Date and Time

Following modules are used to work with date, time, and duration.

- time
- datetime

- Epoch - The epoch is the point where the time starts, and is platform dependent. This point is taken as the January 1st of the current year, 00:00:00. For Unix, the epoch is January 1st 1970, 00:00:00 (UTC)

- UTC - UTC is Coordinated Universal Time (formerly known as Greenwich Mean Time, or GMT). The acronym UTC is not a mistake but a compromise between English and French.

- DST - DST is Daylight Saving Time, an adjustment of the timezone by (usually) one hour during part of the year. DST rules are magic (determined by local law) and can change from year to year.

Time Modules

- time ( ) Function – This function return the time in seconds since the epoch as a floating point number. The specific date of the epoch and the handling of leap seconds is platform dependent.

- ctime ( ) Function – This function is used to get current date and time. When we pass epoch time in seconds to the function, it returns corresponding date and time in string format. When we do not pass epoch time, it returns current date and time in string format.

Time Modules

- localtime ( ) Function – This function is used to convert seconds into date and time. It returns an object *struct_time* which can be used to access the attributes either using an index or using a name.

Index	Attribute	Value
0	tm_year	4 digit year number e.g. 2019
1	tm_mon	Range [1, 12]
2	tm_mday	Range [1, 31]
3	tm_hour	Range [0, 23]
4	tm_min	Range [0, 59]
5	tm_sec	Range [0, 61], including leap seconds
6	tm_wday	Range [0, 6], Monday is 0
7	tm_yday	Range [1, 366]
8	tm_isdst	[0, 1 or -1], 0 = no DST, 1 = DST is in effect, -1 = not known
	tm_zone	Timezone name
	tm_gmtoff	Offset east of UTC in seconds

datetime Module

datetime – It handles date and time. It has year, month, day, hour, minute, second, microsecond and tzinfo attributes

date – It handles dates of gregorian calendar, without taking time zone into consideration. It has year, month and day attributes.

time – It handles time assuming that every day has exactly 24 x 60 x 60 seconds. It has hour, minute, second, microsecond and tzinfo attributes.

timedelta – It handles durations. The duration may be the difference between two date, time or datetime instances.

datetime class

datetime object - A datetime object is a single object containing all the information from a date object and a time object.

Creating Object of datetime Class

object_name = datetime(year, month, day, hour=0, minute=0, second=0, microsecond=0, tzinfo=None, *, fold=0)

The year, month and day arguments are required. tzinfo may be None, or an instance of a tzinfo subclass. The remaining arguments may be integers, in the following ranges:

MINYEAR <= year <= MAXYEAR,

1 <= month <= 12,

1 <= day <= number of days in the given month and year,

0 <= hour < 24,

0 <= minute < 60,

0 <= second < 60,

0 <= microsecond < 1000000,

fold in [0, 1].

Ex:-
```
dt = datetime(year=2019, month=6, day=30, hour=5, minute=34)
```

The **fold** parameter specifies whether there was any fold in time. A fold in time means a reverse back of the clock time. In countries following Daylight Saving time during the end of summer clocks are reversed back by 1 hour. This reverse back is a fold in time.

* means a splat operator. Using a splat operator a tuple can be unpacked and a time object can be constructed out of the values from the tuple.

datetime class’s Methods

now() – This method is used to get the current date and time. We can provide timezone information to this method. If the timezone is not provided, then it takes the local time zone. It returns an object that contains date and time information in any timezone. We can use day, month, year, hour, minute and second.

Ex:- datetime.now()

today() – This method is used to get the current date and time. It returns the date and time information.

Ex:- datetime.today()

date class

date object - A date object is an object containing information of year, month and day

Creating Object of date Class

object_name = date(year, month, day)

All arguments are required. Arguments may be integers, in the following ranges:

MINYEAR <= year <= MAXYEAR

1 <= month <= 12

1 <= day <= number of days in the given month and year

Ex:-

d = date(year=2019, month=6, day=30)

date class’s Method

today() Method – This method is used to get the current date. It returns only date.

Ex:- date.today()

time class

time object - A time object is an object containing information of local time of day, independent of any particular day, and subject to adjustment via a tzinfo object.

Creating Object of time Class

object_name = time(hour=0, minute=0, second=0, microsecond=0, tzinfo=None,*, fold=0)

All arguments are optional. tzinfo may be None, or an instance of a tzinfo subclass. The remaining arguments may be integers, in the following ranges:

0 <= hour < 24,

0 <= minute < 60,

0 <= second < 60,

0 <= microsecond < 1000000,

fold in [0, 1].

All default to 0 except tzinfo, which defaults to None.

Ex:-

t = time(hour=5, minute=34, second=30)

timedelta class

timedelta object - A timedelta object represents a duration, the difference between two dates or times.

It is possible to know the future dates or previous dates using timedelta.

Creating Object of timedelta Class

object_name = timedelta(days=0, seconds=0, microseconds=0, milliseconds=0, minutes=0, hours=0, weeks=0)

All arguments are optional and default to 0. Arguments may be integers or floats, and may be positive or negative.

Only days, seconds and microseconds are stored internally. Arguments are converted to those units:

A millisecond is converted to 1000 microseconds.

A minute is converted to 60 seconds.

An hour is converted to 3600 seconds.

A week is converted to 7 days.

Ex:-

```
td = timedelta(days=10)
```

Comparing Two Dates

We can compare date class and datetime class objects using ==, <, >.

The comparison will return either True or False.

```
d1 = date(year=2019, month=6, day=30)
d2 = date(year=2016, month=6, day=30)
d1 == d2
d1 < d2
d1 > d2
```

Formatting Date and Time

strftime() Method – This method is used to format the content of datetime, date and time class object. strftime represents string format to time. This method convert the object into a specified format and returns the formatted string.

Ex:-

```
dt = datetime.today()
newdt = dt.strftime(“%B, %d, %Y”)
```

Format Code

Format Code

Format Code	Meaning	Example
%a	Weekday in short name	Sun, Mon,…., Sat
%A	Weekday in full name	Sunday, Monday,…, Saturday
%d	Day of month with 0 padded	01, 02,….,30, 31
%b	Month in short Name	Jan, Feb, ……, Dec
%B	Month in full Name	January,….., December
%m	Month in number with 0 padded	01, 02, …., 12
%y	Year in short with 0 padded, without century	00, 01, 02,…., 99
%Y	Year in Full with century	0001, 0002, ….., 9999

Format Code

Format Code	Meaning	Example
%H	Hours with 0 padded (24 hours clock)	00, 01, 02,……, 23
%I	Hours with 0 padded (12 hours clock)	01, 02,…., 12
%p	AM/PM	AM, PM
%M	Minute with 0 padded	00, 01, ….., 59
%S	Second with 0 padded	00, 01, ….., 59
%f	Microsecond with 0 padded	000000,……, 999999
%Z	Time zone name	(empty), UTC, CST, EST
%j	Day number of year with 0 padded	001, 002,……, 366
%U	Week number of the year, Sunday as the first day of week with 0 padded	00, 01, ……., 53

Format Code

Format Code	Meaning	Example
%c	Locale’s appropriate date and time representation	Tue Jan 30 21:30:00 2019
%x	Locale’s appropriate date representation	08/16/88 (None); 08/16/1988 (en_US); 16.08.1988 (de_DE)
%X	Locale’s appropriate time representation	21:30:00 (en_US); 21:30:00 (de_DE)
%%	A literal ‘%’ character	%

https://docs.python.org/3.7/library/datetime.html#strftime-and-strptime-behavior