geoWorldStream.py

```
# -*- coding: UTF-8 -*-
import sys
import time
from urllib2 import URLError
from httplib import BadStatusLine
import twitter
import pymongo
import geocoder


# Auth for Twitter
def oauth_login():

    OAUTH_TOKEN = '477055521-PpCBhLezySPX8CWmSCyUxRmQU7AMZyB5PqyvPNJF'
    OAUTH_TOKEN_SECRET = 'PdPH2nrikeBGiHlfg3vm2dWys6knZSQYYMvnF5xFKy14G'
    CONSUMER_KEY = 'eqOV0Iu2M1114ILJW87nHMoxX'
    CONSUMER_SECRET = '8Gg7KoKpcRJlADXWRXb78PaaHcx6KAIU4Q4sxbeatC0Bpyf9gh'


    # Define Auth
    auth = twitter.oauth.OAuth(OAUTH_TOKEN, OAUTH_TOKEN_SECRET, CONSUMER_KEY, CONSUMER_SECRET)

    # Define Twitter API
    twitter_api = twitter.Twitter(auth=auth)

    return twitter_api

def make_twitter_request(twitter_api_func, max_errors=10, *args, **kw):

    def handle_twitter_http_error(e, wait_period=2, sleep_when_rate_limited=True):

        if wait_period > 3600:  # Seconds
            print >> sys.stderr, 'Too many retries. Quitting.'
            raise e
        # See https://dev.twitter.com/overview/api/response-codes for codes
        if e.e.code == 401:
            print >> sys.stderr, 'Encountered 401 Error (not authorized)'
            return None
        elif e.e.code == 404:
            print >> sys.stderr, 'Encountered 404 Error (not found)'
            return None
        elif e.e.code == 429:
            print >> sys.stderr, 'Encountered 429 Error (rate limit exceeded)'
            if sleep_when_rate_limited:
                print >> sys.stderr, 'Retrying in 15 minutes...ZzZzZzZz...'
                time.sleep(60*15 + 5)
                print >> sys.stderr, '...ZzZzZ...Awake and trying again.'
                return 2
            else:
                raise e  # Caller must handle the rate limiting error
        elif e.e.code in (500, 502, 503, 504):
            print >> sys.stderr, 'Encountered %i Error. Retrying in %i seconds' % (e.e.code, wait_period)
            time.sleep(wait_period)
            wait_period *= 1.5
            return wait_period
        else:
            raise e

        # End of nested helper function

    wait_period = 2
    error_count = 0

    while True:
        try:
            print "frank"
            return twitter_api_func(*args, **kw)
```

```
        except twitter.api.TwitterHTTPError, e:
            error_count = 0
            wait_period = handle_twitter_http_error(e, wait_period)
            if wait_period is None:
                return
        except URLError, e:
            error_count += 1
            print >> sys.stderr, "URLError encountered. Continuing."
            if error_count > max_errors:
                print >> sys.stderr, "Too many consecutive errors...bailing out."
                raise
        except BadStatusLine, e:
            error_count += 1
            print >> sys.stderr, "BadStatusLine encountered. Continuing."
            if error_count > max_errors:
                print >> sys.stderr, "Too many consecutive errors...bailing out."
                raise


def save_to_mongo(data, mongo_db, mongo_db_collection):
    # Connects to the MongoDB server running on
    # localhost: 27017 by default

    # Get a reference to a particular database

    db = client[mongo_db]

    # Reference a particular collection in the database

    coll = db[mongo_db_collection]

    # Perform a bulk insert and return the IDs

    return coll.insert(data)
```

```
def geocode_user_location(location):
    location = location.encode('ascii', 'ignore').decode('ascii')
    g = geocoder.google(location)
    if len(g.latlng) != 0:
        return [g.latlng['lat'], g.latlng['lng']]
    else:
        return [0, 0]

client = pymongo.MongoClient()

# define world bounding box
query = ''
location = '-180,-90,180,90'

twitterAccess = oauth_login()

twitter_stream = twitter.TwitterStream(auth=twitterAccess.auth)

stream = make_twitter_request(twitter_stream.statuses.filter, locations=location)

for response in stream:
    try:
        default = 'null'
        tweet = {}
        tweet['favorited'] = response.get('favorited', default)
        tweet['contributors'] = response.get('contributors', default)
        tweet['text'] = response.get('text', default)
        tweet['in_reply_to_status_id'] = response.get('in_reply_to_status_id', default)
        user = response.get('user', default)
        if user != 'null':
            tweet['user'] = {}
            tweet['user']['id'] = user.get('id', default)
            tweet['user']['followers_count'] = user.get('followers_count', default)
            tweet['user']['listed_count'] = user.get('listed_count', default)
            tweet['user']['utc_offset'] = user.get('utc_offset', default)
```

```
            tweet['user']['statuses_count'] = user.get('statuses_count', default)
            tweet['user']['description'] = user.get('description', default)
            tweet['user']['friends_count'] = user.get('friends_count', default)
            tweet['user']['location'] = user.get('location', default)
            # tweet['user']['geocoded'] = geocode_user_location(tweet['user']['location'])
            tweet['user']['following'] = user.get('following', default)
            tweet['user']['geo_enabled'] = user.get('geo_enabled', default)
            tweet['user']['name'] = user.get('name', default)
            tweet['user']['lang'] = user.get('lang', default)
            tweet['user']['favourites_count'] = user.get('favourites_count', default)
            tweet['user']['screen_name'] = user.get('screen_name', default)
            tweet['user']['account_created'] = user.get('created_at', default)
            tweet['user']['contributors_enabled'] = user.get('contributors_enabled', default)
            tweet['user']['time_zone'] = user.get('time_zone', default)
            tweet['user']['default_profile'] = user.get('default_profile', default)
            tweet['user']['is_translator'] = user.get('is_translator', default)
        tweet['id'] = response.get('id', default)
        tweet['favorite_count'] = response.get('favorite_count', default)
        tweet['lang'] = response.get('lang', default)
        tweet['retweeted'] = response.get('retweeted', default)
        tweet['retweet_count'] = response.get('retweet_count', default)
        tweet['date_tweeted'] = response.get('created_at', default)
        try:
            if response['geo']:
                tweet['geo'] = response.get('geo', default)
                tweet['coordinates'] = response.get('coordinates', default)
                tweet['place'] = response.get('place', default)
                save_to_mongo(tweet, 'geoWorldStreamDatabase', 'geolocated')
            else:
                save_to_mongo(tweet, 'geoWorldStreamDatabase', 'tweets')
        except KeyError:
            pass

    except AttributeError:
        pass
```