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 'fputs.3p' command

$ man fputs.3p

FPUTS(3P) POSIX Programmer's Manual FPUTS(3P)

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

fputs ? put a string on a stream

SYNOPSIS

```
#include <stdio.h>
int fputs(const char *restrict s, FILE *restrict stream);
```

DESCRIPTION

The functionality described on this reference page is aligned with the ISO C standard. Any conflict between the requirements described here and the ISO C standard is unintentional. This volume of POSIX.1?2017 defers to the ISO C standard.

The fputs() function shall write the null-terminated string pointed to by s to the stream pointed to by stream. The terminating null byte shall not be written.

The last data modification and last file status change timestamps of the file shall be marked for update between the successful execution of fputs() and the next successful completion of a call to fflush() or fclose() on the same stream or a call to exit() or abort().

RETURN VALUE

Upon successful completion, fputs() shall return a non-negative number. Otherwise, it shall return EOF, set an error indicator for the stream, and set errno to indicate the error.

ERRORS

Refer to fputc().

The following sections are informative.

EXAMPLES

Printing to Standard Output

The following example gets the current time, converts it to a string using localtime() and asctime(), and prints it to standard output using fputs(). It then prints the number of minutes to an event for which it is waiting.

```
#include <time.h>
#include <stdio.h>
...
time_t now;
int minutes_to_event;
...
time(&now);
printf("The time is ");
fputs(asctime(localtime(&now)), stdout);
printf("There are still %d minutes to the event.\n",
    minutes_to_event);
...
```

APPLICATION USAGE

The puts() function appends a <newline> while fputs() does not.

This volume of POSIX.1?2017 requires that successful completion simply return a non-negative integer. There are at least three known different implementation conventions for this requirement:

* Return a constant value.
* Return the last character written.
* Return the number of bytes written. Note that this implementation

convention cannot be adhered to for strings longer than {INT_MAX} bytes as the value would not be representable in the return type of the function. For backwards-compatibility, implementations can re? turn the number of bytes for strings of up to {INT_MAX} bytes, and return {INT_MAX} for all longer strings.

RATIONALE

The fputs() function is one whose source code was specified in the ref? erenced The C Programming Language. In the original edition, the func? tion had no defined return value, yet many practical implementations would, as a side-effect, return the value of the last character written as that was the value remaining in the accumulator used as a return value. In the second edition of the book, either the fixed value 0 or EOF would be returned depending upon the return value of ferror(); how? ever, for compatibility with extant implementations, several implemen? tations would, upon success, return a positive value representing the last byte written.

FUTURE DIRECTIONS

None.

SEE ALSO

Section 2.5, Standard I/O Streams, fopen(), putc(), puts()

The Base Definitions volume of POSIX.1?2017, <stdio.h>

COPYRIGHT

files to man page format. To report such errors, see https://www.ker?

nel.org/doc/man-pages/reporting_bugs.html .