Unit 21 - Twitter API - Setup and Trends

Introduction

As humans, what are some things that we want that technology might help us to get?

- We want to be heard.
- We want to satisfy our curiosity.
- We want it easy.
- We want it now.

We have a deeply rooted need to share our ideas and experiences, which gives us the ability to connect with other people, to be heard, and to feel a sense of worth and importance. We are curious about the world around us, and how to organize and manipulate it. We use communication to share our observations, ask questions, and engage with other people in meaningful communications about our lives and our situations.

Twitter is as a microblogging service that allows people to communicate with short, 140-character messages called “tweets” that roughly correspond to thoughts or ideas. Think of Twitter as being a free, high-speed, globally reaching texting service.

Users share their thoughts, links to stuff that interests them, and photographs on Twitter. People comment on live events and companies promote their products in an attempt to engage with customers. All of this goes on 500 million times a day.

Learning Outcomes

In this lesson, you’ll use Python to take a peek into the torrent of tweets out there. So, for these next few lessons to work you’ll need a twitter account. If you don’t have one, sign up now and immediately follow someone or something to get some tweets flowing through your account.

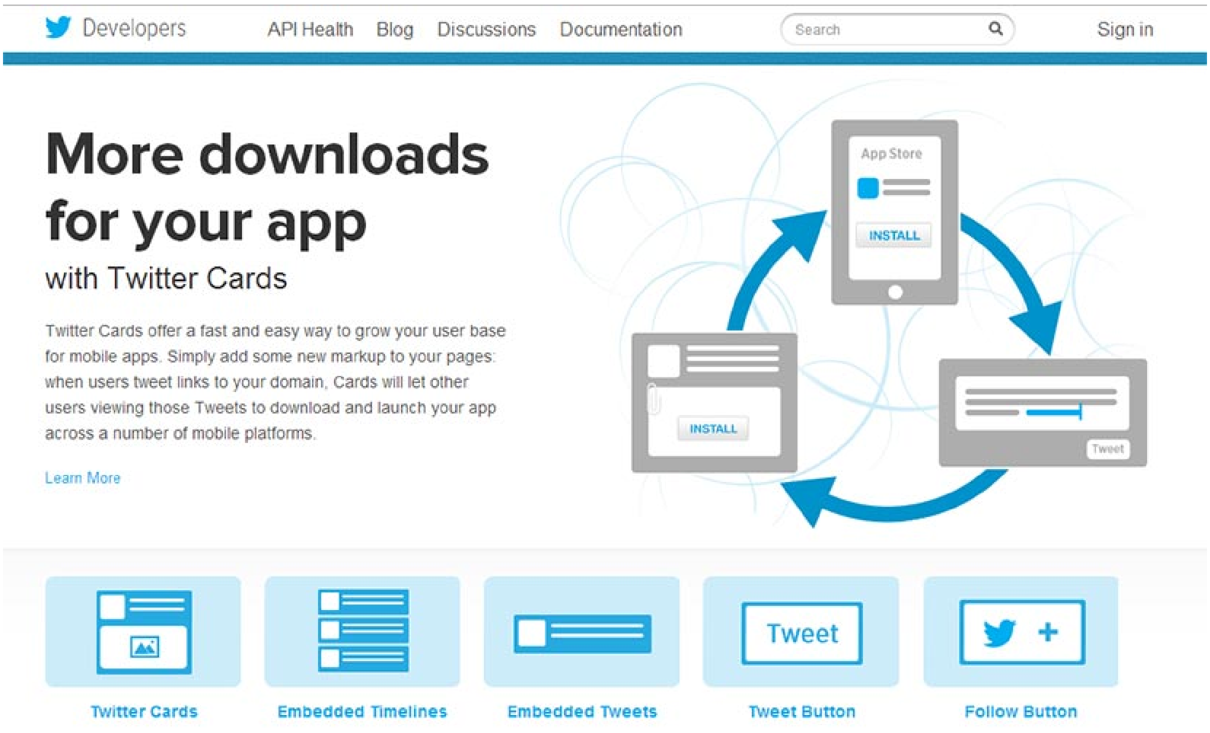
Topics include:

- Register Your App with Twitter
- What’s Trending?
- What’s Trending in Multiple Locations?

Topic 1 : Register Your App with Twitter

1. Go to the Twitter Developers’ Site

   In order to access the Twitter API in your app, you must first register it on the Twitter Developer’s Site.



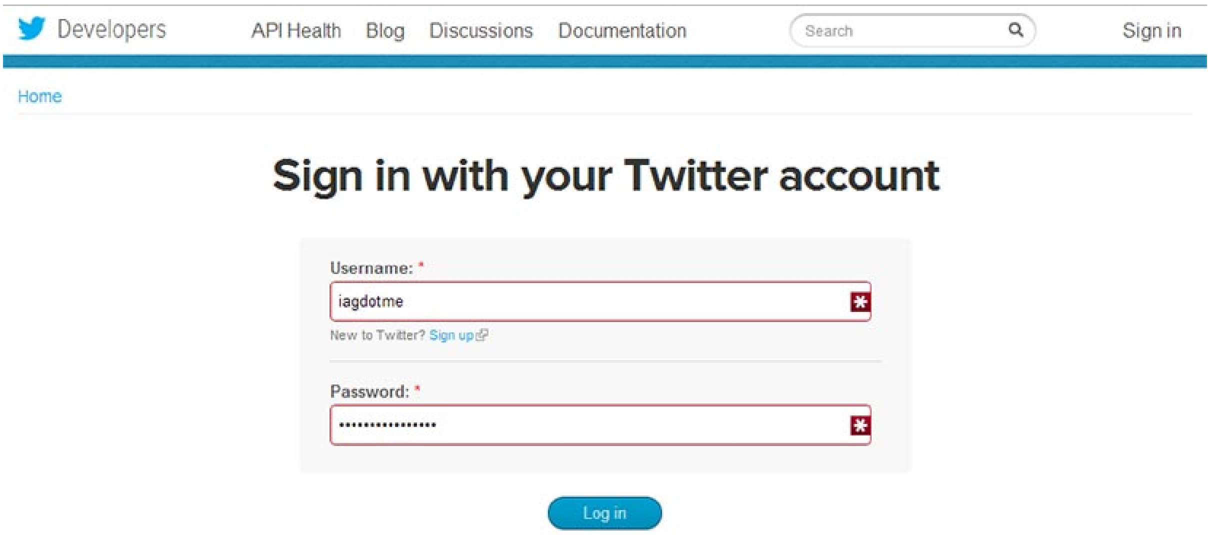
More downloads for your app

with Twitter Cards

Twitter Cards offer a fast and easy way to grow your user base for mobile apps. Simply add some new markup to your pages: when users tweet links to your domain, Cards will let other users viewing those Tweets to download and launch your app across a number of mobile platforms.

Learn More

2. Once you're there, log in with your existing twitter account



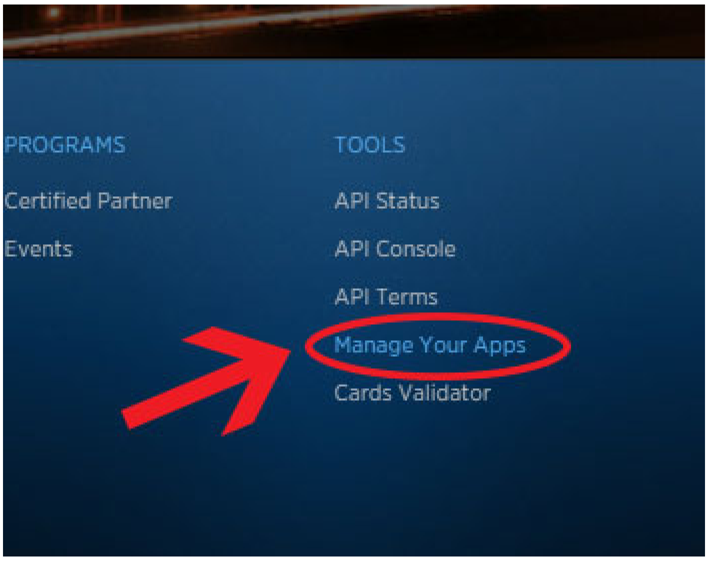
Sign in with your Twitter account

Username: *
iagdotme
New to Twitter? Sign up

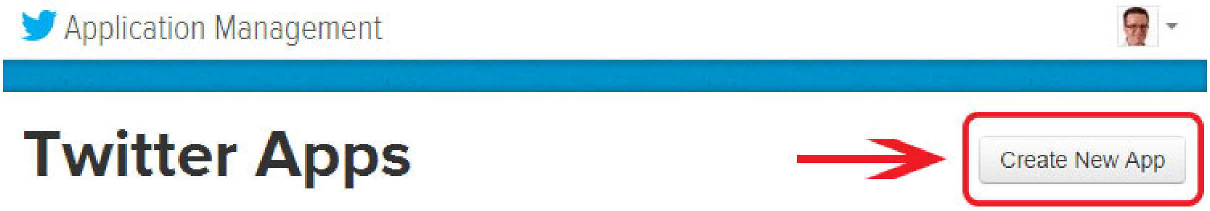
Password: *
••••••••••••••••

3. Navigate to **`apps.twitter.com`**

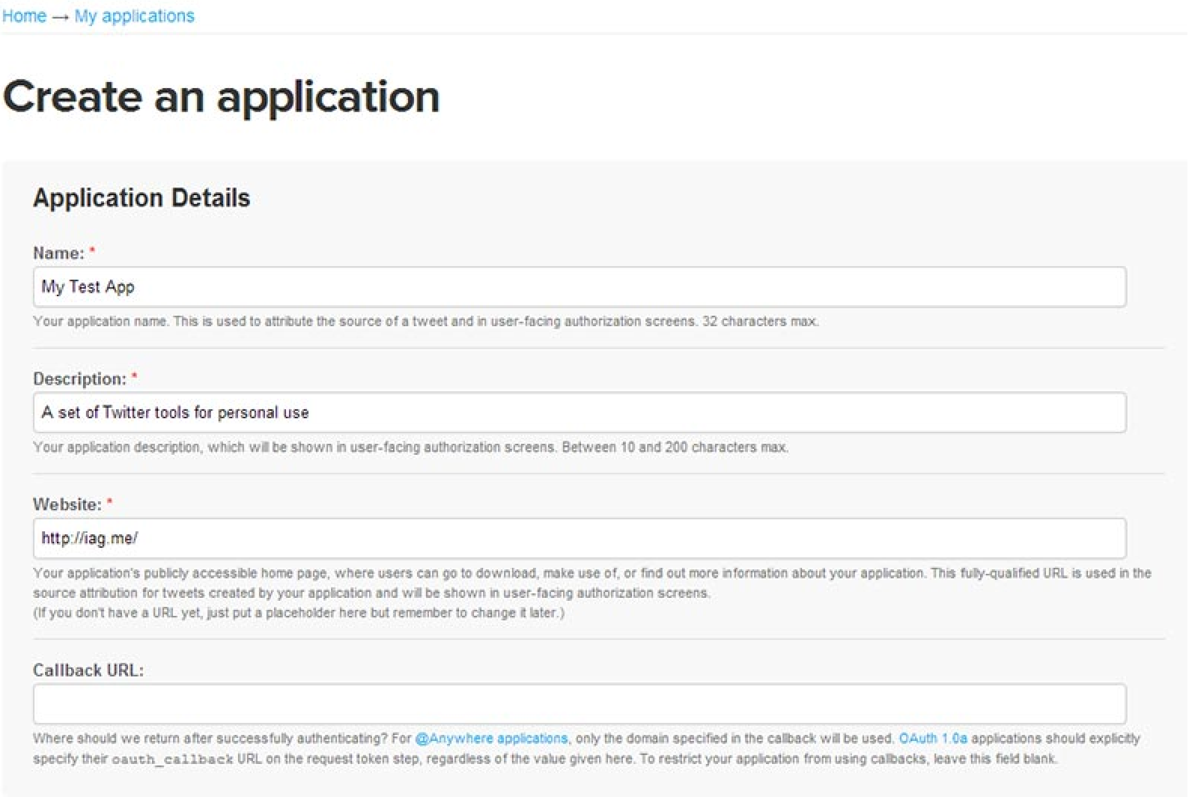
Once you're logged in, go to the Twitter's app website. You can also access it by clicking "manage your apps" in the footer of the Twitter Developer's Site.



4. Create a new app.



5. Fill in the detail of your app.



Home → My applications

Create an application

Application Details

Name: *

My Test App

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description: *

A set of Twitter tools for personal use

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website: *

http://iag.me/

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL:

Where should we return after successfully authenticating? For @Anywhere applications, only the domain specified in the callback will be used. OAuth 1.0a applications should explicitly specify their `oauth_callback` URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

- **Name**:

  First you need to give your app a unique name . As you're going to be creating an app for personal use, you can use your domain name or maybe even your own name.

- **Description**:

  This is just a general explanation of what your app will do. You can enter data mining as an example.

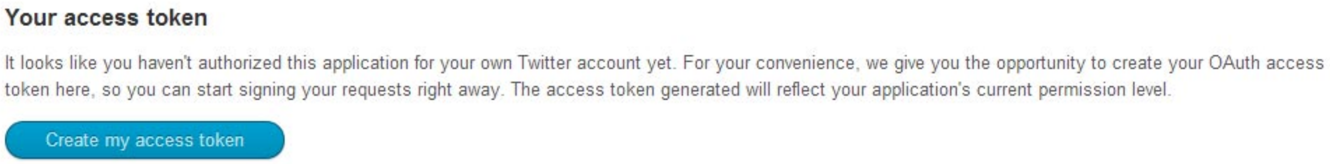
- **Website**:

  If your app is for personal use, this isn't really applicable. Just enter the website address where the app will be hosted or put in a fake placeholder address.

- **Callback URL:**

For now, ignore the Callback URL field. If you are allowing users to log into your app to authenticate themselves, you'd enter the URL where they would be returned after they've given permission to Twitter to use your app. Your app won't include third-party users logging in.

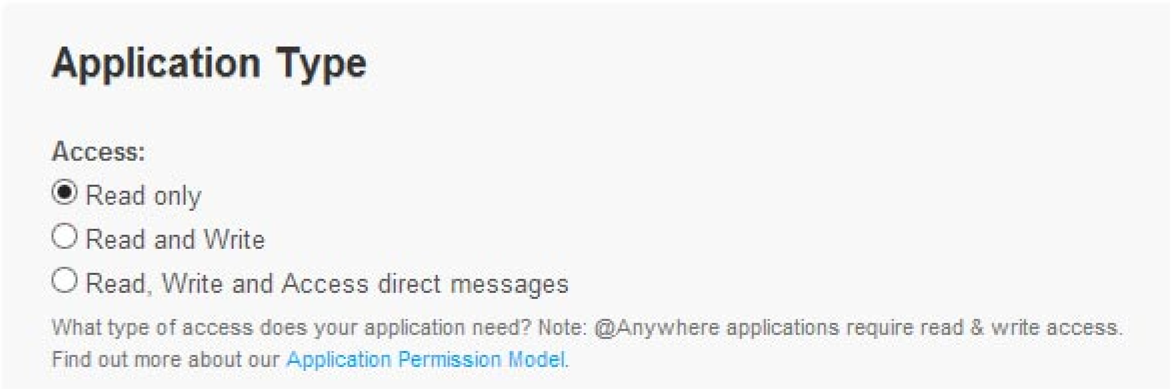
6. Create your access token.



Your access token

It looks like you haven't authorized this application for your own Twitter account yet. For your convenience, we give you the opportunity to create your OAuth access token here, so you can start signing your requests right away. The access token generated will reflect your application's current permission level.

7. Choose your access type.



Application Type

Access:

● Read only
○ Read and Write
○ Read, Write and Access direct messages

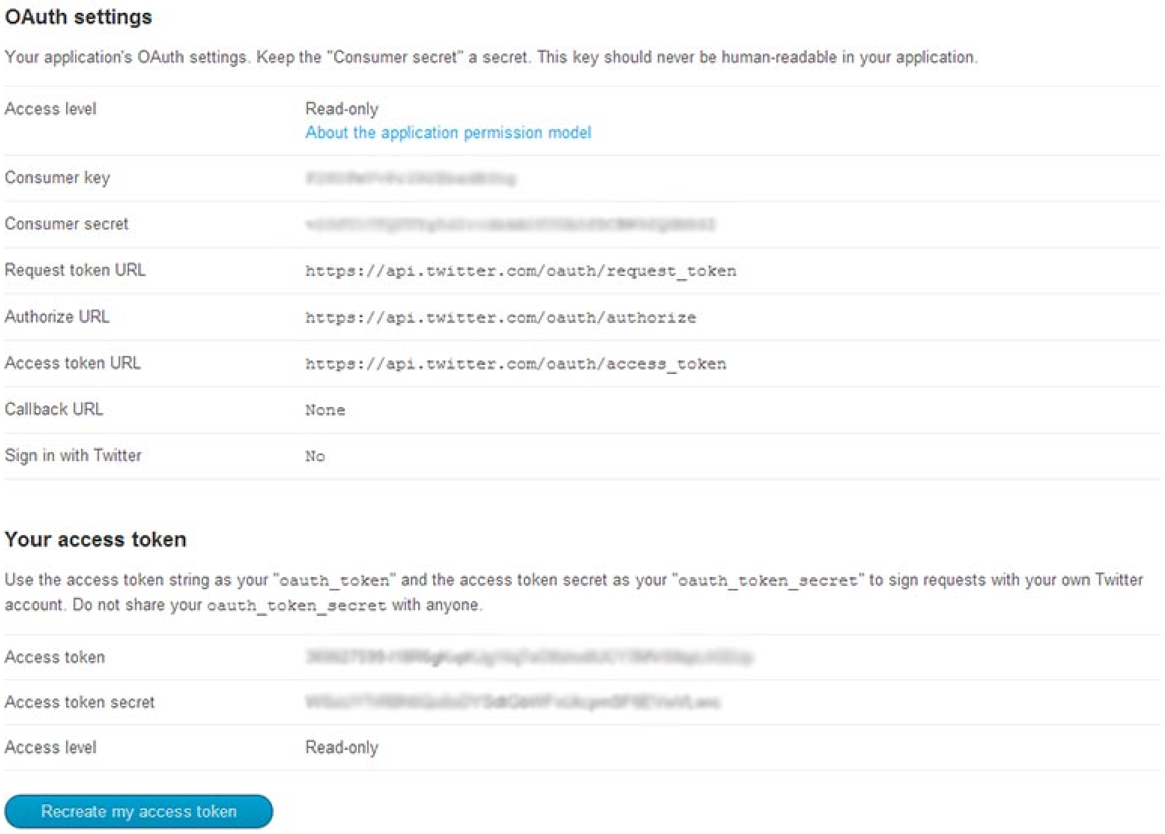
What type of access does your application need? Note: @Anywhere applications require read & write access. Find out more about our Application Permission Model.

>_< Note:

The default access type when you create a Twitter app is read only. As our app will only concerned with reading existing tweets, you can go with the default setting. You can change the access type to "Read and Write" to be able to read Twitter data and send tweets, and select "Read, Write and access direct messages" if you want to have access to your direct messages.

8. Remember your OAUTH Settings

   You can read more about OAUTH here.



OAuth settings

Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

Access level	Read-only About the application permission model
Consumer key	[redacted]
Consumer secret	[redacted]
Request token URL	https://api.twitter.com/oauth/request_token
Authorize URL	https://api.twitter.com/oauth/authorize
Access token URL	https://api.twitter.com/oauth/access_token
Callback URL	None
Sign in with Twitter	No

Your access token

Use the access token string as your "oauth_token" and the access token secret as your "oauth_token_secret" to sign requests with your own Twitter account. Do not share your oauth_token_secret with anyone.

Access token	[redacted]
Access token secret	[redacted]
Access level	Read-only

Make a note of your OAUTH settings. You'll need these in your Python app when communicating with the Twittersphere. You'll need:

- Consumer Key
- Consumer Secret
- OAuth Access Token
- OAuth Access Token Secret

Also, if you intend to use this app in a professional setting, you should keep these settings a secret to avoid having your associated Twitter account being hacked.

>_< Note:

The Twitter API imposes limitations on the amount of data you can use.

- Rate limit

  https://dev.twitter.com/rest/public/rate-limiting

- Agreements

  https://dev.twitter.com/overview/terms/agreement-and-policy

Topic 2 : What's Trending?

Twitter provides REST APIs and Streaming APIs that you can use to interact with their service. As you might imagine, Python has a number of libraries that you can use to interact with Twitter. One of them is Tweepy, which is easy to use and works with Python 3.

Install Tweepy

Let's get started by installing Tweepy:

1. Create a new directory called "twitter" (of course, you can name it anything you like).
2. Run your virtualenv command and give the virtual environment some meaningful name.
3. Install Tweepy as per below:

```
pip install tweepy
```

Create a New Python File

Now, you are ready to start coding your app.

1. Create a file called **`twitter_intro.py`** and save it in the **`twitter`** directory that you just created.
2. The first thing you need to do is give the file access to the Tweepy API. To do this, you add imports to the top of your file. These imports will allow you to the tweepy functionality, including handling OAUTH authentication.

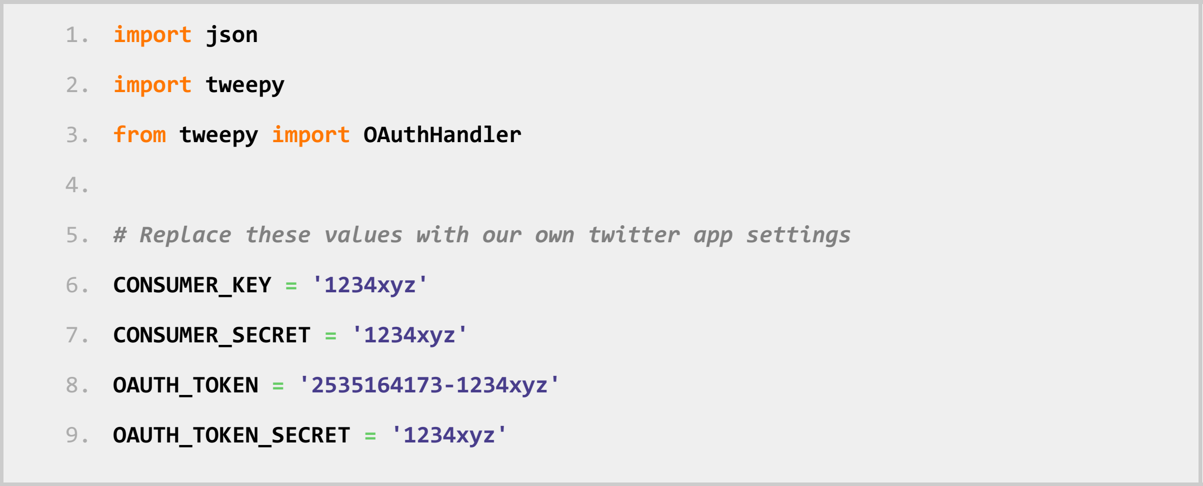
>_< Note:

JSON is a great data format for useful tasks like binding data to visualizations. It's also great for storing the data in files or just a good way to access object attributes and their values.

You can access and parse a tweet easily in JSON format, and Tweepy makes it very simple to do so with just a single change to your code. Because Twitter stream data generally arrives in JSON format, you'll also import JSON directly from python to help you work with the data.

```
import json
import tweepy
from tweepy import OAuthHandler
```

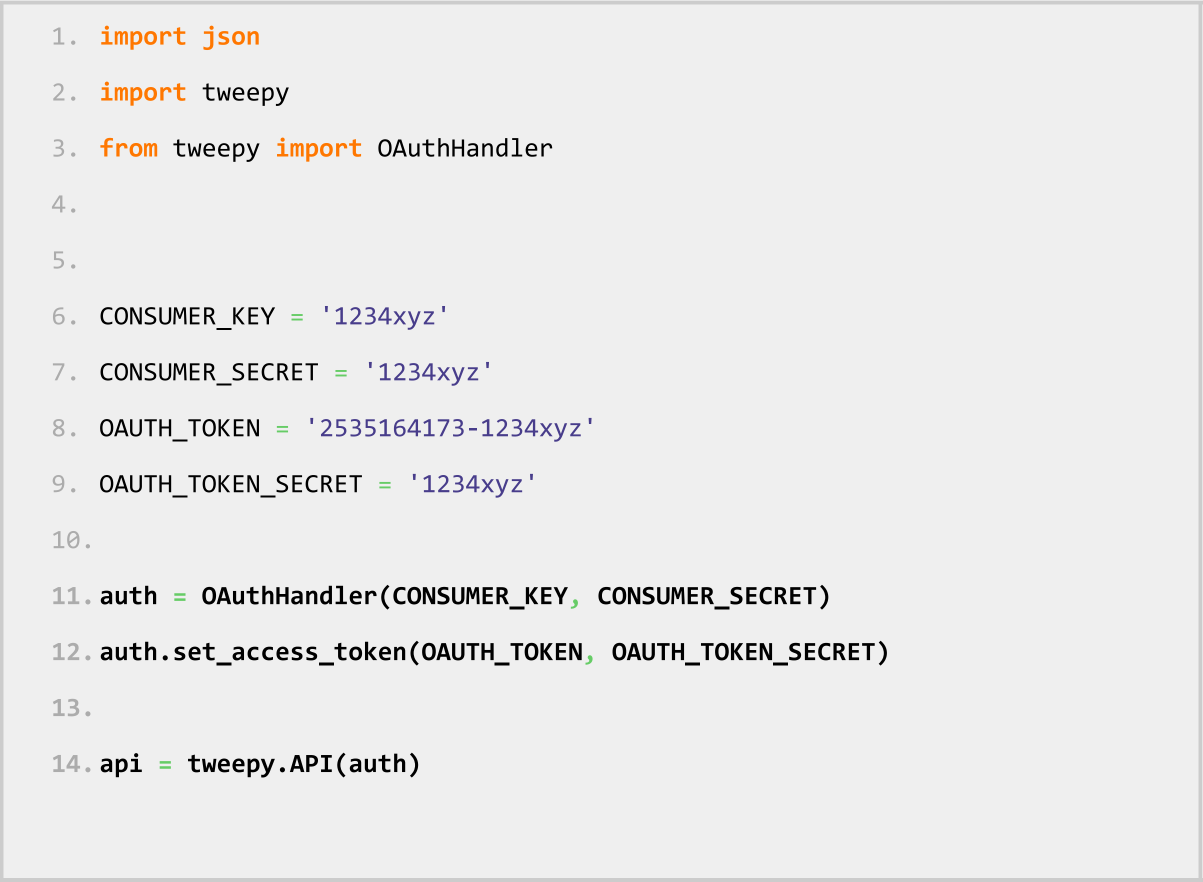
3. Now, you need to include into your app the Twitter API authentication keys that you created earlier. (Replace the "**`123xyz`**" values with your own settings on the API site).



```python
import json
import tweepy
from tweepy import OAuthHandler

# Replace these values with our own twitter app settings
CONSUMER_KEY = '1234xyz'
CONSUMER_SECRET = '1234xyz'
OAUTH_TOKEN = '2535164173-1234xyz'
OAUTH_TOKEN_SECRET = '1234xyz'
```

4. Now that you have the authentication keys in place, you can use them to pull in some tweet data from your account.



```python
import json
import tweepy
from tweepy import OAuthHandler


CONSUMER_KEY = '1234xyz'
CONSUMER_SECRET = '1234xyz'
OAUTH_TOKEN = '2535164173-1234xyz'
OAUTH_TOKEN_SECRET = '1234xyz'

auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN, OAUTH_TOKEN_SECRET)

api = tweepy.API(auth)
```

Line 11: Create an instance of Tweepy's **`OAuthHandler`** class by passing in the **`CONSUMER_KEY`** and **`CONSUMER_SECRET`** values, and assign the instance to the **`auth`** variable.

Line 12: Invoke the **`set_access_token`** function passing in the **`OAUTH_TOKEN`** and **`OAUTH_TOKEN_SECRET`** values as arguments. The **`OAuthHandler`** object now has everything it needs to connect and authenticate with the new Twitter application you just created on the Twitter developers site.

Line 14: Create an instance of the Tweepy API that will do the actual data access. In order for Twitter to allow the access to the API, you pass in the **`OAuthHandler`** object when instantiating it.

5. Now, it's time to get some actual data back from your Twitter account. Let's see what topics are trending right now in our nearest location. The example below uses Dublin, but you can change that to any location you like. Yahoo has created an ever-increasing list of Where On Earth IDs (WOEID) used to identify locations on the planet. Twitter uses these IDs to identify the place origin of its tweets.
    - Read more about WOEID https://developer.yahoo.com/geo/geoplanet/guide/concepts.html
    - Find a WOEID http://woeid.rosselliot.co.nz/lookup/dublin

```
1.  import json
2.  import tweepy
3.  from tweepy import OAuthHandler
4.
5.  # Replace these values with our own twitter app settings
6.  CONSUMER_KEY = '1234xyz'
7.  CONSUMER_SECRET = '1234xyz'
8.  OAUTH_TOKEN = '2535164173-1234xyz'
9.  OAUTH_TOKEN_SECRET = '1234xyz'
10.
```

```
11. auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
12. auth.set_access_token(OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
13. api = tweepy.API(auth)
14.
15. DUB_WOE_ID = 560743
16.
17. dub_trends =  api.trends_place(DUB_WOE_ID)
18.
19. print json.dumps(dub_trends, indent=1)
```

Line 15: Create a constant called **`DUB_WOE_ID`** and assign it the WOEID for Dublin , Ireland.

Line 17: Invoke the **`trends_place`** Tweepy API method passing the **`DUB_WOE_ID`** as an argument, and assign the results to the **`dub_trends`** variable.

Line 19: Print the results to the console. To make it more readable, you format the data using the **`json.dumps`** method.

Build your Python file and, if your code is written correctly and your authentication keys have been copied in correctly you, should get something like below.

```
[
 {
  "created_at": "2015-08-19T09:11:10Z",
  "trends": [
   {
    "url": "http://twitter.com/search?q=%23ShareHumanity",
    "query": "%23ShareHumanity",
    "name": "#ShareHumanity",
    "promoted_content": null
   },
   {
    "url": "http://twitter.com/search?q=%23roseoftralee",
    "query": "%23roseoftralee",
   "name": "#roseoftralee",
   "promoted_content": null
  },
  {
   "url": "http://twitter.com/search?q=%23ShesKindaHotVMA",
   "query": "%23ShesKindaHotVMA",
   "name": "#ShesKindaHotVMA",
   "promoted_content": null
  },
  {
   "url": "http://twitter.com/search?q=%23brilliantirishbands",
   "query": "%23brilliantirishbands",
   "name": "#brilliantirishbands",
   "promoted_content": null
  },
  {
   "url": "http://twitter.com/search?q=%23MakeLifeBetterIn3Words",
   "query": "%23MakeLifeBetterIn3Words",
   "name": "#MakeLifeBetterIn3Words",
   "promoted_content": null
  },
  {
   "url": "http://twitter.com/search?q=%22Michael+Owen%22",
   "query": "%22Michael+Owen%22",
   "name": "Michael Owen",
   "promoted_content": null
  },
  {
   "url": "http://twitter.com/search?q=Depay",
   "query": "Depay",
   "name": "Depay",
   "promoted_content": null
  },
  {
   "url": "http://twitter.com/search?q=Carrick",
   "query": "Carrick",
    "name": "Carrick",
    "promoted_content": null
   },
   {
    "url": "http://twitter.com/search?q=Fellaini",
    "query": "Fellaini",
    "name": "Fellaini",
    "promoted_content": null
   },
   {
    "url": "http://twitter.com/search?q=%22Club+Brugge%22",
    "query": "%22Club+Brugge%22",
    "name": "Club Brugge",
    "promoted_content": null
   }
  ],
  "as_of": "2015-08-19T09:15:17Z",
  "locations": [
   {
    "woeid": 560743,
    "name": "Dublin"
   }
  ]
 }
]
```

Find out more about **`get_places`** here: http://docs.tweepy.org/en/latest/api.html?highlight=trends_place#API.trends_place

Topic 3 : What’s Trending in Multiple Locations?

Now, let’s add another location to our trending places.

```python
import json
import tweepy
from tweepy import OAuthHandler


# Replace these values with our own twitter app settings
CONSUMER_KEY = '1234xyz'
CONSUMER_SECRET = '1234xyz'
OAUTH_TOKEN = '2535164173-1234xyz'
OAUTH_TOKEN_SECRET = '1234xyz'

auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
api = tweepy.API(auth)

DUB_WOE_ID = 560743
LON_WOE_ID = 44418

dub_trends =  api.trends_place(DUB_WOE_ID)
lon_trends = api.trends_place(LON_WOE_ID)

print json.dumps(dub_trends, indent=1)
print json.dumps(lon_trends, indent=1)
```

In the above example, you've added London as an additional location. When you save and run your code, you can look through the JSON output to see if there are common trends between both locations. But there's an easier way…

You can use Python's **set** data structure (https://docs.python.org/2/library/stdtypes.html#set.intersection) to do the work for you. A setwise **`intersection`** computes common items between sets, a setwise **`union`** combines all of the items from sets, and the setwise **`difference`** among sets acts as a subtraction operation in which items from one set are removed from another.

In the example below, you find the intersection of the the two result sets. You also use Python's nifty list comprehension (https://docs.python.org/2/tutorial/datastructures.html#list-comprehensions) mechanism to build the result sets into lists for examination.

```
import json
import tweepy
from tweepy import OAuthHandler


# Replace these values with our own twitter app settings
CONSUMER_KEY = '1234xyz'
CONSUMER_SECRET = '1234xyz'
OAUTH_TOKEN = '2535164173-1234xyz'
OAUTH_TOKEN_SECRET = '1234xyz'

auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
api = tweepy.API(auth)

DUB_WOE_ID = 560743
LON_WOE_ID  =  44418


dub_trends =  api.trends_place(DUB_WOE_ID)
```

```
21. lon_trends = api.trends_place(LON_WOE_ID)
22.
23. dub_trends_set = set([trend['name']
24.                    for trend in dub_trends[0]['trends']])
25.
26. lon_trends_set = set([trend['name']
27.                   for trend in lon_trends[0]['trends']])
28.
 29. common_trends = set.intersection(dub_trends_set, lon_trends_set)
30.
31.
32. print common_trends
```

Line 23/24: Loop through the **`dub_trends`** results and extract the **`name`** attribute for each result and then add it to a set which is then assigned to the **`lon_trends_set`** variable.

Line 26/27: The same is done for **`lon_trends`**.

Line 29: Find trends common to both sets and return these to the common_trends, which is printed out on **line 32**.

> **>_< Note:** You may sometimes get an empty set printed out. That's simply because there happens to be no common trends between the locations at that given moment. Retry a few times and the data may change.

Summary

In this lesson, you learned how create a Twitter application and access it via the twitter API. You used Tweepy to connect to and access data from the API. You also learned how to get lists of trending topics from different WOEID locations and JSON format the output. Finally, you finished up by comparing two sets of result data to see if there were overlapping trends.

Challenge

- Play around with the code you've created.
    - Add new WOEID locations.
    - Add new result sets and find the intersections and differences between them (Hint: You can add more sets to the intersection and difference methods).
- Explore the Twitter API at a high level. Find out about the data limits imposed on the API. https://dev.twitter.com/rest/public/rate-limiting
- Research the Twitter firehose and what it is used for.