NAME

strfmon – convert monetary value to string

SYNOPSIS

#include <monetary.h>

ssize_t strfmon(char **s,* **size_t** *maxsize,* **const char** **format,* **. . .);**

MT-LEVEL

MT-Safe

DESCRIPTION

The **strfmon( )** function places characters into the array pointed to by *s* as controlled by the string pointed to by *format*. No more than *maxsize* bytes are placed into the array.

The format is a character string that contains two types of objects: plain characters, which are simply copied to the output stream, and conversion specifications, each of which results in the fetching of zero or more arguments which are converted and formatted. The results are undefined if there are insufficient arguments for the format. If the format is exhausted while arguments remain, the excess arguments are simply ignored.

A conversion specification consists of the following sequence:

- a % character
- optional flags
- optional field width
- optional left precision
- optional right precision
- a required conversion character that determines the conversion to be performed.

Flags

One or more of the following optional flags can be specified to control the conversion:

=*f* — An = followed by a single character *f* which is used as the numeric fill character. The fill character must be representable in a single byte in order to work with precision and width counts. The default numeric fill character is the space character. This flag does not affect field width filling which always uses the space character. This flag is ignored unless a left precision (see below) is specified.

^ — Do not format the currency amount with grouping characters. The default is to insert the grouping characters if defined for the current locale.

+ or (— Specify the style of representing positive and negative currency amounts. Only one of '+' or '(' may be specified. If '+' is specified, the locale's equivalent of + and '–' are used (for example, in the U.S.A.: the empty string if positive and '–' if negative). If '(' is specified, negative amounts are enclosed within parentheses. If neither flag is specified, the '+' style is used.

! — Suppress the currency symbol from the output conversion.

– — Specify the alignment. If this flag is present all fields are left-justified (padded to the right) rather than right-justified.

Field Width

w A decimal digit string *w* specifying a minimum field width in bytes in which the result of the conversion is right-justified (or left-justified if the flag '–' is specified). The default is zero.

Left Precision

#*n* A '#' followed by a decimal digit string *n* specifying a maximum number of digits expected to be formatted to the left of the radix character. This option can be used to keep the formatted output from multiple calls to the **strfmon( )** aligned in the same columns. It can also be used to fill unused positions with a special character as in **$***123.45**. This option causes an amount to be formatted as if it has the number of digits specified by *n*. If more than *n* digit positions are required, this conversion specification is ignored. Digit positions in excess of those actually required are filled with the numeric fill character (see the =*f* flag above).

If grouping has not been suppressed with the '^' flag, and it is defined for the current locale, grouping separators are inserted before the fill characters (if any) are added. Grouping sparators are not applied to fill characters even if the fill character is a digit.

To ensure alignment, any characters appearing before or after the number in the formatted output such as currency or sign symbols are padded as necessary with space characters to make their positive and negative formats an equal length.

Right Precision

.*p* A period followed by a decimal digit string *p* specifying the number of digits after the radix character. If the value of the right precision *p* is zero, no radix character appears. If a right precision is not included, a default specified by the current locale is used. The amount being formatted is rounded to the specified number of digits prior to formatting.

Conversion Characters

The conversion characters and their meanings are:

i The **double** argument is formatted according to the locale's international currency format (for example, in the U.S.A.: USD 1,234.56**).**

n The **double** argument is formatted according to the locale's national currency format (for example, in the U.S.A.: $1,234.56).

% Convert to a %; no argument is converted. The entire conversion specification must be %%.

Locale Information

The **LC_MONETARY** category of the program's locale affects the behaviour of this function including the monetary radix character (which may be different from the numeric radix character affected by the **LC_NUMERIC** category), the grouping separator, the currency symbols and formats. The international currency symbol should be conformant with the ISO 4217: 1987 standard.

RETURN VALUES

If the total number of resulting bytes (including the terminating null byte) is not more than *maxsize*, **strfmon( )** returns the number of bytes placed into the array pointed to by *s*, not including the terminating null byte. Otherwise, –**1** is returned, the contents of the array are indeterminate, and **errno** is set to indicate the error.

ERRORS

strfmon( ) will fail if:

ENOSYS The function is not supported.

E2BIG Conversion stopped due to lack of space in the buffer.

EXAMPLES

Given a locale for the U.S.A. and the values 123.45, –123.45, and 3456.781:

Conversion Specification	Output	Comments
%n	$123.45 -$123.45 $3,456.78	default formatting
%11n	$123.45 -$123.45 $3,456.78	right align within an 11 character field
%#5n	$ 123.45 -$ 123.45 $ 3,456.78	aligned columns for values up to 99,999
%=*#5n	$***123.45 -$***123.45 $*3,456.78	specify a fill character
%=0#5n	$000123.45 -$000123.45 $03,456.78	fill characters do not use grouping even if the fill character is a digit
%^#5n	$ 123.45 -$ 123.45 $ 3456.78	disable the grouping separator
%^#5.0n	$ 123 -$ 123 $ 3457	round off to whole units
%^#5.4n	$ 123.4500 -$ 123.4500 $ 3456.7810	increase the precision
%(#5n	123.45 ($ 123.45) $ 3,456.78	use an alternative pos/neg style
%!(#5n	123.45 (123.45) 3,456.78	disable the currency symbol

SEE ALSO

localeconv(3C)

NOTES

This interface is expected to be mandatory in a future issue of this document.

Lower-case conversion characters are reserved for future use and upper-case for implementation-dependent use.