NAME

tzset, tzname, timezone, daylight - initialize time conversion information

SYNOPSIS

#include <time.h>

void tzset (void);

extern char ****tzname*[2**];
extern long *timezone*;
extern int *daylight*;

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

tzset(): _POSIX_C_SOURCE >= 1 || _XOPEN_SOURCE || _POSIX_SOURCE
tzname: _POSIX_C_SOURCE >= 1 || _XOPEN_SOURCE || _POSIX_SOURCE
timezone: _SVID_SOURCE || _XOPEN_SOURCE
daylight: _SVID_SOURCE || _XOPEN_SOURCE

DESCRIPTION

The **tzset**() function initializes the *tzname* variable from the **TZ** environment variable. This function is automatically called by the other time conversion functions that depend on the timezone. In a System-V-like environment, it will also set the variables *timezone* (seconds West of UTC) and *daylight* (to 0 if this timezone does not have any daylight saving time rules, or to nonzero if there is a time during the year when daylight saving time applies).

If the **TZ** variable does not appear in the environment, the *tzname* variable is initialized with the best approximation of local wall clock time, as specified by the tzfile(5)-format file *localtime* found in the system timezone directory (see below). (One also often sees */etc/lo caltime* used here, a symlink to the right file in the system timezone directory.)

If the **TZ** variable does appear in the environment but its value is empty or its value cannot be interpreted using any of the formats specified below, Coordinated Universal Time (UTC) is used.

The value of **TZ** can be one of three formats. The first format is used when there is no daylight saving time in the local timezone:

std offset

The *std* string specifies the name of the timezone and must be three or more alphabetic characters. The*offset* string immediately follo ws *std* and specifies the time value to be added to the local time to get Coordinated Universal Time (UTC). The *offset* is positive if the local timezone is west of the Prime Meridian and negative if it is east. The hour must be between 0 and 24, and the minutes and seconds 0 and 59.

The second format is used when there is daylight saving time:

std offset dst [offset],start[/time],end[/time]

There are no spaces in the specification. The initial *std* and *offset* specify the standard timezone, as described above. The*dst* string and *offset* specify the name and offset for the corresponding daylight saving timezone. If the offset is omitted, it default to one hour ahead of standard time.

The *start* field specifies when daylight saving time goes into effect and the *end* field specifies when the change is made back to standard time. These fields may have the following formats:

J*n* This specifies the Julian day with *n* between 1 and 365. Leap days are not counted. In this format, February 29 can't be represented; February 28 is day 59, and March 1 is always day 60.

n This specifies the zero-based Julian day with *n* between 0 and 365. February 29 is counted in leap years.

M*m.w.d*

This specifies day *d* (0 <= *d* <= 6) of week *w* (1 <= *w* <= 5) of month *m* (1 <= *m* <= 12). Week 1 is the first week in which day *d* occurs and week 5 is the last week in which day *d* occurs. Day 0 is a Sunday.

The *time* fields specify when, in the local time currently in effect, the change to the other time occurs. If omitted, the default is 02:00:00.

Here is an example for New Zealand, where the standard time (NZST) is 12 hours ahead of UTC, and daylight saving time (NZDT), 13 hours ahead of UTC, runs from the first Sunday in October to the third Sunday in March, and the changeovers happen at the default time of 02:00:00:

TZ=NZST-12:00:00NZDT-13:00:00,M10.1.0,M3.3.0

The third format specifies that the timezone information should be read from a file:

:[filespec]

If the file specification *filespec* is omitted, the timezone information is read from the file *localtime* in the system timezone directory, which nowadays usually is */usr/share/zoneinfo*. This file is in tzfile(5) format. If *filespec* is given, it specifies another tzfile(5)-format file to read the timezone information from. If *filespec* does not begin with a /, the file specification is relative to the system timezone directory.

Here's an example, once more for New Zealand:

TZ=:Pacific/Auckland

FILES

Under glibc, the system timezone directory is determined using the **TZDIR** the environment variable. If**TZDIR** the default depends on the system setup, but is normally */usr/shar e/zone-info*.

This timezone directory contains the files

localtime local timezone file
posixrules rules for POSIX-style TZ's

Often, */etc/localtime* is a symbolic link to the file *localtime* or to the correct timezone file in the system timezone directory.

CONFORMING TO

SVr4, POSIX.1-2001, 4.3BSD.

NOTES

Note that the variable *daylight* does not indicate that daylight saving time applies right now. It used to give the number of some algorithm (see the variable *tz_dsttime* in gettimeofday(2)). It has been obsolete for many years but is required by SUSv2.

4.3BSD had a function **char *timezone(***zone***,** *dst***)** that returned the name of the timezone corresponding to its first argument (minutes West of UTC). If the second argument was 0, the standard name was used, otherwise the daylight saving time version.

SEE ALSO

date(1), gettimeofday(2), time(2), ctime(3), getenv(3), tzfile(5)

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/.