PYTHON [time](https://docs.python.org/3/library/time.html#module-time) MODULE

This module provides various time-related functions. For related functionality, see also the [datetime](https://docs.python.org/3/library/datetime.html#module-datetime) and [calendar](https://docs.python.org/3/library/calendar.html#module-calendar) modules.

Although this module is always available, not all functions are available on all platforms. Most of the functions defined in this module call platform C library functions with the same name. It may sometimes be helpful to consult the platform documentation, because the semantics of these functions varies among platforms.

* the precision of [time()](https://docs.python.org/3/library/time.html#time.time) and [sleep()](https://docs.python.org/3/library/time.html#time.sleep) is better than their Unix equivalents: times are expressed as floating point numbers, [time()](https://docs.python.org/3/library/time.html#time.time) returns the most accurate time available (using Unix gettimeofday() where available), and [sleep()](https://docs.python.org/3/library/time.html#time.sleep) will accept a time with a nonzero fraction (Unix select() is used to implement this, where available).
* The time value as returned by [gmtime()](https://docs.python.org/3/library/time.html" \l "time.gmtime" \o "time.gmtime), [localtime()](https://docs.python.org/3/library/time.html" \l "time.localtime" \o "time.localtime), and [strptime()](https://docs.python.org/3/library/time.html" \l "time.strptime" \o "time.strptime), and accepted by [asctime()](https://docs.python.org/3/library/time.html" \l "time.asctime" \o "time.asctime), [mktime()](https://docs.python.org/3/library/time.html" \l "time.mktime" \o "time.mktime) and [strftime()](https://docs.python.org/3/library/time.html#time.strftime), is a sequence of 9 integers. The return values of [gmtime()](https://docs.python.org/3/library/time.html" \l "time.gmtime" \o "time.gmtime), [localtime()](https://docs.python.org/3/library/time.html" \l "time.localtime" \o "time.localtime), and [strptime()](https://docs.python.org/3/library/time.html" \l "time.strptime" \o "time.strptime) also offer attribute names for individual fields.

See [struct\_time](https://docs.python.org/3/library/time.html" \l "time.struct_time" \o "time.struct_time) for a description of these objects.

*Changed in version 3.3:*The [struct\_time](https://docs.python.org/3/library/time.html#time.struct_time) type was extended to provide the tm\_gmtoff and tm\_zone attributes when platform supports corresponding struct tm members.

*Changed in version 3.6:*The [struct\_time](https://docs.python.org/3/library/time.html" \l "time.struct_time" \o "time.struct_time) attributes tm\_gmtoff and tm\_zone are now available on all platforms.

* Use the following functions to convert between time representations:

| **From** | **To** | **Use** |
| --- | --- | --- |
| seconds since the epoch | [struct\_time](https://docs.python.org/3/library/time.html#time.struct_time) in UTC | [gmtime()](https://docs.python.org/3/library/time.html#time.gmtime) |
| seconds since the epoch | [struct\_time](https://docs.python.org/3/library/time.html#time.struct_time) in local time | [localtime()](https://docs.python.org/3/library/time.html#time.localtime) |
| [struct\_time](https://docs.python.org/3/library/time.html#time.struct_time) in UTC | seconds since the epoch | [calendar.timegm()](https://docs.python.org/3/library/calendar.html#calendar.timegm) |
| [struct\_time](https://docs.python.org/3/library/time.html#time.struct_time) in local time | seconds since the epoch | [mktime()](https://docs.python.org/3/library/time.html#time.mktime) |

Functions

time.**asctime**([*t*])

Convert a tuple or [struct\_time](https://docs.python.org/3/library/time.html" \l "time.struct_time" \o "time.struct_time) representing a time as returned by [gmtime()](https://docs.python.org/3/library/time.html" \l "time.gmtime" \o "time.gmtime) or [localtime()](https://docs.python.org/3/library/time.html" \l "time.localtime" \o "time.localtime) to a string of the following form: 'Sun Jun 20 23:21:05 1993'. The day field is two characters long and is space padded if the day is a single digit, e.g.: 'Wed Jun  9 04:26:40 1993'.

If *t* is not provided, the current time as returned by [localtime()](https://docs.python.org/3/library/time.html" \l "time.localtime" \o "time.localtime) is used. Locale information is not used by [asctime()](https://docs.python.org/3/library/time.html" \l "time.asctime" \o "time.asctime).

**Note**

Unlike the C function of the same name, [asctime()](https://docs.python.org/3/library/time.html" \l "time.asctime" \o "time.asctime) does not add a trailing newline.

time.**pthread\_getcpuclockid**(*thread\_id*)

Return the *clk\_id* of the thread-specific CPU-time clock for the specified *thread\_id*.

Use [threading.get\_ident()](https://docs.python.org/3/library/threading.html" \l "threading.get_ident" \o "threading.get_ident) or the [ident](https://docs.python.org/3/library/threading.html#threading.Thread.ident) attribute of [threading.Thread](https://docs.python.org/3/library/threading.html" \l "threading.Thread" \o "threading.Thread) objects to get a suitable value for *thread\_id*.

**Warning**

Passing an invalid or expired *thread\_id* may result in undefined behavior, such as segmentation fault.

[Availability](https://docs.python.org/3/library/intro.html#availability): Unix (see the man page for *[pthread\_getcpuclockid(3)](https://manpages.debian.org/pthread_getcpuclockid(3))* for further information).

*New in version 3.7.*

time.**clock\_getres**(*clk\_id*)

Return the resolution (precision) of the specified clock *clk\_id*. Refer to [Clock ID Constants](https://docs.python.org/3/library/time.html#time-clock-id-constants) for a list of accepted values for *clk\_id*.

[Availability](https://docs.python.org/3/library/intro.html#availability): Unix.

*New in version 3.3.*

time.**clock\_gettime**(*clk\_id*) → float

Return the time of the specified clock *clk\_id*. Refer to [Clock ID Constants](https://docs.python.org/3/library/time.html#time-clock-id-constants) for a list of accepted values for *clk\_id*.

[Availability](https://docs.python.org/3/library/intro.html#availability): Unix.

*New in version 3.3.*

time.**clock\_gettime\_ns**(*clk\_id*) → int

Similar to [clock\_gettime()](https://docs.python.org/3/library/time.html" \l "time.clock_gettime" \o "time.clock_gettime) but return time as nanoseconds.

[Availability](https://docs.python.org/3/library/intro.html#availability): Unix.

*New in version 3.7.*

time.**clock\_settime**(*clk\_id*, *time: float*)

Set the time of the specified clock *clk\_id*. Currently, [CLOCK\_REALTIME](https://docs.python.org/3/library/time.html#time.CLOCK_REALTIME) is the only accepted value for *clk\_id*.

[Availability](https://docs.python.org/3/library/intro.html#availability): Unix.

*New in version 3.3.*

time.**clock\_settime\_ns**(*clk\_id*, *time: int*)

Similar to [clock\_settime()](https://docs.python.org/3/library/time.html" \l "time.clock_settime" \o "time.clock_settime) but set time with nanoseconds.

[Availability](https://docs.python.org/3/library/intro.html#availability): Unix.

*New in version 3.7.*

time.**ctime**([*secs*])

Convert a time expressed in seconds since the epoch to a string of a form: 'Sun Jun 20 23:21:05 1993' representing local time. The day field is two characters long and is space padded if the day is a single digit, e.g.: 'Wed Jun  9 04:26:40 1993'.

If *secs* is not provided or [None](https://docs.python.org/3/library/constants.html#None), the current time as returned by [time()](https://docs.python.org/3/library/time.html#time.time) is used. ctime(secs) is equivalent to asctime(localtime(secs)). Locale information is not used by [ctime()](https://docs.python.org/3/library/time.html" \l "time.ctime" \o "time.ctime).

time.**get\_clock\_info**(*name*)

Get information on the specified clock as a namespace object. Supported clock names and the corresponding functions to read their value are:

* 'monotonic': [time.monotonic()](https://docs.python.org/3/library/time.html" \l "time.monotonic" \o "time.monotonic)
* 'perf\_counter': [time.perf\_counter()](https://docs.python.org/3/library/time.html" \l "time.perf_counter" \o "time.perf_counter)
* 'process\_time': [time.process\_time()](https://docs.python.org/3/library/time.html" \l "time.process_time" \o "time.process_time)
* 'thread\_time': [time.thread\_time()](https://docs.python.org/3/library/time.html" \l "time.thread_time" \o "time.thread_time)
* 'time': [time.time()](https://docs.python.org/3/library/time.html" \l "time.time" \o "time.time)

The result has the following attributes:

* *adjustable*: True if the clock can be changed automatically (e.g. by a NTP daemon) or manually by the system administrator, False otherwise
* *implementation*: The name of the underlying C function used to get the clock value. Refer to [Clock ID Constants](https://docs.python.org/3/library/time.html#time-clock-id-constants) for possible values.
* *monotonic*: True if the clock cannot go backward, False otherwise
* *resolution*: The resolution of the clock in seconds ([float](https://docs.python.org/3/library/functions.html#float))

*New in version 3.3.*

time.**gmtime**([*secs*])

Convert a time expressed in seconds since the epoch to a [struct\_time](https://docs.python.org/3/library/time.html" \l "time.struct_time" \o "time.struct_time) in UTC in which the dst flag is always zero. If *secs* is not provided or [None](https://docs.python.org/3/library/constants.html#None), the current time as returned by [time()](https://docs.python.org/3/library/time.html#time.time) is used. Fractions of a second are ignored. See above for a description of the [struct\_time](https://docs.python.org/3/library/time.html" \l "time.struct_time" \o "time.struct_time) object. See [calendar.timegm()](https://docs.python.org/3/library/calendar.html" \l "calendar.timegm" \o "calendar.timegm) for the inverse of this function.

time.**localtime**([*secs*])

Like [gmtime()](https://docs.python.org/3/library/time.html" \l "time.gmtime" \o "time.gmtime) but converts to local time. If *secs* is not provided or [None](https://docs.python.org/3/library/constants.html#None), the current time as returned by [time()](https://docs.python.org/3/library/time.html#time.time) is used. The dst flag is set to 1 when DST applies to the given time.

time.**mktime**(*t*)

This is the inverse function of [localtime()](https://docs.python.org/3/library/time.html" \l "time.localtime" \o "time.localtime). Its argument is the [struct\_time](https://docs.python.org/3/library/time.html" \l "time.struct_time" \o "time.struct_time) or full 9-tuple (since the dst flag is needed; use -1 as the dst flag if it is unknown) which expresses the time in *local* time, not UTC. It returns a floating point number, for compatibility with [time()](https://docs.python.org/3/library/time.html#time.time). If the input value cannot be represented as a valid time, either [OverflowError](https://docs.python.org/3/library/exceptions.html" \l "OverflowError" \o "OverflowError) or [ValueError](https://docs.python.org/3/library/exceptions.html" \l "ValueError" \o "ValueError) will be raised (which depends on whether the invalid value is caught by Python or the underlying C libraries). The earliest date for which it can generate a time is platform-dependent.

time.**monotonic**() → float

Return the value (in fractional seconds) of a monotonic clock, i.e. a clock that cannot go backwards. The clock is not affected by system clock updates. The reference point of the returned value is undefined, so that only the difference between the results of two calls is valid.

*New in version 3.3.*

*Changed in version 3.5:*The function is now always available and always system-wide.

time.**monotonic\_ns**() → int

Similar to [monotonic()](https://docs.python.org/3/library/time.html#time.monotonic), but return time as nanoseconds.

*New in version 3.7.*

time.**perf\_counter**(→ float

Return the value (in fractional seconds) of a performance counter, i.e. a clock with the highest available resolution to measure a short duration. It does include time elapsed during sleep and is system-wide. The reference point of the returned value is undefined, so that only the difference between the results of two calls is valid.

*New in version 3.3.*

time.**perf\_counter\_ns**() → int

Similar to [perf\_counter()](https://docs.python.org/3/library/time.html" \l "time.perf_counter" \o "time.perf_counter), but return time as nanoseconds.

*New in version 3.7.*

time.**process\_time**() → float

Return the value (in fractional seconds) of the sum of the system and user CPU time of the current process. It does not include time elapsed during sleep. It is process-wide by definition. The reference point of the returned value is undefined, so that only the difference between the results of two calls is valid.

*New in version 3.3.*

time.**process\_time\_ns**() → int

Similar to [process\_time()](https://docs.python.org/3/library/time.html" \l "time.process_time" \o "time.process_time) but return time as nanoseconds.

*New in version 3.7.*

time.**sleep**(*secs*)

Suspend execution of the calling thread for the given number of seconds. The argument may be a floating point number to indicate a more precise sleep time. The actual suspension time may be less than that requested because any caught signal will terminate the [sleep()](https://docs.python.org/3/library/time.html#time.sleep) following execution of that signal’s catching routine. Also, the suspension time may be longer than requested by an arbitrary amount because of the scheduling of other activity in the system.

*Changed in version 3.5:*The function now sleeps at least *secs* even if the sleep is interrupted by a signal, except if the signal handler raises an exception (see [**PEP 475**](https://www.python.org/dev/peps/pep-0475) for the rationale).

**strftime**(*format*[, *t*])

Convert a tuple or [struct\_time](https://docs.python.org/3/library/time.html" \l "time.struct_time" \o "time.struct_time) representing a time as returned by [gmtime()](https://docs.python.org/3/library/time.html" \l "time.gmtime" \o "time.gmtime) or [localtime()](https://docs.python.org/3/library/time.html" \l "time.localtime" \o "time.localtime) to a string as specified by the *format* argument. If *t* is not provided, the current time as returned by [localtime()](https://docs.python.org/3/library/time.html" \l "time.localtime" \o "time.localtime) is used. *format* must be a string. [ValueError](https://docs.python.org/3/library/exceptions.html" \l "ValueError" \o "ValueError) is raised if any field in *t* is outside of the allowed range.