NAME

gettimeofday, settimeofday − get / set time

SYNOPSIS

#include <sys/time.h>

int gettimeofday(struct timeval *_tv_, struct timezone *_tz_);

int settimeofday(const struct timeval *_tv_, const struct timezone *_tz_);

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

settimeofday(): _BSD_SOURCE

DESCRIPTION

The functions **gettimeofday**() and **settimeofday**() can get and set the time as well as a timezone. The *tv* argument is a *struct timeval* (as specified in *<sys/time.h>*):

```
struct timeval {
    time_t      tv_sec;     /* seconds */
    suseconds_t tv_usec;    /* microseconds */
};
```

and gives the number of seconds and microseconds since the Epoch (see **time**(2)). The *tz* argument is a *struct timezone*:

```
struct timezone {
    int tz_minuteswest;     /* minutes west of Greenwich */
    int tz_dsttime;         /* type of DST correction */
};
```

If either *tv* or *tz* is NULL, the corresponding structure is not set or returned.

The use of the *timezone* structure is obsolete; the *tz* argument should normally be specified as NULL. The *tz_dsttime* field has never been used under Linux; it has not been and will not be supported by libc or glibc. Each and every occurrence of this field in the kernel source (other than the declaration) is a bug. Thus, the following is purely of historic interest.

The field *tz_dsttime* contains a symbolic constant (values are given below) that indicates in which part of the year Daylight Saving Time is in force. (Note: its value is constant throughout the year: it does not indicate that DST is in force, it just selects an algorithm.) The daylight saving time algorithms defined are as follows :

```
DST_NONE    /* not on dst */
DST_USA     /* USA style dst */
DST_AUST    /* Australian style dst */
DST_WET     /* Western European dst */
DST_MET     /* Middle European dst */
DST_EET     /* Eastern European dst */
DST_CAN     /* Canada */
DST_GB      /* Great Britain and Eire */
DST_RUM     /* Rumania */
DST_TUR     /* Turkey */
DST_AUSTALT /* Australian style with shift in 1986 */
```

Of course it turned out that the period in which Daylight Saving Time is in force cannot be given by a simple algorithm, one per country; indeed, this period is determined by unpredictable political decisions. So

this method of representing timezones has been abandoned. Under Linux, in a call to **settimeofday**() the *tz_dsttime* field should be zero.

Under Linux there are some peculiar "warp clock" semantics associated with the **settimeofday**() system call if on the very first call (after booting) that has a non-NULL *tz* argument, the *tv* argument is NULL and the *tz_minuteswest* field is non-zero. In such a case it is assumed that the CMOS clock is on local time, and that it has to be incremented by this amount to get UTC system time. No doubt it is a bad idea to use this feature.

Macros for operating on *timeval* structures are described in **timeradd**(3).

RETURN VALUE

gettimeofday() and **settimeofday**() return 0 for success, or −1 for failure (in which case *errno* is set appropriately).

ERRORS

EFAULT
One of *tv* or *tz* pointed outside the accessible address space.

EINVAL
Timezone (or something else) is invalid.

EPERM
The calling process has insufficient privilege to call **settimeofday**(); under Linux the **CAP_SYS_TIME** capability is required.

CONFORMING TO

SVr4, 4.3BSD. POSIX.1-2001 describes **gettimeofday**() but not **settimeofday**(). POSIX.1-2008 marks **gettimeofday**() as obsolete, recomending the use of **clock_gettime**(2) instead.

NOTES

Traditionally, the fields of *struct timeval* were of type *long*.

SEE ALSO

date(1), **adjtimex**(2), **time**(2), **ctime**(3), **ftime**(3), **capabilities**(7), **time**(7)

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.