To: MTB Distribution

From: Paul W. Benjamin

Date: 11/16/79

Subject: New Command, Subroutine for Formatting Documents

INTRODUCTION

This MTB describes format_document, an efficient but unsophisticated command for simple fill-and-adjust type formatting. A subroutine interface, format_document_, is also provided. The command is intended for use in situations where the complex features of compose are not required and efficiency is an issue. The need for such a command is amplified by the fact that runoff, which meets to a certain extent, the requirements outlined above, is implemented in a non-supported language, BCPL, and at some point in time will become obsolete.

HISTORY

This command has its roots in benchmarking situations where there is a requirement for formatting consisting solely of fill-and-adjust, indentation and page numbering. There was such a requirement in the Executive Office of the President Office Automation (EOP I) benchmark. A command of this nature was written for that benchmark. For that particular application the amount of cpu time used was roughly 10% that of compose and 20% that of runoff. The MIT PL/I runoff was also analyzed and its performance was slightly worse than that of the installed runoff. In a benchmarking situation, such numbers are crucial. In EOP I it reduced the number of processors proposed from 4 to 3.

GENERAL FEATURES

By reducing the allowable control requests to indent, undent, page_length, page_width, align_both, align_left, fill_on, and fill_off, format_document avoids the complex decision making that bog down its more sophisticated cousins. These controls are implemented in a manner consistent with those used in compose so that a user is able to use his input file as input to compose without conversion. Similarly, the command interface conforms to compose as closely as possible.

Additional compose-like features are available in the new print command which supports the -stop, -wait, -from_page, and -to_page control arguments. Since a user has the option to direct his format_document output to a segment and then print it, these features and their associated overhead are not included in format_document.

Commands such as ted and send_mail have their own fill or fill-and-adjust features. The subroutine interface format_document_ could be used by these and other commands requiring such functionality.

Multi-segment files are allowed as input and output.

Name: format_document, fdoc

This command is used to format text segments. Output lines are built from the left margin by adding text words until no more words fit on the line; the line is then justified by inserting extra blanks to make an even right margin. Control over margins and indentation is provided by control lines that begin with a period. Although the control lines are interspersed within the text, they do not appear in the output.

Usage

```
format_document path {-control_args}
```

where:

1. path

   is the pathname of an input segment or multisegment file named entryname.fdoc. The fdoc suffix must be the last component of the entryname; however the suffix need not be supplied in the command line.

2. control_args

   can be chosen from the following list.

   -indent N, -ind N, -in N

   indents output N spaces from the left margin (default indentation is 0). This space is in addition to any indentation established by the usage of the indent control in the text.

   -output_file {<path>}, -of {<path>}

   directs the output to a file instead of to the user's terminal. If {<path>} is not given, then the output is written to an output file whose name is formed by replacing the "fdoc" suffix of the input file entryname with the suffix "fdout". The default for this feature is OFF.

   -page_numbers, -pgno

   Causes each page to end with two blank lines and a centered page number. The default for this feature is OFF.

Notes

The following is a discussion of each of the control lines.

.alb causes break

Align the text at both the left and right margins according to the current value of the left indentation and undentation. Text is padded by insertion of uniformly distributed white space. The fill mode must be on for this mode to operate. If the fill mode is off, this control is mapped into the align-left (.all) control. This is the default alignment mode.

.all causes break

Align the text on the left margin according to the current values of left indentation and undentation leaving the right margin ragged.

.fif causes break

Set the fill mode off. See the discussion of .fin for details.

.fin causes break

Set the fill mode on. In fill mode, text words are moved from line to line in such a way that the last word does not extend past the right margin. The default for this mode is on.

.in {<±n>} causes break

If <±n> is given without the optional sign, then set the left indentation point to <n> columns to the right of the left margin. If <±n> is given with the optional sign, then change the current left indentation point by <n> columns. Positive values for <±n> cause movement to the right. The default value for <±n> is 0. Any value that results in a zero or negative effective line length will produce an error diagnostic message. The left indentation point is never set to the left of the left margin. The left margin is determined by the -indent control argument.

.pdl {<±n>}   no break

If <±n> is given without the optional sign, then set the page length to <n> lines. If <±n> is given with the optional sign, then change the current page length by <n>. If the resulting page length is zero or negative, an error diagnostic message is produced. The default value for <±n> is 66.

.pdw {<±n>}   no break

If {<±n>} is given without the optional sign, then set the page width to <n> columns. If <±n> is given with the optional sign, then change the current page width by <n>. If the resulting page width is zero or negative, an error diagnostic message is produced. The default value for <±n> is 65.

.un {<±n>}   causes break

Adjust the indentation point for only the next output line If <±n> is unsigned or has the + sign the indentation point is moved n columns to the left. If <±n> has the - sign, the indentation point is moved n columns to the right. The default value for <±n> is the value of the indentation value.

Any lines that begin with ".x", where x is not a space or a ".", and are not listed above are discarded so that a compin file could be processed by fdoc.

Text lines contain the material to be printed. If an input line is too short or too long to fill an output line, material is taken from or deferred to the next text line. A line beginning with a space is interpreted as a break in the text (e.g., the beginning of a new paragraph) and the previous line is printed as is.

Name: format_document_

The format_document_ entry point, given directory names and entry names, is used to fill and adjust text. The entry names can reference segments, links, or multi-segment files. Certain control lines can be imbedded in the text. See the description of the format_document command in the MPM Commands and Active Functions for information on those control lines.

Usage

```
     declare format_document_ entry (char(*), char(*), char(*),
          char(*), fixed bin, fixed bin, bit(*), fixed  bin(35));

     call format_document_ (dir_name_in, entry_name_in,
          dir_name_out, entry_name_out, indentation,
          line_length, options, code);
```

where:

1. dir_name_in (input)
   is the pathname of the containing directory of the input.

2. entry_name_in (input)
   is the entry name of the input segment, link or multi-segment file.

3. dir_name_out (input)
   is the pathname of the containing directory of the output.

4. entry_name_out (input)
   is the entry name of the output segment, link or multi-segment file. If the entry does not exist it will be created.

5. indentation (input)
   is the indentation value, causing indentation from the left margin. This space is in addition to any indentation established by the usage of the indent control in the text.

6. line_length (input)
   is the initial line length value. It is the equivalent of the ".pdw" control in the text, and can be over-ridden in the text.

7. options (input)

are two switches that specify the actions to be taken. The switches must be given in the order listed below.

pgno_sw

"1"b enables page numbering. Each page is to end with two blank lines and a centered page number.

"0"b indicates that no page numbering is requested.

adj_sw

"1"b causes adjust mode to be on initially. This is the equivalent of a ".alb" in the text. It can be over-ridden in the text.

"0"b causes adjust mode to be off initially. This is the equivalent of a ".all" in the text. It can be over-ridden in the text.

8. code (output)

is a standard status code.

Notes

The format_document_$seg_ptr entrypoint performs the same operation, for segments only, given pointer and length rather than directory name and entry name.

Name: format_document_$seg_ptr

The format_document_$seg_ptr entry point, given pointers to input and output segments, is used to fill and adjust text. Certain control lines can be imbedded in the text. See the description of the format_document command in the MPM Commands and Active Functions for information on those control lines.

Usage

```
     declare  format_document_$seg_ptr entry (ptr, fixed bin(21),
          ptr, fixed bin(21), fixed bin, fixed bin, bit(*),
          fixed bin(35));

     call format_document_$seg_ptr (inptr, inlen, outptr, outlen,
          indentation, line_length, options,  code);
```

where:

1. inptr (input)
   is a pointer to the input segment.

2. inlen (input)
   is the length in bytes of the input segment.

3. outptr (input)
   is a pointer to the output segment.

4. outlen (output)
   is the length in bytes of the output segment.

5. indentation (input)
   is the indentation value, causing indentation from the left margin. This space is in addition to any indentation established by the usage of the indent control in the text.

6. line_length (input)
   is the initial line length value. It is the equivalent of the ".pdw" control in the text, and can be over-ridden in the text.

7. options (input)
   are two switches that specify the actions to be taken. The switches must be given in the order listed below.
   pgno_sw
   "1"b enables page numbering. Each page is to end with two blank lines and a centered page number.

"0"b indicates that no page numbering is requested.

adj_sw
"1"b causes adjust mode to be on initially. This is the equivalent of a ".alb" in the text. It can be over-ridden in the text.
"0"b causes adjust mode to be off initially. This is the equivalent of a ".all" in the text. It can be over-ridden in the text.

8. code (output)
is a standard status code.

Notes

This entrypoint is for segments only. Use format_document_ for multi-segment files.

The format_document_ entrypoint performs the same operation for segments or multi-segment files given directory and entry names rather than pointers.