

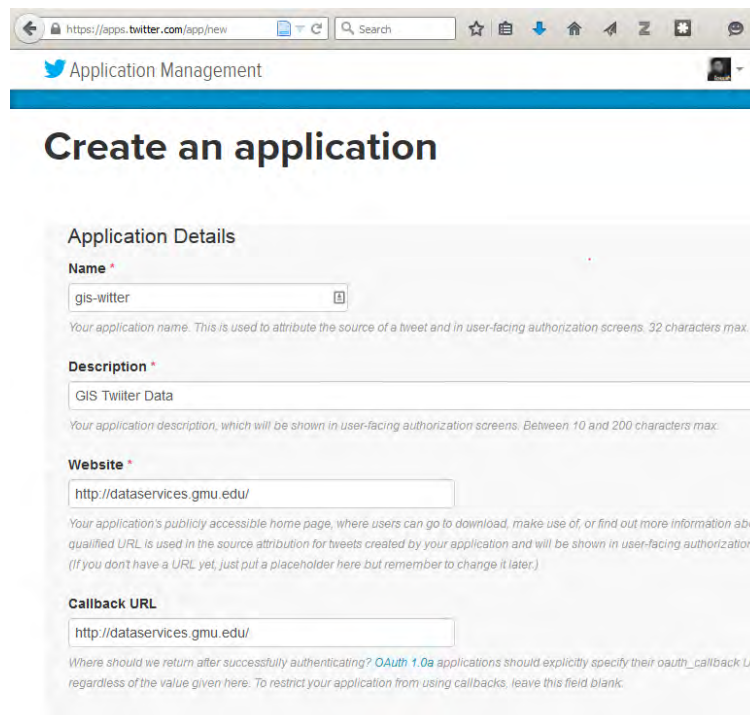
Getting Started with Twitter Data Collection

Note: For this exercises you will need some programming experience and should know how to install libraries. We will need **tweepy** (<http://www.tweepy.org/>) library to install on your computer. You also need get API Keys in your twitter account prior to collect data from twitter. We have installed these on computers in the lab.

The aim of this tutorial is to collect data from twitter with “GeoTagged” information by a keyword(s) and by a geographic area and visualize them using ArcGIS in CSV format.

Steps

1. You need to create account at twitter.com. Create it now (at <https://twitter.com/signup>) if you don't have it now.
2. Once you have a working account, we need few tokens from twitter.
 - a. Go to <https://apps.twitter.com/app/new>



The screenshot shows a web browser window with the URL <https://apps.twitter.com/app/new>. The page title is "Application Management" and the main heading is "Create an application". Below the heading is a form titled "Application Details" with the following fields:

- Name ***: A text input field containing "gis-witter". Below it is a note: "Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max."
- Description ***: A text input field containing "GIS Twitter Data". Below it is a note: "Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max."
- Website ***: A text input field containing "http://dataservices.gmu.edu/". Below it is a note: "Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about 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**: A text input field containing "http://dataservices.gmu.edu/". Below it is a note: "Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL regardless of the value given here. To restrict your application from using callbacks, leave this field blank."

- b. Enter a name for the script that we are going to come up with, for this exercise we need not worry about website and callback url. Use any valid URL as shown above. Agree to the terms and click create application

3. After creation, application management page is displayed.

The screenshot shows the Twitter Application Management interface for an application named 'gis-twitter'. At the top, there is a blue header with the Twitter logo and the text 'Application Management'. Below this, the application name 'gis-twitter' is displayed in a large, bold font. A navigation bar contains four tabs: 'Details', 'Settings', 'Keys and Access Tokens', and 'Permissions'. The 'Keys and Access Tokens' tab is selected and highlighted in blue. Below the navigation bar, there is a profile picture of a blue gear with a Twitter bird, followed by the text 'Just to tryout twitter API' and the URL 'http://dataservices.gmu.edu/'. The 'Organization' section follows, with the text 'Information about the organization or company associated with your application. This information is optional.' Below this, there are two rows of information: 'Organization' with the value 'Developer' and 'Organization website' with the value 'None'. The 'Application Settings' section is next, with the text 'Your application's Consumer Key and Secret are used to authenticate requests to the Twitter Platform.' Below this, there are three rows of settings: 'Access level' with the value 'Read-only (modify app permissions)', 'Consumer Key (API Key)' with a redacted value and a link '(manage keys and access tokens)', and 'Callback URL' with the value 'None'. A blue box highlights the 'Consumer Key (API Key)' field, and a black arrow points from this field to the 'Keys and Access Tokens' tab in the navigation bar.

From this page copy **Consumer Key** to a note pad. Click on “Keys and Access Token Tab”

4. From Keys And Access Tokens, Copy **Consumer Key, Consumer Secret**

gis-twitter

Details Settings **Keys and Access Tokens** Permissions

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application

Consumer Key (API Key)	[Redacted]
Consumer Secret (API Secret)	[Redacted]
Access Level	Read-only (modify app permissions)
Owner	[Redacted]
Owner ID	[Redacted]

[Give Your Own API](#)

5. Now scroll down the same page, will have something as shown below> copy **Access Token, Access Token Secret**

Your Access Token

This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.

Access Token	[Redacted]
Access Token Secret	[Redacted]
Access Level	Read-only
Owner	[Redacted]
Owner ID	[Redacted]

Token Actions

[Regenerate My Access Token and Token Secret](#) [Revoke Token Access](#)

6. Now we have everything that is needed to get started . Go ahead and download the script from https://gist.githubusercontent.com/anupkalburgi/227944dfc5b618e3d087/raw/be17a7685e7636ab8909cb0a3bb344e3a5998adf/twitter_data_collection

After downloading the file, save it to twitter_data_collection.py. And open up in “Idle” editor (in Python)and put in the keys that were copied from the twitter site



```
72 twitter_data_collection.py - C:\Dropbox\Workshop Files\GIS for...
File Edit Format Run Options Windows Help
import tweepy
import json
import csv

# Authentication details. To obtain these visit dev.twitter.com
consumer_key =
consumer_secret =
access_token =
access_token_secret =

# This is the listener, responsible for receiving data

class StdOutListener(tweepy.StreamListener):
    def on_data(self, data):
        # Twitter returns data in JSON format - we need to decode it first
        decoded = json.loads(data)
        if decoded['geo']:
            print "random"
            row = []
            row.append(decoded['id'])
            row.append(decoded['text'].encode('ascii', 'ignore'))
            row.append(decoded['geo']['coordinates'][0])
            row.append(decoded['geo']['coordinates'][1])
            with open("geo_twitter_ebola.csv", "a+") as fp:
                writer = csv.writer(fp)
                writer.writerow(row)
            print "Write Done"
            return True
    def on_error(self, status):
        print "Error"
        print status

if __name__ == '__main__':
    l = StdOutListener()
    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)

    # There are different kinds of streams: public stream, user stream, multi-user stre
    # In this example follow #programming tag
    # For more details refer to https://dev.twitter.com/docs/sr
    stream = tweepy.Stream(auth, l)
    stream.filter(track=['ipl'])
    #stream.filter(track=["django"])
    #locations=[-121.37,27.241,-64.07,48.30]
```

Pick a proper filename.

Type a keyword in the text field that you want to harvest

Type a location boundary

Ln: 14, Col: 27

After getting the variables right, got to **Run** (green circle) and click on Run Module. That should start the stream. You might see SSL related warnings coming up, we don't worry about those for this given exercise

NOTE: This will bring you #hashtag word (eg, #django) and any word with “Django”

Locations field is to filter a geographic boundary box to populate twitter data.