Модуль datetime

Модуль `datetime` предназначен для работы с датами и временем и предоставляет, кроме функций, несколько новых типов данных. Библиотека чаще всего применяется для того, чтобы узнать текущую дату или время:

```
import datetime as dt



print(dt.datetime.now())

print(dt.datetime.now().time())

print(dt.datetime.now().date())
```

```
2022-09-04 18:34:42.071614

18:34:42.071614

2022-09-04
```

Но ее можно использовать и для более интересных вещей, она предоставляет несколько интересных типов данных: `time`, `date`, `datetime` — для хранения времени, даты и совместно даты и времени.

Давайте посмотрим, какие функции есть в этом модуле:

```
import datetime



print(dir(datetime))
```

```
['MAXYEAR', 'MINYEAR', '__builtins__', '__cached__', '__doc__',
'__file__',

'__loader__', '__name__', '__package__', '__spec__', 'date',
'datetime',

'datetime_CAPI', 'sys', 'time', 'timedelta', 'timezone', 'tzinfo']
```

Кроме различных специальных функций, видим несколько объектов без нижнего подчеркивания. Разберем подробнее, что можно с ними делать.

Объекты модуля datetime

`date` — тип для хранения даты. При создании новой даты нужно указать год, месяц и день.

```
import datetime as dt
```

```
# тип данных 'дата' (год, месяц, день)

new_date = dt.date(2022, 6, 5)

print(new_date)
```

```
2022-06-05
```

У этого объекта есть атрибуты: `day, month, year`:

```
import datetime



my_date = datetime.date(2022, 5, 17)

        print(my_date.year, my_date.month, my_date.day)
```

```
2022 5 17
```

С помощью функции `today()` можно узнать текущую дату на компьютере, а с помощью функции `weekday()` — день недели (нумерация с 0):

```
import datetime as dt



print(dt.date.today())

print(dt.date.today().weekday())
```

```
2022-04-06

1
```

`time` — тип для хранения времени. При создании объекта данного типа надо указать час, минуту, секунду.

```
import datetime as dt



# тип данных 'время' (часы, минуты, секунды, миллисекунды)

my_time = dt.time(23, 15, 29)

print(my_time)
```

```
23:15:29
```

`datetime` — тип для объединения даты и времени. Можно объединить `date` и `time` в один `datetime` с помощью функции `combine(date, time)`.

```
import datetime as dt


# тип данных дата + время
my_datetime = dt.datetime(2022, 12, 15, 13, 25, 9)
print(my_datetime)
# можно объединить дату и время
my_date = dt.date(2022, 11, 5)
my_time = dt.time(23, 15, 29)
my_datetime = dt.datetime.combine(my_date, my_time)
print(my_datetime)
```

```
2022-12-15 13:25:09
2022-11-05 23:15:29
```

Обратиться к отдельным частям объекта `datetime` можно через точку и название части: например, `hour` (аналогично можно поступить для `date` или `time`).

```
import datetime as dt


my_datetime = dt.datetime(2021, 12, 31, 23, 59, 59)
print(my_datetime.hour, my_datetime.minute)
```

```
23 59
```

С помощью функции `now()` можно узнать текущее дату и время.

```
# получить текущую дату и время
print(dt.datetime.now())
```

```
2022-04-06 19:55:28.406986
```

Тип `timedelta` необходим для создания объектов, содержащих некоторый временной интервал. При создании таких объектов можно указать, сколько недель, дней, часов, минут, секунд содержит временной интервал. Как и с другими типами модуля `datetime`, с частями временного интервала можно обращаться аналогично. С помощью функции `total_seconds()` можно узнать длину интервала в секундах.

```
import datetime as dt
```

```
# тип данных временной интервал

delta_time1 = dt.timedelta(seconds=10, weeks=2)

print(delta_time1)

print(delta_time1.days)

print(delta_time1.total_seconds())
```

```
14 days, 0:00:10

14

1209610.0
```

Интервал, в отличие от предыдущих типов, может быть отрицательным.

```
import datetime as dt


# интервал может быть отрицательным

delta_time2 = dt.timedelta(seconds=-10, weeks=-2)

print(delta_time2)
```

```
-15 days, 23:59:50
```

Операции с временными интервалами

Для интервалов, дат, времени и типа `datetime` поддерживаются математические и логические операции. Мы можем сложить два интервала времени, вычесть один из другого, увеличить интервал в n раз, уменьшить интервал в n раз, а с помощью деления узнать, сколько раз один интервал помещается в другой.

```
import datetime as dt


# поддерживаются математические и логические операции

# для интервалов времени

delta_time1 = dt.timedelta(days=4, hours=4)

delta_time2 = dt.timedelta(days=1, hours=1)

print(delta_time1 - delta_time2)

print(delta_time1 + delta_time2)

print(delta_time1 * 10)
```

```
print(delta_time1 / 10)

print(delta_time1 / delta_time2)

print(delta_time1 > delta_time2)
```

```
3 days, 3:00:00

5 days, 5:00:00

41 days, 16:00:00

10:00:00

4.0

True
```

К датам мы можем прибавлять временной интервал или удалять интервал для получения новой даты, которая раньше/позже на этот интервал. При вычитании одной даты из другой мы получим интервал времени между датами. С помощью логических функций мы можем узнать, какая дата позже (т. е. больше) или раньше (т. е. меньше).

```
import datetime as dt



# поддерживаются математические и логические операции для дат

date1 = dt.date(2019, 11, 5)

date2 = dt.date(2018, 10, 1)

delta_time1 = dt.timedelta(days=4, hours=4)

print(date1 + delta_time1)

print(date1 - delta_time1)

print(date1 - date2)

print(date2 > date1)
```

```
2019-11-09

2019-11-01

400 days, 0:00:00

False
```

Аналогичные действия работают и для `datetime`.

```
import datetime as dt
```

```
# поддерживаются математические и логические операции для типа datetime

datetime1 = dt.datetime(2020, 12, 15, 13, 25, 9)

datetime2 = dt.datetime(2019, 12, 15, 13, 25, 9)

delta_time1 = dt.timedelta(days=4, hours=4)

print(datetime1 + delta_time1)

print(datetime1 - delta_time1)

print(datetime1 - datetime2)

print(datetime2 < datetime1)
```

```
2020-12-19 17:25:09

2020-12-11 09:25:09

366 days, 0:00:00

True
```

Даты можно перебирать в цикле. Например, на далекий остров самолеты летают только по четвергам, да и то, если сумма номера дня и месяца в этот день нечетная. Выведем все даты вылетов в некотором временном диапазоне:

```
import datetime as dt start_date = dt.date(2021, 5, 1) end_date = dt.date(2021, 7, 31) date = start_date while date < end_date: if date.weekday() == 3 and \ (date.month + date.day) % 2: print(date) date += dt.timedelta(days=1)
```

```
2021-05-06

2021-05-20

2021-06-03

2021-06-17

2021-07-08

2021-07-22
```

Форматированный вывод даты и времени

Для времени, даты и типа `datetime` поддерживается форматированный вывод, который позволяет представлять информацию в удобном для нас виде. Для этого есть метод `strftime()`, который возвращает форматированную строку.

```
import datetime as dt


# поддерживается форматированный вывод
my_format = "%B"
print(dt.datetime.now().strftime("%A %d-%B-%y %H:%M:S"))
print(dt.date.today().strftime(my_format))
```

```
Wednesday 04-September-22 18:50:S

September
```

Основные обозначения для строки-формата приведены в таблице:

Команда	Значение	Пример
%a	Аббревиатура дня недели	Sun, Mon, …, Sat (en_US);So, Mo, …, Sa (de_DE)
%A	Полное название дня недели	Sunday, Monday, …, Saturday (en_US);Sonntag, Montag, …, Samstag (de_DE)
%w	День недели как десятичное число, где 0 — воскресенье, а 6 — суббота	0, 1, …, 6
%d	День месяца в формате из двух цифр	01, 02, …, 31
%b	Аббревиатура месяца	Jan, Feb, …, Dec (en_US);Jan, Feb, …, Dez (de_DE)
%B	Полное название месяца	January, February, …, December (en_US);Januar, Februar, …, Dezember (de_DE)
%m	Номер месяца в формате из двух цифр	01, 02, …, 12
%y	Последние 2 цифры года (год без века)	00, 01, …, 99
%Y	Год полностью	0001, 0002, …, 2013, 2014, …, 9998, 9999
%H	Час в 24-часовом формате из двух цифр	00, 01, …, 23
%I	Час в 12-часовом формате из двух цифр	01, 02, …, 12
%p	AM или PM	AM, PM (en_US);am, pm (de_DE)
%M	Минута в формате из двух цифр	00, 01, …, 59
%S	Секунда в формате их двух цифр	00, 01, …, 59
%f	Микросекунды в формате из 6 цифр	000000, 000001, …, 999999
%j	День в году в формате из 3 цифр	001, 002, …, 366
%U	Неделя в году	00, 01, …, 53
%c	Принятое, согласно локальным настройкам, представление даты и времени	Tue Aug 16 21:30:00 1988 (en_US);Di 16 Aug 21:30:00 1988 (de_DE)
%x	Принятое, согласно локальным настройкам, представление даты	08/16/88 (None);08/16/1988 (en_US);16.08.1988 (de_DE)
%X	Принятое, согласно локальным настройкам, представление времени	21:30:00 (en_US);21:30:00 (de_DE)
%%	Знак '%'	%

Более подробно можно посмотреть в документации.

Преобразование строки в объект даты

Как есть метод преобразования даты-времени в строку, так есть и обратный метод `strptime()` получения объекта даты-времени из строки, только нужно указать, что в строке является какой частью возвращаемого объекта. Пусть дату нужно получить из строки, в которой день и месяц разделены пробелом, а год записан через запятую и пробел:

```
import datetime as dt



date_string = "17 May, 2022"

date = dt.datetime.strptime(date_string, "%d %B, %Y")

print(date)

print(date.date())
```

```
2022-05-17 00:00:00

2022-05-17
```

Мы рассмотрели только несколько встроенных библиотек — их намного больше. И у рассмотренных мы разобрали далеко не все функции. Пользуйтесь функциями `dir` и `help` и читайте документацию, в ней приведены примеры и объяснены нюансы.

https://docs.python.org/3.7/library/datetime.html