Collecting User Info

User Profiles, Friends & Followers

Outline

- User Data in Tweet record
- HCDE user module UserLookup.py
- HCDE user module FriendsFollowers.py

Tweet JSON Structure

- When requesting tweets the JSON structure includes some user data
- The "user" structure within the tweet data
- Consider the sample_tweets_json.txt file

Almost everything

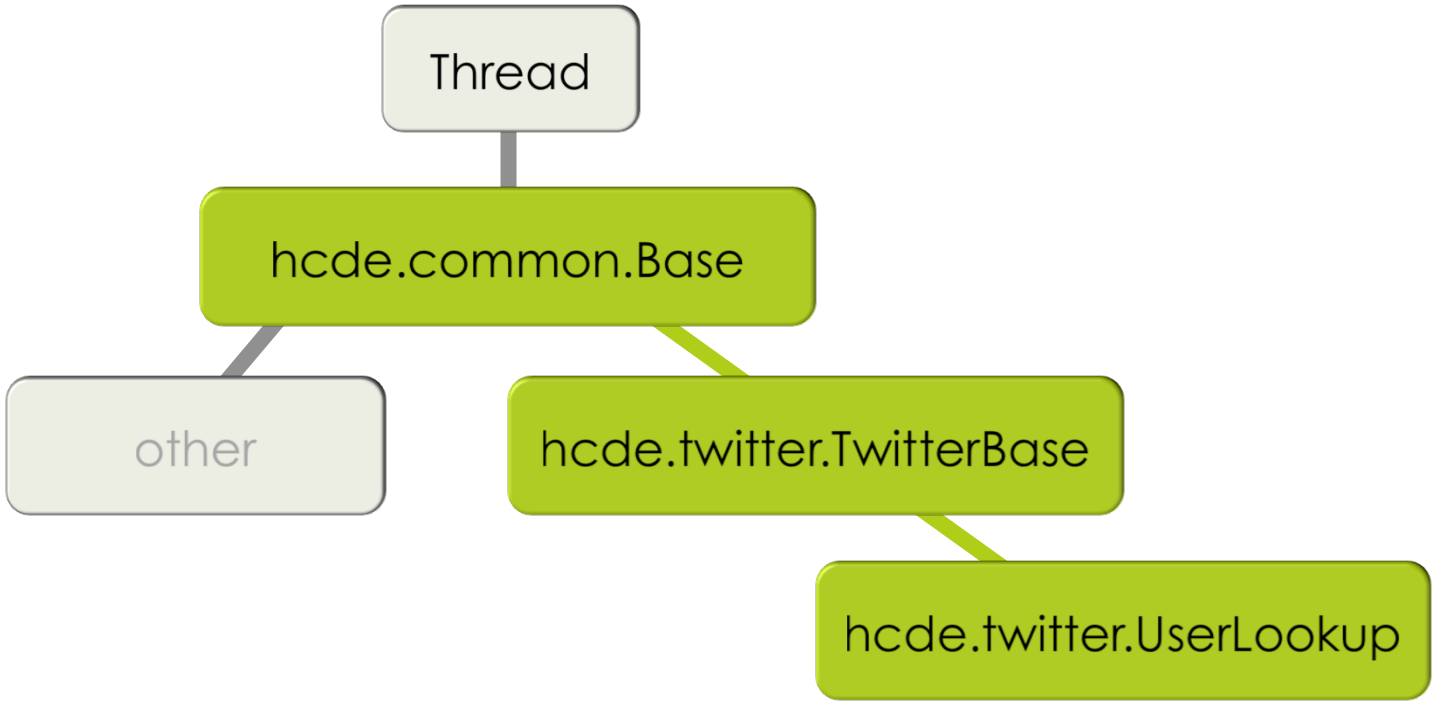

```
"user": {
    "contributors_enabled": false,
    "created_at": "Sat Apr 09 02:08:26 +0000 2011",
    "default_profile": true,
    "default_profile_image": false,
    "description": "Just a young White boy trying to live a life like Young Jeezy.",
    "entities": {
      "description": {
        "urls": []
      }
    },
    "favourites_count": 1097,
    "follow_request_sent": false,
    "followers_count": 310,
    "following": false,
    "friends_count": 400,
    "geo_enabled": false,
    "id": 279330988,
    "id_str": "279330988",
    "is_translator": false,
    "lang": "en",
    "listed_count": 1,
    "location": "",
    "name": "Michael Dropsit",
    "notifications": false,
    "profile_background_color": "C0DEED",
    "profile_background_image_url": "http://abs.twimg.com/images/themes/theme1/bg.png",
    "profile_background_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.png",
    "profile_background_tile": false,
    "profile_banner_url": "https://pbs.twimg.com/profile_banners/279330988/1387392271",
    "profile_image_url": "http://pbs.twimg.com/profile_images/421677287676080128/ZObRiBf3_normal.jpeg",
    "profile_image_url_https": "https://pbs.twimg.com/profile_images/421677287676080128/ZObRiBf3_normal.jpeg",
    "profile_link_color": "0084B4",
    "profile_sidebar_border_color": "C0DEED",
    "profile_sidebar_fill_color": "DDEEF6",
    "profile_text_color": "333333",
    "profile_use_background_image": true,
    "protected": false,
    "screen_name": "MichaelDros",
    "statuses_count": 10490,
    "time_zone": "Eastern Time (US & Canada)",
    "url": null,
    "utc_offset": -18000,
    "verified": false}
```

Embedded Objects

- User data in the tweet is an embedded object
  - Other embedded objects, “media”, “user_mentions”, “retweeted_status” …
- Embedded user data may be enough
- These may be stale, out of date
  - Useful, but you may not want to trust this data

Reviewing UserLookup.py code

- UserLookup.py object hierarchy



Command Line usage

- Part of the HCDE User Module
  - Look in hcde/twitter

- Like the other code get a little help – run without command line parameters

```
python UserLookup.py

USAGE: python UserLookup.py -auth <appname> -user <auth_user> [-n <username>
| -id <userid>] [-json] [-limits] [-test names|ids|both]
```

User Lookup API Specification

- Twitter Dev Site
  - https://dev.twitter.com/docs/api/1.1/get/users/lookup
- Returns full user data for up to 100 users in one request
- Two possible parameters
  - user_id
  - screen_name

Demo (setup oauth login)

```
# You should all be really used to this by now
import sys, json
from hcde.twitter.Login import Login
from hcde.twitter.UserLookup import UserLookup
from hcde.twitter.auth_settings import *
app = "HCDE530Test01"
user = "dwmcphd"
app_keys = TWITTER_APP_OAUTH_PAIR(app=app)
app_token_fname = TWITTER_APP_TOKEN_FNAME(app=app)
lg = Login(app_name=app, app_user=user, token_fname=app_token_fname)
lg.set_consumer_key(consumer_key=app_keys['consumer_key'])
lg.set_consumer_secret(consumer_secret=app_keys['consumer_secret'])
lg.login()
```

Demo (setup UserLookup)

```
# Assuming imports and lg (Login object from prior page)
ul = UserLookup()
ul.set_auth_obj(obj=lg) # all Twitter requests require auth
ul.set_user_agent(agent="safari")
ul.set_throttling(True)
# You can add up to 100 usernames or user ids
ul.add_username("ThePSF")
ul.add_username("jimmyfallon")
ul.add_username("timoreilly")
# Make the request
ul.make_request()
# like the lecture demos before, no threading, just a request
```

Demo (getting profile results)

```
# Check for messages
print ul.messages()
# Get the message from the message queue
resp = ul.get_message()
# The response is a list of dictionary items, json response was converted
print len(resp)
#
for u in resp:
          print json.dumps(u,indent=4,sort_keys=True)
```

Demo (add more users)

```
# You can add up to 100 usernames or user ids
ul.add_username("dwmcphd")
# Make the next request
ul.make_request()
print ul.messages()
resp = ul.get_message()
print len(resp)
print json.dumps(resp[0],indent=4,sort_keys=True)
```

Excerpt of the profile response

- Consider friends and followers (jan 2014)

```
"screen_name": "ThePSF",
"followers_count": 38126,
"friends_count": 124,
#
"screen_name": "jimmyfallon",
"followers_count": 11462845,
"friends_count": 5417,
#
"screen_name": "timoreilly",
"followers_count": 1748270,
"friends_count": 1178,
#
"screen_name": "dwmcphd",
"followers_count": 349,
"friends_count": 159,
```

Excerpt of the profile response

- Consider friends and followers (feb 2016)

```
#
# extracted from a –json dump using UserLookup.py at the command line
#
"name": "Python Software",
"screen_name": "ThePSF",
"followers_count": 98305,
"friends_count": 120,
#
"name": "David McDonald",
"screen_name": "dwmcphd",
"followers_count": 483,
"friends_count": 171,
```

Excerpt of the profile response

- Consider friends and followers (feb 2016)

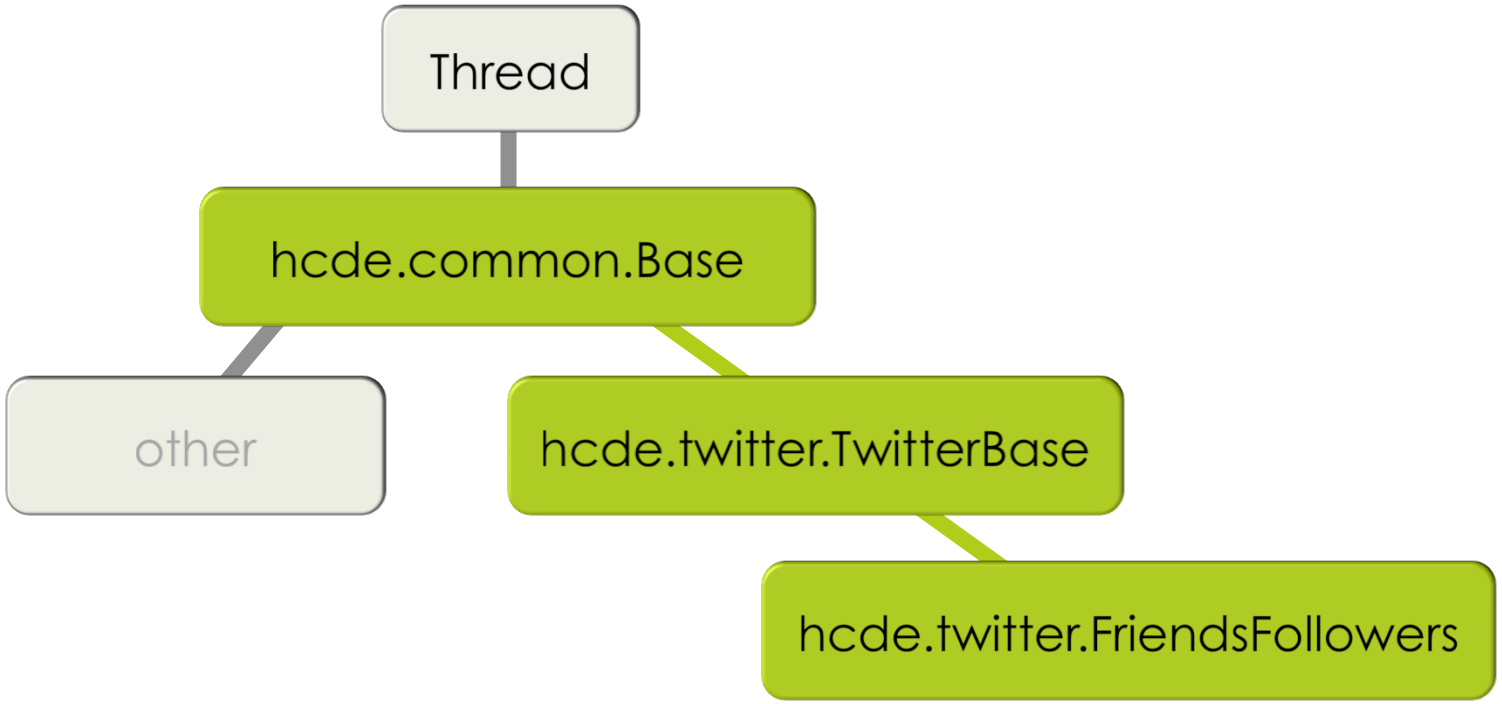
```
#
"name": "jimmy fallon",
"screen_name": "jimmyfallon",
"followers_count": 35595222,
"friends_count": 6783,
#
"name": "Tim O'Reilly",
"screen_name": "timoreilly",
"followers_count": 1964609,
"friends_count": 1536,
#
"name": "Barack Obama",
"screen_name": "BarackObama",
"followers_count": 69922822,
"friends_count": 637782,
```

Friends & Followers

- The user profile contains a count of friends and a count of followers
  - The counts could be useful as a simple (simplistic) measure of a users' influence
  - follower_count / (friend_count + follower_count)
    - dwmcphd = (483/(171+483)) = 483/654 = 0.7385
    - jimmyfallon = (35595222/35602005) = 0.9998
- How do you get the list of people who are friends/ followers?

Reviewing FriendsFollowers.py code

- FriendsFollowers.py object hierarchy



Command Line usage

- Part of the HCDE User Module
  - Look in hcde/twitter

- Like the other code get a little help – run without command line parameters

```
python FriendsFollowers.py

USAGE: python FriendsFollowers.py -auth <appname> -user <auth_user> [-friends | -followers] -n <username> | -id <userid> [-count <count_per_request>] [-json]
```

User Lookup API Specification

- Twitter Dev Site
  - https://dev.twitter.com/docs/api/1.1/get/friends/ids
  - https://dev.twitter.com/docs/api/1.1/get/followers/ids

- Request by either user_id or screen_name
- Returns up to 5000 friends/followers per response
- Cursor to page through the friends/followers

Consider Rate Limits

- Our five users from before:

```
ThePSF "followers_count": 98305
jimmyfallon "followers_count": 35595222
timoreilly "followers_count": 1964609
barackobama"followers_count": 69922822
dwmcphd "followers_count": 483
```

- Consider “ThePSF” a little over 98K followers
  - 98305/5000 = 20 requests
  - 20 minutes, best case
  - Assuming current Twitter limits
    - 15 minute windows, 15 requests per window

Consider Rate Limits

- Consider “timoreilly” about 1.9 million followers
  - 1964609/5000 = 393 requests
  - 393 minutes = 6.75 hours (best case)
- Jimmy Fallon 35.5 million followers
  - 35,595,222/5000 = 7120 = 118.75 hours = 4.95 days
- Barack Obama 69.9 million followers
  - 69,922,822/5000 = 13984.75 = 582.75 hours = 24.3 days

Demo (setup oauth login)

```
import sys, json
from hcde.twitter.Login import Login
from hcde.twitter.FriendsFollowers import FriendsFollowers
from hcde.twitter.auth_settings import *
app = "HCDE530Test01"
user = "dwmcphd"
app_keys = TWITTER_APP_OAUTH_PAIR(app=app)
app_token_fname = TWITTER_APP_TOKEN_FNAME(app=app)
lg = Login(app_name=app, app_user=user, token_fname=app_token_fname)
lg.set_consumer_key(consumer_key=app_keys['consumer_key'])
lg.set_consumer_secret(consumer_secret=app_keys['consumer_secret'])
lg.login()
```

Demo (setup FriendsFollowers)

```
# Assuming imports and lg (Login object from prior page)
ff = FriendsFollowers()
ff.set_auth_obj(obj=lg) # all Twitter requests require auth
ff.set_user_agent(agent="ie")
# The object implements BOTH friends and follower requests
ff.set_request_type_as_friends()
#ff.set_request_type_as_followers()
ff.set_throttling(True)
ff.set_count(30)
ff.set_username("dwmcphd")
print ff.get_rate_limit()
ff.make_request()
print ff.messages()
resp = ff.get_message()
print ff.messages()
print json.dumps(resp["ids"],indent=4,sort_keys=True)
```

Caveats

- Current implementation of FriendsFollowers.py
  - Will request *all* friends/followers with the cursor
- Friends and Followers requests have very strict rate limits
  - Max 15 requests in each 15 minute window
- With throttling turned on (the default) code will make about 1 request per minute
  - If you know that you're requesting fewer than 75,000 then you could just go for it
  - but then you'll have to wait 15 minutes

Revisit collect_tweets.py

- hcde/data/example
- Current collection
  - tweets
  - minimal user records
- Could modify this to collect the full user record with a UserLookup() object
- Could modify to collect a list of friends/followers with FriendsFollowers() object – would need to hook to the Friends/Followers table in the DB