S. Chard Ave.
Topanga CA 90290
(213) 455-1407

23 Dec 1980

1. Description

Text is a text formatting program for Heath's HDOS operating system and for CP/M, greatly extended from format [1], by Kernighan and Plauger. It will format documents (such as this software description) for printing on a typewriter or line printer. Text contains all of the basic functions that a text formatter should have, including filling, spacing, indenting, right margin justification, underlining, and pagination. In addition, it provides many sophisticated features, such as the ability to include files from any number of diskettes (resetting drives as necessary), a string replacement capability, and three-part titles with automatic page numbering.

Note: Throughout this document we will use the HDOS convention of referring to the second disk drive as SY1:. CP/M users should read B: for SY1: wherever the latter appears.

2. Usage

Preparing formatted documents using Text is a two step process. First, an input file is prepared. This file contains the text to be formatted, together with command lines which indicate to Text the formatting operations to be performed.

The input file may be prepared using any standard editor, such as the EDIT program supplied with HDOS, the ED program under CP/M, or the more convenient and efficient PIE (TM) Full Screen Editor from The Software Toolworks.

After an input file has been prepared, the Text program is used to read the input file and produce a formatted version of the document. The formatted document is usually written out on a printer, but may also be stored on another file or scrolled onto the terminal screen for proofreading.

To invoke Text from HDOS or CP/M in order to format an input file, give the command

```
          text infile
or
          text infile outfile
or, most generally,
          text [switches] [commands] infile [outfile]
or
          text -v
```

where infile is the raw Text input file, outfile is the desired location of the finished document, and the brackets surround optional material. Omitting outfile sends output to the terminal. The filename may be "LP:" (HDOS) or "LST:" (CP/M) to direct output to the line printer.

The -v switch prints Text's version number. Other switches are:

-w - wait between pages for the user to type a newline on the controlling terminal; beeps between pages

-t - controlling terminal is not an H89/H19: this will disable the H19 feature that puts underlining or other overstrikes in inverse video

-o3 - start output at page 3 (use any number here)

The switches can be used together, e.g.

```
        text -w -o7 doc.txt lp:
```

to output to the line printer starting at page 7, waiting between pages.

Any Text command may be included on the HDOS or CP/M command line. For example,

```
        text ".rm +20" syl:heath.ltr heath.lp
```

will process the file "heath.ltr" from syl: just as if the command ".rm +20" had been included as the first command in the file. (The .rm command is explained in Section 8 below.) A Text command on the HDOS or CP/M command line will not override a contradictory command in the input file, since the command in the input file is executed after the one in the command line.

If your printer or printer driver understand about page lengths, and provide automatic top of form after a certain number of lines, you must either convince them to stop doing this, or use a .pl command to tell Text to use the same page length as the printer. Text uses a default page length of 66 lines.

Under HDOS, the printer can be convinced to let Text do the paging by the HDOS command

```
     SET LP: PAGE 0
```

In order to provide a comprehensive example of the proper use of all the Text features, the Text distribution disk contains the source files for this documentation. To produce a copy of this manual on your terminal, mount a copy of the distribution disk on SY1: and execute the command

```
     sy1:text sy1:text.txt
```

(On a single drive HDOS system, ONECOPY the files from the distribution disk to a bootable disk and omit every "sy1:" from these examples.)

You can also produce a manual on your printer. (Please note, however, that making copies other than for your own use violates the copyright on this document.) To output the manual on the printer under HDOS, use the command

```
     sy1:text sy1:text.txt lp:
```

or, if your printer is a Selectric or similar device which performs underlining by backspacing a character at a time,

```
     sy1:text ".bs 1" sy1:text.txt lp:
```

3. Inputting Text

If no special formatting is desired, the input text file can simply be typed more or less as it should appear on the output. Text will normally fill the text (i.e. the line lengths are evened out) and right-justify it (i.e., provide straight margins).

Two-letter commands beginning with a period invoke special processing features of Text. For example, underlined words or phrases appear on a separate line preceded by the command ".ul". A list of all the legal commands appears in Section 15. Each command must be on a separate line and the period must be in the leftmost column.

Text honors indentations, so that a paragraph can be started with an arbitrary number of spaces. If the usual 5 spaces are desired, the .pp command can be used to separate paragraphs. Extra spaces typed within a line are retained, and a period at the end of an input line is followed by two spaces on output.

4. Text Filling and Justifying

Text fills each line with words until it runs out of room, ignoring the line endings of the input text. To prevent filling, use the command

```
        .nf
```

This is useful for including tables such as the command list of Section 15 exactly as they were input. The inside address of a letter is typed in no-fill mode. The command

```
        .fi
```

restores fill mode. Both .nf and .fi stop processing the current output line when they are encountered. That is, they cause a "break".

A break can be explicitly requested using the command

```
        .br
```

which terminates the current output line and begins a new one with the text from the next input line.

While filling, Text right-justifies the lines by adding extra spaces between words to make the right margin line up exactly. The spaces are added alternately from each side to prevent obtrusive rivers of spaces from appearing in the output text. To stop justifying, use the command

```
        .nj
```

and to resume justifying use the command

```
        .ju
```

5. Indenting

Text normally prints the entire document starting each line at the first available character position. This can be changed using the command

```
        .in value
```

The value can be an absolute number of columns to indent, or it can be a relative value. For example,

```
        .in 6
```

would indent subsequent text by 6 character positions (giving a 1-inch left margin for many printers).

```
        .in +5
```

would indent 5 characters deeper than the current indentation level, as for an indented paragraph.

Hanging indents This paragraph was created by giving the commands

```
          .in +17
          .hi "Hanging indents" -17
```

to achieve an indented and labeled block paragraph. The .hi command can be followed with a value to specify the indentation level for the label:

```
          .hi string value
```

If the value is omitted, the label will begin at the left margin.

To get an indentation only for the next input line, use the command

```
     .ti value
```

where the value can again be either relative or absolute.

The initial line of a paragraph can be indented by any number of spaces. To get the standard 5-letter indentation, use the command

```
     .pp
```

which is equivalent to

```
     .ti +5
```

6. Centering, Underlining and Overstriking

The command

```
     .ce integer
```

will center the specified number of input lines without filling them. If no number is specified, only the next input line is centered. To center a large number of lines without counting them, say

```
     .ce 1000          or some other large number
     line 1 to be centered
     more text to be centered
     ...
     .ce 0             to stop the centering
```

The text is centered between the current indentation level and the right margin (see Section 8).

The commands

```
.ul integer
.cu integer
```

cause the specified number of input lines to be underlined. ".ul" is used for underlining individual words, and ".cu" is used for continuous underlining, including interword spacing. If no number is specified only the next input line will be underlined. The same trick used to center many lines (above) can be used to underline a lot of text. Filling and justifying are not affected by underlined text. On an H19 terminal (e.g. in an H89) Text will use the inverse video function to highlight underlined text directed to the screen.

The underline character, "_", can be changed to any other character. ~~A popular choice is the hyphen, which can be used to cross out text.~~ To change it to a hyphen, give the command

```
.uc -
```

It can subsequently be changed back to a "_" to resume normal underlining.

Another way to highlight text is to overstrike it one or more times. The command

```
.bd integer
```

will embolden the specified number of input lines, just as the .ul and .cu commands underline input lines. Note that a single overstrike may not be enough to yield a striking difference in appearance. To overstrike several times, give the command

```
.bn integer
```

to tell Text to strike any further .bd text that many times.

Text underlines or emboldens text by outputting one line containing only the overstrike characters, then a carriage return without a line feed, and finally the text to be underlined. If the output device does not recognize a carriage return, the command

```
.bs 1
```

will cause subsequent underlining to type the character followed by a backspace character and an underline for every character. This is slower than the default method, which can be re-entered with a

```
.bs 0
```

To overstrike an individual character, the string "\b" can be used on input to represent one backspace. For example, the Welsh word "môr-leider" (pirate) is input as "mo\b^r-leider".

7. Page Control

The standard page length is 66 lines. To change this to a different length, use the command

```
     .pl value
```

The value can be either absolute or relative.

During the first draft of a document, the page breaks may appear in inconvenient places. To force a page break (e.g. to keep a paragraph together), use the command

```
     .bp
```

which will immediately eject the current page. If it is followed by an integer, e.g.

```
     .bp 7
```

the next page will be page 7. (Page numbers are discussed in Section 10.)

A conditional page break allows more sophisticated control of the page ejection. The command

```
     .ne integer                    ("need")
```

tells Text to eject the page if there are less than the specified number of lines remaining on the page. Thus, for example, the command

```
     .ne 4
```

might be used in no-fill mode to keep a four-line table together.

8. Setting the Line Length

The line length is controlled by setting the left and right margins. Indentation handles the left margin. E.g.

```
     .in 6
```

would specify a typical left margin for a line printer. Indentation is discussed in more detail in Section 5. The command

```
     .rm value
```

modifies the right margin. The value may be relative or absolute. Initially the right margin is set at 60.

9. Vertical Spacing

To get blank vertical space in the output file, either leave blank lines in the input (each becomes a blank output line) or use the command

```
     .sp integer
```

to get the specified number of blank lines. If no integer is specified, one blank line is output.

To get more than one newline between each output line, use the command

```
     .ls integer
```

For example,

```
     .ls 2
```

forces double spaced output.

10. Headers and Footers

Each page may have a 1-line header and/or a 1-line footer, either of which may include automatically counted page numbers (see also ".bp integer", Section 7). These titles have three parts: left-adjusted, centered, and right-adjusted. The commands

```
     .he "left string"centered string"right string"
     .fo "left string"centered string"right string"
```

are used to define the header or footer lines. Leading or trailing blanks may appear in any string, and will be counted in the adjusted or centering. The quoting character may be any non-blank character in place of the double quote shown in the examples. If any title string includes a # sign it is replaced with the page number of the current page. The margins used for adjusting the three-part titles are those in effect at the time the last title was defined.

The top and bottom margins are initially set at 5 blank lines. A one-line header and/or footer, if defined, becomes the third of the five lines. The space above and below the header and footer lines are initially set at 2 lines above the header, 3 lines from header to text (including the header), 3 lines from text to footer (including the footer), and 2 lines below the footer. These may be changed by the user:

.h1 value	lines above and including the header
.h2 value	lines below the header
.f1 value	lines above the footer
.f2 value	lines below and including the footer

The values may be either relative or absolute.

11. Including other files

To include the text from another file use the command

```
.rf filename                ("read file")
```

Text will interrupt processing of the original file, process all the text from the specified file, and then switch back to continue the original file. The new file may include another .rf command, to a nesting limit of about 5 files.

This feature is particularly useful for including boilerplate such as the copyright notice at the beginning of this document. It may also be used to include files from another drive, if the document is too big to fit on one diskette.

Unless the filename explicitly specifies the drive (e.g. SY2:BOILER.TXT under HDOS or B:BOILER.TXT under CP/M), the .rf file will be taken from the same drive as the original file.

If the filename is not found on any drive (HDOS) or the specified drive (CP/M), *Text* enters an interactive mode, where the user can request one of several options:

HDOS:

Option	Action
1	reset drive SY1:
2	reset drive SY2:
A	reset drive DK0:
B	reset drive DK1:
C	reset drive DK2:
D	reset drive DK3:
i	ignore the bad .rf file and continue
n	user specifies new filename
a	abort all processing

CP/M:

Any drive from a: through p: may be selected by using the single lower case letter identifying the drive. The diskette may be replaced before the drive is searched.

Option	Action
a-p	check the specified drive
I	ignore the bad .rf file and continue
N	user specifies new filename
A	abort all processing

This menu will return until Text finds the file or is told to give up.

A user with two or more drives can use this feature to process arbitrarily large documents by putting the initial file on one drive and requesting each desired filename in order. Subsequent files need not be on the same diskette; as unmounted files are requested with the .rf, Text will offer the opportunity to bring them in.

12. Terminal Messages

While processing text to a file or printer, it is sometimes useful to see confirmation on the screen that something is happening. The command

```
.tm Any text can appear here
```

will cause the text to be printed on the controlling terminal when the command is encountered.

13. String Substitution

Text provides for 26 user-defined strings, identified by the letters a-z. To define string x, use the command

```
.sb x string
```

To invoke the string, embed \sx in the text wherever the string is to occur. For example, at the beginning of this file is the definition

```
.sb t _\bT_\be_\bx_\bt
```

which defines the word Text. Each underlined occurrence of Text in the text is obtained by using the abbreviation \st.

14. Escapes

The escape symbol "\" is used to evoke special *Text* processing features. In earlier sections the \b (backspace) and \s (string substitution) features were described. The other escapes are:

```
\\ - give a single \ on output
\" - everything on the line after this is ignored (comment)
\& - null character, mainly used for getting a "." at the
     beginning of a line without having it introduce
     a command.
\  - (backslash space) - produce a space that will not be
     processed further; it is not paddable and will not be
     broken across a line boundary.
\w - stop printing the output until a <RETURN> is typed on
     the controlling terminal. This is useful for switching
     the type ball or print wheel on output devices like
     IBM Selectrics or Diablo printers.
```

15. COMMAND SUMMARY

The commands implemented in Text are:

Command	Description	Default	Section
.bd n	embolden n lines	1	6
.bn n	overstrike .bd n times	1	6
.bp [n]	begin page numbered n	---	7
.br	start new output line	---	4
.bs n	n=1: use <BS> for underline	0	6
.ce n	center n input lines	1	6
.cu n	continuous underline n lines	1	6
.f1 [+/-]n	lines above & incl footer	3	10
.f2 [+/-]n	lines below footer	2	10
.fi	fill mode	YES	4
.fo "str"str"str"	three-part footer	none	10
.h1 [+/-]n	lines above header	2	10
.h2 [+/-]n	lines below & incl header	3	10
.he "str"str"str"	three-part header	---	10
.hi string [[-]n]	hanging indent	n=0	5
.in [+/-]n	indent n characters	---	5,8
.ju	right justify (must fill)	YES	4
.ls n	double spacing or more	1	9
.ne n	need n more lines on page	---	7
.nf	stop filling text	NO	4
.nj	end right justification	NO	4
.pl [+/-]n	set page length to n	66	7
.pp	paragraph indent of 5	---	5
.rf filename	read specified file	---	11
.rm [+/-]n	set right margin to n	60	8
.sb x string	substitute string for \sx	---	13
.sp n	space down n output lines	1	6
.ti [+/-]n	indent next output line	---	5
.tm message	print message on screen	---	12
.uc char	change underline to char	"_"	6
.ul [n]	underline n input lines	1	6
\ escapes	special character functions	---	14

Reference

[1] Kernighan, B.W. and Plauger, P.J. Software Tools, Addison-Wesley Publishing Company, Reading MA, pp. 219-250.