NAME
>   strtod, strtof, strtold − convert ASCII string to floating-point number

SYNOPSIS
>   **#include <stdlib.h>**
>
>   **double strtod(const char ***nptr**, char ****endptr**);**
>   **float strtof(const char ***nptr**, char ****endptr**);**
>   **long double strtold(const char ***nptr**, char ****endptr**);**

Feature Test Macro Requirements for glibc (see **feature_test_macros**(7)):

>   **strtof**(), **strtold**():
>   _XOPEN_SOURCE >= 600 || _ISOC99_SOURCE || _POSIX_C_SOURCE >= 200112L;
>   or *cc -std=c99*

DESCRIPTION
>   The **strtod**(), **strtof**(), and **strtold**() functions convert the initial portion of the string pointed to by *nptr* to
>   *double*, *float*, and *long double* representation, respectively.
>
>   The expected form of the (initial portion of the) string is optional leading white space as recognized by **isspace**(3), an optional plus ('+') or minus sign ('−') and then either (i) a decimal number, or (ii) a hexadecimal number, or (iii) an infinity, or (iv) a NAN (not-a-number).
>
>   A *decimal number* consists of a nonempty sequence of decimal digits possibly containing a radix character (decimal point, locale-dependent, usually '.'), optionally followed by a decimal exponent. A decimal exponent consists of an 'E' or 'e', followed by an optional plus or minus sign, followed by a nonempty sequence of decimal digits, and indicates multiplication by a power of 10.
>
>   A *hexadecimal number* consists of a "0x" or "0X" followed by a nonempty sequence of hexadecimal digits possibly containing a radix character, optionally followed by a binary exponent. A binary exponent consists of a 'P' or 'p', followed by an optional plus or minus sign, followed by a nonempty sequence of decimal digits, and indicates multiplication by a power of 2. At least one of radix character and binary exponent must be present.
>
>   An *infinity* is either "INF" or "INFINITY", disregarding case.
>
>   A *NAN* is "NAN" (disregarding case) optionally followed by a string, *(n-char-sequence)*, where *n-char-sequence* specifies in an implementation-dependent way the type of NAN (see NOTES).

RETURN VALUE
>   These functions return the converted value, if any.
>
>   If *endptr* is not NULL, a pointer to the character after the last character used in the conversion is stored in the location referenced by *endptr*.
>
>   If no conversion is performed, zero is returned and the value of *nptr* is stored in the location referenced by *endptr*.
>
>   If the correct value would cause overflow, plus or minus **HUGE_VAL** (**HUGE_VALF**, **HUGE_VALL**) is returned (according to the sign of the value), and **ERANGE** is stored in *errno*. If the correct value would cause underflow, zero is returned and **ERANGE** is stored in *errno*.

ERRORS
>   **ERANGE**
>   Overflow or underflow occurred.

ATTRIBUTES

Multithreading (see pthreads(7))

The **strtod**(), **strtof**(), and **strtold**() functions are thread-safe with exceptions. These functions can be safely used in multithreaded applications, as long as **setlocale**(3) is not called to change the locale during their execution.

CONFORMING TO

C89 describes **strtod**(), C99 describes the other two functions.

NOTES

Since 0 can legitimately be returned on both success and failure, the calling program should set *errno* to 0 before the call, and then determine if an error occurred by checking whether *errno* has a nonzero value after the call.

In the glibc implementation, the *n-char-sequence* that optionally follows "NAN" is interpreted as an integer number (with an optional '0' or '0x' prefix to select base 8 or 16) that is to be placed in the mantissa component of the returned value.

EXAMPLE

See the example on the **strtol**(3) manual page; the use of the functions described in this manual page is similar.

SEE ALSO

atof(3), **atoi**(3), **atol**(3), **nan**(3), **nanf**(3), **nanl**(3), **strtol**(3), **strtoul**(3)

COLOPHON

This page is part of release 3.74 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at http://www.kernel.org/doc/man−pages/.