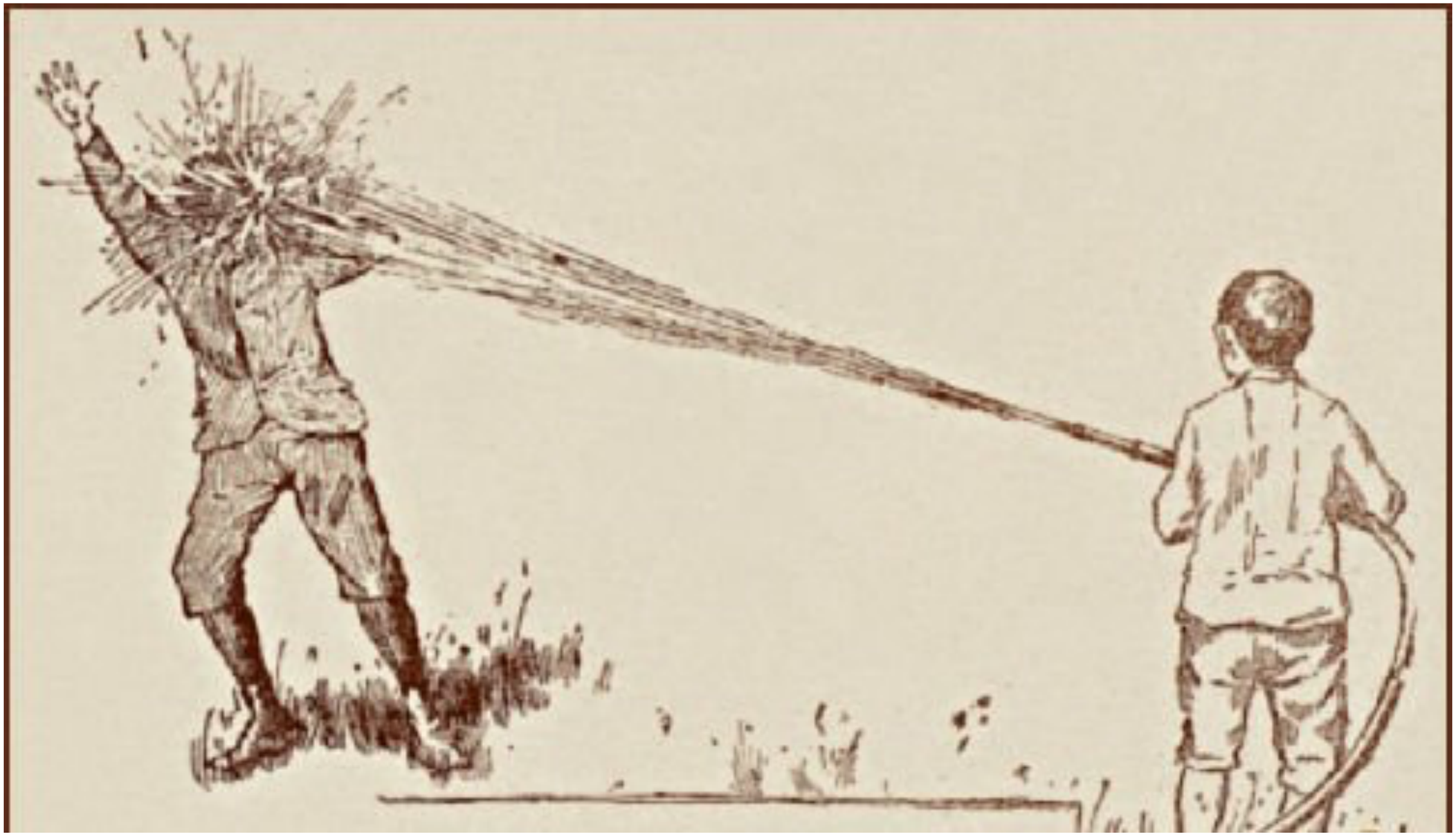
# Let’s Drink from the Firehose



# Mongo and Twitter

- To demonstrate a simple usage for MongoDB with Jupyter, you will implement a basic Twitter streamer that inserts captured tweets into a MongoDB collection.

- Twitter data represents an ideal use case for the NoSQL MongoDB.

- Each tweet obtained via the Twitter API is received as an unstructured nested JSON object.

# Mongo and Twitter

- Adding such an object to a SQL database would be a non-trivial task by any measure involving numerous foreign keys and JoinTables as the user seeks to manage each of the one-to-one, one-to-many, and many-to-one relationships built into the tweet.

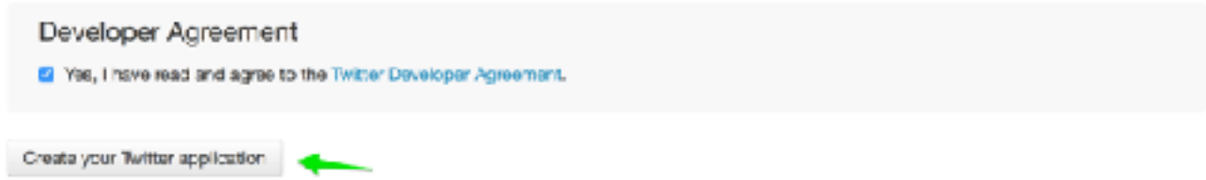
- Adding such an object to Mongo, on the other hand, is a trivial task.

- MongoDB's native Binary JSON (BSON) format was designed precisely to accept such an object.

# Obtain Twitter Credentials

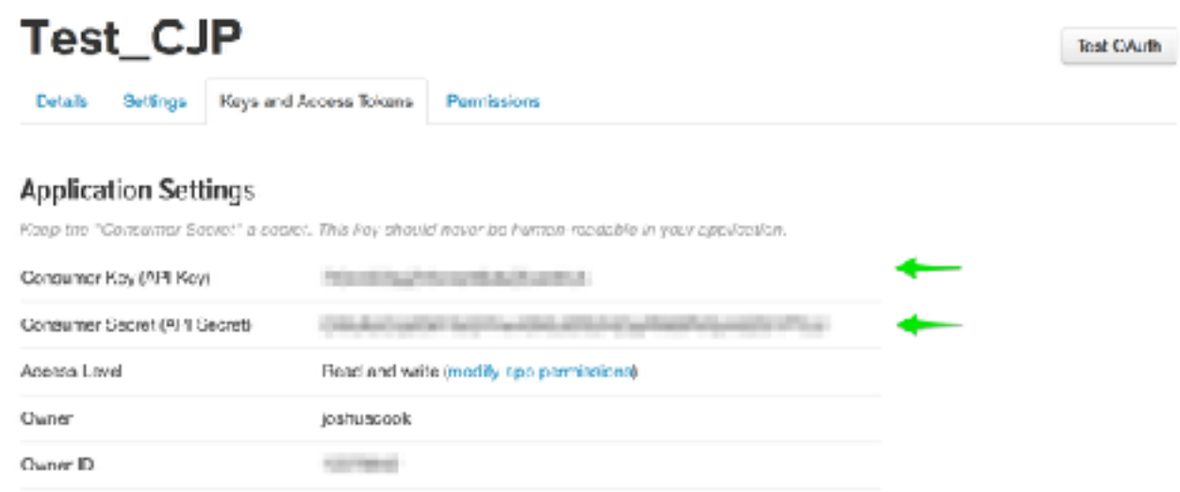
- In order to follow along, you must obtain API credentials for accessing the Twitter API. This is done by creating a Twitter application.

- In order to do this, follow these steps:
  1. Visit https://apps.twitter.com and sign in.
  2. Choose “Create New App”.
  3. Give the new app a name, description, and website. For your purposes, the values of these responses are irrelevant, although the website will need to have a valid URL structure.
  4. Agree to the Developer Agreement and click “Create your Twitter Application”.

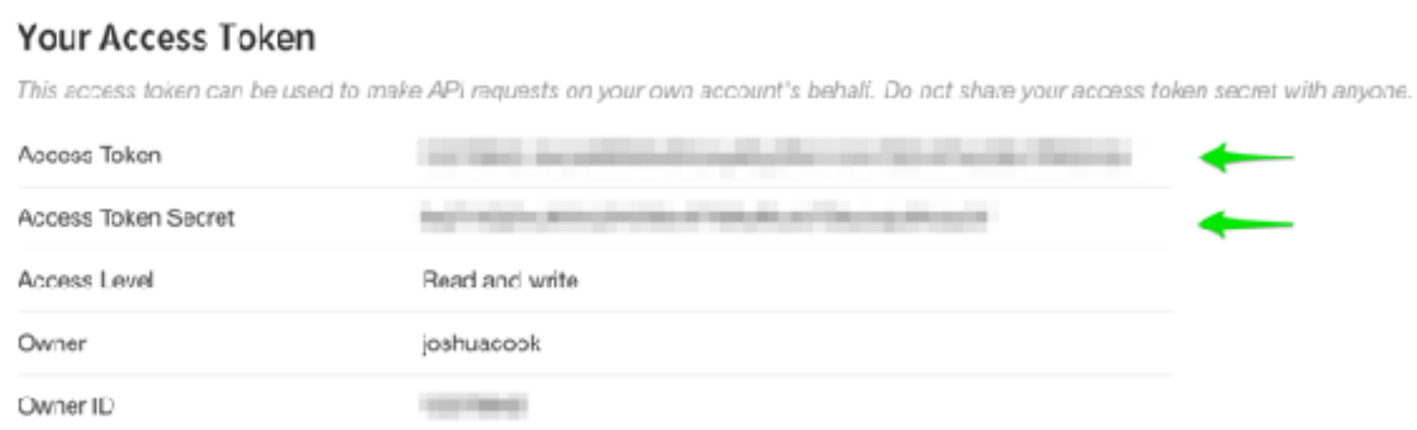



# Obtain Twitter Credentials

- Next, you will need to access your credentials on the “Keys and Access Tokens” tab.




- You will need a total of four values:
  1. A consumer key (API Key)
  2. A consumer secret (API Secret)
  3. An access token
  4. An access token secret



# Load Twitter Credentials

- Load Twitter Credentials as Strings
- Replace this with your credentials:

```
CONSUMER_KEY = None
CONSUMER_SECRET = None
ACCESS_TOKEN = None
ACCESS_SECRET = None
```

# Install the `twitter` library

- I prefer the `twitter` library over `tweepy`. I've found it to be better for streaming. Others have found tweeps better for historical data.

```
!pip install twitter
```

# Authentication

- You next instantiate a `twitter.OAuth` object using the Python twitter module and the credentials you have just loaded.

- You will use this object to facilitate your connection to Twitter's API.

```
from twitter import OAuth
oauth = OAuth(ACCESS_TOKEN, ACCESS_SECRET,
CONSUMER_KEY, CONSUMER_SECRET)
```

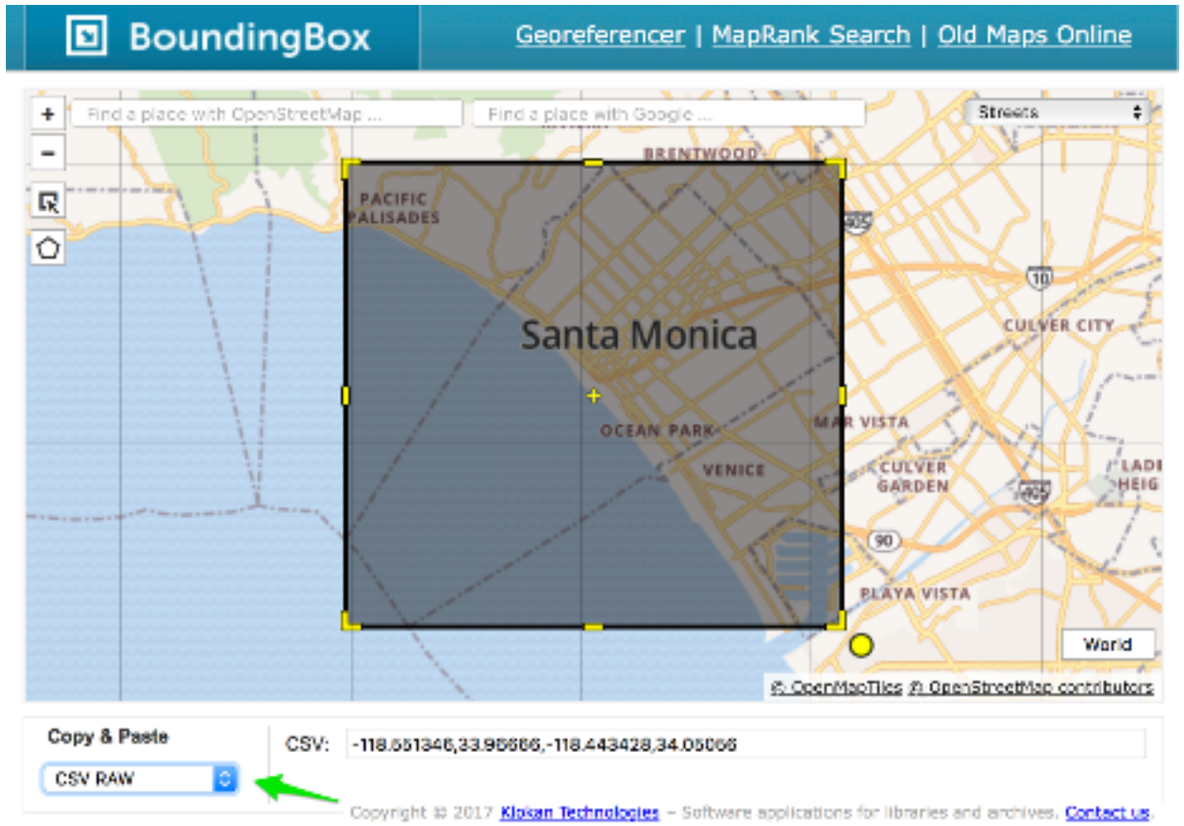
# Collect Tweets by Geolocation

- For this example, you will be using Twitter's Public Stream.
- Applications that are able to connect to a streaming endpoint will receive a sample of public data flowing through Twitter and will be able to do so without polling or concern of API rate limits.
- In other words, the Public Stream is a safe and sanctioned way to collect a sample of live public tweets.
- That said, even this sample will return a great deal of unordered data.

# Collect Tweets by Geolocation

- In order to provide a modicum of order to your Twitter stream, you will restrict incoming tweets using a geolocation bounding box, or bbox. You can easily obtain a bbox for a location of interest using the Klokantech BoundingBox Tool.

- Let's obtain a bbox for Santa Monica, California in the United States, making sure to select CSV Raw as the copy and paste format.




```
los_angeles_bbox = "-118.551346,33.96666,-118.443428,34.05056"
```

# Instantiate a `TwitterStream`

- Finally, you instantiate a `twitter.TwitterStream` object you will use to collect tweets.

- `twitter.TwitterStream` provides an interface to the Twitter Stream API in Python.

- The result of calling a method on this object is an iterator that yields tweets decoded from the Twitter stream as JSON objects.

```
from twitter import TwitterStream

twitter_stream = TwitterStream(auth=oauth)
twitterator =
twitter_stream.statuses.filter(locations=los_angeles_bbox)
```

# Insert Tweets Into Mongo

- Twitter is a wonderful source of messy, “real” data.
- Wrangling it into a database is where MongoDB truly shines.
- Using your `twitterator` object and the `.insert_one()` class function this can be done in a single line of code.

```
coll_ref.insert_one(next(twitterator))
coll_ref.count()
coll_ref.find_one()
```

# Tweet to Mongo

```
from pymongo import MongoClient
from twitter import TwitterStream
from twitter import OAuth

oauth = OAuth(ACCESS_TOKEN,
              ACCESS_SECRET,
              CONSUMER_KEY,
              CONSUMER_SECRET)
client = MongoClient('34.213.186.97', 27016)
twitter_stream = TwitterStream(auth=oauth)
santa_monica_bbox = "-118.551346,33.96666,-118.443428,34.05056"
twitterator = (twitter_stream
               .statuses
               .filter(locations=santa_monica_bbox))

db_ref = client.twitter
coll_ref = db_ref.tweets
coll_ref.insert_one(next(twitterator))
```