Rocky Enterprise Linux 9.2 Manual Pages on command 'strptime.3'

$ man strptime.3

STRPTIME(3)   Linux Programmer's Manual   STRPTIME(3)

NAME

    strptime - convert a string representation of time to a time tm struc?

    ture

SYNOPSIS

    #define _XOPEN_SOURCE   /* See feature_test_macros(7) */

    #include <time.h>

    char *strptime(const char *s, const char *format, struct tm *tm);

DESCRIPTION

    The strptime() function is the converse of strftime(3); it converts the

    character string pointed to by s to values which are stored in the

    "broken-down time" structure pointed to by tm, using the format speci?

    fied by format.

    The broken-down time structure tm is defined in <time.h> as follows:

        struct tm {

            int tm_sec;   /* Seconds (0-60) */

            int tm_min;   /* Minutes (0-59) */

            int tm_hour;   /* Hours (0-23) */

```
    int tm_mday;   /* Day of the month (1-31) */

    int tm_mon;    /* Month (0-11) */

    int tm_year;   /* Year - 1900 */

    int tm_wday;   /* Day of the week (0-6, Sunday = 0) */

    int tm_yday;   /* Day in the year (0-365, 1 Jan = 0) */

    int tm_isdst;  /* Daylight saving time */

};
```

For more details on the tm structure, see ctime(3).

The format argument is a character string that consists of field de‐
scriptors and text characters, reminiscent of scanf(3). Each field de‐
scriptor consists of a % character followed by another character that
specifies the replacement for the field descriptor. All other charac‐
ters in the format string must have a matching character in the input
string, except for whitespace, which matches zero or more whitespace
characters in the input string. There should be whitespace or other
alphanumeric characters between any two field descriptors.

The strptime() function processes the input string from left to right.
Each of the three possible input elements (whitespace, literal, or for‐
mat) are handled one after the other. If the input cannot be matched
to the format string, the function stops. The remainder of the format
and input strings are not processed.

The supported input field descriptors are listed below. In case a text
string (such as the name of a day of the week or a month name) is to be
matched, the comparison is case insensitive. In case a number is to be
matched, leading zeros are permitted but not required.

%%   The % character.

%a or %A

     The name of the day of the week according to the current locale,

     in abbreviated form or the full name.

%b or %B or %h

     The month name according to the current locale, in abbreviated

     form or the full name.

%c   The date and time representation for the current locale.

%C   The century number (0?99).

%d or %e

     The day of month (1?31).

%D   Equivalent to %m/%d/%y. (This is the American style date, very

     confusing to non-Americans, especially since %d/%m/%y is widely

     used in Europe. The ISO 8601 standard format is %Y-%m-%d.)

%H   The hour (0?23).

%I   The hour on a 12-hour clock (1?12).

%j   The day number in the year (1?366).

%m   The month number (1?12).

%M   The minute (0?59).

%n   Arbitrary whitespace.

%p   The locale's equivalent of AM or PM. (Note: there may be none.)

%r   The 12-hour clock time (using the locale's AM or PM). In the

     POSIX locale equivalent to %I:%M:%S %p. If t_fmt_ampm is empty

     in the LC_TIME part of the current locale, then the behavior is

     undefined.

%R   Equivalent to %H:%M.

%S   The second (0?60; 60 may occur for leap seconds; earlier also 61

     was allowed).

%t   Arbitrary whitespace.

%T   Equivalent to %H:%M:%S.

%U   The week number with Sunday the first day of the week (0?53).

     The first Sunday of January is the first day of week 1.

%w   The ordinal number of the day of the week (0?6), with Sunday =

     0.

%W   The week number with Monday the first day of the week (0?53).

     The first Monday of January is the first day of week 1.

%x   The date, using the locale's date format.

%X   The time, using the locale's time format.

%y   The year within century (0?99). When a century is not otherwise

     specified, values in the range 69?99 refer to years in the twen?

     tieth century (1969?1999); values in the range 00?68 refer to

years in the twenty-first century (2000?2068).

%Y   The year, including century (for example, 1991).

Some field descriptors can be modified by the E or O modifier charac?

ters to indicate that an alternative format or specification should be

used. If the alternative format or specification does not exist in the

current locale, the unmodified field descriptor is used.

The E modifier specifies that the input string may contain alternative

locale-dependent versions of the date and time representation:

%Ec   The locale's alternative date and time representation.

%EC   The name of the base year (period) in the locale's alternative

      representation.

%Ex   The locale's alternative date representation.

%EX   The locale's alternative time representation.

%Ey   The offset from %EC (year only) in the locale's alternative rep?

      resentation.

%EY   The full alternative year representation.

The O modifier specifies that the numerical input may be in an alterna?

tive locale-dependent format:

%Od or %Oe

      The day of the month using the locale's alternative numeric sym?

      bols; leading zeros are permitted but not required.

%OH   The hour (24-hour clock) using the locale's alternative numeric

      symbols.

%OI   The hour (12-hour clock) using the locale's alternative numeric

      symbols.

%Om   The month using the locale's alternative numeric symbols.

%OM   The minutes using the locale's alternative numeric symbols.

%OS   The seconds using the locale's alternative numeric symbols.

%OU   The week number of the year (Sunday as the first day of the

      week) using the locale's alternative numeric symbols.

%Ow   The ordinal number of the day of the week (Sunday=0),

      using the locale's alternative numeric symbols.

%OW   The week number of the year (Monday as the first day of the

week) using the locale's alternative numeric symbols.

%Oy   The year (offset from %C) using the locale's alternative numeric symbols.

RETURN VALUE

The return value of the function is a pointer to the first character not processed in this function call. In case the input string contains more characters than required by the format string, the return value points right after the last consumed input character. In case the whole input string is consumed, the return value points to the null byte at the end of the string. If strptime() fails to match all of the format string and therefore an error occurred, the function returns NULL.

ATTRIBUTES

For an explanation of the terms used in this section, see attributes(7).

???
?Interface ? Attribute   ? Value   ?
???
?strptime() ? Thread safety ? MT-Safe env locale ?
???

CONFORMING TO

POSIX.1-2001, POSIX.1-2008, SUSv2.

NOTES

In principle, this function does not initialize tm but stores only the values specified. This means that tm should be initialized before the call. Details differ a bit between different UNIX systems. The glibc implementation does not touch those fields which are not explicitly specified, except that it recomputes the tm_wday and tm_yday field if any of the year, month, or day elements changed.

The 'y' (year in century) specification is taken to specify a year in the range 1950?2049 by glibc 2.0. It is taken to be a year in 1969?2068 since glibc 2.1.

For reasons of symmetry, glibc tries to support for strptime() the same format characters as for strftime(3). (In most cases, the correspond‐ ing fields are parsed, but no field in tm is changed.) This leads to

%F   Equivalent to %Y-%m-%d, the ISO 8601 date format.

%g   The year corresponding to the ISO week number, but without the century (0?99).

%G   The year corresponding to the ISO week number. (For example, 1991.)

%u   The day of the week as a decimal number (1?7, where Monday = 1).

%V   The ISO 8601:1988 week number as a decimal number (1?53). If the week (starting on Monday) containing 1 January has four or more days in the new year, then it is considered week 1. Other‐ wise, it is the last week of the previous year, and the next week is week 1.

%z   An RFC-822/ISO 8601 standard timezone specification.

%Z   The timezone name.

Similarly, because of GNU extensions to strftime(3), %k is accepted as a synonym for %H, and %l should be accepted as a synonym for %I, and %P is accepted as a synonym for %p. Finally

%s   The number of seconds since the Epoch, 1970-01-01 00:00:00 +0000 (UTC). Leap seconds are not counted unless leap second support is available.

The glibc implementation does not require whitespace between two field descriptors.

EXAMPLES

The following example demonstrates the use of strptime() and strf‐ time(3).

```
#define _XOPEN_SOURCE
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
int
```

```
   main(void)

   {

      struct tm tm;

      char buf[255];

      memset(&tm, 0, sizeof(tm));

      strptime("2001-11-12 18:31:01", "%Y-%m-%d %H:%M:%S", &tm);

      strftime(buf, sizeof(buf), "%d %b %Y %H:%M", &tm);

      puts(buf);

      exit(EXIT_SUCCESS);

   }
```

SEE ALSO

   time(2), getdate(3), scanf(3), setlocale(3), strftime(3)

COLOPHON

   This page is part of release 5.10 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
   https://www.kernel.org/doc/man-pages/.

GNU   2020-11-01   STRPTIME(3)