05 Datetime and Timedeltas

April 14, 2019

0.0.1 Datetime and Timedetas

datetime(year, month, day[, hour[, minute[, second[, microsecond[, tzinfo]]]]])

```
In [2]: import datetime
```

A good practice is to use the Universal Time

```
In [3]: datetime.datetime.utcnow()
```

```
Out[3]: datetime.datetime(2017, 8, 29, 13, 34, 31, 874849)
```

To get the local time:

```
In [4]: datetime.datetime.now()
```

```
Out[4]: datetime.datetime(2017, 8, 29, 15, 34, 59, 671541)
```

Create a specific timestamp

```
In [7]: datetime.datetime(
            year=2017, month=8, day=29,
            hour=9, minute=0, second=0)
```

```
Out[7]: datetime.datetime(2017, 8, 29, 9, 0)
```

```
In [30]: datetime.datetime(year=2017, month=9, day=1)
```

```
Out[30]: datetime.datetime(2017, 9, 1, 0, 0)
```

0.0.2 Timedelta

Substraction of dates from one another will return a timedelta:

```
In [31]: datetime.datetime(2017, 8, 29, 9) - datetime.datetime(2017, 8, 29 - 1, 9)
```

```
Out[31]: datetime.timedelta(1)
```

Timedeltas can also be created

```
In [32]: drei_tage = datetime.timedelta(days=3)
```

```
In [33]: drei_tage.total_seconds()

Out[33]: 259200.0
```

...and used for calculations

```
In [ ]: datetime.datetime.utcnow() + drei_tage
```

- Get a datimetime for the last day of the month:

```
In [34]: datetime.datetime(
             year=2017, month=9, day=1) - datetime.timedelta(days=1)

Out[34]: datetime.datetime(2017, 8, 31, 0, 0)

In [35]: # even more precise
         datetime.datetime(
             year=2017, month=9, day=1) - datetime.timedelta(seconds=1)

Out[35]: datetime.datetime(2017, 8, 31, 23, 59, 59)
```

- Access attributes directly

```
In [15]: now = datetime.datetime.now()

In [25]: now.year

Out[25]: 2017

In [26]: now.month

Out[26]: 8

In [16]: now.day

Out[16]: 29

In [17]: now.hour

Out[17]: 15

In [18]: now.minute

Out[18]: 40
```

0.0.3 Convert datetime to string and vice versa

```
In [19]: now.strftime("%Y-%m-%d %H:%M:%S")

Out[19]: '2017-08-29 15:40:33'

In [20]: now.strftime("%A %d.%m.%y %H:%M:%S")

Out[20]: 'Tuesday 29.08.17 15:40:33'

In [21]: datetime.datetime.strptime('2017-08-28 23:03:20', "%Y-%m-%d %H:%M:%S")

Out[21]: datetime.datetime(2017, 8, 28, 23, 3, 20)
```

note: the pattern must match exatly.

```
In [22]: datetime.datetime.strptime(
             '2017-08-28 23:03:20.7865',
             "%Y-%m-%d %H:%M:%S")


         ---------------------------------------------------------------------------

         ValueError                                Traceback (most recent call last)

         <ipython-input-22-c5cc1a2d8c57> in <module>()
     ----> 1 datetime.datetime.strptime('2017-08-28 23:03:20.7865', "%Y-%m-%d %H:%M:%S")


         /Users/hendorf/anaconda/envs/introduction_to_python/lib/python2.7/_strptime.pyc in _st
         333     if len(data_string) != found.end():
         334         raise ValueError("unconverted data remains: %s" %
     --> 335                          data_string[found.end():])
         336
         337     year = None


         ValueError: unconverted data remains: .7865


In [37]: datetime.datetime.strptime(
             '2017-08-28 23:03:20.7865',
             "%Y-%m-%d %H:%M:%S.%f")

Out[37]: datetime.datetime(2017, 8, 28, 23, 3, 20, 786500)

In [25]: datetime.datetime.strptime(
             '2017-08-28 23:03:20.7865Z',
             "%Y-%m-%d %H:%M:%S.%fZ")

Out[25]: datetime.datetime(2017, 8, 28, 23, 3, 20, 786500)
```

```
In [38]: datetime.datetime.strptime(
             '2017-08-28',
             "%Y-%m-%d")
```

```
Out[38]: datetime.datetime(2017, 8, 28, 0, 0)
```

All directives for date-string formatting can be found here: https://docs.python.org/2/library/datetime.html?highlight=datetime#strftime-and-strptime-behavior

Standard library recommendation is to use thrid party OS pytz for handling timezones

0.0.4 Alternative libraries with probably better timezone management:

- Maya - https://github.com/kennethreitz/maya
- Arrow - http://arrow.readthedocs.io/en/latest/