**NAME**

getdate – convert user format date and time

**SYNOPSIS**

**#include <time.h>**

**struct tm *getdate(const char ****string***);**

**extern int getdate_err;**

**MT-LEVEL**

MT-Safe

**DESCRIPTION**

**getdate( )** converts user-definable date and/or time specifications pointed to by *string* into a **tm** structure. The **tm** structure declaration is in the **<time.h>** header file.

User-supplied templates are used to parse and interpret the input string. The templates are text files created by the user and identified via the environment variable **DATEMSK**. Each line in the template represents an acceptable date and/or time specification using conversion specifications similar to those used by **strftime**(3C) and **strptime**(3C). The first line in the template that matches the input specification is used for interpretation and conversion into the internal time format. If successful, the function **getdate( )** returns a pointer to a **tm** structure; otherwise, it returns **NULL** and sets the global variable **getdate_err** to indicate the error.

The following conversion specifications are supported:

| | |
|---|---|
| %% | same as % |
| **%a** | locale's abbreviated weekday name |
| **%A** | locale's full weekday name |
| **%b** | locale's abbreviated month name |
| **%B** | locale's full month name |
| **%c** | locale's appropriate date and time representation |
| **%C** | century number [0,99]; leading zero is permitted but not required |
| **%d** | day of month [01,31]; leading zero is permitted but not required |
| **%D** | date as **%m/%d/%y** |
| **%e** | same as **%d** |
| **%h** | locale's abbreviated month name |
| **%H** | hour (24-hour clock) [0,23]; leading zero is permitted but not required |
| **%I** | hour (12-hour clock) [1,12]; leading zero is permitted but not required |
| **%j** | day number of the year [1,366]; leading zeros are permitted but not required |
| **%m** | month number [1,12]; leading zero is permitted but not required |
| **%M** | minute [0,59]; leading zero is permitted but not required |
| **%n** | any white space |
| **%p** | locale's equivalent of either a.m. or p.m. |
| **%r** | appropriate time representation in the 12-hour clock format with **%p** |
| **%R** | time as **%H**:**%M** |
| **%S** | seconds [0,61]; leading zero is permitted but not required |
| **%t** | any white space |
| **%T** | time as **%H**:**%M**:**%S** |

**%U** week number of the year as a decimal number [0,53], with Sunday as the first day of the week; leading zero is permitted but not required
**%w** weekday as a decimal number [0,6], with 0 representing Sunday
**%W** week number of the year as a decimal number [0,53], with Monday as the first day of the week; leading zero is permitted but not required
**%x** locale's appropriate date representation
**%X** locale's appropriate time representation
**%y** year within the century [0,99]; leading zero is permitted but not required
**%Y** year, including the century (for example, 1993)
**%Z** time zone name or no characters if no time zone exists

**Modified Conversion Specifications**

Some conversion specifications can be modified by the **E** and **O** modifier characters to indicate that an alternative format or specification should be used rather than the one normally used by the unmodified specification. If the alternative format or specification does not exist in the current locale, the behaviour will be as if the unmodified conversion specification were used.

**%Ec** locale's alternative appropriate date and time representation
**%EC** name of the base year (period) in the locale's alternative representation
**%Ex** locale's alternative date representation
**%EX** locale's alternative time representation
**%Ey** offset from **%EC** (year only) in the locale's alternative representation
**%EY** full alternative year representation
**%Od** day of the month using the locale's alternative numeric symbols; leading zeros are permitted but not required
**%Oe** same as **%Od**
**%OH** hour (24-hour clock) using the locale's alternative numeric symbols
**%OI** hour (12-hour clock) using the locale's alternative numeric symbols
**%Om** month using the locale's alternative numeric symbols
**%OM** minutes using the locale's alternative numeric symbols
**%OS** seconds using the locale's alternative numeric symbols
**%OU** week number of the year (Sunday as the first day of the week) using the locale's alternative numeric symbols
**%Ow** number of the weekday (Sunday=0) using the locale's alternative numeric symbols
**%OW** week number of the year (Monday as the first day of the week) using the locale's alternative numeric symbols
**%Oy** year (offset from **%C**) in the locale's alternative representation and using the locale's alternative numeric symbols

**Internal Format Conversion**

The following rules are applied for converting the input specification into the internal format:

If only the weekday is given, today is assumed if the given day is equal to the current day and next week if it is less.

If only the month is given, the current month is assumed if the given month is equal to the current month and next year if it is less and no year is given. (The first day of month is assumed if no day is given.)

If no hour, minute, and second are given, the current hour, minute, and second are assumed.

If no date is given, today is assumed if the given hour is greater than the current hour and tomorrow is assumed if it is less.

**General Specifications**

A conversion specification that is an ordinary character is executed by scanning the next character from the buffer. If the character scanned from the buffer differs from the one comprising the conversion specification, the specification fails, and the differing and subsequent characters remain unscanned.

A series of conversion specifications composed of **%n**, **%t**, white space characters, or any combination is executed by scanning up to the first character that is not white space (which remains unscanned), or until no more characters can be scanned.

Any other conversion specification is executed by scanning characters until a character matching the next conversion specification is scanned, or until no more characters can be scanned. These characters, except the one matching the next conversion specification, are then compared to the locale values associated with the conversion specifier. If a match is found, values for the appropriate *tm* structure members are set to values corresponding to the locale information. If no match is found, **getdate( )** fails and no more characters are scanned.

The month names, weekday names, era names, and alternative numeric symbols can consist of any combination of upper and lower case letters. The user can request that the input date or time specification be in a specific language by setting the **LC_TIME** category using **setlocale**(3C).

## RETURN VALUES

On failure **getdate( )** returns **NULL** and sets the variable **getdate_err** to indicate the error.

The following is a complete list of the **getdate_err** settings and their meanings.

| | |
|---|---|
| **1** | The **DATEMSK** environment variable is null or undefined. |
| **2** | The template file cannot be opened for reading. |
| **3** | Failed to get file status information. |
| **4** | The template file is not a regular file. |
| **5** | An error is encountered while reading the template file. |
| **6** | **malloc( )** failed (not enough memory is available). |
| **7** | There is no line in the template that matches the input. |
| **8** | The input specification is invalid (for example, **February 31**). |

**EXAMPLES**

The following example shows the possible contents of a template:

**%m**
**%A %B %d %Y, %H:%M:%S**
**%A**
**%B**
**%m/%d/%y %I %p**
**%d,%m,%Y %H:%M**
**at %A the %dst of %B in %Y**
**run job at %I %p,%B %dnd**
**%A den %d. %B %Y %H.%M Uhr**

The following are examples of valid input specifications for the above template:

**getdate("10/1/87 4 PM")**
**getdate("Friday")**
**getdate("Friday September 19 1987, 10:30:30")**
**getdate("24,9,1986 10:30")**
**getdate("at monday the 1st of december in 1986")**
**getdate("run job at 3 PM, december 2nd")**

If the **LANG** environment variable is set to **de** (German), the following is valid:

**getdate("freitag den 10. oktober 1986 10.30 Uhr")**

Local time and date specification are also supported. The following examples show how local date and time specification can be defined in the template.

| Invocation | Line in Template |
|---|---|
| **getdate("11/27/86")** | **%m/%d/%y** |
| **getdate("27.11.86")** | **%d.%m.%y** |
| **getdate("86-11-27")** | **%y-%m-%d** |
| **getdate("Friday 12:00:00")** | **%A %H:%M:%S** |

The following examples illustrate the Internal Format Conversion rules. Assume that the current date is Mon Sep 22 12:19:47 EDT 1986 and the **LANG** environment variable is not set.

| Input | Line in Template | Date |
|---|---|---|
| **Mon** | **%a** | Mon Sep 22 12:19:48 EDT 1986 |
| **Sun** | **%a** | Sun Sep 28 12:19:49 EDT 1986 |
| **Fri** | **%a** | Fri Sep 26 12:19:49 EDT 1986 |
| **September** | **%B** | Mon Sep 1 12:19:49 EDT 1986 |
| **January** | **%B** | Thu Jan 1 12:19:49 EST 1987 |
| **December** | **%B** | Mon Dec 1 12:19:49 EST 1986 |
| **Sep Mon** | **%b %a** | Mon Sep 1 12:19:50 EDT 1986 |
| **Jan Fri** | **%b %a** | Fri Jan 2 12:19:50 EST 1987 |
| **Dec Mon** | **%b %a** | Mon Dec 1 12:19:50 EST 1986 |
| **Jan Wed 1989** | **%b %a %Y** | Wed Jan 4 12:19:51 EST 1989 |
| **Fri 9** | **%a %H** | Fri Sep 26 09:00:00 EDT 1986 |
| **Feb 10:30** | **%b %H:%S** | Sun Feb 1 10:00:30 EST 1987 |
| **10:30** | **%H:%M** | Tue Sep 23 10:30:00 EDT 1986 |
| **13:30** | **%H:%M** | Mon Sep 22 13:30:00 EDT 1986 |

## FILES

**/usr/lib/locale/*locale*/LC_TIME/time**
locale specific date and time information

**/usr/lib/locale/*locale*/LC_CTYPE/ctype**
character characterization information

## SEE ALSO

**setlocale**(3C), **strftime**(3C), **strptime**(3C), **environ**(5)

## NOTES

Subsequent calls to **getdate( )** alter the contents of **getdate_err**.

Dates before 1970 and after 2037 are illegal.

The range of values for **%S** is [00,61] rather than [00,59] to allow for the occasional leap second and even more occasional double leap second.

**getdate( )** makes explicit use of macros described in **ctype**(3C).