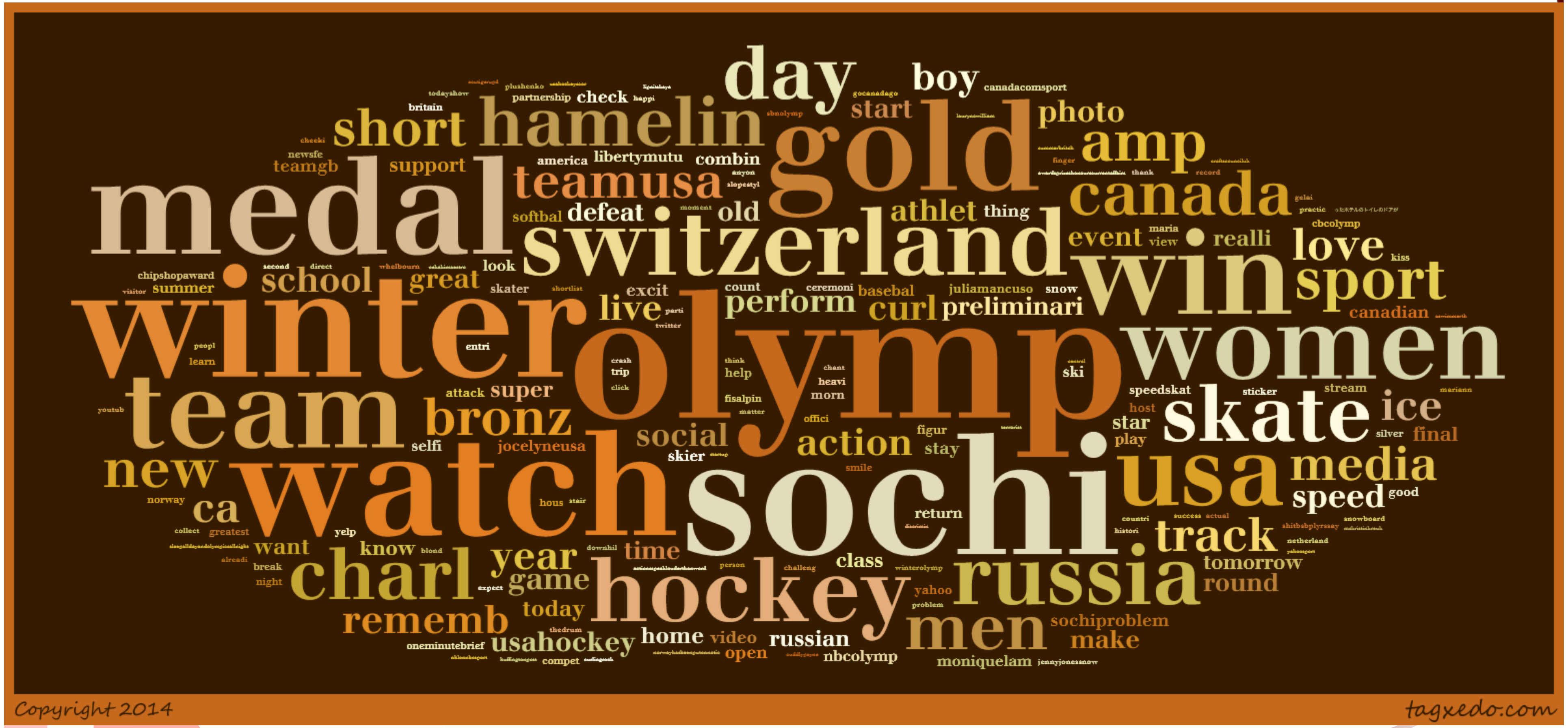
Social Computing
제 3 강
Facebook

2014/3/21

주요일정

- 3월 28일 광교에서 수업
  - 광교 융대원 D동 122호
  - 1:30분 관악 출발 버스가 도착하면 시작 (약 2시20분쯤 예상)
- 4월 18일 중간고사
- 5월 2일 휴강: CHI 학회, CHI 주요논문 video 시청 report로 대체 예정
- 5월 16일 휴강: 융대원 행사
- 6월 6일 휴강: 현충일
- 6월 20일 기말고사

Goal: Build a Word Cloud Summarizing 500 Tweets



Examining Patterns in Retweets

```
status_texts = [ status['text']
              for status in statuses ]
```

Collect text

```
screen_names = [ status['user']['screen_name']
              for status in statuses ]
```

Collect screen names

```
hashtags = [ hashtag['text']
           for status in statuses
              for hashtag in status['entities']['hashtags'] ]
```

Collect hashtags

```
# Compute a collection of all words from all tweets
words = [ w
        for t in status_texts
           for w in t.split() ]
```

Collect words in tweets

```
print json.dumps(status_texts[0:5], indent=1)
print json.dumps(screen_names[0:5], indent=1)
print json.dumps(hashtags[0:5], indent=1)
print json.dumps(words[0:5], indent=1)
```

지난주 숙제

- 오늘 배운 것을 바탕으로 다음을 수행하세요.

1. Word Cloud
   - 개인의 기호에 따라 영어 서치 키워드를 정하고 해당 트윗을 500개 이상 수집
   - 수업에 사용된 방법에 따라서 트윗 Word Cloud 작성
   - Word Cloud를 이미지로 저장/캡처해서 wordcloud.jpg로 제출
   - 해당 코드를 wordcloud.py로 저장해서 제출

2. Retweet 통계
   - 수집된 트윗을 분석하여 가장 많이 retweet된 트윗 10개를 prettytable을 이용해 리트윗 횟수, 원 작성자, 트윗내용 등을 출력
   - 해당 코드를 retweet.py 저장해서 제출
   - 결과를 이미지로 screen capture 해서 retweet.jpg로 제출

3. hashtag 통계
   - 수집된 트윗에서 몇 %의 트윗이 hashtag을 포함하고 있는가?
   - 가장 많이 사용된 hashtag 10개를 prettytable을 사용하여 빈도수와 더불어 출력
   - 해당 코드를 hashtag.py 저장해서 제출
   - 결과를 이미지로 screen capture 해서 hashtag.jpg로 제출

- 3월20일 자정까지 모든 파일을 zip으로 압축해서 etl 을 통해 제출

Having Fun with Facebook API

- Facebook Graph API
- Access Data Using the Graph API

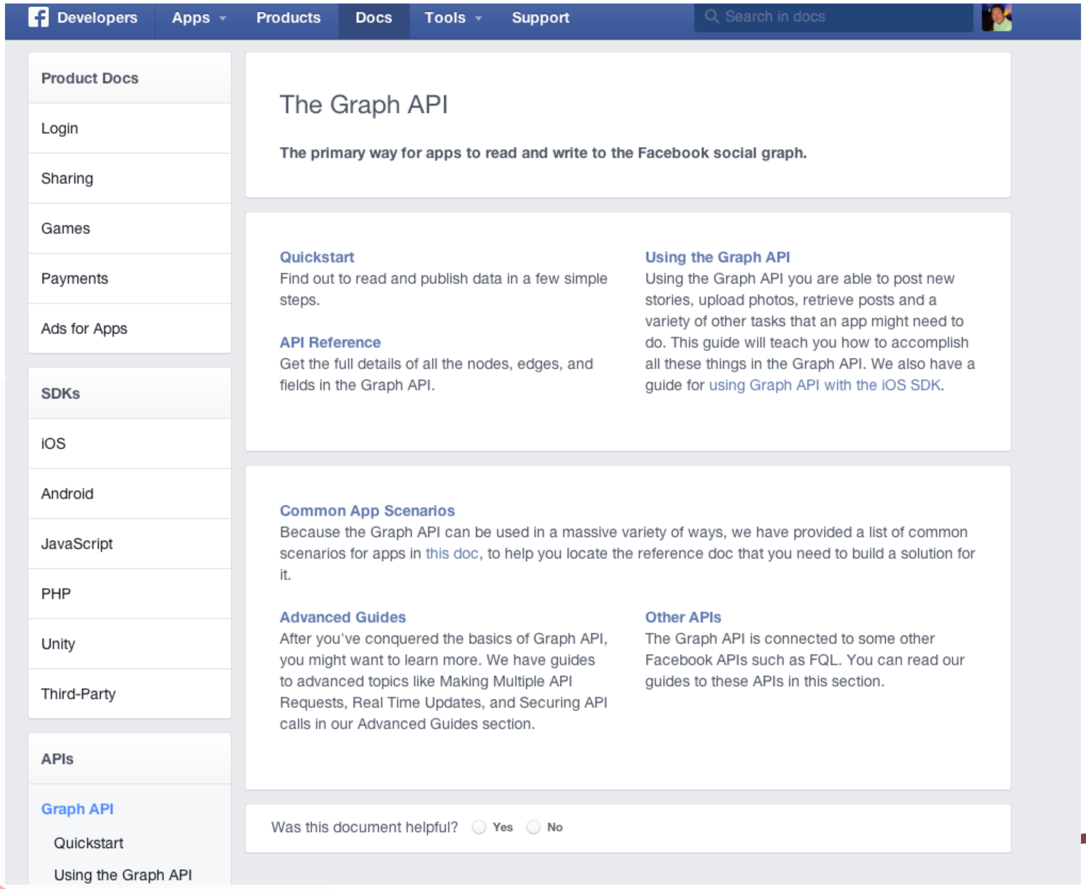
- Open Graph Objects
- Comparing two fan pages

- Friends' Likes
- Common Likes

- Mutual Friendships
- Cliques
- Visualizing a Mutual Friendship Graph

Facebook Graph API

- API란?
- Facebook Social Graph를 읽고 쓰는 API
- https://developers.facebook.com/docs/graph-api



The Graph API

The primary way for apps to read and write to the Facebook social graph.

Quickstart
Find out to read and publish data in a few simple steps.

API Reference
Get the full details of all the nodes, edges, and fields in the Graph API.

Using the Graph API
Using the Graph API you are able to post new stories, upload photos, retrieve posts and a variety of other tasks that an app might need to do. This guide will teach you how to accomplish all these things in the Graph API. We also have a guide for using Graph API with the iOS SDK.

Common App Scenarios
Because the Graph API can be used in a massive variety of ways, we have provided a list of common scenarios for apps in this doc, to help you locate the reference doc that you need to build a solution for it.

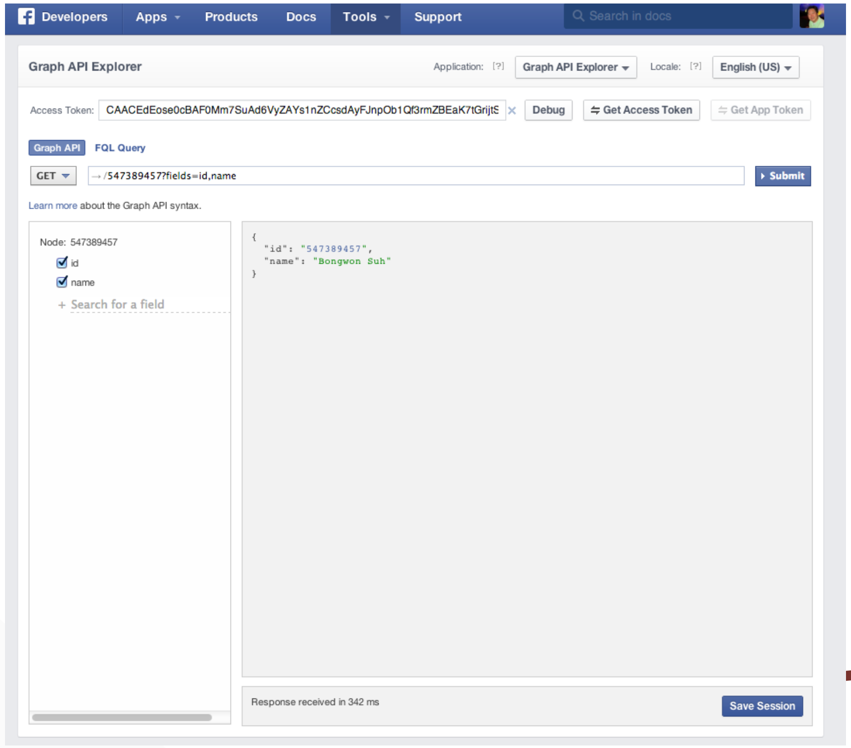
Advanced Guides
After you've conquered the basics of Graph API, you might want to learn more. We have guides to advanced topics like Making Multiple API Requests, Real Time Updates, and Securing API calls in our Advanced Guides section.

Other APIs
The Graph API is connected to some other Facebook APIs such as FQL. You can read our guides to these APIs in this section.

Was this document helpful? Yes No

Facebook Graph Explorer

- API를 직접 수행시키기 전에 원하는 내용이 실제로 접속되는지 확인해본다!
- https://developers.facebook.com/tools/explorer
- 나의 이메일 등록정보 보기
- 친구리스트 보기
- 친구들의 이름과 생년월일 보기



Access Token: CAACEdEose0cBAF0Mm7SuAd6VyZAYs1nZCcsdAyFJnpOb1Qf3rmZBEaK7tGrijtS

→ /547389457?fields=id,name

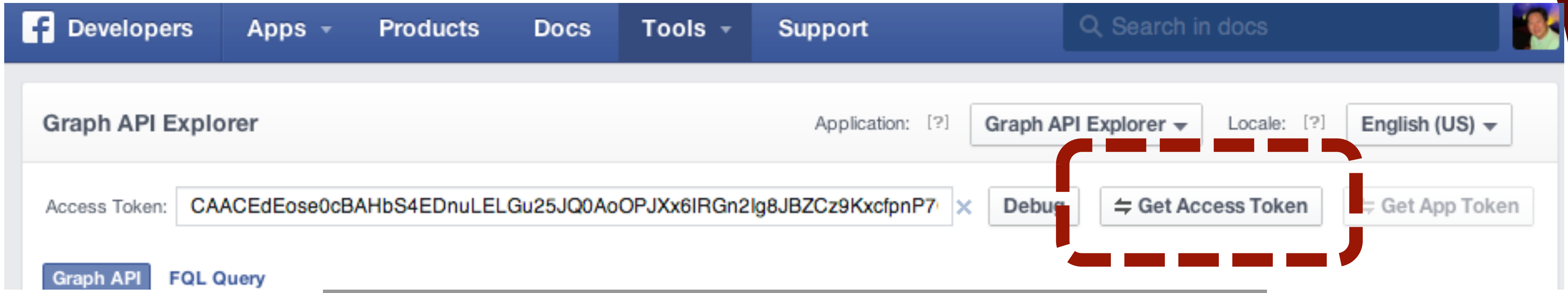
Learn more about the Graph API syntax.

Node: 547389457
- ✓ id
- ✓ name

```
{
  "id": "547389457",
  "name": "Bongwon Suh"
}
```

Response received in 342 ms

Select Permissions

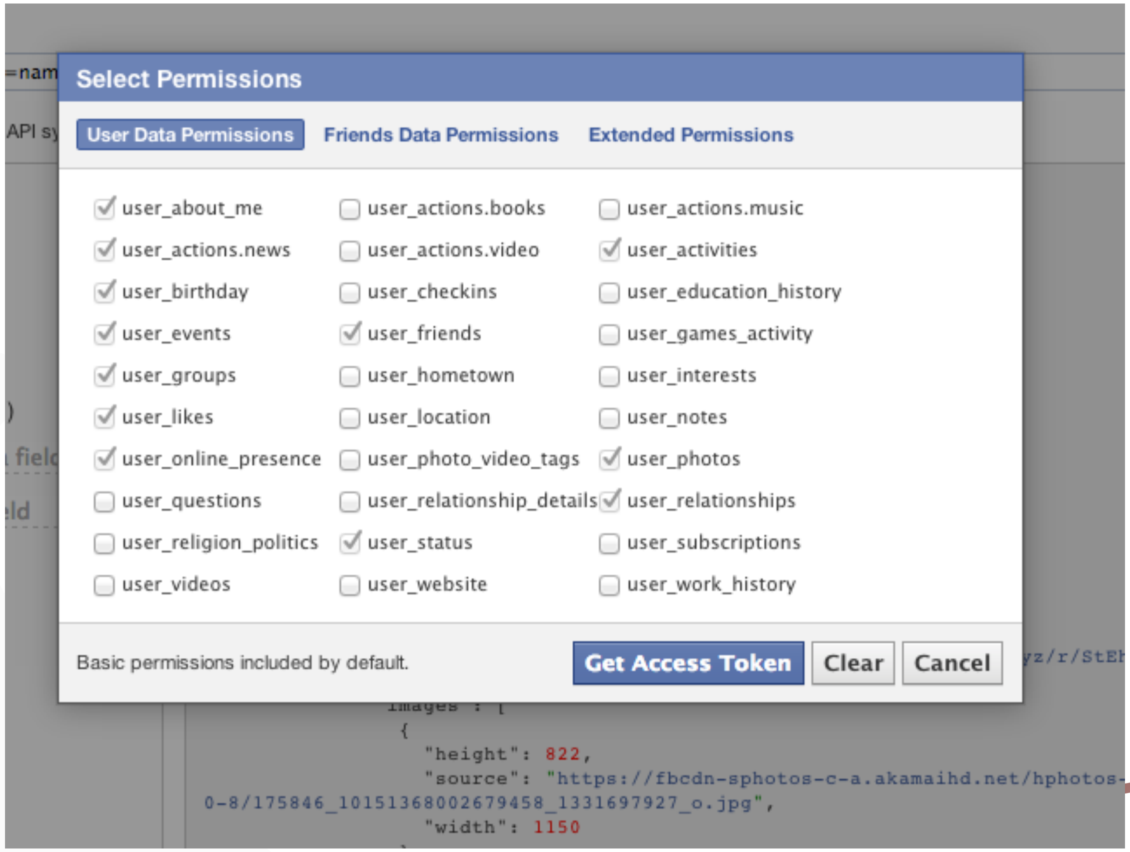


Graph API Explorer

Application: [?] Graph API Explorer

Locale: [?] English (US)

Access Token: CAACEdEose0cBAHbS4EDnuLELGu25JQ0AoOPJXx6lRGn2lg8JBZCz9KxcfpnP7

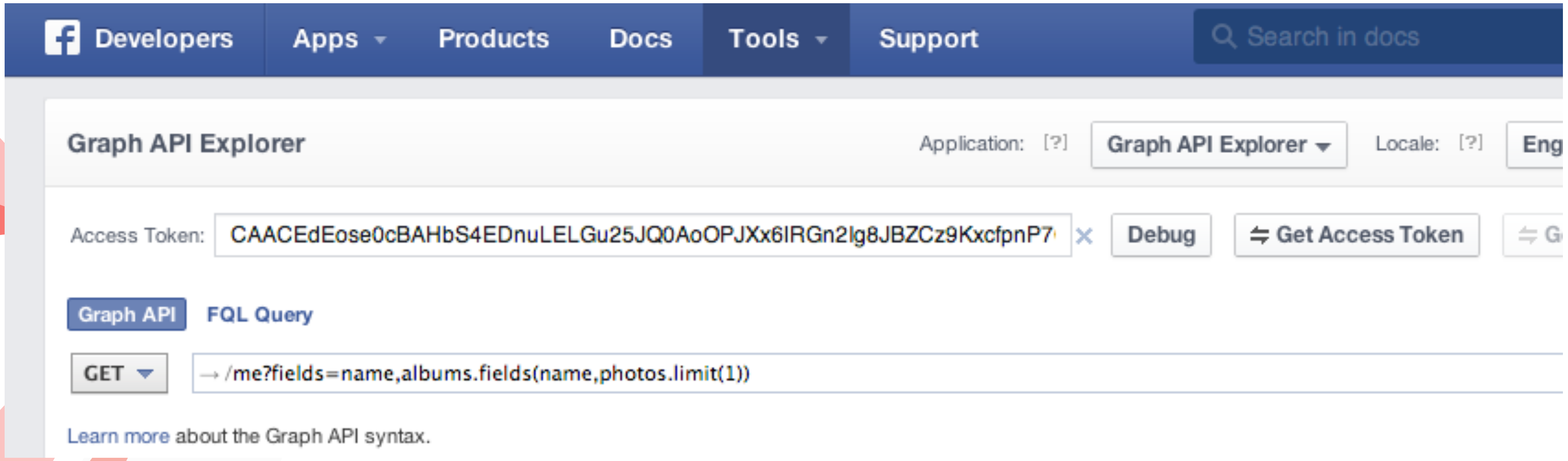

Select Permissions

✓ user_about_me	☐ user_actions.books	☐ user_actions.music
✓ user_actions.news	☐ user_actions.video	✓ user_activities
✓ user_birthday	☐ user_checkins	☐ user_education_history
✓ user_events	✓ user_friends	☐ user_games_activity
✓ user_groups	☐ user_hometown	☐ user_interests
✓ user_likes	☐ user_location	☐ user_notes
✓ user_online_presence	☐ user_photo_video_tags	✓ user_photos
☐ user_questions	☐ user_relationship_details	✓ user_relationships
☐ user_religion_politics	✓ user_status	☐ user_subscriptions
☐ user_videos	☐ user_website	☐ user_work_history

Basic permissions included by default.

```
"images": [
  {
    "height": 822,
    "source": "https://fbcdn-sphotos-c-a.akamaihd.net/hphotos-
0-8/175846_10151368002679458_1331697927_o.jpg",
    "width": 1150
```

Get Access Token



Access Token: CAACEdEose0cBAHbS4EDnuLELGu25JQ0AoOPJXx6IRGn2lg8JBZCz9KxcfpnP7

→ /me?fields=name,albums.fields(name,photos.limit(1))

Learn more about the Graph API syntax.

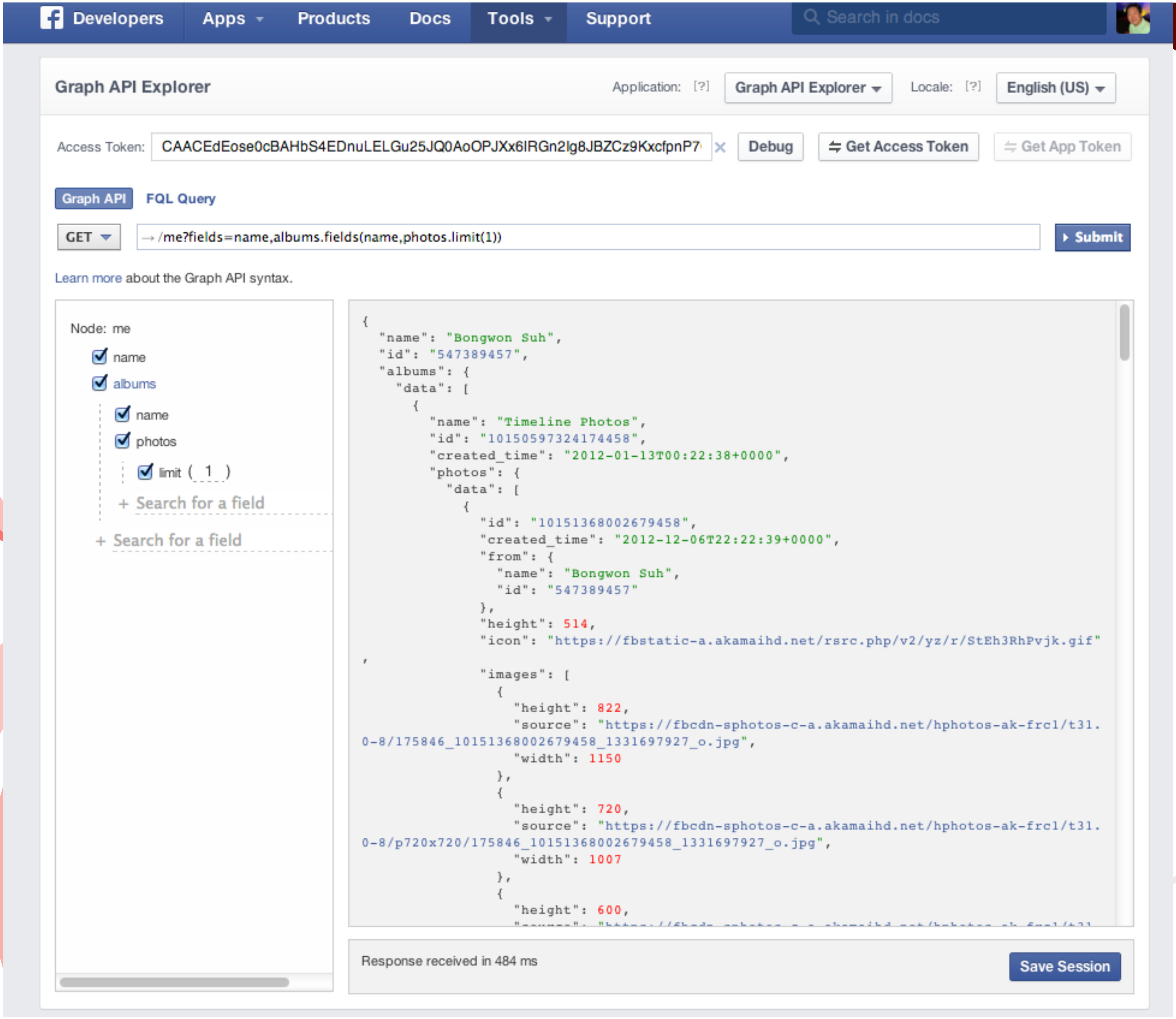
Using Facebook Graph API

- https://developers.facebook.com/docs/graph-api/using-graph-api/

- GET graph.facebook.com
- /{node-id}? fields={first-level}.fields({second-level})

- GET graph.facebook.com
- /me?fields=albums.limit(5),posts.limit(5)

- GET graph.facebook.com
- /me?fields=albums.limit(5).fields(name, photos.limit(2)),posts.limit(5)



Access Token: CAACEdEose0cBAHbS4EDnuLELGu25JQ0AoOPJXx6IRGn2Ig8JBZCz9KxcfpnP7

/me?fields=name,albums.fields(name,photos.limit(1))

Learn more about the Graph API syntax.

```
Node: me
  ☑ name
  ☑ albums
    ☑ name
    ☑ photos
      ☑ limit ( 1 )
    + Search for a field
  + Search for a field
```

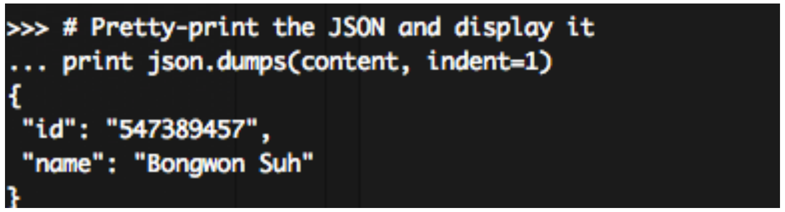
```
{
  "name": "Bongwon Suh",
  "id": "547389457",
  "albums": {
    "data": [
      {
        "name": "Timeline Photos",
        "id": "10150597324174458",
        "created_time": "2012-01-13T00:22:38+0000",
        "photos": {
          "data": [
            {
              "id": "10151368002679458",
              "created_time": "2012-12-06T22:22:39+0000",
              "from": {
                "name": "Bongwon Suh",
                "id": "547389457"
              },
              "height": 514,
              "icon": "https://fbstatic-a.akamaihd.net/rsrc.php/v2/yz/r/StEh3RhPvjk.gif"
,
              "images": [
                {
                  "height": 822,
                  "source": "https://fbcdn-sphotos-c-a.akamaihd.net/hphotos-ak-frc1/t31.
0-8/175846_10151368002679458_1331697927_o.jpg",
                  "width": 1150
                },
                {
                  "height": 720,
                  "source": "https://fbcdn-sphotos-c-a.akamaihd.net/hphotos-ak-frc1/t31.
0-8/p720x720/175846_10151368002679458_1331697927_o.jpg",
                  "width": 1007
                },
                {
                  "height": 600,
```

Response received in 484 ms

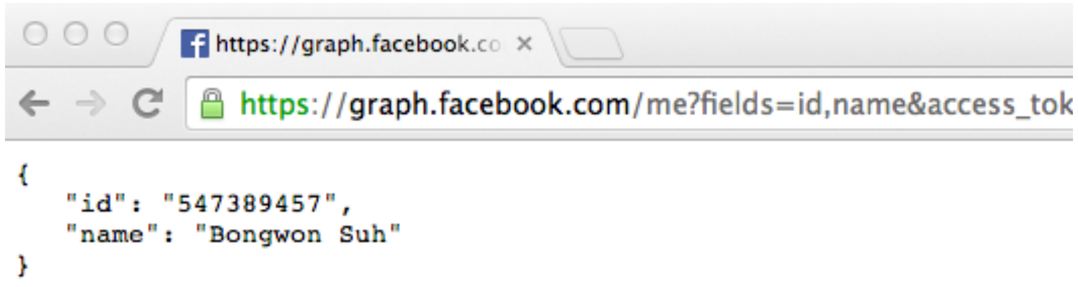
Making Graph API Request over HTTP

- 설치
  - Python 밖에서 다음을 타이핑
  - pip install requests
  - 웹 브라우저에서 처럼 로드하는 방식

```
import requests # pip install requests
import json

base_url = 'https://graph.facebook.com/me'
ACCESS_TOKEN = 'XXXXX'

# Get id & name
fields = 'id,name'

url = '%s?fields=%s&access_token=%s' % (base_url, fields, ACCESS_TOKEN,)

print url

# Interpret the response as JSON
content = requests.get(url).json()

# Pretty-print the JSON and display it
print json.dumps(content, indent=1)
```



```
>>> # Pretty-print the JSON and display it
... print json.dumps(content, indent=1)
{
 "id": "547389457",
 "name": "Bongwon Suh"
}
```



```
https://graph.facebook.com/me?fields=id,name&access_tok
```

```
{
   "id": "547389457",
   "name": "Bongwon Suh"
}
```

Facebook API for Python

- https://github.com/pythonforfacebook/facebook-sdk
- 설치
  - Python 밖에서 다음을 타이핑
  - pip install facebook-sdk

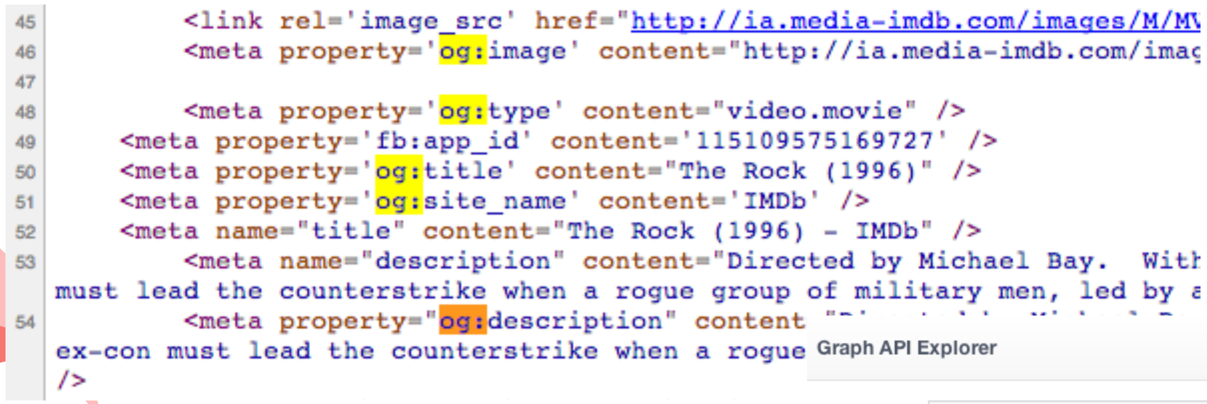
```
import facebook # pip install facebook-sdk
import json

ACCESS_TOKEN = 'XXXXX'

# Create a connection to the Graph API with your access token
g = facebook.GraphAPI(ACCESS_TOKEN)

obj = g.get_object('me')
print json.dumps(obj, indent=1)

friends = g.get_connections('me', 'friends')
print json.dumps(friends, indent=1)

socialweb = g.request("search", {'q' : 'social web', 'type' : 'page'})
print json.dumps(socialweb, indent=1)
```

Access Open Graph Objects by Their URLs

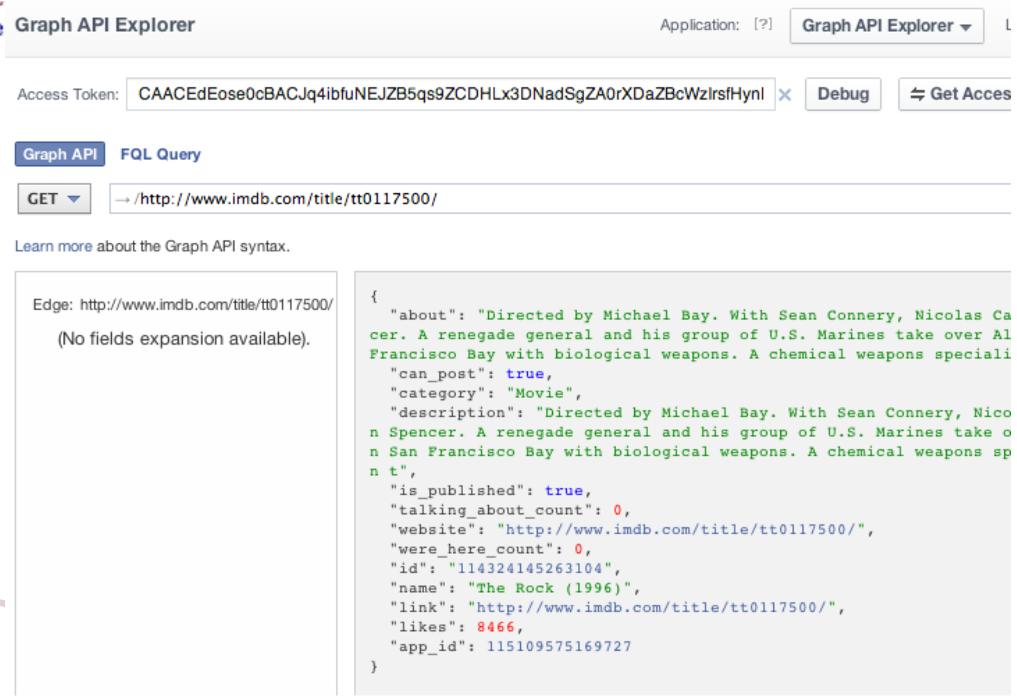
- Search ‘pepsi’ on Facebook & check the name in the URL
  - print g.get_object('pepsi')
  - print g.get_object('339150749455906')
  - print "Pepsi likes:", g.get_object('pepsi')['likes']
  - print "Coke likes:", g.get_object('cocacola')['likes']
- Any URL can be Id when it is shared
  - print g.get_object('http://shop.oreilly.com/product/0636920030195.do')
  - print g.get_object('http://www.snu.ac.kr')
- For groups
  - print g.get_object('109742322436123')

Open Graph Protocol

- http://www.imdb.com/title/tt0117500/



```
        <link rel='image_src' href="http://ia.media-imdb.com/images/M/MV
        <meta property='og:image' content="http://ia.media-imdb.com/imag

        <meta property='og:type' content="video.movie" />
    <meta property='fb:app_id' content='115109575169727' />
    <meta property='og:title' content="The Rock (1996)" />
    <meta property='og:site_name' content='IMDb' />
    <meta name="title" content="The Rock (1996) - IMDb" />
        <meta name="description" content="Directed by Michael Bay.  With
must lead the counterstrike when a rogue group of military men, led by a
        <meta property="og:description" content
ex-con must lead the counterstrike when a rogue
/>
```



Access Token: CAACEdEose0cBACJq4ibfuNEJZB5qs9ZCDHLx3DNadSgZA0rXDaZBcWzIrsfHynl

→ /http://www.imdb.com/title/tt0117500/

Edge: http://www.imdb.com/title/tt0117500/

(No fields expansion available).

```
{
  "about": "Directed by Michael Bay. With Sean Connery, Nicolas Ca
cer. A renegade general and his group of U.S. Marines take over Al
Francisco Bay with biological weapons. A chemical weapons speciali
  "can_post": true,
  "category": "Movie",
  "description": "Directed by Michael Bay. With Sean Connery, Nico
n Spencer. A renegade general and his group of U.S. Marines take o
n San Francisco Bay with biological weapons. A chemical weapons sp
n t",
  "is_published": true,
  "talking_about_count": 0,
  "website": "http://www.imdb.com/title/tt0117500/",
  "were_here_count": 0,
  "id": "114324145263104",
  "name": "The Rock (1996)",
  "link": "http://www.imdb.com/title/tt0117500/",
  "likes": 8466,
  "app_id": 115109575169727
}
```

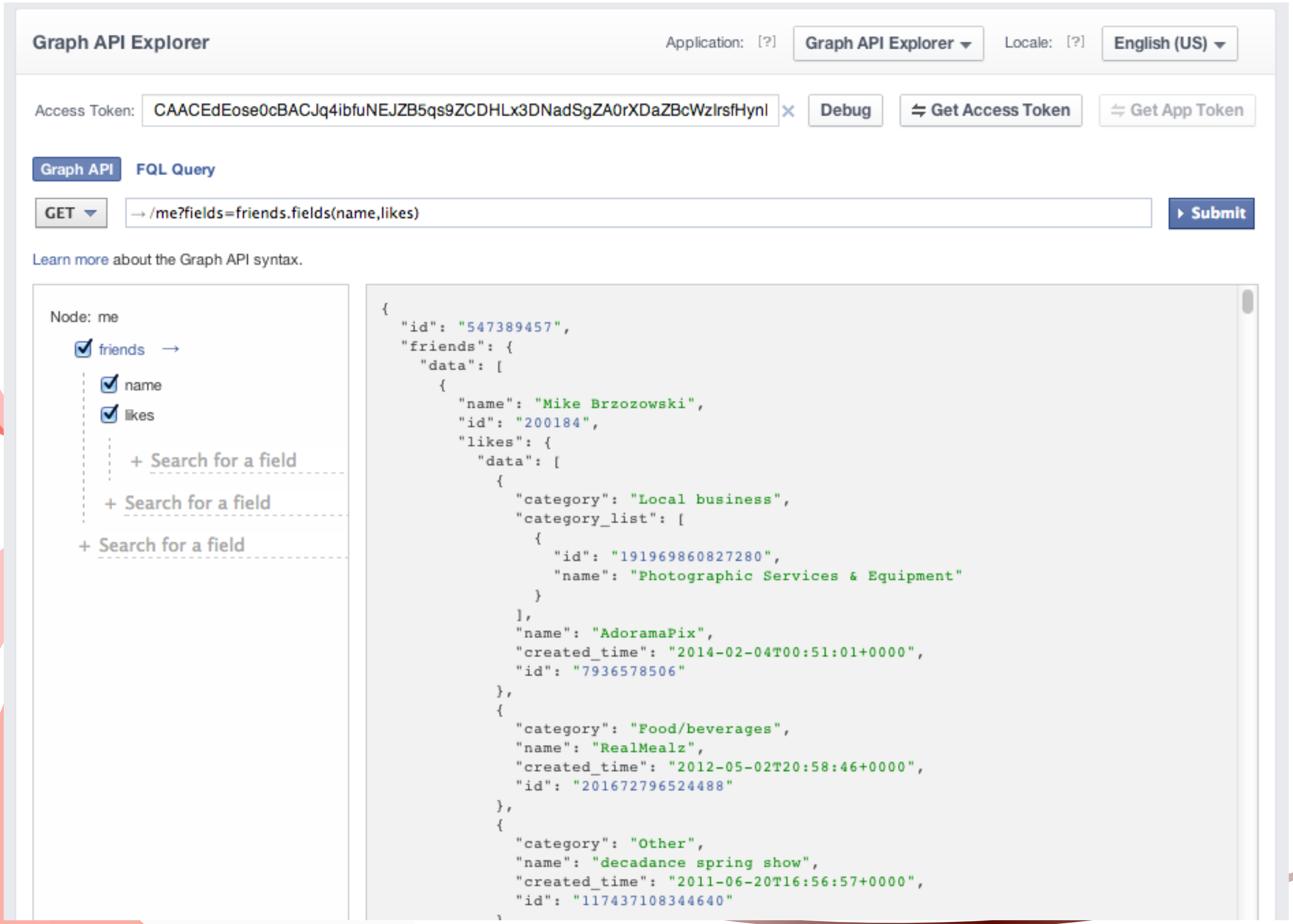
What do they say on “Fan Pages”?

```
def pp(o):
   print json.dumps(o, indent=1)

pp(g.get_connections('pepsi', 'feed'))
pp(g.get_connections('pepsi', 'links'))

pp(g.get_connections('cocacola', 'feed'))
pp(g.get_connections('cocacola', 'links'))

pepsi = g.get_connections('pepsi', 'feed')
pepsi['data'][4]['message']
```

- Feed vs. Link vs. Status

- Which posts in the feed are the most popular?
- Posts with links more popular than posts with photos?
- What makes a post go viral?

What my friends like in Facebook? (Graph API Explorer)



```
/me?fields=friends.fields(name,likes)
```

```
{
  "id": "547389457",
  "friends": {
    "data": [
      {
        "name": "Mike Brzozowski",
        "id": "200184",
        "likes": {
          "data": [
            {
              "category": "Local business",
              "category_list": [
                {
                  "id": "191969860827280",
                  "name": "Photographic Services & Equipment"
                }
              ],
              "name": "AdoramaPix",
              "created_time": "2014-02-04T00:51:01+0000",
              "id": "7936578506"
            },
            {
              "category": "Food/beverages",
              "name": "RealMealz",
              "created_time": "2012-05-02T20:58:46+0000",
              "id": "201672796524488"
            },
            {
              "category": "Other",
              "name": "decadance spring show",
              "created_time": "2011-06-20T16:56:57+0000",
              "id": "117437108344640"
```

What my friends like in Facebook

```
flike = g.get_object('me', fields='id,name,friends.fields(id,name,likes)')
flike = g.get_object('me', fields='id,name,friends.fields(id,name,likes).limit(2)')

flike['friends']['data'][0]['name']
flike['friends']['data'][0]['likes']
flike['friends']['data'][0]['likes']['data']
flike['friends']['data'][0]['likes']['data'][0]
flike['friends']['data'][0]['likes']['data'][1]
flike['friends']['data'][0]['likes']['data'][1]['name']

flike['friends']['data'][1]['likes']['data'][0]
flike['friends']['data'][1]['likes']['data'][1]
```

Different Ways to Acquire Like Data

```
flike = g.get_object('me', fields='id,name,friends.fields(id,name,likes)')

likes = {}
for friend in flike['friends']['data']:
    friendname = friend['name']
    if friend.has_key('likes'):
        likes[friendname] = friend['likes']['data']
```

- Compare the above with the following lines

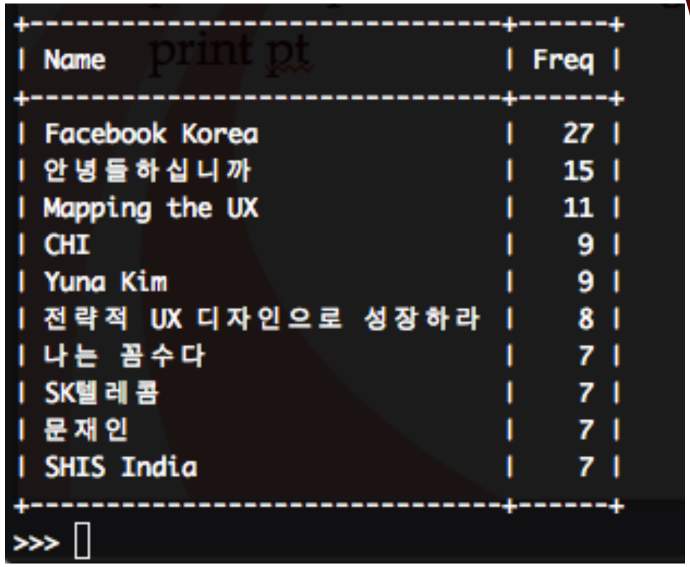
```
friends = g.get_connections("me", "friends")['data’]
likes = { friend['name'] : g.get_connections(friend['id'], "likes")['data']
          for friend in friends }
```

- Or

```
friends = g.get_connections("me", "friends")['data']
likes = {}
for friend in friends:
    like = g.get_connections(friend['id'], "likes")['data']
    likes[friend['name']] = like
```

Calculating the most popular likes among friends

```
from prettytable import PrettyTable
from collections import Counter

friends_likes = Counter([like['name']
                         for friend in likes
                          for like in likes[friend]
                            if like.get('name')])


pt = PrettyTable(field_names=['Name', 'Freq'])
pt.align['Name'] = 'l'
pt.align['Freq'] = 'r'

for fl in friends_likes.most_common(10):
   pt.add_row(fl)

print 'Top 10 likes amongst friends'
print pt
```



```
+-----------------------------+------+
| Name                        | Freq |
+-----------------------------+------+
| Facebook Korea              |   27 |
| 안녕들하십니까              |   15 |
| Mapping the UX              |   11 |
| CHI                         |    9 |
| Yuna Kim                    |    9 |
| 전략적 UX 디자인으로 성장하라 |    8 |
| 나는 꼼수다                 |    7 |
| SK텔레콤                    |    7 |
| 문재인                      |    7 |
| SHIS India                  |    7 |
+-----------------------------+------+
>>> 
```

For Loop – 이제는 알아야 한다!

```
friends_likes = Counter([like['name']
                        for friend in likes
                          for like in likes[friend]
                             if like.get('name')])
```

- Compare the above and the below

```
friends_likes_list = []
for friend in likes:
   for like in likes[friend]:
      if like.has_key ('name'):
         friends_likes_list.append(like['name'])

friends_likes = Counter(friends_likes_list)
```

Calculating the number of likes for each friend

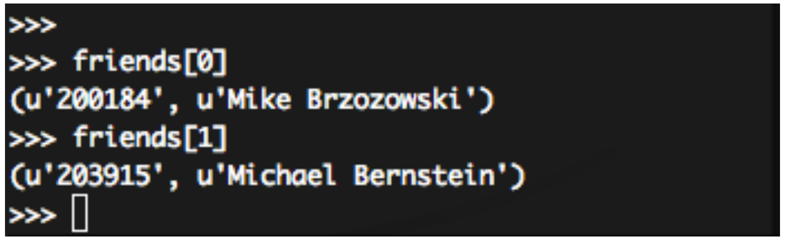
```
from operator import itemgetter

num_likes_by_friend = { friend : len(likes[friend])
                        for friend in likes }

pt = PrettyTable(field_names=['Friend', 'Num Likes'])
pt.align['Friend'], pt.align['Num Likes'] = 'l', 'r'

sorted = sorted(num_likes_by_friend.items(), key=itemgetter(1), reverse=True)

size = min(50, len(sorted))
for i in range(size):
    pt.add_row(sorted[i])

print "Number of likes per friend"
print pt
```

Mutual Friendship in Facebook

```
import facebook # pip install facebook-sdk
import json
import networkx as nx # pip install networkx
import requests # pip install requests

base_url = 'https://graph.facebook.com/me'
ACCESS_TOKEN = ''XXXXXXXX'
g = facebook.GraphAPI(ACCESS_TOKEN)

friends = [ (friend['id'], friend['name'],)
            for friend in g.get_connections('me', 'friends')['data'] ]

url = 'https://graph.facebook.com/me/mutualfriends/%s?access_token=%s'

mutual_friends = {}

for friend_id, friend_name in friends:
   r = requests.get(url % (friend_id, ACCESS_TOKEN,) )
   response_data = json.loads(r.content)['data']
   mutual_friends[friend_name] = [ data['name']
                           for data in response_data ]

nxg = nx.Graph()

for mf in mutual_friends:
   nxg.add_edge('me', mf)

for f1 in mutual_friends:
   for f2 in mutual_friends[f1]:
      nxg.add_edge(f1, f2)
```

Mutual Friendship in Facebook

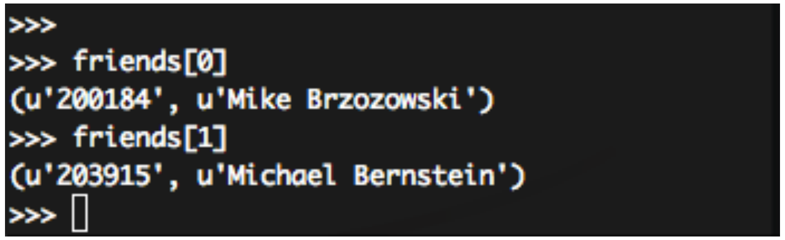
```
import facebook # pip install facebook-sdk
import json
import networkx as nx # pip install networkx
import requests # pip install requests

base_url = 'https://graph.facebook.com/me'
ACCESS_TOKEN = ''XXXXXXXX'
g = facebook.GraphAPI(ACCESS_TOKEN)

friends = [ (friend['id'], friend['name'],)
            for friend in g.get_connections('me', 'friends')['data'] ]

url = 'https://graph.facebook.com/me/mutualfriends/%s?access_token=%s'

mutual_friends = {}

for friend_id, friend_name in friends:
    r = requests.get(url % (friend_id, ACCESS_TOKEN,) )
    response_data = json.loads(r.content)['data']
    mutual_friends[friend_name] = [ data['name']
                                    for data in response_data ]

nxg = nx.Graph()

for mf in mutual_friends:
    nxg.add_edge('me', mf)

for f1 in mutual_friends:
    for f2 in mutual_friends[f1]:
        nxg.add_edge(f1, f2)
```



```
>>>
>>> friends[0]
(u'200184', u'Mike Brzozowski')
>>> friends[1]
(u'203915', u'Michael Bernstein')
>>> 
```

Mutual Friendship in Facebook

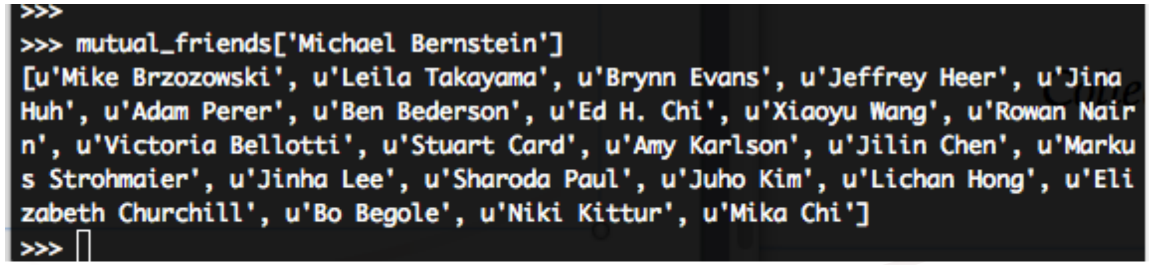
```
import facebook # pip install facebook-sdk
import json
import networkx as nx # pip install networkx
import requests # pip install requests

base_url = 'https://graph.facebook.com/me'
ACCESS_TOKEN = ''XXXXXXXX'
g = facebook.GraphAPI(ACCESS_TOKEN)

friends = [ (friend['id'], friend['name'],)
            for friend in g.get_connections('me', 'friends')['data'] ]
```



```
>>>
>>> friends[0]
(u'200184', u'Mike Brzozowski')
>>> friends[1]
(u'203915', u'Michael Bernstein')
>>> 
```

```
url = 'https://graph.facebook.com/me/mutualfriends/%s?access_token=%s'

mutual_friends = {}

for friend_id, friend_name in friends:
   r = requests.get(url % (friend_id, ACCESS_TOKEN,) )
   response_data = json.loads(r.content)['data']
   mutual_friends[friend_name] = [ data['name']
                                   for data in response_data ]
```

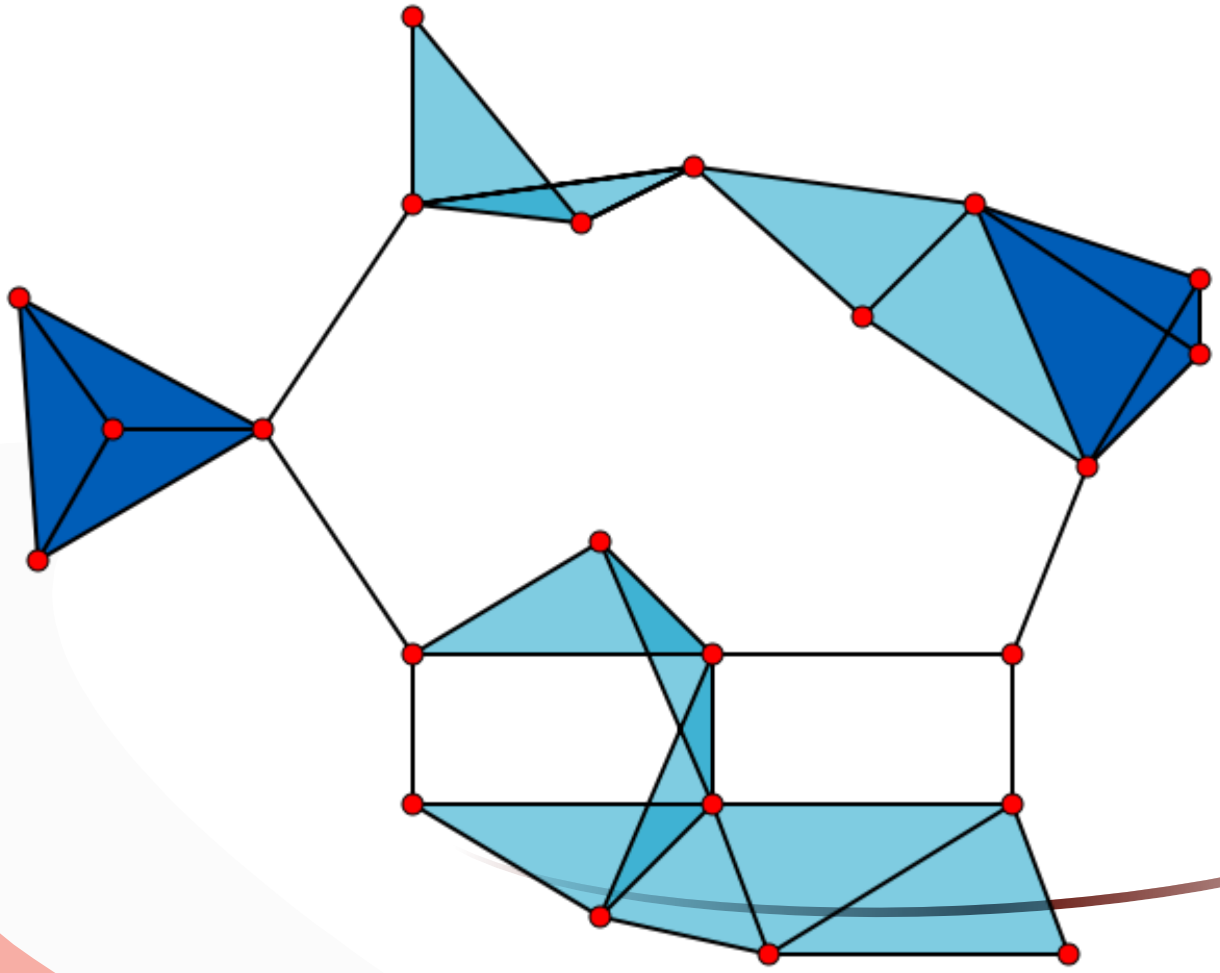
```
nxg = nx.Graph()

for mf in mutual_friends:
  nxg.add_edge('me', mf)

for f1 in mutual_friends:
  for f2 in mutual_friends[f1]:
    nxg.add_edge(f1, f2)
```



```
>>>
>>> mutual_friends['Michael Bernstein']
[u'Mike Brzozowski', u'Leila Takayama', u'Brynn Evans', u'Jeffrey Heer', u'Jina
Huh', u'Adam Perer', u'Ben Bederson', u'Ed H. Chi', u'Xiaoyu Wang', u'Rowan Nair
n', u'Victoria Bellotti', u'Stuart Card', u'Amy Karlson', u'Jilin Chen', u'Marku
s Strohmaier', u'Jinha Lee', u'Sharoda Paul', u'Juho Kim', u'Lichan Hong', u'Eli
zabeth Churchill', u'Bo Begole', u'Niki Kittur', u'Mika Chi']
>>> 
```

Clique

- A subset of a graph of which vertices are completed connected
- 모든 사람이 다 친구인 소그룹



Finding Cliques

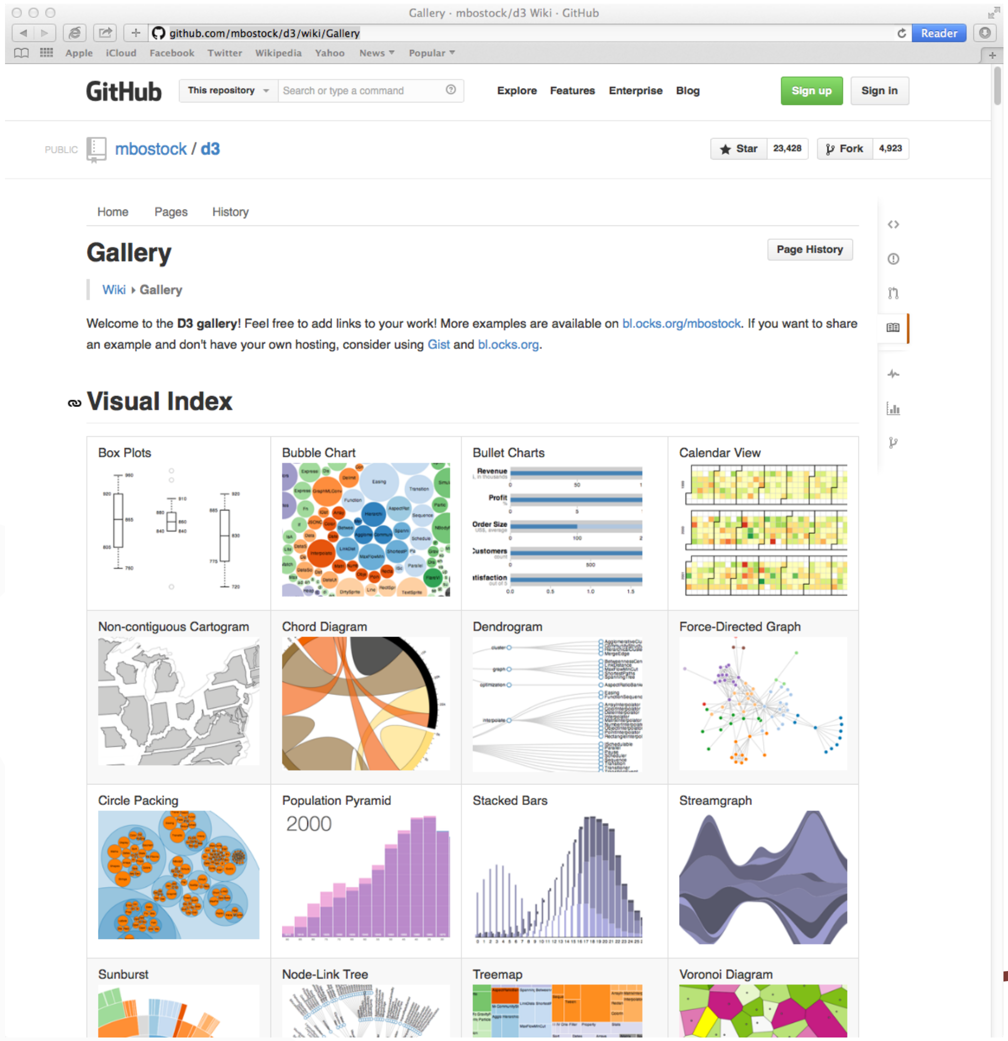
```
cliques = [c for c in nx.find_cliques(nxg)]

num_cliques = len(cliques)

clique_sizes = [len(c) for c in cliques]
max_clique_size = max(clique_sizes)
avg_clique_size = sum(clique_sizes) / num_cliques

max_cliques = [c for c in cliques if len(c) == max_clique_size]

num_max_cliques = len(max_cliques)


print 'Num cliques:', num_cliques
print 'Avg clique size:', avg_clique_size
print 'Max clique size:', max_clique_size
print 'Num max cliques:', num_max_cliques
Print
print 'Max cliques:'
print json.dumps(max_cliques, indent=1)
```

Saving the graph into a JSON file

```
from networkx.readwrite import json_graph

nld = json_graph.node_link_data(nxg)
json.dump(nld, open('force.json','w'))
```

D3.js - Visualization Tookit

- https://github.com/mbostock/d3/wiki/Gallery



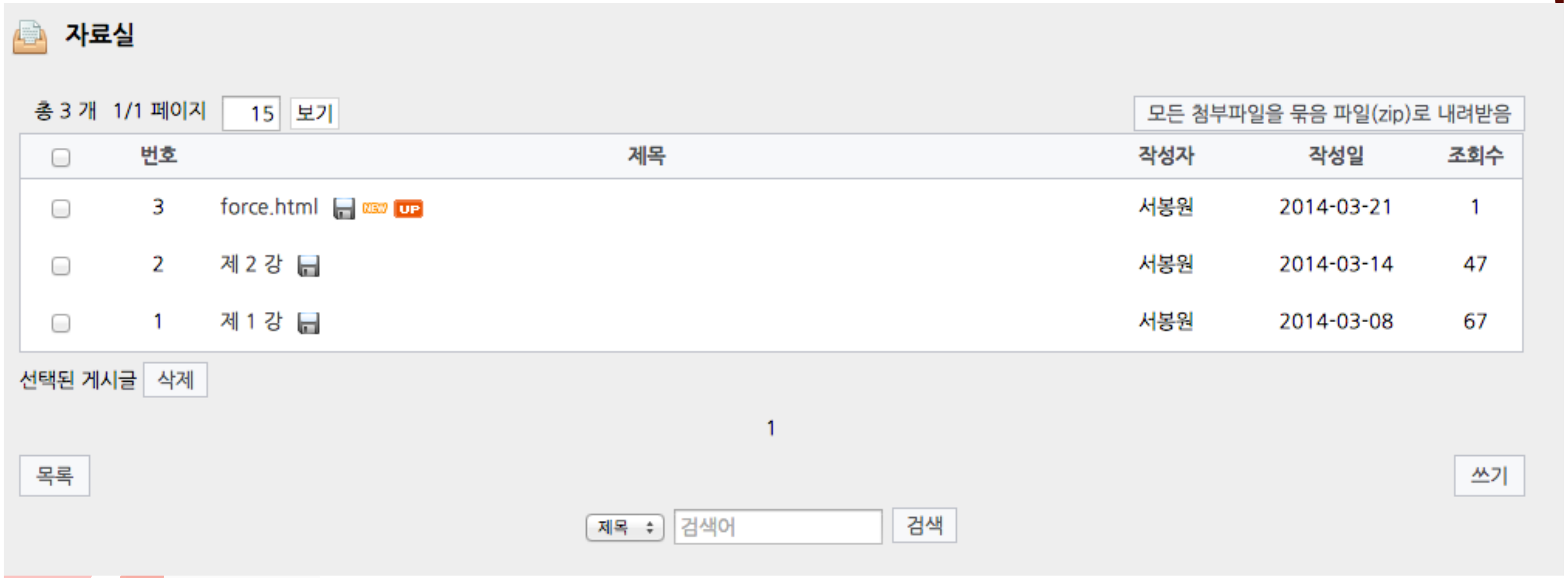
Gallery

Wiki › Gallery

Welcome to the **D3 gallery**! Feel free to add links to your work! More examples are available on bl.ocks.org/mbostock. If you want to share an example and don't have your own hosting, consider using Gist and bl.ocks.org.

Visual Index

Download force.html file from etl board



자료실

총 3 개 1/1 페이지 15 보기 모든 첨부파일을 묶음 파일(zip)로 내려받음

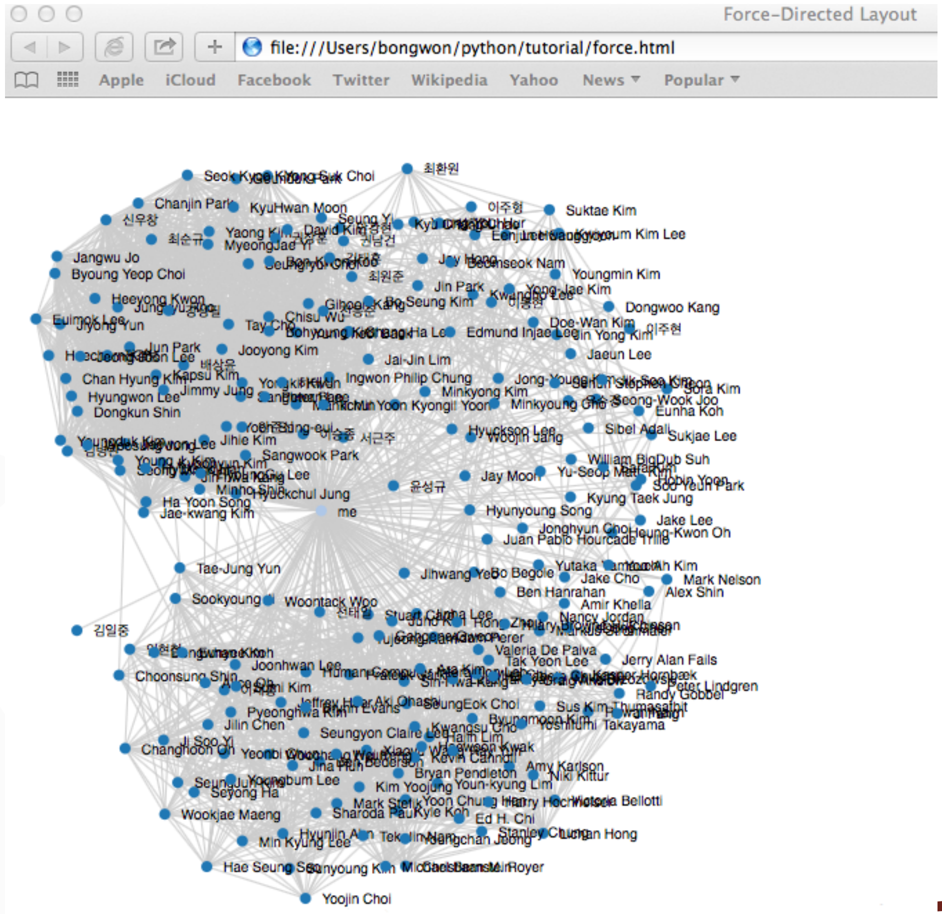
	번호	제목	작성자	작성일	조회수
☐	3	force.html	서봉원	2014-03-21	1
☐	2	제 2 강	서봉원	2014-03-14	47
☐	1	제 1 강	서봉원	2014-03-08	67

선택된 게시글 삭제

1

Firefox나 Safari에서 force.html을 실행

- force.json 과 force.html을 같은 directory에 넣고 Firefox나 Safari browser에서 force.html을 로드



Homework – etl로 제출

- 오늘 배운 것을 바탕으로 다음을 수행하세요.

1. Word Cloud
   - 자신이 좋아하는 페이지를 찾고 feed의 'message'를 수집
   - 전수업에 사용된 방법에 따라서 feed의 Word Cloud 작성
   - Word Cloud를 이미지로 저장/캡처해서 facebookfanpage.jpg로 제출
   - 해당 코드를 facebookfanpage.py로 저장해서 제출

2. Mutual Friendship Graph
   - 자신의 페이스 친구들의 관계를 수집해서 force.json 화일을 작성
   - 해당 코드를 force.py 저장해서 제출
   - force.html에서 로드한 결과를 이미지로 screen capture 해서 force.jpg로 제출

3. Most Popular Like
   - 페이스북 친구들이 누른 like를 수집해서 가장 Like수가 높은 아이템 20개를 prettytable을 사용하여 빈도수와 더불어 출력
   - 해당 코드를 popular.py 저장해서 제출
   - 결과를 이미지로 screen capture 해서 popular.jpg로 제출

- 3월27일 자정까지 모든 파일을 zip으로 압축해서 etl 을 통해 제출