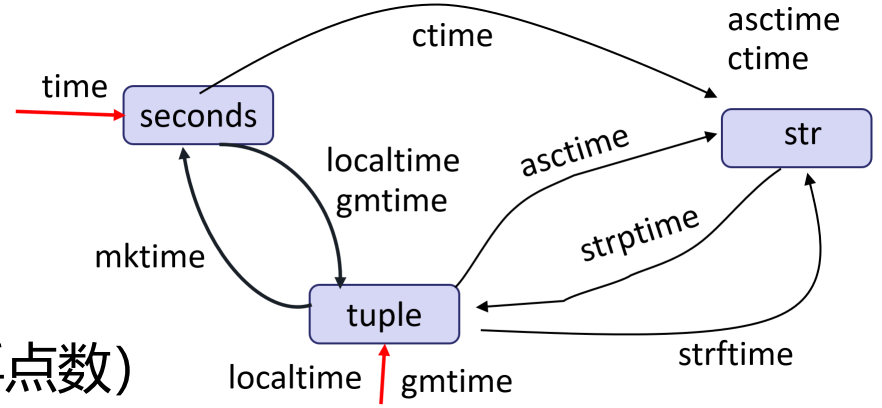
时间相关的模块:time和datetime

time模块： 04-有序对象-Page48

拓展的内容



- 得到当前时刻，进行不同格式的时间之间的转换
- 两种格式 + 字符串
  - 从epoch(即标准时1970年1月1日8点）开始到现在的秒数（浮点数）
  - 包含9个整数的tuple: 年/月/日/小时/分钟/秒/星期几(0-6, 0表示星期一)/当年第几天(1-366)/是否夏时制

time()	**返回从epoch开始到现在的秒数**
localtime([seconds])	将从epoch开始的秒数格式转换为本地时间的tuple格式，缺省为现在
gmtime([seconds])	将从epoch开始的秒数格式转换为UTC时间的tuple格式，缺省为现在
ctime([seconds])	将从epoch开始的秒数按照系统缺省方式转换为字符串，缺省为现在
mktime(tuple)	将本地时间的tuple格式转换为秒数
asctime([tuple])	将本地时间的tuple格式按系统缺省方式转换为字符串，缺省为现在，即相当于传递参数localtime()
strftime(format[,tuple])	按照format描述的格式将本地时间的tuple格式转换为字符串，缺省为现在
strptime(string, format)	按照format所描述的格式分析字符串string中的时间，返回tuple格式
sleep(seconds)	睡眠指定的秒数(浮点数)

time模块

拓展的内容

```
>>> import time
>>> start = time.time()
>>> start
1585987981.4362593
>>> stop = time.time()
>>> stop - start
190.97639966011047
>>> time.localtime()
time.struct_time(tm_year=2020, tm_mon=4, tm_mday=4, tm_hour=16, tm_min=13,
tm_sec=21, tm_wday=5, tm_yday=95, tm_isdst=0)
>>> time.gmtime()
time.struct_time(tm_year=2020, tm_mon=4, tm_mday=4, tm_hour=8, tm_min=13, tm_sec=31,
tm_wday=5, tm_yday=95, tm_isdst=0)
>>> start_tuple = time.localtime(start)
>>> time.mktime(start_tuple)
1585987981.0
>>> time.asctime(start_tuple)
'Sat Apr  4 16:13:01 2020'
>>> time.ctime(start)
'Sat Apr  4 16:13:01 2020'
```

time模块： help(time.strftime)

拓展的内容

格式化字符	描述	格式化字符	描述
%Y	四位年份	%a	星期几的简写，如Wed
%m	月份[1,12]	%A	星期几的全名，如Wednesday
%d	当月第几天	%b	月份的简写
%H	小时[0,23]	%B	月份的全名
%I	小时[1,12]	%c	缺省的日期和时间格式
%M	分钟[0,59]	%u	星期几[1,7]
%S	秒数[0,59]	%w	星期几[0,6]
%p	上午或下午描述 PM/AM	%U	今年的第几个星期 [0, 53]
%z	包含时区偏移	%j	今年的第几天[1,366]
%s	距离Epoch的秒数		

```
>>> now = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime())
>>> now
'2018-04-09 23:20:26‘
>>> time.strptime(now, "%Y-%m-%d %H:%M:%S")
time.struct_time(tm_year=2018, tm_mon=4, tm_mday=9, tm_hour=23, tm_min=20, tm_sec=26,
tm_wday=0, tm_yday=99, tm_isdst=-1)
```

datetime模块

- 在time模块的基础上提供日期时间等信息的访问，支持时间运算。可通过属性访问相关信息
- 支持时间运算：
  - 时刻与时刻可以比较大小
  - 时刻 - 时刻 = 时间间隔
  - 时刻 + 时间间隔 = 时刻
- 4个主要类型
  - datetime类型：某日的某个时刻
  - date：某日（day/month/year)，没有明确时间部分
  - time: 时间(hour/minute/second/microsecond)，没有明确日期部分
  - timedelta：时间间隔

datetime.date类型

- 某日（day/month/year），没有明确时间部分

	主要方法	描述
date对象	date(year, month, day)	构造一个date对象。只读属性：year/month/day
	dateobj.replace(year=None, month=None, day=None)	在dateobj的基础上，改变日期的某些部分，得到一个新的date对象
	date.today()	类方法，得到一个表示今天的date对象
	date.fromtimestamp(ts)	类方法，根据距离epoch的秒数得到date对象
	date.fromisoformat(date_str)	类方法，根据ISO8601格式(YYYY-MM-DD)解析得到date对象
	date.fromordinal(n)	类方法，根据前公历序数得到date对象
	date.fromisocalendar(year, week, weekday)	类方法，根据ISO日历：年，第几个星期，星期几解析得到date对象
星期几?	dateobj.ordinal()	前公历序数，1年1月1日为第1天
	dateobj.weekday()	星期几，取值[0, 6]，0表示星期一
	dateobj.isoweekday()	星期几，取值[1, 7]，1表示星期一
	dateobj.isocalendar()	返回一个元组，（年，第几个星期，星期几）
格式化	dateobj.ctime()	字符串，类似于： 'Mon May 22 00:00:00 2023'
	dateobj.isoformat()； str(dateobj)	字符串，ISO 8601格式(YYYY-MM_DD)。
	dateobj.strftime(fmt)	按照fmt进行格式化，参见time模块的strftime方法
	__format__(fmt)	字符串的格式化方法的实现，fmt参见time模块的strftime方法
运算符	dateobj.timetuple()	返回一个time模块中的表示时间的9元元组对象
	dateobj + timedelta, dateobj - timedelta	时刻 加减 时间间隔，得到一个新的date对象
	dateobj1 - dateobj2	两个时刻相减，得到一个新的timedelta对象
	dateobj op dateobj， op为比较运算符	支持两个date对象的比较，比较日期的先后顺序

datetime.time类型

- 时间(hour/minute/second/microsecond)，没有明确日期部分

主要方法	描述
time(hour, minute, second, microsecond)	构造一个time对象。hour和minute必须，其他可选。 只读属性：hour, minute, second, microsecond
timeobj.replace(hour=None, minute=None, second=None, microsecond=None)	在timeobj的基础上，改变时间的某些部分，得到一个新的time对象
timeobj.fromisoformat(time_str)	类方法，根据ISO8601格式(HH:MM:SS) 解析得到time对象
timeobj.isoformat()； str(dateobj)	字符串，ISO8601格式(HH:MM:SS)。
timeobj.strftime(fmt)	按照fmt进行格式化，参见time模块的strftime方法
__format__(fmt)	字符串的格式化方法的实现，fmt参见time模块的strftime方法
timeobj op timeobj， op为比较运算符	支持两个time对象的比较，比较时间的先后顺序

datetime.datetime类型：某日的某个时刻

- 继承了datetime.date，原来date类型的方法仍然可使用，其实现也基本类似（考虑了小时分钟等）

datetime对象	datetime(year, month, day, hour, minute, second, microsecond)	构造一个datetime对象。 只读属性：year, month, day, hour, minute, second, microsecond
	datetimeobj.replace(year=None, month=None, day=None, hour=None, minute=None, second=None, microsecond=None)	在datetimeobj的基础上，改变某个或某些部分，得到一个新的datetime对象
	datetime.now(); datetime.today()	类方法，得到一个表示现在时刻的datetime对象
	datetime.fromtimestamp(ts)	类方法，根据距离ephoch的秒数构造代表该时刻的datetime对象
	datetime.fromisoformat(datetime_str)	类方法，根据ISO8601格式(YYYY-MM-DD HH:MM:SS)解析得到datetime对象
	datetime.fromordinal(n)	类方法，根据前公历序数得到datetime对象，时间为0点
	datetime.fromisocalendar(year, week, weekday)	类方法，根据ISO日历（年，第几个星期，星期几）解析得到datetime对象，时间为0点
	datetime.strptime(datetime_str, fmt)	**类方法，基于fmt给出的格式，解析字符串，得到datetime对象**
	datetime.combine(dateobj, timeobj)	**类方法，将日期和时间组合**
	datetimeobj.date()　datetimeobj.time()	**根据日期或者时间得到一个date对象或time对象**
星期几?	datetimeobj.ordinal()	前公历序数
	datetimeobj.weekday()	星期几，取值[0, 6]，0表示星期一
	datetimeobj.isoweekday()	星期几，取值[1, 7]，1表示星期一
	datetimeobj.isocalendar()	返回一个元组，（年，第几个星期，星期几）
格式化	datetimeobj.ctime()	字符串，类似于：'Mon May 22 00:00:00 2023'
	datetimeobj.isoformat()； str(datetimeobj)	字符串，ISO8601格式(YYYY-MM-DD HH:MM:SS)。
	datetimeobj.strftime(fmt)	按照fmt进行格式化，参见time模块的strftime方法
	__format__(fmt)	字符串的格式化方法的实现，fmt参见time模块的strftime方法
	datetimeobj.timetuple()	返回一个time模块中的表示时间的9元元组对象
	datetimeobj.timestamp()	**返回距离epoch的秒数**
运算符	datetimeobj + timedelta, datetimeobj - timedelta	时刻 加减 时间间隔，得到一个新的datetime对象
	datetimeobj1 - datetimeobj2	两个时刻相减，得到一个新的timedelta对象
	datetimeobj op datetimeobj，　op为比较运算符	支持两个datetime对象的比较，比较日期的先后顺序

datetime.timedelta类型

- 时间间隔：datetime或date对象与时间间隔之间的时间运算：
  - 时刻 - 时刻 = 时间间隔
  - 时刻 + 时间间隔 = 时刻
  - 时刻 – 时间间隔 = 时刻
- 时间间隔本身的运算：
  - - 时间间隔 = 时间间隔， abs(时间间隔） = 时间间隔
  - 时间间隔 + 时间间隔 = 时间间隔； 时间间隔 – 时间间隔 = 时间间隔
  - 时间间隔 * 整数或浮点数 = 时间间隔； 时间间隔 / 整数或浮点数 = 时间间隔
  - 时间间隔 // 时间间隔 = 整数； 时间间隔 % 时间间隔 = 时间间隔

主要方法	**描述**
timedelta(weeks, days, hours, minutes, seconds, milliseconds, microseconds)	构造函数，描述时间间隔，所有参数都是可选的。 只读属性：days, seconds, microseconds。时间间隔内部通过这3个属性表示，即表示多少天多少秒多少微秒
timedeltaobj.total_seconds()	该时间间隔可以表示为多少秒
str(timedeltaobj)	类似格式的输出：10 days, 14:48:00
repr(timedeltaobj)	类似格式的输出：datetime.timedelta(days=10, seconds=53280)

datetime模块示例

- 可通过属性访问相关信息
- toordinal(): 转换为**前公历**序数，1年1月1日为第1天

datetime_demo.py

```
import datetime
now = datetime.datetime.now()
print('现在时刻: %d-%02d-%02d %02d:%02d' % (now.year, now.month,
        now.day, now.hour, now.minute))
delta_now = now - datetime.datetime(year=now.year, month=1, day=1)
print('今天是今年的第%d天' % (delta_now.days + 1) )

if now.month == 12:
    next_month = now.replace(year=now.year + 1, month=1, day=1)
else:
    next_month = now.replace(month=now.month + 1, day=1)
print('还有%d天到下个月' % (next_month - now).days)

interval = float(input('请输入一个时间间隔（分钟）'))
future = now + datetime.timedelta(minutes=interval)
print(future.ctime())
print(future.strftime('%Y-%m-%d %H:%M:%S'))
print('过去了%d天' % (future.toordinal() - now.toordinal()))
print('那一天是', future.strftime('%A'))
```