Klout API Documentation

Release 0.1.4

Irfan Ahmad

Jan 21, 2018

Contents

1 Design Philosophy **3**
- 1.1 Klout API URL Structure . . . 3
- 1.2 Designing python functions . . . 4
- 1.3 Klout API URL to Python Function mapping examples . . . 4

2 Quickstart **9**

3 Design Philosoph **11**

4 API Documentation **13**

Python Module Index **15**


build
error

CHAPTER 1

Design Philosophy

This API wrapper tries to generalize the API calls. First we must find out the generalization in API URL structure.

1.1 Klout API URL Structure

Klout's API URL structure is RESTful with a little cutomization. Almost all API URLs can be generalized in a way.

<protocol>://<domain>/<version>/<resource>.<format>/<input_name>/<input_value>/<action>?<query_string>

- <protocol>: http or https
- <domain>: api.klout.com
- <version>: v2 for the new API
- <resource>: Currently there are only 2 resources *identity* and *user*
- <format>: Klout no longer supports *XML*. Only *json* is supported.
- <input_name>: (Optional) Name of the field to query. Required only when the input field is not *Primary Key*. Possible values for *identity* resource are:
  - tw - Twitter
  - klout - (Written as ks in official Klout API documentation but it doesn't work)
  - gp - Google+
  - twitter (when using twitter screenName as input)
- <input_value>: Valule of the <input_name> specified. Can be twitter id, google plus id or klout id.
- <action>: Name of the action to be performed.
- <query_string>: Must contain a key obtained from <http://klout.com/s/developers/v2>. MUST also contain *screenName* when <input_name> is twitter

Note: <input_name> MUST be skipped when query a resource with *Primary Key* e.g. when call user resource with *kloutId*, the <*input_name*> should be skipped.

Note: <input_value> MUST be skipped when <input_name> is *twitter* The value should be passed in <query_string> instead.

Note: <action> MUST be skipped when expected result is *Primary Key* e.g. when call identity resource and expected result is *kloutId*

1.2 Designing python functions

After generalizing the API URLs, next step is to make a generalized pythonic function calls:

Here is what I propose:

```
k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxx')
k.<resource>.<action>(<input_name>=<input_value>)
```

1.3 Klout API URL to Python Function mapping examples

Here are all possible combinations of the URLs. Lets try to map them to our generalized structure defined above:

- http://api.klout.com/v2/identity.json/gp/112975106809988327760?key=xxxxxxxxxxxxxxxxxxxxxxx
  - protocol: http
  - domain: api.klout.com
  - version: v2
  - resource: identity
  - format: json
  - input_name: gp - Google+
  - input_value: 112975106809988327760 - Google+ Id
  - action: Empty - We need kloutId as output
  - query_string: key=xxxxxxxxxxxxxxxxxxxxxxx

  Code Example:

  ```
  k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxx')
  k.identity.klout(gp='112975106809988327760')
  # Note that we replaced the empty action with klout as we want to be consistent
  ```

- http://api.klout.com/v2/identity.json/tw/11158872?key=xxxxxxxxxxxxxxxxxxxxxxx
  - protocol: http
  - domain: api.klout.com

  - version: v2
  - resource: identity
  - format: json
  - input_name: tw - Twitter
  - input_value: 11158872 - Twitter Id
  - action: Empty - We need kloutId as output
  - query_string: key=xxxxxxxxxxxxxxxxxxxxxxx

  Code Example:

```
k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxx')
k.identity.klout(tw='11158872')
# Note that we replaced the empty action with klout as we want to be consistent
```

- http://api.klout.com/v2/identity.json/twitter?key=xxxxxxxxxxxxxxxxxxxxxxx&screenName=erfaan
  - protocol: http
  - domain: api.klout.com
  - version: v2
  - resource: identity
  - format: json
  - input_name: twitter
  - input_value: Empty - We want to query using twitter screenName
  - action: Empty - We need kloutId as output
  - query_string: key=xxxxxxxxxxxxxxxxxxxxxxx&screenName=erfaan - screenName is only required in this case

  Code Example:

```
k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxx')
k.identity.klout(screenName='erfaan')
# Note that we replaced the empty action with klout as we want to be consistent
# Also the input parameters query string are used as function parameters. (again␣
↪consistency)
```

- http://api.klout.com/v2/identity.json/klout/11747/gp?key=xxxxxxxxxxxxxxxxxxxxxxx
  - protocol: http
  - domain: api.klout.com
  - version: v2
  - resource: identity
  - format: json
  - input_name: klout
  - input_value: 11747
  - action: gp - Google+
  - query_string: key=xxxxxxxxxxxxxxxxxxxxxxx

Code Example:

```
k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxx')
k.identity.gp(klout='11747')
```

- http://api.klout.com/v2/identity.json/klout/11747/tw?key=xxxxxxxxxxxxxxxxxxxxxxx
  - protocol: http
  - domain: api.klout.com
  - version: v2
  - resource: identity
  - format: json
  - input_name: klout
  - input_value: 11747
  - action: tw - Twitter
  - query_string: key=xxxxxxxxxxxxxxxxxxxxxxx

Code Example:

```
k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxx')
k.identity.tw(klout='11747')
```

- https://api.klout.com/v2/user.json/11747/score?key=xxxxxxxxxxxxxxxxxxxxxxx
  - protocol: https
  - domain: api.klout.com
  - version: v2
  - resource: user
  - format: json
  - input_name: Empty - We are inputting kloutId
  - input_value: 11747
  - action: score
  - query_string: key=xxxxxxxxxxxxxxxxxxxxxxx

Code Example:

```
k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxx')
k.user.scrore(kloutId='11747')
```

- https://api.klout.com/v2/user.json/11747/influence?key=xxxxxxxxxxxxxxxxxxxxxxx
  - protocol: https
  - domain: api.klout.com
  - version: v2
  - resource: user
  - format: json
  - input_name: Empty - We are inputting kloutId

  - input_value: 11747
  - action: influence
  - query_string: key=xxxxxxxxxxxxxxxxxxxxxxxx

  Code Example:

```
k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxxx')
k.user.influence(kloutId='11747')
```

- https://api.klout.com/v2/user.json/11747/topics?key=xxxxxxxxxxxxxxxxxxxxxxxx
  - protocol: https
  - domain: api.klout.com
  - version: v2
  - resource: user
  - format: json
  - input_name: Empty - We are inputting kloutId
  - input_value: 11747
  - action: topics
  - query_string: key=xxxxxxxxxxxxxxxxxxxxxxxx

  Code Example:

```
k = Klout(key='xxxxxxxxxxxxxxxxxxxxxxxx')
k.user.topics(kloutId='11747')
```

A minimalist klout API interface. Use of this API requires klout *developer key*. You can get registered and get a key at

<http://klout.com/s/developers/v2>

Supports Python >= 2.5 and Python 3

CHAPTER 2

Quickstart

Install the PyPi package:

```
pip install Klout
```

This short example shows how to get a kloutId first and fetch user's score using that kloutId:

```
from klout import *

# Make the Klout object
k = Klout('YOUR_KEY_HERE')

# Get kloutId of the user by inputting a twitter screenName
kloutId = k.identity.klout(screenName="erfaan").get('id')

# Get klout score of the user
score = k.user.score(kloutId=kloutId).get('score')

print "User's klout score is: %s" % (score)

# Optionally a timeout parameter (seconds) can also be sent with all calls
score = k.user.score(kloutId=kloutId, timeout=5).get('score')
```

CHAPTER 3

Design Philosoph

See *Design Philosophy*

CHAPTER 4

API Documentation

class klout.api.**Klout**(*key*, *domain='api.klout.com'*, *secure=False*, *api_version=<class 'klout.api._DEFAULT'>*)

A minimalist yet fully featured klout API interface.

Get RESTful data by accessing members of this class. The result is decoded python objects (dicts and lists).

The klout API is documented at:

http://klout.com/s/developers/v2

Examples:

We need a *developer key* to call any Klout API function

```
>>> f = open('key')
>>> key= f.readline().strip()
>>> f.close()
```

By default all communication with Klout API is not secure (HTTP). It can be made secure (HTTPS) by passing an optional *secure=True* to *Klout* constructor like this:

```
>>> k = Klout(key, secure=True)
```

Identity Resource

All calls to the Klout API now require a unique kloutId. To facilitate this, you must first translate a {network}/{networkId} into a kloutId.

- Get kloutId by twitter id

```
>>> k.identity.klout(tw="11158872")
{u'id': u'11747', u'network': u'ks'}
```

- Get kloutId by twitter screenName

```
>>> k.identity.klout(screenName="erfaan")
{u'id': u'11747', u'network': u'ks'}
```

- Get kloutId by google plus id

```
>>> k.identity.klout(gp="112975106809988327760")
{u'id': u'11747', u'network': u'ks'}
```

User Resource

Once we have kloutId, we can use this resource to lookup user's score, influcent or topics

- Get user score

```
>>> k.user.score(kloutId='11747')
...
{u'score': ..., u'scoreDelta': {u'dayChange': ..., u'monthChange': ...}}
```

- Get user influences

```
>>> k.user.influence(kloutId='11747')
...
{u'myInfluencersCount': ..., u'myInfluenceesCount': ...,     u'myInfluencers': [..
↪.], u'myInfluencees': [...]}
```

- Get user topics

```
>>> k.user.topics(kloutId='11747')
...
[{u'displayName': ..., u'name': ..., u'imageUrl': ..., u'id': ...,     u
↪'displayType': ..., u'slug': ...}, ...]
```

exception klout.api.**KloutError**(*errors*)
Base Exception thrown by Klout object when there is a general error interacting with the API.

exception klout.api.**KloutHTTPError**(*errors*, *uri*)
Exception thrown by Klout object when there is an HTTP error interacting with api.klout.com.

Python Module Index

k

klout.api, 7

Index

K

Klout (class in klout.api), 13
klout.api (module), 7
KloutError, 14
KloutHTTPError, 14