Web Scraping

with Beautiful Soup and MongoDB

```python
from pymongo import MongoClient
from bson.objectid import ObjectId

# connect to database
client = MongoClient('mongodb://localhost:27017')

# get the collection
restaurants = client.test.restaurants

# find all restaurants in Manhattan
for r in restaurants.find({'borough': 'Manhattan'}):
    print(r['name'], r['borough'])

# find only one restaurant by its _id
restaurants.find_one({'_id': ObjectId('5aaad67ec7d922f116bed455')})

# insert a restaurant into the database
r_id = restaurants.insert_one({'name': 'chyld'}).inserted_id

# delete a single restaurant in Queens
restaurants.delete_one({'borough': 'Queens'}).deleted_count

# update a property in the restaurant
restaurants.update_one({'name': 'Bob Deli'}, {'$set': {'name': 'Sue Deli'}})
```

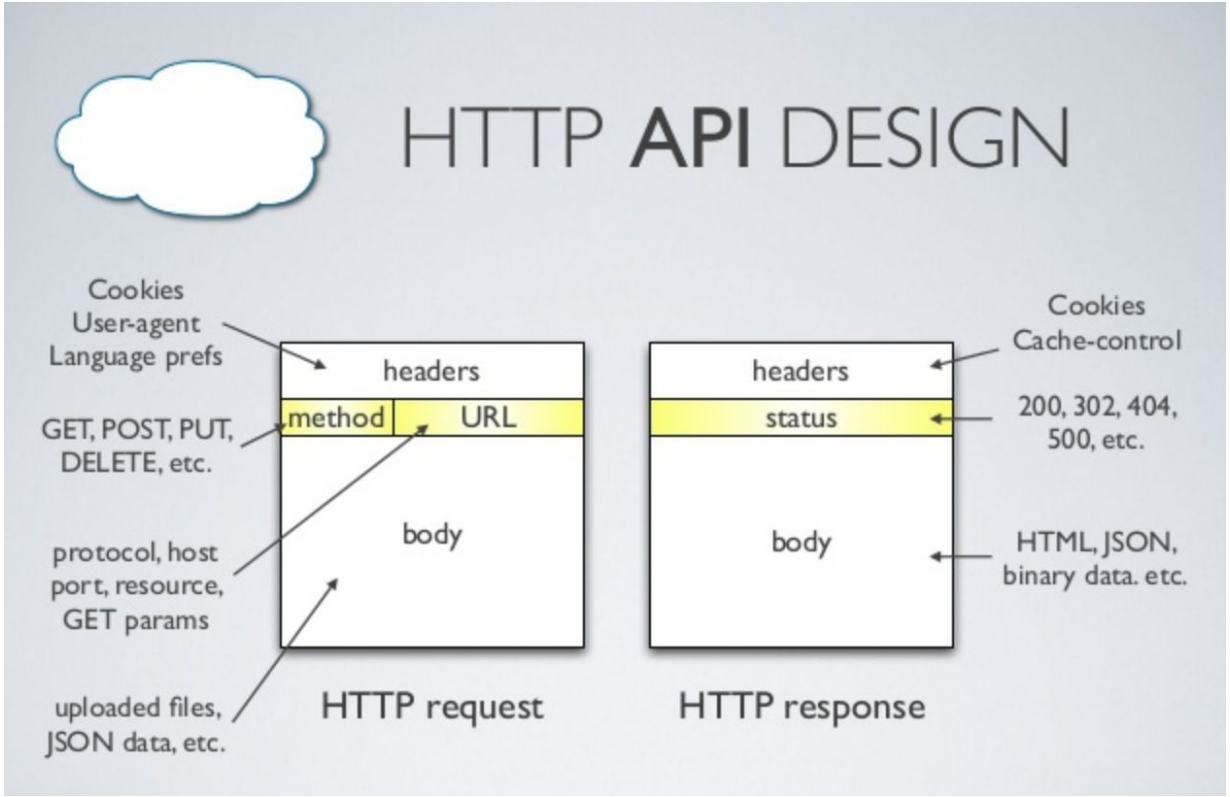
PYMONGO

HTTP METHODS

HTTP Methods and Their Meaning

Method	Meaning
GET	Read data
POST	Insert data
PUT or PATCH	Update data, or insert if a new id
DELETE	Delete data

HTTP ARCHITECTURE



HURL.IT - HTTP DEBUGGING

Hurl.it — Make HTTP Requests

Destination

Follow redirects: Off

GET

yourapihere.com

Authentication

+ Add Authentication

Headers

+ Add Header(s)

Parameters

+ Add Parameter(s)

I'm not a robot

reCAPTCHA
Privacy - Terms

Launch Request

Sorry about the reCAPTCHA! People were using the site for bad things. We're working to improve things so you don't have to check it every time.
Use this free tool if you need to make a lot of requests.

JSON API REQUESTS

```python
# https://data.sfgov.org/Transportation/Air-Traffic-Passenger-Statistics/rkru-6vcg

import requests
import json
from pymongo import MongoClient

# connect to database
client = MongoClient('mongodb://localhost:27017')

# send http request to json endpoint, with authentication
text = requests.get(
    'https://data.sfgov.org/resource/rptz-7xyh.json', params={
        '$$app_token': 'Vw2D4R6437×4OT3wdM0nC445z',
        '$limit': 5,
        '$offset': 20}).text

# turn json into a dict
data = json.loads(text)

# create a database and a collection
flights = client.sf.flights

# insert the flight data into the database
flights.insert_many(data)
```

SCRAPING BOX OFFICE MOJO

Box Office Mojo

IMDbPro

Adjuster: Actuals Go

Search Site

Search...

Features
- News
- Release Sched.
- Showtimes at IMDb

Box Office
- Daily
- **Weekend**
- Weekly
- Monthly
- Quarterly
- Seasonal
- Yearly
- All Time
- International

Indices
- Studios
- People
- Genres
- Franchises
- Showdowns
- Theater Counts

Weekend Box Office

Weekend Index

By Year | **By Weekend**

1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | **10** | 11 | 12 | 13
14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26
27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39
40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53

Year	Weekend (Click to view)	Top 12 Gross	Change LY	Overall Gross	Change LY	Total Movies	#1 Movie
2018	Mar. 9–11	$128,205,007	-15.7%	$139,937,460	-14.7%	97	Black Panther
2017	Mar. 10–12	$151,990,204	+3.4%	$163,997,836	+1.6%	107	Kong: Skull Island
2016	Mar. 4–6	$146,949,733	+87.7%	$161,499,059	+80.6%	99	Zootopia
2015	Mar. 6–8	$78,304,316	-40.3%	$89,415,791	-37.8%	94	Chappie
2014	Mar. 7–9	$131,212,139	+2.3%	$143,854,240	+2.3%	109	300: Rise of An Empire
2013	Mar. 8–10	$128,276,687	+6.0%	$140,687,760	+2.8%	112	Oz The Great and Powerful
2012	Mar. 9–11	$120,993,059	+2.9%	$136,877,195	+5.8%	127	The Lorax
2011	Mar. 11–13	$117,561,730	-36.8%	$129,391,313	-34.3%	118	Battle: Los Angeles
2010	Mar. 5–7	$185,857,643	+82.8%	$196,996,205	+68.6%	107	Alice in Wonderland
2009	Mar. 6–8	$101,682,847	+11.2%	$116,835,237	+8.9%	111	Watchmen

WEB SCRAPING WITH BEAUTIFUL SOUP

```python
import requests
import json
from pymongo import MongoClient
from bs4 import BeautifulSoup

# connect to database
client = MongoClient('mongodb://localhost:27017')

# get html from website
text = requests.get('http://www.boxofficemojo.com/weekend/?view=wknd').text

# turn html into bs4 object
soup = BeautifulSoup(text, 'lxml')

# create empty movies list for storage
movies = []

# find the data on the webpage, using css selectors
for i, tr in enumerate(soup.select('td > center > table > tr')):
    if i != 0:
        # creating a movie dictionary
        movie = {}
        tds = tr.select('td')
        movie['yr'] = tds[0].text
        movie['ov'] = tds[4].text
        movie['mv'] = tds[7].text
        # appending the movie dictionary to the list
        movies.append(movie)

# getting the database and collection
coll = client.galvanize.movies

# inserting the movie data into mongodb
coll.insert_many(movies)
```