Session 2 Lesson 1 - Web API

1. Write a CSV

Lets continue where we left off. We will write a small code to write into a csv file.

Write a code to append data row wise to a csv file

In [12]:

```
import csv
WRITE_CSV = "/Users/medapa/Dropbox/HEC/Teaching/Python Course for PhDs/Scri
pts/python4phd/write_csv.csv"
with open(WRITE_CSV, 'at',encoding = 'utf-8') as csv_obj:
    write = csv.writer(csv_obj) # Note it is csv.writer not reader

    write.writerow(['Poonacha','32','OM'])
    write.writerow(['XXX','29','DS'])
    write.writerow(['XXX','18','S'])
```

What do you think will happen if we use 'wt' as mode instead of 'at' ?

Can you write a program to read the word_sentiment.csv file and copy paste all the words that have a positive sentiment (row[1] > 0) in a new excel sheet ?

In [15]:

```
import csv
SENTIMENT_CSV = "/Users/medapa/Dropbox/HEC/Teaching/Python Course for PhDs/
Scripts/python4phd/word_sentiment.csv"
GOOD_SENTIMENT_CSV = "/Users/medapa/Dropbox/HEC/Teaching/Python Course for
PhDs/Scripts/python4phd/good_sentiment_csv.csv"

"""You should be very careful while indenting the code when you use with. I
t automatically
closes the file when you move out of the indent """

with open(GOOD_SENTIMENT_CSV, 'at',encoding = 'utf-8') as csv_obj:
    write = csv.writer(csv_obj) # Note it is csv.writer not reader

    with open(SENTIMENT_CSV, 'rt',encoding = 'utf-8') as senti_data:
        sentiment = csv.reader(senti_data)

        for row in sentiment:
            if int(row[1]) >0:
                write.writerow([row[0],row[1]])
```

2. Connecting to the web

Let us use the requests module. This module provides functions to send a HTTP request and get the response from the server. Documentation on the same can be found here - http://docs.python-requests.org/en/master/user/quickstart/#make-a-request (http://docs.python-requests.org/en/master/user/quickstart/#make-a-request)

Requests is a third party module. If not installed, we will need to do "$pip install requests" in the mac terminal or in the command pronpt of windows.

In [24]:

```
import requests
url = 'https://www.yelp.com/xxx'
resp = requests.get(url)
print(resp.status_code)

if resp.status_code == 200:
    print('Success')
else:
    print('Failed to get a response from the url. Error code: ',resp.status
_code )
```

```
404
Failed to get a response from the url. Error code:  404
```

In [1]:

```
import requests
url = 'https://www.yelp.com/search?find_desc=jouy+en+josas&start=0&sortby=r
ating&cflt=restaurants&l=g:2.32935905457,48.8534209294,2.35493659973,48.870
3607224'
resp = requests.get(url)
#print(resp.text)
```

3. Using a Web API to Collect Data

An application programming interface is a set of functions that you call to get access to some service.An API is basically a list of instructions for interfacting with websites's data. The way these work is similar to viewing a web page. When you point your browser to a website, you do it with a URL (http://www.yelp.com (http://www.yelp.com) for instance). Yelp sends you back data containing HTML, CSS, and Javascript. Your browser uses this data to construct the page that you see. The API works similarly, you request data with a URL (http://api.yelp.com/stuff (http://api.yelp.com/stuff)), but instead of getting HTML and such, you get data formatted as JSON.

All the details regarding the data format and the different API's are available on the yelp website : https://www.yelp.com/developers/documentation/v2/search_api (https://www.yelp.com/developers/documentation/v2/search_api)

Lets see how the data given by the api looks like. In the following code we use the api to search for restaurants 500 mtrs around jouy en josas. We get 10 results. The data is packaged in JSON format and is given as below

In [2]:

```
import requests
import json # This library helps us format data into JSON and helps read JS
ON data
url = 'https://api.yelp.com/v2/search/?oauth_nonce=11906484&radius_filter=1
000&location=jouy+en+josas&oauth_timestamp=1485912240&oauth_consumer_key=Is
R5jptq8d5x_KLho9FHnw&oauth_signature_method=HMAC-SHA1&oauth_signature=GFzn4
MHT3Xf0BlgzIFZqHRQDicg%3D&category_filter=restaurants&oauth_body_hash=2jmj7
l5rSw0yVb%2FvlWAYkK%2FYBwk%3D&limit=40&oauth_token=0aR02eWkvx_e7ORc0Lm7pSLF
nr9mrBBK&cc=FR'
response = requests.get(url)
jason_data = json.loads(response.text)
#print(jason_data)
```

Lets use a better representation to see what data they have actually given us:
https://www.yelp.com/developers/api_console (https://www.yelp.com/developers/api_console)

Step 1: Authentication

OAuth 2 is an authorization framework that enables a user to connect to their account using a third party application. It works by delegating user authentication to the service that hosts the user account, and authorizing third-party applications to access the user account. While this is more secure thant the basic authentication (i.e. passing the userid and password while you make a http request), it is a little more difficult to code.

It needs a token and a consumer key to be generated and passed to the webserver. Unfortunately different websites have different ways of generating and using the token and consumer keys. Hence we will need to write the authorization code for each website seperately. HOwever, every website provides detailed information on how you can generate and send the token and keys.

You can get your OAuth2 token and consumer keys at :
https://www.yelp.fr/developers/v2/manage_api_keys
(https://www.yelp.fr/developers/v2/manage_api_keys)

The authentication code is packaged into a small function.

In [33]:

```
def search_yelp(url):
        ''' USE OAuth to authenticate the application. Make a request with
Yelp's
    API and return the result in JSON format'''
```

Step 2 - Create the URL

In this step we build the search URL that we need to use. The yelp API has been restricted to return only 40 search results. So we will need to change the location parameters in the URL so that we get less than 40 results during each search.

To understand the format of the URL we can check the examples given here: https://www.yelp.com/developers/documentation/v2/search_api (https://www.yelp.com/developers/documentation/v2/search_api) In yelp, we can specify the search based on latitude and longitude so it is possible to restrict the search to a small area and then iterate by changing the latitude and longitude values.

Let us now create the search URL to find all the restaurants located in jouy en josas with 1000 mtrs radius

In [35]:

```
search_url = 'https://api.yelp.com/v2/search/?location=jouy en josas&limit=
40&radius_filter=1000&cc=FR&category_filter=restaurants'
```

Step 3 - We now get the JSON data and collect the relevant fields

Let us now write the function main() that creates the search URL and sends it to the search_yelp() function. The search_yelp() function returns the search results in JSON format. We will use the for loops to traverse through the restaurants and find the relevant information

In [5]:

```
TOKEN = 'XXX'
TOKEN_SECRET = 'XXX'
CONSUMER_KEY = 'XXX'
CONSUMER_SECRET = 'XXX'

import json
import requests
import oauth2

# This function performs a Yelp API request, taken from Yelp's python examp
le
def search_yelp(url):
    ''' USE OAuth to authenticate the application. Make a request with Yel
p's
    API and return the result in JSON format'''
    consumer = oauth2.Consumer(CONSUMER_KEY, CONSUMER_SECRET)
    token = oauth2.Token(TOKEN, TOKEN_SECRET)

    oauth_request = oauth2.Request(method="GET", url=url)
    oauth_request.update({'oauth_nonce': oauth2.generate_nonce(),
                          'oauth_timestamp': oauth2.generate_timestamp(),
                          'oauth_token': TOKEN,
                          'oauth_consumer_key': CONSUMER_KEY})

    oauth_request.sign_request(oauth2.SignatureMethod_HMAC_SHA1(),
                               consumer, token)
    signed_url = oauth_request.to_url()
#    print(signed_url)
    try:
        response = requests.get(signed_url)
        print(response.status_code)
        if response.status_code == 200:
            return json.loads(response.text)
        else:
            print('HTTP status code not ok: ',response.status_code)
            return 0

    except:
        print('Error running the search, see if everything within the "try"
statement is working')
403
        return 0
HTTP status code not ok:  403
```

```
# OUR CODE STARTS HERE
def main():
    ''' This function collects sends the URL to the search API and collects
the JSON data back from
    it. From the JSON data we pick the relevant fields tah twe want to disp
layed'''
# Code Here
```

Let us now try and save these results in a CSV sheet

In [ ]:

```
    search_url = 'https://api.yelp.com/v2/search/?location=jouy en josas&li
mit=40&radius_filter=1000&cc=FR&category_filter=restaurants'

    search_results = search_yelp(search_url)
    if search_results == 0 : return

    print('The number of search results are: ', search_results['total'])
    i = 1
```

```
    '''The search comes ip with 15 results to iterate through each of these
restaurants,
    we use the for loop'''

    for biz in  search_results['businesses']:
        print(i,'. ',biz['name'],' ',biz['id'],' ',biz['rating'],' ',biz['r
eview_count'])
        i=i+1

main()
```