

Workshop in Methods (WIM)

Introduction to APIs for Social Scientists

Helge Marahrens

Indiana University Bloomington

Department of Sociology

Email correspondence to: hmarahre@iu.edu



The New York Times



The New York Times

Science

CLIMATE | SPACE & COSMOS | HEALTH | TRILOBITES | SCIENTETAKE | OUT THERE



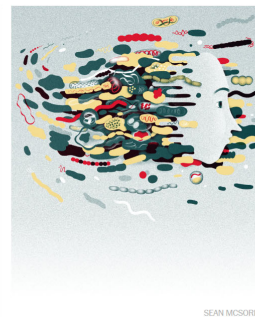
ARNO BURGI/PICTURE ALLIANCE, VIA GETTY IMAGES

TRILOBITES

Seeking Superpowers in the Axolotl Genome

The smiling salamanders can regrow most of their body parts, so researchers are building improved maps of their DNA.

1d ago • By STEPH YIN



SEAN MCSORLEY

MATTER

Germes in Your Gut Are Talking to Your Brain. Scientists Want to Know What They're Saying.

The body's microbial community may influence the brain and behavior, perhaps even playing a role in dementia, autism and other disorders.

1d ago • By CARL ZIMMER

A Closer Look at the Polar Vortex's Dangerously Cold Winds

Chicago will be as cold as the Arctic on Wednesday. We'll show you why.



11h ago • By YULIYA PARSHINA-KOTTAS, KARTHIK PATANJALI, JEREMY WHITE, BENJAMIN WILHELM and EVAN GROTHJAN

This Is Your Brain Off Facebook

Planning on quitting the social platform? A major new study offers a glimpse of what unplugging might do for your life. (Spoiler: It's not so bad.)



1h ago • By BENEDICT CAREY

Option 1: Webscrapping

- Flexible
- Difficult
- Understand html
 - e.g. where NYT article saves title information
- Ethical concerns

Option 2: API

- Organized
- Easier Access
- Limited
 - Not all information sources have APIs
 - Information restricted by whoever maintains API

API (Application Programming Interface)

- A set of protocols and routines for building and interacting with software applications.
 - tool that allows computers to exchange information
 - tool that allows you easy access to new datasets

Examples:

<https://locationiq.com/docs>

<https://projects.propublica.org/api-docs/congress-api/>

<https://developer.nytimes.com/docs/archive-product/1/overview>

A few things to consider

- How is the data sampled?
 - What kind of data are returned?
 - What is the request limit?
 - Do I need a key?
-
- read the documentation
 - trial & error on small tasks
 - step by step building

<https://developer.nytimes.com/docs/archive-product/1/overview>

FAQ

If you aren't sure whether your plans constitute "commercial purposes," please contact us at code@nytimes.com. For commercial use please visit <https://nytlicensing.com/>.

11. Is there an API call limit?

Yes, there are two rate limits per API: 4,000 requests per day and 10 requests per minute. You should sleep 6 seconds between calls to avoid hitting the per minute rate limit. If you need a higher rate limit, please contact us at code@nytimes.com.

12. What response formats do you support?

Data is returned as JSON. Specific APIs may also return other formats. See the documentation for each API for more details.

13. Can I make suggestions for future development?

Yes, please! You can email us at code@nytimes.com.



Example APIs

LocationIQ

ProPublica

New York Times

- Python 3.X (IDLE)
 - data types
 - lists and dictionaries
 - function vs. method
 - indentation is key (4 spaces)
 - counting begins at zero

pseudoscript

1. read API documentation
2. import packages
3. authentication
4. build get request
5. send get request – check server response
 - 200 – OK
 - 401 – unauthorized
 - 404 – data not found
 - 429 – too many requests
6. explore data structures
 - lists, dictionaries
7. save data
 - csv

1. read API documentation (<https://locationiq.com/docs>) & get your key

LocationIQ

2

Excited?! Get a developer token!

You'll need an unique token to identify your requests. We'll send you an email with an activation link .



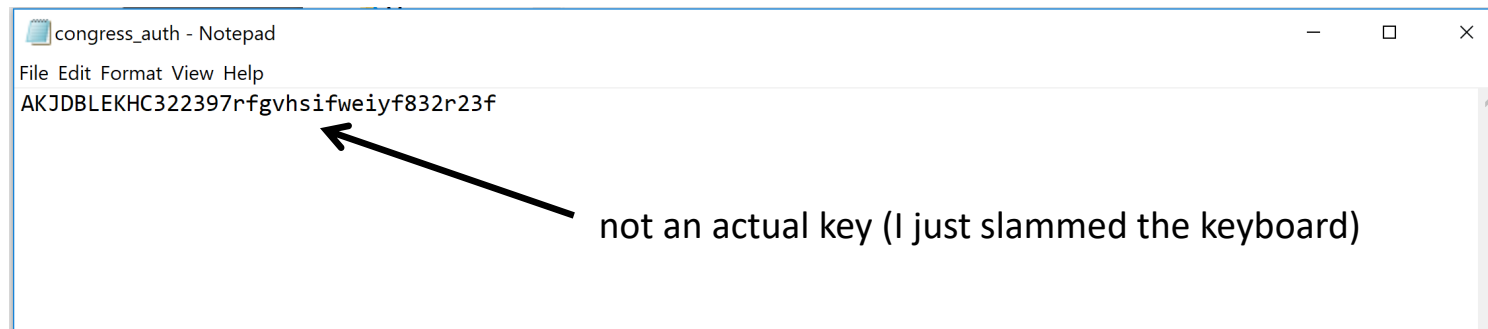
Your Name

Your E-mail

Sign Up

LocationIQ_2020-01-31_hmarahre.py

Do not put your authentication key in your script



Instead, call it from a .txt file in your working directory

Open Python – Import packages

- Open IDLE Python 3.X
- Command line / Script file

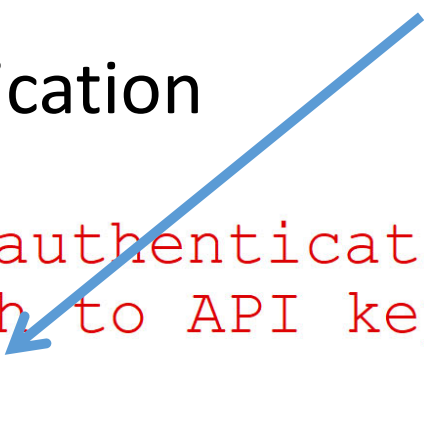
```
import requests
import json
import time
from collections import defaultdict

import csv
import pandas as pd
import matplotlib.pyplot as plt
```

your working directory (path) here

3. authentication

```
# //- 3. authentication
# set path to API key directory
path = ""
local_file = path + 'locationIQ_auth.txt'
with open(local_file, "r") as txtfile:
    API_key = txtfile.readline().strip('\n')
```



4. build get request

```
# we use a dictionary to specify search parameters
host = "https://us1.locationiq.com/v1/search.php"
data = {
    'key': API_key,
    'format': 'json',
    'city': 'Bloomington',
    'country': 'USA',
    'state': 'IN'
}
```

5. send get request – check server response

```
response = requests.get(host, params=data)
assert(response.status_code==200)
print(response)
location_data = response.json()
```

```
# loop through list to collect location data
# create dictionary to save results
cities = ['Bloomington, IN, USA',
          'Hannover, Germany',|
          'Chicago, IL, USA',
          'New York, NY, USA']
city loc json = defaultdict()
```



```
for city in cities:
    data = {'key': API_key,
            'format': 'json',
            'q':city}
    time.sleep(1)
    response = requests.get(host, params=data)
    if response.status_code!=200:
        break
    city_loc_json[city] = response.json()
```

6. explore data structures

```
[len(val) for key, val in city_loc_json.items()]
type(city_loc_json['Bloomington, IN, USA'])
city_loc_json['Bloomington, IN, USA'][0]
city_loc_json['Bloomington, IN, USA'][1]
for key, val in city_loc_json.items():
    print(key + ": " + val[0]['display_name'] + ", coord: " + \
          str(val[0]['lat']) + ", " + str(val[0]['lon']))
```

7. save data

```
# create a dataframe
city_loc_clean = defaultdict()
for key, val in city_loc_json.items():
    city_loc_clean[key] = (val[0]['display_name'], \
                           val[0]['lat'], val[0]['lon'])
df = pd.DataFrame.from_dict(city_loc_clean, \
                             orient='index', \
                             columns = ['name', 'lat', 'lon'])
df.to_csv("locationIQ.csv")
```

3. authentication

ProPublica_2020-01-31_hmarahre.py



FREE

ProPublica Congress API

Source	Various
Date Released	April 2016
Updates	At least daily

[VIEW DOCUMENTATION -->](#)

The Congress API returns the following types of data:

- **Roll-call vote data:** Only roll-call votes (not voice votes or division votes) are tracked by official Congressional data sources. Along with basic vote data, the ProPublica API returns additional information that is less readily available, such as party totals. Votes are available from 1991 for the House of Representatives and from 1989 for the Senate.
- **Member data:** Along with general biographical information for current and past members of Congress, the API returns data about members' Congressional roles. Role data includes the Congress number and chamber, as well as the member's title, state and party. A single member may have more than one role in a particular Congress (for example, the member may switch parties or move from the House to the Senate). The API also helps you compare member data, including vote positions and bill cosponsorships. Member data is available for every member who has served in Congress, but those who have served more recently (since 1995) have more information.
- **Personal explanations:** Statements inserted into the Congressional Record explaining why lawmakers missed votes and what their votes would have been (the statements are merely explanations and have no effect on the vote itself).
- **Bill data:** Along with standard bill summaries and details, the API returns bill subjects, amendments and related bills. You can also retrieve bills by member, view all the cosponsors of a bill, and search for bill subjects. Bill information is available for bills from 1995 onward. Full-text search of bills is also available.
- **Nomination data:** The API returns presidential civilian nomination lists and details. Military nominations are not included. For general information about presidential nominations, see [Congress.gov](#). Nomination data is available from 2001 onward.
- **Floor actions:** Descriptions of legislative activity that are updated in near real-time throughout the day. This endpoint exists for both the House and the Senate.
- **Committee data:** The API returns data about House and Senate committees, along with the memberships of those committees.
- **Other data:** The API returns various supplemental information, including party counts by state; and

REQUEST AN API KEY

To request an API key, please provide the following information.

Name *,required

Email *,required

Organization

☐ I intend to use this API for a commercial application

Purpose

Tell us how you'd like to use this API!

☐ Sign up for our newsletter to hear about new and updated data products.

SUBMIT

<https://www.propublica.org/datastore/api/propublica-congress-api>

1. read documentation / 2. Import packages

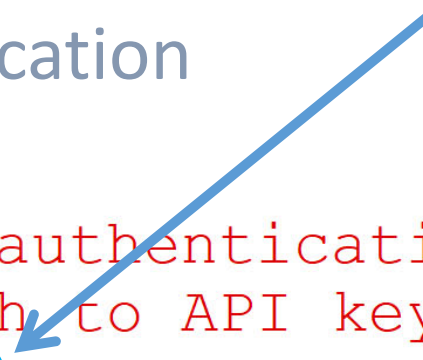
```
# //- 1. read API documentation
# https://www.propublica.org/datastore/api/propublica-congress-api
# "Usage is limited to 5000 requests per day
# (rate limits are subject to change)."
```



```
# //- 2. import packages
import requests
import json
import time
import pandas as pd
import csv
import matplotlib.pyplot as plt
```

3. authentication

your working directory (path) here



```
# //- 3. authentication
# set path to API key directory
path = ""
local_file = path + 'congress_auth.txt'
with open(local_file, "r") as txtfile:
    content = txtfile.readline().strip('\n')
# create dictionary with API key
credentials = {'X-API-Key':content}
```

4. build get request / 5. send get request

```
# //- 4. build get request
# list of all members of the 114th house of representatives
host = "https://api.propublica.org/congress/v1/114"
chamber = "/house"
data_section = "/members.json"

# //- 5. send get request - check server response
response = requests.get(host + chamber + data_section, headers=credentials)
assert(response.status_code==200)
members = response.json()
```


6. explore data structures

```
# //- 6. explore data structures
print(len(members))
print(type(members))
print(members.keys())
print(len(members['results']))
print(members['results'][0].keys())
print(members['results'][0]['congress'])
#print([print(members['results'][0][key]) for\
#       key in ['congress', 'chamber', 'num_results', 'offset']])
print(type(members['results'][0]['members']))
print(len(members['results'][0]['members']))
print(json.dumps(members['results'][0]['members'][0],\
                  indent=4, sort_keys=True))
```

7. save dataframe as csv file

```
# //- 7. save data
# create a dataframe
df_114 = pd.DataFrame(members['results'][0]['members'])
df_114.shape
list(df_114)

# analyze data
plt.hist(pd.to_datetime(df_114['date_of_birth']))
plt.show()

# save as csv
df_114.to_csv("congress_house_114.csv")
```


- Email correspondence to: hmarahre@iu.edu
- For those who like to learn via video/MOOC:
 - <https://www.datacamp.com>
 - <https://www.udemy.com>
- For those who prefer books:
 - <https://packtpub.com>