Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'read.1p' command

$ man read.1p

READ(1P) POSIX Programmer's Manual READ(1P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

read ? read from standard input into shell variables

SYNOPSIS

read [-r] var...

DESCRIPTION

The read utility shall read a single logical line from standard input into one or more shell variables.

By default, unless the -r option is specified, <backslash> shall act as an escape character. An unescaped <backslash> shall preserve the lit? eral value of the following character, with the exception of a <new? line>. If a <newline> follows the <backslash>, the read utility shall interpret this as line continuation. The <backslash> and <newline> shall be removed before splitting the input into fields. All other un? escaped <backslash> characters shall be removed after splitting the in? put into fields.

If standard input is a terminal device and the invoking shell is inter? active, read shall prompt for a continuation line when it reads an in?

put line ending with a <backslash> <newline>, unless the -r option is specified.

The terminating <newline> (if any) shall be removed from the input and the results shall be split into fields as in the shell for the results of parameter expansion (see Section 2.6.5, Field Splitting); the first field shall be assigned to the first variable var, the second field to the second variable var, and so on. If there are fewer fields than there are var operands, the remaining vars shall be set to empty strings. If there are fewer vars than fields, the last var shall be set to a value comprising the following elements:

* The field that corresponds to the last var in the normal assignment sequence described above
* The delimiter(s) that follow the field corresponding to the last var
* The remaining fields and their delimiters, with trailing IFS white space ignored

The setting of variables specified by the var operands shall affect the current shell execution environment; see Section 2.12, Shell Execution Environment. If it is called in a subshell or separate utility execu? tion environment, such as one of the following:

```
(read foo)
nohup read ...
find . -exec read ... \;
```

it shall not affect the shell variables in the caller's environment.

OPTIONS

The read utility shall conform to the Base Definitions volume of POSIX.1?2017, Section 12.2, Utility Syntax Guidelines.

The following option is supported:

-r Do not treat a <backslash> character in any special way. Con? sider each <backslash> to be part of the input line.

OPERANDS

The following operand shall be supported:

var The name of an existing or nonexisting shell variable.

STDIN

The standard input shall be a text file.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of read:

IFS Determine the internal field separators used to delimit fields; see Section 2.5.3, Shell Variables.

LANG Provide a default value for the internationalization vari? ables that are unset or null. (See the Base Definitions vol? ume of POSIX.1?2017, Section 8.2, Internationalization Vari? ables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard er? ror.

NLSPATH Determine the location of message catalogs for the processing of LC_MESSAGES.

PS2 Provide the prompt string that an interactive shell shall write to standard error when a line ending with a <backslash> <newline> is read and the -r option was not specified.

ASYNCHRONOUS EVENTS

Default.

STDOUT

Not used.

STDERR

The standard error shall be used for diagnostic messages and prompts

for continued input.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0 Successful completion.

>0 End-of-file was detected or an error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The -r option is included to enable read to subsume the purpose of the line utility, which is not included in POSIX.1?2008.

EXAMPLES

The following command:

```
while read -r xx yy
do
    printf "%s %s\n$yy$xx"
done < input_file
```

prints a file with the first field of each line moved to the end of the line.

RATIONALE

The read utility historically has been a shell built-in. It was sepa? rated off into its own utility to take advantage of the richer descrip? tion of functionality introduced by this volume of POSIX.1?2017.

Since read affects the current shell execution environment, it is gen? erally provided as a shell regular built-in. If it is called in a sub? shell or separate utility execution environment, such as one of the following:

```
(read foo)
nohup read ...
```

find . -exec read ... \;

it does not affect the shell variables in the environment of the caller.

Although the standard input is required to be a text file, and there? fore will always end with a <newline> (unless it is an empty file), the processing of continuation lines when the -r option is not used can re? sult in the input not ending with a <newline>. This occurs if the last line of the input file ends with a <backslash> <newline>. It is for this reason that ``if any'' is used in ``The terminating <newline> (if any) shall be removed from the input'' in the description. It is not a relaxation of the requirement for standard input to be a text file.

FUTURE DIRECTIONS

None.

SEE ALSO

Chapter 2, Shell Command Language

The Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables, Section 12.2, Utility Syntax Guidelines