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 'wcstod.3p' command

$ man wcstod.3p

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

wcstod, wcstof, wcstold ? convert a wide-character string to a double-precision number

SYNOPSIS

```
#include <wchar.h>
double wcstod(const wchar_t *restrict nptr, wchar_t **restrict endptr);
float wcstof(const wchar_t *restrict nptr, wchar_t **restrict endptr);
long double wcstold(const wchar_t *restrict nptr,
    wchar_t **restrict endptr);
```

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.

These functions shall convert the initial portion of the wide-character string pointed to by nptr to double, float, and long double representa? tion, respectively. First, they shall decompose the input wide-charac?

ter string into three parts:

1. An initial, possibly empty, sequence of white-space wide-character codes (as specified by iswspace())
2. A subject sequence interpreted as a floating-point constant or representing infinity or NaN
3. A final wide-character string of one or more unrecognized wide-character codes, including the terminating null wide-character code of the input wide-character string

Then they shall attempt to convert the subject sequence to a floating-point number, and return the result.

The expected form of the subject sequence is an optional '+' or '-' sign, then one of the following:

* A non-empty sequence of decimal digits optionally containing a radix character; then an optional exponent part consisting of the wide character 'e' or the wide character 'E', optionally followed by a '+' or '-' wide character, and then followed by one or more decimal digits
* A 0x or 0X, then a non-empty sequence of hexadecimal digits optionally containing a radix character; then an optional binary exponent part consisting of the wide character 'p' or the wide character 'P', optionally followed by a '+' or '-' wide character, and then followed by one or more decimal digits
* One of INF or INFINITY, or any other wide string equivalent except for case
* One of NAN or NAN(n-wchar-sequenceopt), or any other wide string ignoring case in the NAN part, where:

    n-wchar-sequence:

        digit

        nondigit

        n-wchar-sequence digit

        n-wchar-sequence nondigit

The subject sequence is defined as the longest initial subsequence of the input wide string, starting with the first non-white-space wide

character, that is of the expected form. The subject sequence contains no wide characters if the input wide string is not of the expected form.

If the subject sequence has the expected form for a floating-point num? ber, the sequence of wide characters starting with the first digit or the radix character (whichever occurs first) shall be interpreted as a floating constant according to the rules of the C language, except that the radix character shall be used in place of a period, and that if neither an exponent part nor a radix character appears in a decimal floating-point number, or if a binary exponent part does not appear in a hexadecimal floating-point number, an exponent part of the appropri? ate type with value zero shall be assumed to follow the last digit in the string. If the subject sequence begins with a <hyphen-minus>, the sequence shall be interpreted as negated. A wide-character sequence INF or INFINITY shall be interpreted as an infinity, if representable in the return type, else as if it were a floating constant that is too large for the range of the return type. A wide-character sequence NAN or NAN(n-wchar-sequenceopt) shall be interpreted as a quiet NaN, if supported in the return type, else as if it were a subject sequence part that does not have the expected form; the meaning of the n-wchar sequences is implementation-defined. A pointer to the final wide string shall be stored in the object pointed to by endptr, provided that endptr is not a null pointer.

If the subject sequence has the hexadecimal form and FLT_RADIX is a power of 2, the conversion shall be rounded in an implementation-de? fined manner.

The radix character shall be as defined in the current locale (category LC_NUMERIC). In the POSIX locale, or in a locale where the radix char? acter is not defined, the radix character shall default to a <period> ('.').

In other than the C or POSIX locale, additional locale-specific subject sequence forms may be accepted.

If the subject sequence is empty or does not have the expected form, no

conversion shall be performed; the value of nptr shall be stored in the object pointed to by endptr, provided that endptr is not a null pointer.

These functions shall not change the setting of errno if successful.

Since 0 is returned on error and is also a valid return on success, an application wishing to check for error situations should set errno to 0, then call wcstod(), wcstof(), or wcstold(), then check errno.

RETURN VALUE

Upon successful completion, these functions shall return the converted value. If no conversion could be performed, 0 shall be returned and er? rno may be set to [EINVAL].

If the correct value is outside the range of representable values, ?HUGE_VAL, ?HUGE_VALF, or ?HUGE_VALL shall be returned (according to the sign of the value), and errno shall be set to [ERANGE].

If the correct value would cause underflow, a value whose magnitude is no greater than the smallest normalized positive number in the return type shall be returned and errno set to [ERANGE].

ERRORS

The wcstod() function shall fail if:

ERANGE The value to be returned would cause overflow or underflow.

The wcstod() function may fail if:

EINVAL No conversion could be performed.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

If the subject sequence has the hexadecimal form and FLT_RADIX is not a power of 2, and the result is not exactly representable, the result should be one of the two numbers in the appropriate internal format that are adjacent to the hexadecimal floating source value, with the extra stipulation that the error should have a correct sign for the current rounding direction.

If the subject sequence has the decimal form and at most DECIMAL_DIG

(defined in <float.h>) significant digits, the result should be cor?
rectly rounded. If the subject sequence D has the decimal form and more than DECIMAL_DIG significant digits, consider the two bounding, adja?
cent decimal strings L and U, both having DECIMAL_DIG significant dig?
its, such that the values of L, D, and U satisfy "L<=D<=U". The result should be one of the (equal or adjacent) values that would be obtained by correctly rounding L and U according to the current rounding direc?
tion, with the extra stipulation that the error with respect to D should have a correct sign for the current rounding direction.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

fscanf(), iswspace(), localeconv(), setlocale(), wcstol()

The Base Definitions volume of POSIX.1?2017, Chapter 7, Locale, <float.h>, <wchar.h>