IAE 101 – Introduction to Digital Intelligence
Fall, 2019

Documentation for simple_twit, v.3

0. What is simple_twit?

simple_twit is an interface that has been wrapped around the Tweepy Python library package for accessing Twitter's Application Programming Interface (API). The purpose of simple_twit is to give users a simpler, safer, and more restricted way to access some of the methods inside the Tweepy library and so access Twitter functionality from inside their programs.

1. Dependencies

simple_twit depends upon Tweepy in order to function. The Tweepy library must be installed before simple_twit can be used. Tweepy can be installed with the following commands run from the command-line:

On Windows:

1. Open an instance of CMD – the command prompt:
   -If you are having trouble finding it, try:
   Windows key + R to open the run utility, and then enter "cmd" (without the quotes).

2. At the prompt enter: `py -3 -m pip install tweepy --user`

On Mac:

1. Open an instance of the terminal.

2. At the command line enter: `python3 -m pip install tweepy --user`

2. Tweepy Classes

The Tweepy library uses several classes to represent data returned by Twitter in response to requests made through Twitter's API.

2.a. The Status Class

In Twitter terminology "status" or "status update" is the thing that the rest of us all call "tweets". When a tweet is returned as the result of a request sent to Twitter's API through Tweepy, the tweet is contained in an instance of the Status class—a Status object.

The Status object has many attributes that can be used to access significant information about the tweet it represents.

Status Class Attributes:

```
.author – A User object representing the user that wrote
          the tweet.

.favorite_count – An integer that represents how many times
                  the tweet has been liked.

.full_text – A string containing the text of the tweet.

.id – An integer that is a numerical identifier for this
      tweet.

.retweet – A Boolean describing whether this is a retweet
           or not.

.retweet_count – An integer describing how many times the
                 tweet has been retweeted.

.retweeted – A Boolean that represents whether this tweet
             has been retweeted.

.user – A User object representing the user that posted
        this status to your timeline (may not be the same
        as author in the case of retweets).
```

2.b The User Class

The User class is used to represent Twitter's users. It contains information about a single Twitter account holder

User Class Attributes:

```
.created_at - The date the user's account was created.

.description - A string giving the user's description of
               their account.

.favourites_count - An integer giving a count of how many
                    tweets the user has liked.

.followed_by - A Boolean that represents whether the home
               user is followed by this user.

.followers_count - An integer representing how many
                   followers this user has.

.following - A Boolean that represents whether the home
             user is following this user.

.friends_count - An integer representing how many accounts
                 this user follows.

.id - An integer expressing a numerical ID for this user.

.location - A string giving the location of the user.

.name - A string giving the account name of the user

.screen_name - A string giving the screen_name of the user.

.statuses_count - An integer representing how many tweets
                  the user has posted.

.verified - A boolean representing whether the user is
            verified or not.
```

3. simple_twit Functions

A. General Functions

simple_twit.create_api()

This is the initialization function for the simple_twit module. This function must be called before calling any other simple_twit functions.

This function will authorize your program to make requests to Twitter's API through Tweepy on behalf of a specific Twitter user. create_api() will look for home user authorization credentials in a file called "twitterbot.config". If that file exists, then it will read the access token and secret from the file and use them to complete the authorization process. If not, then it will launch a web browser to complete the authorization process, and then store the new access token and secret in the file "twitterbot.config".

When authorization is complete, create_api() will create and initialize a Tweepy API object and return it. This Tweepy API object must be passed as the first argument to other simple_twit functions in order for them to make authorized requests to Twitter's API.

Parameters: None
Return Value: An initialized Tweepy API object

simple_twit.version()

This will print out and return a string stating the version of the simple_twit module being used.

Parameters: None
Return Value: A string stating the current version number.

B. Tweet Functions

simple_twit.send_tweet(api, text)

Updates the authenticating user's current status, also known as Tweeting. If the text duplicates a recent tweet posted by the home user, then this function will return an error. If the user has posted too many tweets recently, then this function will return an error.
Otherwise, the return value is a Status object representing the tweet just posted by this function call.

Parameters:
api : A Tweepy api object; default value is None
text : A string containing the status update text to be posted to Twitter; default value is an empty string.
Return Value: A Status object representing the new tweet.

simple_twit.send_reply_tweet(api, text, tweet_id)

Same as send_tweet(), but in reply to another tweet. The text of the status update must include the @username of the author of the tweet to which this is a reply. The return value is a Status object representing the tweet just posted by this function call.

Parameters:
api : A Tweepy api object; default value is None
text : A string containing the status update text to be posted to twitter; it must include the @username for the author of the tweet to which this tweet is a reply; default value is an empty string.
tweet_id : An int that is a unique identifier for the tweet to which this is a reply.

Return Value: A Status object representing the new tweet.

simple_twit.send_media_tweet(api, text, filename)

Same as send_tweet(), but the status update can include an image. Return value is a Status object representing the tweet just posted by this function call.

Parameters:
api : A Tweepy api object; default value is None
text : A string containing the status update text to be posted to Twitter; default value is an empty string.
filename : A string naming an image file. The image will be included in the status update posted to twitter.

Return Value: A Status object representing the new tweet.

simple_twit.retweet(api, id)

Retweets a tweet. Requires the id of the tweet you are retweeting.

Parameters:
api : A Tweepy api object; default value is None
id : This argument is the numerical identifier for the tweet that will be retweeted by the home user; default value is None
Return Value: A Status object representing the retweet.

simple_twit.get_tweet(api, id)

Returns a single status (a tweet) specified by the ID parameter.

Parameters:
api : A Tweepy api object; default value is None
id : This argument is the numerical identifier for the tweet that will be retrieved; default value is None
Return Value: A Status object representing the retweet.

simple_twit.get_retweets(api, id, count)

Returns up to *count* of the first retweets of a given tweet.

Parameters:
api : A Tweepy api object; default value is None
id : This argument is the numerical identifier for the tweet that will be retrieved; default value is None
count : The number of retweet status objects to retrieve; default value is 20.
Return Value: A Status object representing the retweet.

simple_twit.get_retweeters(api, id, count)

Return up to *count* user ids for users who have retweeted the given tweet. It does not return a list of User objects. It returns the numerical identifiers of the users, which can be used to get the User object for that user.

Parameters:
api : A Tweepy api object; default value is None
id : This argument is the numerical identifier for the tweet that will be retrieved; default value is None
count : The number of user ids to retrieve; the default value is 20.
Return Value: A Status object representing the retweet.

C. Timeline Functions

simple_twit.get_home_timeline(api, count)

Returns the most recent statuses, including retweets, posted by the authenticating user and that user's friends. This is the equivalent of /timeline/home on the Web. The number of statues returned is determined by the value of the count parameter. Tweets are returned as a list of Status objects.

Parameters:
- api : A Tweepy api object; default value is None
- count : An integer that specifies the number of statuses(tweets) to be returned from the home user's timeline; default value is 20.

Return Value: A list of Status objects

simple_twit.get_user_timeline(api, user, count)

Returns the most recent statuses posted from the specified user's timeline. We identify the user by their screen name or account name as a string. We specify how many recent tweets to return through the count parameter.

Parameters:
- api : A Tweepy api object; default value is None
- user : A string giving the account name or screen name of the user whose timeline will be returned; default value is an empty string.
- count : An integer specifying how many recent tweets will be returned from the user's timeline; default value is 20

Return Value: A list of Status objects.

simple_twit.get_retweets_of_me(api, count)

Returns the most recent tweets of yours that have been retweeted by other accounts.

Parameters:
- api : A Tweepy api object; default value is None
- count : An integer that specifies the number of statuses(tweets) to be returned from the home user's timeline; default value is 20.

Return Value: A list of Status objects

simple_twit.get_mentions(api, count)

```
   Returns the most recent mentions of the authenticating
   user, including retweets.

   Parameters:
      api   : A Tweepy api object; default value is None
      count : An integer that specifies the number of
              statuses(tweets) to be returned from the home
              user's timeline; default value is 20.
   Return Value: A list of Status objects
```

D. User Functions

simple_twit.get_user(api, user)

```
Returns information about the specified user.

Parameters:
  api  : A Tweepy api object; default value is None
  user : A string giving the account name or screen name of
         a user.
Return Value: A User object representing data about this
              Twitter account holder.
```

simple_twit.get_user_friends(api, user, count)

```
Returns a user's friends in the order in which they were
added 100 at a time.

Parameters:
  api   : A Tweepy api object; default value is None
  user  : A string giving the account name or screen name
          of a user; default value is an empty string.
  count : An integer that specifies how many of the user's
          friends to return; default value is 100.
Return Value: A list of User objects.
```

simple_twit.get_my_friends(api, count)

```
Returns the home user's friends in the order in which they
were added 100 at a time.

Parameters:
  api   : A Tweepy api object; default value is None
  count : An integer that specifies how many of the home
          user's friends to return; default value is 100.
Return Value: A list of User objects.
```

simple_twit.get_user_followers(api, user, count)

```
Returns a user's followers in the order in which they were
added.

Parameters:
  api   : A Tweepy api object; default value is None
  user  : A string giving the account name or screen name
          of a user; default value is an empty string.
  count : An integer that specifies how many of the user's
          followers to return; default value is 100.
Return Value: A list of User objects.
```

simple_twit.get_my_followers(api, count)

```
Returns a user's followers in the order in which they were
added.

Parameters:
  api   : A Tweepy api object; default value is None
  count : An integer that specifies how many of the user's
          followers to return; default value is 100.
Return Value: A list of User objects.
```

E. Follow and Unfollow Functions

simple_twit.follow_user(api, user)

```
Create a new friendship with the specified user (aka
follow).

Parameters:
  api  : A Tweepy api object; default value is None
  user : A string giving the account name or screen name of
         a user; default value is an empty string.
Return Value: A User object representing the new friend.
```

simple_twit.unfollow_user(api, user)

```
Destroy a friendship with the specified user (aka
unfollow).

Parameters:
  api  : A Tweepy api object; default value is None
  user : A string giving the account name or screen name of
         a user; default value is an empty string.
Return Value: A User object representing the former friend.
```

F. Search Functions

simple_twit.search(api, query, count)

Returns a search object containing information responsive to the query string.

Parameters:
api : A Tweepy api object; default value is None
query : A string describing what you want to search for; default value is an empty string.
count : An integer representing how many search results to return.
Return Value: A SearchResults object representing the response to the search.

simple_twit.search_users(api, query, count)

Returns a list of User objects responsive to the query string.

Parameters:
api : A Tweepy api object; default value is None
query : A string describing what users you want to search for; the default value is an empty string.
Return Value: A list of User objects responsive to your query.

4. Guidelines for Using simple_twit and Tweepy

4A. Rate Limiting

Twitter places restrictions on how many times an application can make requests. These limits are imposed in 15 minute windows—so there is a maximum number of requests per 15 minute interval.

If you exceed the rate limits, your requests will be blocked until the next interval opens (after 15 minutes. If you try to evade the rate limits, Twitter will ban the application from accessing its API.

You are using my application credentials to access Twitter. Please be careful. Keep your tests small and try to space them out. Do not spam Twitter's API with requests.

IMPORTANT: For now, do not put function calls to Twitter's API inside loops.

IMPORTANT: If you use loops, you must slow them down by using the `time.sleep()` function to force your program to wait before executing any more code.

-For example `time.sleep(900)` will force your program to wait 900 seconds (15 minutes) before executing the next line of code.

4B. Acceptable Behavior

Do not post profane, lewd, obscene, malicious, hateful, bigoted, or harassing content to Twitter.

Be very careful about how you interact with other users.

You must follow all of twitter's policies for the use of their API, especially their policies regarding automated behavior:

https://help.twitter.com/en/rules-and-policies/twitter-automation