Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'time.h.0p' command

$ man time.h.0p

time.h(0P) POSIX Programmer's Manual time.h(0P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

time.h ? time types

SYNOPSIS

#include <time.h>

DESCRIPTION

Some of the functionality described on this reference page extends the ISO C standard. Applications shall define the appropriate feature test macro (see the System Interfaces volume of POSIX.1?2017, Section 2.2, The Compilation Environment) to enable the visibility of these symbols in this header.

The <time.h> header shall define the clock_t, size_t, time_t, types as described in <sys/types.h>.

The <time.h> header shall define the clockid_t and timer_t types as de? scribed in <sys/types.h>.

The <time.h> header shall define the locale_t type as described in <lo? cale.h>.

The <time.h> header shall define the pid_t type as described in

<sys/types.h>.

The tag sigevent shall be declared as naming an incomplete structure type, the contents of which are described in the <signal.h> header.

The <time.h> header shall declare the tm structure, which shall include at least the following members:

int tm_sec Seconds [0,60].

int tm_min Minutes [0,59].

int tm_hour Hour [0,23].

int tm_mday Day of month [1,31].

int tm_mon Month of year [0,11].

int tm_year Years since 1900.

int tm_wday Day of week [0,6] (Sunday =0).

int tm_yday Day of year [0,365].

int tm_isdst Daylight Savings flag.

The value of tm_isdst shall be positive if Daylight Savings Time is in effect, 0 if Daylight Savings Time is not in effect, and negative if the information is not available.

The <time.h> header shall declare the timespec structure, which shall include at least the following members:

time_t tv_sec Seconds.

long tv_nsec Nanoseconds.

The <time.h> header shall also declare the itimerspec structure, which shall include at least the following members:

struct timespec it_interval Timer period.

struct timespec it_value Timer expiration.

The <time.h> header shall define the following macros:

NULL As described in <stddef.h>.

CLOCKS_PER_SEC

A number used to convert the value returned by the clock() function into seconds. The value shall be an ex? pression with type clock_t. The value of CLOCKS_PER_SEC shall be 1 million on XSI-conformant systems. However, it may be variable on other systems, and it should not be

assumed that CLOCKS_PER_SEC is a compile-time constant.

The <time.h> header shall define the following symbolic constants. The values shall have a type that is assignment-compatible with clockid_t.

CLOCK_MONOTONIC

The identifier for the system-wide monotonic clock, which is defined as a clock measuring real time, whose value cannot be set via clock_settime() and which cannot have negative clock jumps. The maximum possible clock jump shall be implementation-defined.

CLOCK_PROCESS_CPUTIME_ID

The identifier of the CPU-time clock associated with the process making a clock() or timer*() function call.

CLOCK_REALTIME

The identifier of the system-wide clock measuring real time.

CLOCK_THREAD_CPUTIME_ID

The identifier of the CPU-time clock associated with the thread making a clock() or timer*() function call.

The <time.h> header shall define the following symbolic constant:

TIMER_ABSTIME Flag indicating time is absolute. For functions taking timer objects, this refers to the clock associated with the timer.

The <time.h> header shall provide a declaration or definition for get? date_err. The getdate_err symbol shall expand to an expression of type int. It is unspecified whether getdate_err is a macro or an identifier declared with external linkage, and whether or not it is a modifiable lvalue. If a macro definition is suppressed in order to access an ac? tual object, or a program defines an identifier with the name get? date_err, the behavior is undefined.

The following shall be declared as functions and may also be defined as macros. Function prototypes shall be provided.

```
char      *asctime(const struct tm *);
char      *asctime_r(const struct tm *restrict, char *restrict);
```

```
clock_t    clock(void);
int        clock_getcpuclockid(pid_t, clockid_t *);
int        clock_getres(clockid_t, struct timespec *);
int        clock_gettime(clockid_t, struct timespec *);
int        clock_nanosleep(clockid_t, int, const struct timespec *,
               struct timespec *);
int        clock_settime(clockid_t, const struct timespec *);
char      *ctime(const time_t *);
char      *ctime_r(const time_t *, char *);
double     difftime(time_t, time_t);
struct tm *getdate(const char *);
struct tm *gmtime(const time_t *);
struct tm *gmtime_r(const time_t *restrict, struct tm *restrict);
struct tm *localtime(const time_t *);
struct tm *localtime_r(const time_t *restrict, struct tm *restrict);
time_t     mktime(struct tm *);
int        nanosleep(const struct timespec *, struct timespec *);
size_t     strftime(char *restrict, size_t, const char *restrict,
           const struct tm *restrict);
size_t     strftime_l(char *restrict, size_t, const char *restrict,
               const struct tm *restrict, locale_t);
char      *strptime(const char *restrict, const char *restrict,
               struct tm *restrict);
time_t     time(time_t *);
int        timer_create(clockid_t, struct sigevent *restrict,
               timer_t *restrict);
int        timer_delete(timer_t);
int        timer_getoverrun(timer_t);
int        timer_gettime(timer_t, struct itimerspec *);
int        timer_settime(timer_t, int, const struct itimerspec *restrict,
               struct itimerspec *restrict);
void       tzset(void);
```

The <time.h> header shall declare the following as variables:

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

Inclusion of the <time.h> header may make visible all symbols from the <signal.h> header.

The following sections are informative.

APPLICATION USAGE

The range [0,60] for tm_sec allows for the occasional leap second.

tm_year is a signed value; therefore, years before 1900 may be repre? sented.

To obtain the number of clock ticks per second returned by the times() function, applications should call sysconf(_SC_CLK_TCK).

RATIONALE

The range [0,60] seconds allows for positive or negative leap seconds.

The formal definition of UTC does not permit double leap seconds, so all mention of double leap seconds has been removed, and the range shortened from the former [0,61] seconds seen in earlier versions of this standard.

FUTURE DIRECTIONS

None.

SEE ALSO

<locale.h>, <signal.h>, <stddef.h>, <sys_types.h>

The System Interfaces volume of POSIX.1?2017, Section 2.2, The Compila? tion Environment, asctime(), clock(), clock_getcpuclockid(), clock_ge? tres(), clock_nanosleep(), ctime(), difftime(), getdate(), gmtime(), localtime(), mktime(), mq_receive(), mq_send(), nanosleep(), pthread_getcpuclockid(), pthread_mutex_timedlock(), pthread_rwlock_timedrdlock(), pthread_rwlock_timedwrlock(), sem_timed? wait(), strftime(), strptime(), sysconf(), time(), timer_create(), timer_delete(), timer_getoverrun(), tzset(), utime()

COPYRIGHT

table Operating System Interface (POSIX), The Open Group Base Specifi? cations Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at http://www.opengroup.org/unix/online.html .

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.ker? nel.org/doc/man-pages/reporting_bugs.html .