Zajęcia 7 - zapytania i praca z plikami JSON

In [2]:
```
import urllib
import json
```

Przykład 1 - API serwisu Airly

Każde zapytanie należy poprzedzić czytaniem dokumentacji i odnajdywaniem kolejnych linków!!!

In [2]:
```
apikey = '3Im4ALn4pqbRihkybONpv4Ey700CQzxA'
url = 'https://airapi.airly.eu/v2/installations/nearest?lat=52.229770&lng=21.011780&maxDistanceKM=10&maxResults=5'

req = urllib.request.Request(url, headers={'apikey': apikey})
resp = urllib.request.urlopen(req)
content = resp.read()

data=json.loads(content)
```

In [3]:
```
print(data)
```

```
[{'id': 39528, 'location': {'latitude': 52.23483, 'longitude': 21.014997}, 'locationId': 39212, 'address': {'countr
y': 'Poland', 'city': 'Warsaw', 'street': 'Warecka', 'number': '13/15', 'displayAddress1': 'Warsaw', 'displayAddress
2': 'Warecka'}, 'elevation': 116.14, 'airly': True, 'sponsor': {'id': 872, 'name': 'Biznes Spot', 'description': "Air
ly Sensor's sponsor", 'logo': 'https://cdn.airly.eu/logo/BiznesSpot_1613376321905_1762686420.png', 'link': 'https://b
iznesspot.pl', 'displayName': 'Biznes Spot'}}, {'id': 337, 'location': {'latitude': 52.235619, 'longitude': 21.0112},
'locationId': 337, 'address': {'country': 'Poland', 'city': 'Warsaw', 'street': 'Jasna', 'number': '14/16A', 'display
Address1': 'Warsaw', 'displayAddress2': 'Świętokrzyska'}, 'elevation': 116.59, 'airly': True, 'sponsor': {'id': 259,
'name': 'Anonymous SmogFighter', 'description': '', 'logo': 'https://cdn.airly.eu/logo/SmogFighter.jpg', 'link': 'htt
ps://www.facebook.com/airlyeu/', 'displayName': 'Anonymous SmogFighter'}}, {'id': 18791, 'location': {'latitude': 52.
223896, 'longitude': 21.017094}, 'locationId': 9944, 'address': {'country': 'Poland', 'city': 'Warsaw', 'street': 'Pi
ękna', 'number': '24/26', 'displayAddress1': 'Warsaw', 'displayAddress2': 'Piękna'}, 'elevation': 116.72, 'airly': Tr
ue, 'sponsor': {'id': 767, 'name': 'Huawei', 'description': "Airly Sensor's sponsor", 'logo': 'https://cdn.airly.eu/l
ogo/Huawei_1601292158328_559620620.jpg', 'link': None, 'displayName': 'Huawei'}}, {'id': 6864, 'location': {'latitud
e': 52.227118, 'longitude': 21.024308}, 'locationId': 6864, 'address': {'country': 'Poland', 'city': 'Warsaw', 'stree
t': 'Wiejska', 'number': '19', 'displayAddress1': 'Warsaw', 'displayAddress2': 'Wiejska'}, 'elevation': 111.24, 'airl
y': True, 'sponsor': {'id': 49, 'name': 'Anonimowy', 'description': "Airly Sensor's sponsor", 'logo': 'https://cdn.ai
rly.eu/logo/Anonimowy_1602689776821_829611598.png', 'link': None, 'displayName': 'Anonimowy'}}, {'id': 9064, 'locatio
n': {'latitude': 52.221714, 'longitude': 21.009337}, 'locationId': 9064, 'address': {'country': 'Poland', 'city': 'Wa
rsaw', 'street': 'Plac Politechniki', 'number': '1', 'displayAddress1': 'Warsaw', 'displayAddress2': 'Plac Politechni
ki'}, 'elevation': 119.33, 'airly': True, 'sponsor': {'id': 22, 'name': 'Aviva', 'description': "Airly Sensor's spons
or", 'logo': 'https://cdn.airly.eu/logo/Aviva_1538146740542_399306786.jpg', 'link': 'https://wiemczymoddycham.pl/',
'displayName': 'Aviva'}}]
```

Każde zapytanie należy poprzedzić czytaniem dokumentacji i odnajdywaniem kolejnych linków!!!

In [3]:
```
apikey = '3Im4ALn4pqbRihkybONpv4Ey700CQzxA'
url = 'https://airapi.airly.eu/v2/measurements/point?lat={}&lng={}'.format(52.221714, 21.009337)
req = urllib.request.Request(url, headers={'apikey': apikey})
resp = urllib.request.urlopen(req)
content = resp.read()
```

In [ ]:
```
data=json.loads(content)
print(data)
```

Teraz należy przefiltrować dane!

Przykład 2 - API serwisu Luftdaten: adres linku do API

Każde zapytanie należy poprzedzić czytaniem dokumentacji i odnajdywaniem kolejnych linków!!!

In [5]:
```
url = 'http://api.luftdaten.info/static/v1/data.json'

req = urllib.request.Request(url)
resp = urllib.request.urlopen(req)
content = resp.read()

data=json.loads(content)
```

In [6]:
```
print(data)
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

Zbyt duża ilość danych do pokazania, w takim wypadku należy przeprowadzić ich filtrację a podgląd danych zrobić w przeglądarce!

Przykład 3 - API serwisu USGS: trzęsienia ziemi

Każde zapytanie należy poprzedzić czytaniem dokumentacji i odnajdywaniem kolejnych linków!!!

In [2]:
```
import requests
```

In [13]:
```
r = requests.get('https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_day.geojson')
print(r.status_code)
print(r.encoding)
#print(r.text)
data = r.json()
#print(data)
```

```
200
utf-8
```

Teraz należy przefiltrować dane!

Przykład 4 - API serwisu exchangerates: kursy walut

Każde zapytanie należy poprzedzić czytaniem dokumentacji i odnajdywaniem kolejnych linków!!!

In [14]:
```
APIadress = 'https://api.exchangeratesapi.io/latest'

resp = requests.get(APIadress)
jsonFile = resp.json()
print(jsonFile)
```

```
{'rates': {'CAD': 1.5319, 'HKD': 9.4153, 'ISK': 155.2, 'PHP': 59.021, 'DKK': 7.4366, 'HUF': 359.14, 'CZK': 25.906, 'A
UD': 1.5378, 'RON': 4.876, 'SEK': 10.0747, 'IDR': 17129.52, 'INR': 88.0575, 'BRL': 6.6062, 'RUB': 89.9219, 'HRK': 7.5
77, 'JPY': 127.98, 'THB': 36.453, 'CHF': 1.0946, 'SGD': 1.6048, 'PLN': 4.5066, 'BGN': 1.9558, 'TRY': 8.5681, 'CNY':
7.8506, 'NOK': 10.319, 'NZD': 1.6585, 'ZAR': 17.8599, 'USD': 1.2143, 'MXN': 25.1611, 'ILS': 3.9655, 'GBP': 0.86398,
'KRW': 1351.72, 'MYR': 4.91}, 'base': 'EUR', 'date': '2021-02-23'}
```

Teraz należy przefiltrować dane!

Przykład 5 - API serwisu coindesk: kursy bitcoina

Każde zapytanie należy poprzedzić czytaniem dokumentacji i odnajdywaniem kolejnych linków!!!

In [16]:
```
APIadress = 'https://api.coindesk.com/v1/bpi/currentprice.json'

resp = requests.get(APIadress)
jsonFile = resp.json()
print(jsonFile)
```

```
{'time': {'updated': 'Feb 24, 2021 09:29:00 UTC', 'updatedISO': '2021-02-24T09:29:00+00:00', 'updateduk': 'Feb 24, 20
21 at 09:29 GMT'}, 'disclaimer': 'This data was produced from the CoinDesk Bitcoin Price Index (USD). Non-USD currenc
y data converted using hourly conversion rate from openexchangerates.org', 'chartName': 'Bitcoin', 'bpi': {'USD': {'c
ode': 'USD', 'symbol': '&#36;', 'rate': '50,817.5409', 'description': 'United States Dollar', 'rate_float': 50817.540
9}, 'GBP': {'code': 'GBP', 'symbol': '&pound;', 'rate': '35,854.9258', 'description': 'British Pound Sterling', 'rate
_float': 35854.9258}, 'EUR': {'code': 'EUR', 'symbol': '&euro;', 'rate': '41,762.6682', 'description': 'Euro', 'rate_
float': 41762.6682}}}
```

Teraz należy przefiltrować dane!

Przykład 6 - otwarte dane m. st. Warszawy

Każde zapytanie należy poprzedzić czytaniem dokumentacji i odnajdywaniem kolejnych linków!!!

In [ ]:
```
apiKey = 'c0061569-71b4-4d5e-ab23-feb21d5a350f'
url = r'https://api.um.warszawa.pl/api/action/dbstore_get/?id=ab75c33d-3a26-4342-b36a-6e5fef0a3ac3&apikey={}'.format
(apiKey)

resp = requests.get(url)
jsonFile = resp.json()
print(jsonFile)
```

Teraz należy przefiltrować dane!

Przykład 7 - otwarte dane m. st. Warszawy

Każde zapytanie należy poprzedzić czytaniem dokumentacji i odnajdywaniem kolejnych linków!!!

Wysyłanie maili

Najpierw należy skonfigurować konto gmail tak, aby nie posiadało największych zabezpieczeń link

In [6]:
```
import smtplib

MailFrom = "Prowadzacy BootCamp"
MailTo = 'antek9797@gmail.com'
MailSubject = "Testowe wysłanie maila"
MailBody =\
'''Dzien dobry,

to probny mail. Prosze na niego nie odpowiadac.

Z powazaniem
Antoni Dziedziejko
'''
message = '''From: {}
Subject: {}

{}'''.format(MailFrom, MailSubject, MailBody)

user = 'antek9797@gmail.com'
password = input("Podaj hasło do poczty dla użytkownika {}: ".format(user))


server = smtplib.SMTP_SSL('smtp.gmail.com', 465)
server.ehlo()
server.login(user, password)
server.sendmail(user, MailTo, message)
server.close()
print("Pomyślnie wysłano wiadomość")
```

```
Pomyślnie wysłano wiadomość
```

Pobieranie stron internetowych

In [9]:
```
def save_url_file(url, directory, file, msg='Please wait - the file {} will be downloaded'):
    import os
    print(msg.format(file))

    r = requests.get(url, stream = True)
    file_path = os.path.join(directory, file)

    with open(file_path, "wb") as f:
        f.write(r.content)
```

In [10]:
```
save_url_file(r'https://smartfactor.pl/', r'C:\SmartFactor\BootCamp\zajecia7', 'stronaSF.html')
```

```
Please wait - the file stronaSF.html will be downloaded
```

Praca domowa: wprowadzenie do analiz przestrzennych

In [17]:
```
from shapely.geometry import Point, Polygon
import matplotlib.pyplot as plt

# Create Point objects
p1 = Point(24.952242, 60.1696017)
p2 = Point(24.976567, 60.1612500)

# Create a Polygon
coords = [(24.950899, 60.169158), (24.953492, 60.169158), (24.953510, 60.170104), (24.950958, 60.169990)]
poly = Polygon(coords)
```

In [18]:
```
# Check if p1 is within the polygon using the within function
In [4]: p1.within(poly)
Out[4]: True

# Check if p2 is within the polygon
In [5]: p2.within(poly)
Out[5]: False
```

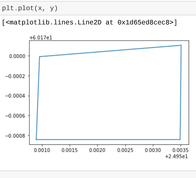
In [21]:
```
x,y = poly.exterior.xy
```

In [22]:
```
print(x)
print(y)
```

```
array('d', [24.950899, 24.953492, 24.95351, 24.950958, 24.950899])
array('d', [60.169158, 60.169158, 60.170104, 60.16999, 60.169158])
```

In [23]:
```
plt.plot(x, y)
```

Out[23]:
```
[<matplotlib.lines.Line2D at 0x1d65ed8cec8>]
```



In [ ]:
```
```