
mv.ENTERPRISE

Online User Reference

Manual, Chapter 6



Release 4.0
August 1997

This edition of the mv. ENTERPRISE manual set is to be used with all mv. ENTERPRISE systems operating under release 4.0 or higher. This electronic document combines the four hard copy volumes into a single file to be viewed with Adobe's Acrobat Reader, 3.01 or later. Content is identical between online and hard copy editions of this manual set.

It is the policy of PICK Systems, Inc. to improve products as new technology, components, software and firmware become available. PICK Systems, therefore, reserves the right to change specifications without prior notice.

Copyright © 1998, 2000 by PICK Systems, Inc.

Irvine, CA 92614

All rights reserved. Printed in U.S.A.

mv. ENTERPRISE is a trademark of PICK Systems, Inc.
MultiValue is a trademark of Spectrum International, Inc.
UNIX is a registered trademark in the USA and other countries, licensed exclusively through X/Open Company, Limited.

SCO and UnixWare are registered trademarks of Santa Cruz Operation, Inc.

AIX is a trademark of International Business Machines Corp.

MP-RAS is a licensed product of NCR Corporation.

PICK is registered trademark of PICK Systems.

All other brand or product names are trademarks or registered trademarks of their respective companies or organizations.

Part Numbers: This online document
combines the hard copy
manuals with these part
numbers:

84-00014A01

84-00014A02

84-00014A03

84-00014A04

Software Release: 4.0

Document Release: AA

The material contained in this document is furnished for customer reference only, and is subject to change without notice. PICK Systems, Inc. makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. PICK Systems assumes no responsibility for any errors that may appear in this document and makes no commitment to update nor to keep current the information contained in this document.

The techniques described here are proprietary and should be treated accordingly. No part of this document may be reproduced in any form or by any means without the prior written consent of PICK Systems.

Note that PICK Systems appreciates receiving suggestions and comments on its publications. Comments may be sent to the publisher:

PICK Systems, Inc.
1691 Browning
Irvine, CA 9606
Tel (800) 367-7425
E-mail: documentation@picksys.com
ATTN: Manager, Technical Publications

This page intentionally left blank.

Chapter 6 Table of Contents

Runoff	6-1
Introduction to Runoff	6-1
Invoking Runoff	6-2
Runoff Source File Format	6-4
Runoff Commands	
Runoff Commands	6-5
.BEGIN PAGE Command	6-5
.BOX and .BOX OFF Commands	6-5
.BREAK Command	6-6
.CAPITALIZE SENTENCES Command	6-6
.CENTER Command	6-7
.CHAIN Command	6-7
.CHAPTER Command	6-8
Comments	6-8
.CONTENTS Command	6-9
.CRT Command	6-9
.ENDCASE Command	6-9
.FILL Command	6-10
.FOOTING Command	6-10
.HEADING Command	6-11
.HILITE and .HILITE OFF Commands	6-12
Hyphenation	6-12
.INDENT Command	6-13
.INDENT MARGIN Command	6-13
.INDEX Command	6-13
.INPUT Command	6-14
.JUSTIFY Command	6-14
.LEFT MARGIN Command	6-15
.LINE LENGTH Command	6-15
.LOWER CASE Command	6-15

.LPTR Command	6-16
.NOCAPITALIZE SENTENCES Command	6-16
.NOFILL Command	6-16
.NOJUSTIFY Command	6-17
.NOPAGING Command	6-17
.NOPARAGRAPH Command	6-17
.PAGE NUMBER Command	6-17
.PAPER LENGTH Command	6-18
.PARAGRAPH Command	6-18
.PRINT INDEX Command	6-19
.PRINT Command	6-20
.READ Command	6-20
.READNEXT Command	6-21
.SAVE INDEX Command	6-25
.SECTION Command	6-25
.SET TABS Command	6-26
.SKIP Command	6-26
.SPACE Command	6-27
.SPACING Command	6-27
.STANDARD Command	6-28
.TEST PAGE Command	6-28
.UPPER CASE Command	6-29

Special Characters

Special Control Characters	6-31
Uppercase and Lowercase Controls	6-31
Underlining	6-32
Tab Settings	6-33
Boldface Printing	6-34
Special Character Override	6-34

Summary

Summary of Commands	6-35
---------------------	------

Chapter 6: Runoff

The Runoff text processor provides text formatting and output control for documents created with the Editor or the JED verb.

Introduction to Runoff

Runoff source text contains commands that control justification, page titling and numbering, spacing and capitalization. Textual material prepared with Runoff may be edited and corrected with the Editor and then reprinted with **RUNOFF**. Material may be inserted or deleted while maintaining the integrity of text that does not need retyping. Runoff can also merge separate documents into a single report and insert copied text into multiple reports.

The **RUNOFF** verb processes one or more source text file items in Runoff format. Multiple input items are treated as a single source text file. A source text item may contain a command that **CHAINs** Runoff to another file item. This makes it possible to **CHAIN** file items without doing a **SELECT** or **SSELECT**. Items included in Runoff's item-list may chain to other items. When the chain ends, processing continues with the next item from the item-list.

A source text item may also contain a Runoff command to read a second file item and then resume processing of the first item. This makes it possible to insert the text from a single file item into the output of many other file items.

Invoking Runoff

The RUNOFF verb invokes the Runoff text processor.

Format

RUNOFF file-name item-list {(options)}

Parameter(s)

file-name	Specifies the name of a file.
item-list	Specifies one or more source text items in Runoff format.
options	Are as follows:
C	Suppresses the .CHAIN and .READ commands.
I	Outputs the name of the next item to be Runoff. (Helpful for tracing CHAINed sequences.)
J	Suppresses .HILITE.
N	Outputs continuously to the terminal. That is, Runoff does not pause at the bottom of a page and wait for a carriage-return if the N option is used (see the “.NOPAGING Command”).
Nnn	Sets the number of times boldface letters are overprinted when output is directed to a line-printer.
P	Directs output to a printer.
S	Suppresses underlining and boldface when Runoff output is directed to a CRT.
U	Forces the entire Runoff output to uppercase.

Invoking Runoff

Description

The **RUNOFF** verb invokes the Runoff text processor allowing one or more source text file items in Runoff format to be processed.

Multiple input items are treated as a single source text file. A source text item may contain a command that **CHAINS** Runoff to another file item. This makes it possible to **CHAIN** file items without doing a **SELECT** or **SSELECT**. Items included in Runoff's item-list may chain to other items. When the chain ends, processing continues with the next item from the item-list.

A source text item may also contain a Runoff command to read a second file item and then resume processing of the first item. This makes it possible to insert the text from a single file item into the output of many other file items.

Runoff Source File Format

The source file contains the textual material that appears on the final copy, plus command information to specify formatting and alternate sources of input.

Each line of input source text is processed in the text mode, except those beginning with a period. A line beginning with a period is assumed to be a command line and is processed in the command mode. A command line may contain one or more commands, each starting with a period. The commands provide formatting information and select various modes of operation.

Runoff fills each output line by adding successive words from the source text until one more word overflows the line. The line is then justified by inserting blank spaces between words at random until the last word in the line exactly meets the right margin.

Runoff may be set to fill output lines without justifying the right margin. When filling lines, spaces and end-of-lines are treated only as word separators. Multiple-word separators are stripped from the input.

Runoff may be set to transmit the input source text to the output without filling lines or justifying margins. In this mode, multiple spaces and end-of-lines are not stripped from the input. Some of the commands cause a **BREAK** in the output. A **BREAK** means that the current line is output without justification. This occurs at the end of paragraphs.

Section 1: Runoff Commands

Runoff Commands

Runoff commands are stored along with the textual material in the source file. These commands are distinguished by a period at the start of a command line. A command line may contain one or more commands, each starting with a period. The commands provide formatting information and select various modes of operation.

.BEGIN PAGE Command

`.BEGIN PAGE` causes a `.BREAK` followed by an advance to a new page. The page number is incremented and the page heading (if set) is printed.

Format

```
.BEGIN PAGE  
.BP
```

.BOX and .BOX OFF Commands

The `.BOX` and `.BOX OFF` commands are used to enclose text in a box.

Format

```
.BOX n,m  
.BOX OFF
```

Description

The `.BOX` command encloses text that follows the command in a box. The width parameters are specified by `n` (right margin) and `m` (left margin). The text continues to be boxed until a `.BOX OFF` command is met.

For example:

```
001 .BOX 4,74
002 This is an example of a BOX.
003 .BOX OFF
```

```
-----
|This is an example of a BOX.      |
-----
```

.BREAK Command

.BREAK outputs any partially filled line before processing the next input line.

Format

```
.BREAK
.B
```

.CAPITALIZE SENTENCES Command

Sets Runoff in the capitalize sentences mode.

Format

```
.CAPITALIZE SENTENCES
.CS
```

Description

In this mode the first letter of each sentence is capitalized. The first letter after a period (.), question mark (?), or exclamation point (!), followed immediately by either a space or an end-of-line (attribute mark) is capitalized. The .CAPITALIZE SENTENCES mode also follows the period (.), question mark (?), exclamation point (!), colon (:), and semicolon (;) characters with a double space or an end-of-line. .CAPITALIZE SENTENCES is one of the .STANDARD settings (see “.STANDARD Command”). This command is ignored in the .NOFILL (NF) mode.

Runoff Commands

.CENTER Command

.CENTER inputs the next line in .NOFILL mode and centers it on the next line of output. This command causes a BREAK to occur.

Format

```
.CENTER  
.C
```

.CHAIN Command

Chains Runoff to the indicated input text file item.

Format

```
.CHAIN {DICT} {file-name} item-ID
```

Description

The DICT and file-name are each optional. If DICT is not specified, the data section of the file is assumed. If no file name is given, the item is read from the same file as the item being processed. The text input from this item is processed and output without any parameter or mode changes. Runoff does not resume processing text from the current source of input. This command does not cause a BREAK.

The .CHAIN command scans the string following the command looking for an item-ID or a file name. The legal delimiter for the item-ID or file name is a blank. They may include a period. If there is more than one string following the .CHAIN command that is delimited by a blank, the next to the last field is understood as the file name, and that file is opened. The last field delimited with a blank is the item-ID. It is retrieved by the Runoff processor to be executed next. However, you can include a comment statement after the .CHAIN. Therefore, with .CHAIN commands the line is exhausted when the processor meets an end-of-line mark, or when it meets a period preceded by a space.

If the processor opens a file when executing a .CHAIN statement, that file is the one from which all succeeding items are retrieved, until the file is respecified by another .CHAIN statement.

The `C` option suppresses the `.CHAIN` command if it is necessary to Runoff only one element of a chained or treed structure. The `l` option causes the name of the next item to be output by Runoff to be placed in the last line of the last item output by Runoff. This is useful with relatively large documents to locate the names of various source items used.

.CHAPTER Command

Use this command to number and format chapter headings. Chapter numbers increment automatically.

Format

```
.CHAPTER text
```

For example, the command `.CHAPTER RUNOFF` produces the heading:

```
CHAPTER  
RUNOFF
```

Comments

The asterisk (`*`) command informs the Runoff processor that the remainder of text in the line is a comment.

An asterisk must be entered either at the beginning of the line or after another command in a command line. It is always the last command in a line. Use it to comment out text and to note the intent of `.READS` and `.CHAINS`.

Runoff Commands

.CONTENTS Command

Use the .CONTENTS command at the end of the Runoff source file to print the table of contents. The table of contents is accumulated by preceding .CHAPTER and .SECTION commands.

Format

```
.CONTENTS
```

✓**NOTE** The LINE LENGTH and LEFT MARGIN of the table of contents is determined by those settings that are in effect when the first .CHAPTER or .SECTION command is met.

.CRT Command

Directs the Runoff output to the user's terminal.

Format

```
.CRT
```

.CRT is one of the .STANDARD settings (see the “.STANDARD Command”).

.ENDCASE Command

.EC negates the UC and LC conditions, outputting the text in its original format.

Format

```
.EC
```

Description

The forms `^^` and `\\` switch the text to uppercase or lowercase in the same way that `UC` and `LC` switch, except that `^^` and `\\` may be embedded in a line. Turning off the condition `^^` or `\\` requires the use of `EC`.

.FILL Command

`.FILL` puts Runoff into the line fill mode.

Format

```
.FILL  
.F
```

Description

Words are processed until there are enough to fill a line without overflowing it. If justification mode is on, Runoff randomly inserts spaces in the line to make the right margin line up. `.FILL` is one of the `.STANDARD` settings (see “`.STANDARD` Command”).

.FOOTING Command

`.FOOTING` inputs the next line in nofill mode and stores it in a page footing buffer. The page footing buffer is output at the bottom of each page. The page footing may be changed with successive `.FOOTING` commands.

Format

```
.FOOTING
```

The following characters have special meaning in page footings and headings, and they *must* be enclosed in single quotes, as shown. The parameter `n` is a number supplied by the user.

Runoff Commands

- 'C' Centers the line.
- 'D' Prints the date in 01 JAN 1977 format (11 characters).
- 'F' Prints the file name.
- 'Fn' Prints the file name, left-justified in a field of n spaces.
- 'I' Prints the item-ID.
- 'In' Prints the item-ID, left-justified in a field of n spaces.
- 'L' Performs a carriage-return.
- 'P' Prints the page number, right-justified in a field of four spaces with blank fill.
- 'Pn' Prints the page number, left-justified in a field of n spaces.
- 'T' Prints the time and date (22 characters).

.FOOTING causes a BREAK. (See the “.STANDARD Command” which resets footings to a null condition.)

.HEADING Command

.HEADING inputs the next line in NOFILL mode and stores it in a page heading buffer. The page heading buffer is output at the top of each page.

Format

.HEADING

The page heading may be changed with successive .HEADING commands. The special characters described in the .FOOTING command may also be used in page headings.

The .HEADING command causes a BREAK. (See “.STANDARD Command” which resets headings to a null condition.)

.HILITE and .HILITE OFF Commands

.HILITE prints the character specified by *c* at the extreme right margin for every line of text until a .HILITE OFF command is met.

Format

```
.HILITE c
.HILITE OFF
```

Description

An example of the .HILITE command may be seen at the right of the next paragraph.

The .HILITE command does not cause a break in the text. This highlights parts of paragraphs in justify or fill mode. To align the .HILITE command with a paragraph, it may be necessary to put the .HILITE command after the first line of filled or justified text, and to put the form **BREAK** at the end of the paragraph.

The execution of the .HILITE command without a character *C* is equivalent to .HILITE OFF. The *J* option suppresses .HILITE.

Hyphenation

Hyphens (-) surrounded by alphabetic characters allow a word-break on the hyphen in fill and justify modes. That is, if a term is a concatenation of two words separated by a hyphen, and the line overflows within the second part of the term, the first part and the hyphen are left in the line, and the next line continues with the second part of the word.

Similarly, if a line in the source text ends with a hyphen preceded by an alphabetic character, and the first character in the next line is an alphabetic character, the last word in the line and the hyphen are concatenated with the first word in the next line. They are then output together in a line with the hyphen between the two parts. If there is a line overflow that occurs during this process, the hyphenated word is handled as above. The processor does not remove the hyphen. If the hyphen does not have this meaning, the back-arrow character may be placed in front of it to suppress this action.

Runoff Commands

.INDENT Command

.INDENT indents the next line of output.

Format

.INDENT n

.I n

Description

The .INDENT command indents the next line of output by n spaces to the right of the left margin. The n may be negative, beginning the line to the left of the left margin. If n is missing, n=1 is assumed. This command causes a **BREAK** to occur.

.INDENT MARGIN Command

Use this command to increase/decrease the left margin.

Format

.INDENT MARGIN n

.IM n

Description

Increases the left margin by n spaces and decreases the line length by n. Negative n decreases the left margin and increases the line length. This command causes a **BREAK** to occur.

.INDEX Command

.INDEX command stores specified text in an index list.

Format

.INDEX text

Description

The text may be more than one word, or several words enclosed in single quotes. The word, or words, along with the current page

number, are put in a sorted index list. The index can be printed by the PRINT INDEX command.

.INPUT Command

The .INPUT command reads the next line of source text from the user's terminal.

Format

```
.INPUT
```

Description

The text input from the terminal is processed and output without a BREAK or mode change. This is useful for inserting words or sentences into form letters, thus customizing them.

.JUSTIFY Command

.JUSTIFY places Runoff in the FILL and JUSTIFY modes.

Format

```
.JUSTIFY  
.J
```

Description

Runoff fills each output line by adding successive words from the source text until one more words overflows the line. The line is justified by randomly inserting blank spaces between words until the last word in the line exactly meets the right margin. .JUSTIFY is one of the .STANDARD settings (see “.STANDARD Command”).

Runoff Commands

.LEFT MARGIN Command

The `.LEFT MARGIN` command sets the left margin spacing.

Format

```
.LEFT MARGIN n
```

Description

This command sets the left margin to `n` spaces. If `n` plus the current line length exceeds the maximum line length, this command is ignored. A `.LEFT MARGIN` of 0 is one of the `.STANDARD` settings (see “`.STANDARD Command`”).

.LINE LENGTH Command

This command sets the line length to `n` characters (not counting the left margin).

Format

```
.LINE LENGTH n
```

Description

If `n` plus the current left margin exceeds the maximum line length, this command is ignored. A `.LINE LENGTH` of 60 is one of the `.STANDARD` settings (see “`.STANDARD Command`”).

.LOWER CASE Command

This command places Runoff into lowercase mode.

Format

```
.LOWER CASE  
.LC
```

In lowercase mode all letters are automatically made lowercase. They may then be changed to uppercase by various text commands or control characters (see “Special Control Characters”).

.LPTR Command

Directs the Runoff output to the line-printer.

Format

```
.LPTR
```

.NOCAPITALIZE SENTENCES Command

Resets the .CAPITALIZE SENTENCES mode.

Format

```
.NOCAPITALIZE SENTENCES  
.NCS
```

.NOFILL Command

Resets the JUSTIFY and FILL modes.

Format

```
.NOFILL  
.NF
```

Description

Input text lines are output as they are, (after possible elimination of special control characters) without removal of extra spaces. Output lines are not filled nor are right margins justified. .NOFILL causes a BREAK.

Runoff Commands

.NOJUSTIFY Command

Resets JUSTIFY mode. Has no effect on the FILL mode.

Format

```
.NOJUSTIFY  
.NJ
```

.NOPAGING Command

The .NOPAGING command eliminates the need for terminal input at the end of each output page.

Formats

```
.NOPAGING  
.N
```

Description

Functions the same as the Runoff N option except that .NOPAGING can be restricted to a portion of a document.

.NOPARAGRAPH Command

Resets the paragraph mode. Blank input text lines and spaces at the beginning of a line are ignored in justify mode.

Format

```
.NOPARAGRAPH
```

.PAGE NUMBER Command

Sets the current page number.

Format

```
.PAGE NUMBER n
```

Description

This command sets the current page number to n. If n is missing, n=1 is assumed.

.PAPER LENGTH Command

Use this command to set the paper length.

Format

```
.PAPER LENGTH n
```

Description

Sets the paper length to *n* lines.

.PARAGRAPH Command

Specifies the number of spaces to indent paragraphs.

Format

```
.PARAGRAPH n
```

Description

Any line that begins with a space is recognized as the first line of a paragraph and is indented according to the number of spaces specified by the `.PARAGRAPH` command.

The number of spaces to indent is specified by the *n* parameter, and can be a positive or negative number. A positive number specifies indenting that number of spaces from the left margin. The command `.PARAGRAPH 5`, for example, indents the first line of a paragraph five spaces, and is one of the `.STANDARD` settings (see `“.STANDARD Command”`).

A negative number specifies extending the first line of the paragraph that number of spaces into the left margin. The command `.PARAGRAPH -5`, for example, indents the first line of the paragraph five spaces into the left margin. An example of negative paragraph spacing is shown in the example below.

The `.PARAGRAPH` command also causes a `BREAK` followed by $(\text{line spacing} + 1) / 2$ blank lines to be inserted between paragraphs.

Runoff Commands

001.NJ.PARAGRAPH -4 .LEFT MARGIN 13 .LINE LENGTH 50
002 1. The .PARAGRAPH command specifies the number of spaces
003for paragraph indentation. Any line that begins with a space
004is recognized as the first line of a paragraph and is indented
005according to the number of spaces specified by the .PARAGRAPH
006command.
007 2. Notice that in this example, text lines 002 and 007
008begin with a space. This indicates that these two lines are
009the first lines of new paragraphs and are affected by the
010.PARAGRAPH -4 command. Indenting by -4 spaces allows these
011two lines to extend into the left margin by four spaces.

The source text above prints as follows:

1. The .PARAGRAPH command specifies the number of spaces for paragraph indentation. Any line that begins with a space is recognized as the first line of a paragraph and is indented according to the number of spaces specified by the .PARAGRAPH command.
2. Notice that in this example, text lines 002 and 007 begin with a space. This indicates that these two lines are the first lines of new paragraphs and are affected by the .PARAGRAPH -4 command. Indenting by -4 spaces allows these two lines to extend into the left margin by four spaces.

.PRINT INDEX Command

This command prints a sorted index list of words and page numbers. The index is sorted into alphabetical order and printed in two columns per page.

Format

`.PRINT INDEX`

✓NOTE This command changes the tab settings and performs a .BEGIN PAGE command.

.PRINT Command

The .PRINT command prints the next line of input text on the user's terminal.

Format

```
.PRINT
```

This command may be used as a prompt before an input command.

.READ Command

Reads the specified file item.

Format

```
.READ {DICT} {file-name} item-ID
```

The DICT and file-name are each optional. If DICT is not specified, the data section of the file is used. If file-name is not specified, the item is read from the same file as the item being processed. The text input from this item is processed and output without any parameter or mode changes. After processing this item, Runoff resumes input with the next line of the current source of input. Options are:

- C Suppresses the .READ command if it is necessary to Runoff only one element of a chained or treed structure.
- I Causes the name of the next item to be output by Runoff to be placed in the last line of the last item output by Runoff. This is useful to locate source text boundaries within large documents.

Description

The .READ command scans the string following the command, looking for an item-ID or a file-name. The legal delimiter for the item-ID or file-name is a blank. They may include periods. If more than one string follows a .READ command that is delimited by a blank, the next-to-the-last field is taken to be the file-name, where the file is opened. The last field delimited with a blank is considered the

Runoff Commands

item-ID, is retrieved by the Runoff processor, and is executed next. The processor eventually returns to this item to continue processing it after finishing the processing of the item read. As it returns, it starts at the beginning of the next line in the item. Therefore, no statements are executed that occur beyond the READ statement. However, you can include a comment statement after the .READ. Therefore, regarding the .READ command, the line is considered exhausted when the processor meets an End-Of-Line mark, or when it meets a period preceding a space.

This command does not cause a BREAK.

.READNEXT Command

Reads data from a pre-selected list. It has an effect only if a SELECT, SSELECT, or GLIST statement is entered (before entering Runoff) that selects a list of values.

Format

.READNEXT

Description

Each .READNEXT command in Runoff extracts one value from the select-list and places it in the text stream. .READNEXT does not cause a break. If there is no preselected list, or when the list is exhausted, .READNEXT stops Runoff and returns to TCL.

This command is especially useful for generating form-letters. For example, it may be necessary to insert from a separate file the name and address of each recipient of the letter. A SSELECT statement extracts the relevant data from the file and saves it in a list. A series of .READNEXT statements inserts data into the text of the letter. At the end of the letter a .CHAIN statement restarts the next letter. When the list is exhausted, Runoff stops. The commands necessary to generate a form letter are:

```
{S}SELECT file-name {selection-criteria} attribute-list  
.READNEXT  
.CHAIN item-name
```

Runoff Commands

The selected attribute-list contains all the variable information to be written into the form letter. The use of .READNEXT command is that it reads each of these variables and writes them into the letter. The .CHAIN command repeats the letter as long as there is variable information in the selected attribute list. The following .READNEXT example demonstrates the generation of a form letter.

Example

Assume the dictionary of the accounts payable file contains the following three attribute defining items:

<u>NAME</u>	<u>ACCOUNT</u>	<u>AMOUNT</u>
001 A	001 A	001 A
002 1	002 2	003 3
003 CUSTOMER NAME	003 ACCOUNT TYPE	003 AMOUNT DUE
004	004	004
005	005	005
006	006	006
007	007	007
008 A1:","	008	008 A3(MR2\$):"."
009 L	009 L	009 R
010 25	010 30	010 10

The dictionary also contains the following form letter written in Runoff:

LETTER

```
001 .SK 8
002 Dear Mr.
003 .READNEXT
004 Our records show that your
005 .READNEXT
006 account is overdrawn by the amount of
007 .READNEXT
008 We would appreciate prompt payment.
009 Thank you,
010           Indiana Jones
011 .SK 2
012 President CELEBRITY SERVICES CO.
013 .SK 3
```

Runoff Commands

```
014 .BP
015 .CHAIN LETTER
```

The data file contains items such as the following three:

250	251	252
001 Magic Johnson	001 Eddie Van Halen	001 Boy George
002 Basketball Shoes	002 Guitar String	002 Voice Lesson
003 25000	003 12345	003 452359

To generate the form letter the data file is first sort selected by name with the attribute list of NAME ACCOUNT and AMOUNT:

```
SSELECT ACC-PAYABLE BY NAME NAME ACCOUNT AMOUNT
```

This command generates a selected list:

```
001 Boy George,
002 Voice Lesson
003 $4,523.59.
004 Eddie Van Halen,
005 Guitar String
006 $123.45.
007 Magic Johnson,
008 Basketball Shoes
009 $250.00.
```

Runoff Commands

Note that correlatives are performed on the names and amounts. Issuing the following Runoff command generates the three form letters shown below.

RUNOFF DICT ACC-PAYABLE LETTER (P)

The form letters print as follows:

Dear Mr. Boy George,

Our records show that your Voice Lesson account is overdrawn by the amount of \$4,523.59. We would appreciate prompt payment.

Thank You,

Indiana Jones

President CELEBRITY SERVICES CO.

Dear Mr. Eddie Van Halen,

Our records show that your Guitar String account is overdrawn by the amount of \$123.45. We would appreciate prompt payment.

Thank You,

Indiana Jones

President CELEBRITY SERVICES CO.

Dear Mr. Magic Johnson,

Our records show that your Basketball Shoes account is overdrawn by the amount of \$250.00. We would appreciate prompt payment.

Thank You,

Indiana Jones

President CELEBRITY SERVICES CO.

Runoff Commands

.SAVE INDEX Command

Saves all chapter and page number information of indexed text data in a separate file.

Format

```
.SAVE INDEX file-name
```

file-name is the name of the file in which the chapter and page information is to be stored.

Description

Each indexed data is stored as an individual item using the data as the item-ID, the chapter (where that data is referenced) as the first attribute, and the page number as the second attribute. Multiple values are stored in these attributes as multiple references to the same indexed data. The resulting file may then be operated on by the ACCESS processor to generate listings for the chapter and page number information of all indexed text data.

✓NOTE This must be a separate file from the text file, otherwise data in the text file is destroyed.

The .SAVE INDEX command is placed in the text item and must precede the .INDEX commands. Only indexed data preceded by the .SAVE INDEX command is saved in the specified file.

.SECTION Command

This command may be used with the .CHAPTER command to handle automatic chapter section numbering and formatting.

Format

```
.SECTION n text
```

The .SECTION command automatically starts the next section at depth n, where n is the range 1-5.

The text is printed following where the section number and **SKIP** occurs. The text is recorded as the section heading in the table of contents. If no text appears on the **.SECTION** command, no **SKIP** occurs, and the section is not recorded in the table of contents. Section numbers are incremented automatically, and the section number is printed in the form i.j.k.l.m with n digits printed.

Conventionally, the **.SECTION** command is followed by a blank line before the next paragraph starts. Since the **.SECTION** command causes a break that ends the preceding paragraph, and since the text following the **.SECTION** command is placed immediately into an output line and output before considering the next line, the blank line after the **.SECTION** command can be avoided by not indenting the first line of the next paragraph. That is, if the processor does not know that the next line starts a paragraph, it does not skip a line. However, it may be necessary to use an **.INDENT MARGIN** if paragraph indentation is wanted.

.SET TABS Command

Clears previous tab stops and sets new tab stops as indicated by the numeric tab positions.

Format

.SET TABS n,n,n ...

Description

The tab stops (up to 30) must be greater than zero and in increasing order. They indicate tab stop positions relative to the left margin. **.TABS** are only in effect in **NOFILL** mode. The left-tab character (<) starts the next word at the next tab position. The right-tab character (>) ends the next word at the next tab position. If a tab character appears at a point in the line where no further tab stops are set, the tab character is ignored.

.SKIP Command

The **.SKIP** command causes a **BREAK**.

Runoff Commands

Formats

.SKIP n
.SK n

Description

.SKIP causes a BREAK after which $n \times (\text{SPACING } n)$ lines are left blank. If the skip advances past the end of the page, the output is advanced to the top of the next page. If n is missing, $n=1$ is assumed.

.SPACE Command

This command functions the same as .SKIP, except that n lines are left blank regardless of the .SPACING n command.

Formats

.SPACE n
.SP n

Description

.SPACE is used where space is to be left independent of the line spacing. .SKIP is used where space should be relative to the .SPACING command. If n is missing, $n=1$ is assumed.

.SPACING Command

Sets the line spacing.

Format

.SPACING n

Description

This command sets the line spacing to n . For double spacing use the command .SPACING 2 .

.STANDARD Command

Sets the standard (default) parameters and modes.

Format

```
.STANDARD
```

Description

The `.STANDARD` command is equivalent to the following commands:

```
.CS.F.J.LEFT MARGIN 0.CRT.HEADING  
.FOOTING  
.PARAGRAPH 5.LINE LENGTH 60
```

.TEST PAGE Command

Causes a `BREAK` followed by an advance to a new page.

Format

```
.TEST PAGE n
```

Description

This command causes a `BREAK` followed by an advance to a new page when there are less than `n` lines remaining on the current page. Use this command to ensure that the `n` lines following the command are all output on the same page.

Runoff Commands

.UPPER CASE Command

Places Runoff into uppercase mode.

Format

.UPPER CASE

.UC

Description

In uppercase mode all letters are automatically made uppercase. They may then be changed to lowercase by various text commands or control characters (see “Special Control Characters”). .UC is one of the .STANDARD settings (see “.STANDARD Command”).

This page intentionally left blank.

Section 2: Special Characters

Special Control Characters

Runoff features include special control characters for underlining, tab setting, uppercase and lowercase, and special character override. Unlike commands, these characters may be embedded in the text lines.

Uppercase and Lowercase Controls

The uppercase, lowercase control structure specifies the case of the text. `.ENDCASE` or `.EC` turns off uppercase or lowercase, reverting the text to its original case.

The forms `^^` and `\\` switch the text between uppercase and lowercase in the same way as do `.UC` and `.LC`, except that `^^` and `\\` may be embedded in a line. Turning off the condition `^^` or `\\` requires the use of `.EC`.

The forms `^`, `\`, `&` and `@` produce, respectively, one character of uppercase, lowercase, underline, or boldface. Each is treated as the character itself if it is followed by a blank. The underline character (`_`) assigns the following character to be a text character rather than a control character. This means that if you have existing Runoff text with forms such as `40# @$1.28/#`, the dollar sign is made bold and the `@` disappears, unless the underline is inserted in front of the `@` sign.

Refer to the following example, which displays the interactions of several of the above commands. The first part is the text sent to Runoff, the second part is the output from Runoff. Note, the `l` in `is` is always capitalized by the single-character `^`, and that the `a` is always in lowercase due to the single-character `\` command.

.LINE LENGTH 66.PARAGRAPH 5.J

This ^is \a test of UC AND LC.

.UC

This ^is \a test of UC AND LC.

.LC

This ^is \a test of UC AND LC.

This ^is \a ^test of UC AND LC.

This ^is \a test\\ of UC AND LC.

.EC

This ^is \a test of UC AND LC.

This Is a test of UC AND LC.

THIS IS a TEST OF UC AND LC.

This Is a test of uc and lc.

This Is a TEST OF UC AND LC.

THIS IS a TEST of uc and lc.

This Is a test of UC AND LC.

The first line is in its natural form. The second line is uniformly capitalized by the .UC command, except the a. The third line is uniformly sent to lowercase, except for the is. The fourth and fifth lines contain a string of the form ^text \\, which is uniformly capitalized, except the a. After \\ the string reverts to lowercase. The only way to retrieve the capitalization of the string UC AND LC is by using the .EC command. Thus, the sixth line is in its natural form.

Underlining

The ampersand (&) underlines the letter immediately following it. For example:

.LC

THE LETTER &A IS FIRST IN THE ALPHABET

the letter a is first in the alphabet

Special Control Characters

The ampersand may be used with the up-arrow (^) and backslash (\) to underline a series of characters. An ampersand followed immediately by an up-arrow (&^) puts Runoff in the underline mode until an ampersand followed immediately by a backslash (&\) is met.

```
&^SPECIAL CONTROL CHARACTERS&\ ARE NEEDED...
```

This example of Runoff source prints as

```
SPECIAL CONTROL CHARACTERS ARE NEEDED ...
```

The S option suppresses underlinings and overstriking.

Tab Settings

The less-than (<) and greater-than (>) characters are used for tabbing. The left-tab character (<) starts the next word at the next tab position as set by the SET TABS command. The right-tab character (>) ends the next word at the next tab position.

```
.NF
.SET TABS 5,8,25
.SK 1
<&^NAME<CONVENTIONAL DATA PROCESSING NAME&
.SK 1
>1.<Item<Record
>1a.<Attribute<Field
>1b.<Item-ID<Record Key
```

This example of Runoff source prints as

	<u>NAME</u>	<u>CONVENTIONAL DATA PROCESSING NAME</u>
1.	Item	Record
1a.	Attribute	Field
1b.	Item-ID	Record Key

✓**NOTE** Tab characters are only in effect in the NOFILL (NF) mode.

Boldface Printing

The at sign (@) is used to indicate **BOLDFACE** type. An at sign (@) followed immediately by an up-arrow (@^) puts Runoff in the boldface mode until an at sign (@) followed immediately by a backslash (@\) is met. The number of times the boldface letters are overprinted may be set by using the numeric option of the RUNOFF verb.

```
@^SPECIAL CONTROL CHARACTERS@\ ARE NEEDED ...
```

This example of Runoff source prints as

```
SPECIAL CONTROL CHARACTERS ARE NEEDED ...
```

Special Character Override

Underscore (_) displays one of the special control characters. The letter immediately following the back-arrow transmits to the output without special processing.

```
_ ^SPECIAL _ ^CONTROL _ ^CHARACTERS ARE NEEDED ...
```

This example of Runoff source prints as

```
^SPECIAL ^CONTROL ^CHARACTERS ARE NEEDED ...
```

To quote the underscore (_) character, precede it with an up-arrow character.

```
^ _UNDERSCORE ^ _CHARACTERS ^ _ARE ^ _NEEDED ...
```

This example of Runoff source prints as

```
_ _UNDERSCORE _ _CHARACTERS _ _ARE _ _NEEDED ...
```

Section 3: Summary

Summary of Commands

The following is a summary of the Runoff commands described in this chapter.

```
RUNOFF file-name item-list {(options)}  
.BEGIN PAGE  
.BP  
  
.BOX n,m  
.BOX OFF  
  
.BREAK  
.B  
  
.CAPITALIZE SENTENCES  
.CS  
  
.CENTER  
.C  
  
.CHAIN {DICT} {file-name} item-ID  
  
.CHAPTER text  
  
.CONTENTS  
  
.CRT  
  
.EC  
  
.FILL  
  
.F  
  
.FOOTING  
.HEADING  
  
.HILITE c
```

.HILITE OFF

.INDENT n
.I n

.INDENT MARGIN n
.IM n

.INDEX text

.INPUT

.JUSTIFY
.J

.LEFT MARGIN n

.LINE LENGTH n

.LOWER CASE
.LC

.LPTR

.NOCAPITALIZE SENTENCES
.NCS

.NOFILL
.NF

.NOJUSTIFY
.NJ

.NOPAGING
.N

.NOPARAGRAPH

.PAGE NUMBER n

.PAPER LENGTH n

.PARAGRAPH n

.PRINT INDEX

.PRINT

Summary of Commands

.READ {DICT} {file-name} item-ID
.READNEXT
.SAVE INDEX file-name
.SECTION n text
.SET TABS n,n,n ...
.SKIP n
.SK n
.SPACE n
.SP n
.SPACING n
.STANDARD
.TEST PAGE n
.UPPER CASE
.UC

This page intentionally left blank.