

01_search_tweets

September 19, 2016

1 Buscando Tweets

```
In [ ]: import tweepy
import json

consumer_key = ""
consumer_secret = ""
access_token = "-"
access_token_secret = ""

auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)

API = tweepy.API(auth)
```

1.1 GET search/tweets

Devuelve una colección de tweets relevantes que coincidan con una consulta especificada.

API.search(q[, lang][, locale][, rpp][, page][, since_id][, geocode][, show_user])

```
In [ ]: # Buscando la felicidad

query = "#felicidad"

tweets = API.search(query)
for tweet in tweets:
    #print (tweet)
    #print (json.dumps(tweet._json, indent=2))
    js = tweet._json
    t = js['text']
    if t.startswith('RT @'):
        print (t)
        print (json.dumps(js, indent=2, sort_keys=True))
        print ("\n"*3)
        break
    #print (tweet._json['text'])
    #print (tweet.user._json)
```

```

        #print (json.dumps(tweet.user._json, indent=2))
        #print (dir(tweet))

    print ('\n', len(tweets), 'tweets recuperados')

In [ ]: # buscando la felicidad sin amor

query = "felicidad -amor"
tweets = API.search(query)

print (tweets[0].created_at)
print (tweets[0].text)
print (tweets[0].user)
for tweet in tweets:
    #print (tweet)
    #print (json.dumps(tweet._json, indent=2))
    js = tweet._json
    print (js['id'], js['text'])
    print ()

In [ ]: # recuperando hasta 100 tweets

query = "Marc Gasol"
tweets = API.search(query, count=100)

print ('\n', len(tweets), 'tweets recuperados')

In [ ]: # sobrepasando la barrera de los 100 tweets

query = "elecciones"
#MAX = 10000
MAX = 500

cr = tweepy.Cursor(API.search, q=query, count=100).items(MAX) # muy importante
tweets = [t for t in cr] # almacenamos los tweets en una lista para procesarlos
print ('\n', len(tweets), 'tweets recuperados')

In [ ]: dir (cr)
print (type(cr))
print (type(cr.current_page))
print (dir(cr.current_page))

#print (json.dumps(cr.current_page, indent=1))

print (cr.limit)
print (cr.next)
print (cr.num_tweets)

```

```

print (cr.page_index)
print (cr.page_iterator)
print (cr.prev)
print ('-'*20)
print (cr.current_page.max_id)
print (cr.current_page.since_id)
a = [x._json['id'] for x in cr]

In [ ]: limits = API.rate_limit_status()

In [ ]: print (limits['resources']['search'])

In [ ]: # cualquier tweet sobre champions

query = "champions"
tweets = API.search(query, count=50)

for t in tweets:
    j = t._json
    city = '-'
    if j['place'] != None:
        city = j['place']['name']
        print (j['id'], ': ', city, ': ', t.text.replace('\n', ' ').replace('\r', ' '))

print ('\n', len(tweets), 'tweets recuperados')

In [ ]: # cualquier tweet sobre Barcelona

lat_valencia = "39.47" # 39°28'00"N
log_valencia = "0.37" # 0°22'30"O
radio = "80km"
geo = lat_valencia+', '+log_valencia+', '+radio

print (geo)

query = "champions"
tweets = API.search(query, geocode=geo, count=50)

for t in tweets:
    j = t._json
    city = '-'
    if j['place'] != None:
        city = j['place']['name']
        print (j['id'], ': ', city, ': ', t.text.replace('\n', ' ').replace('\r', ' '))

print ('\n', len(tweets), 'tweets recuperados')

In [ ]: # son retweets?

```

```

for tweet in tweets:
    js = tweet._json
    #print (tweet)
    #print (json.dumps(tweet._json, indent=2))
    #print (js['text'])
    print (js['id'], ": retweeted?:", js['retweeted'])
    if 'retweeted_status' in js:
        print (json.dumps(js['retweeted_status'], indent=2))
    #print ('-'*20)
    #print (tweet.user._json)

```

1.2 Se ha retuiteado un tweet vs he retuiteado un tweet

El campo **retweeted** en un tweet recuperado indica si el usuario propietario de la aplicación ha retuiteado el tweet.

La existencia del campo **retweeted_status** en un tweet recuperado indica si el tweet es un retuit.

2 Minando Twitter

2.0.1 Lluís F. Hurtado (lhurtado at dsic.upv.es)

In []: