PYT TwitterWall Documentation

Release 0.5

Marek Suchánek

Nov 07, 2018

Contents

Welcome in **PYT TwitterWall** documentation. Continue by choosing desired topic from the list of contents. If you are not sure, try starting with *Introduction*. You can also visit the repository PYT-TwitterWall@GitHub.

CHAPTER 1

Contents

1.1 Introduction

PYT TwitterWall is simple Python powered app for displaying Twitter tweets in CLI or simple web page. It has the ability to show number of queried tweets (i.e. selected by query) at start and then check if new tweets are publish and display them as well.

This project is created as series of tasks for subject MI-PYT, CTU in Prague (more in *Credits*).

Project is open-source (under MIT license) published @GitHub.

1.2 Installation

1.2.1 Requirements

You need **Python 3.4+** to run this app and then there are two options (examples are for Linux systems, for Windows or OSX find appropriate equivalents):

Python environment

Check your `python3 --version` first, maybe you will need to use `python3` or `python3.5` instead or update finally!

You can use system-wide as well as virtual Python environments to work with **PYT TwitterWall**. See this two options below, all other commands in this documentation are not dependent on your choice (we do not remark used environment).

System-wide environment

```
$ python3 setup.py install
$ twitterwall ...
```

Virtual environment

```
$ python3 -m venv env
$ . env/bin/activate
(env) $ python3 setup.py install
(env) $ twitterwall ...

(env) $ deactivate
$ rm -r env
```

Packages

You can see what packages are required for install, testing and docs in files:

- setup.py (`install_requires`, `setup_requires`, `tests_require`)
- requirements.txt (read next section)
- docs/requirements.txt

Install tested environment

Thi project is tested in environment with packages & versions noted in the `requirements.txt` file (made by `pip freeze`). So you can install identical environment by:

```
python -m pip -r requirements.txt
```

Be sure to use correct version of Python (tested with 3.5).

1.2.2 Twitter API keys

You need to set-up your Twitter app on apps.twitter.com and create configuration file containing API key & secret. For that you need Twitter user account with filled phone number in the first place. Provided `config/auth.example.cfg` serves as example of this configuration file.

> **!!!** Never ever publish file with your Twitter API key & secret!
>
> We also recommend NOT to use personal Twitter account.

auth.cfg layout

```
[twitter]
key = YourTwitterAPIKeyComesHere
secret = YourTwitterAPISecretComesHere
```

1.2.3 setup.py

You can use standard `setup.py` file to install the package, after installation you can run Twitter Wall by just `twitterwall` or as Python module `python3 -m twitterwall` (watch you Python version). Installation can be done system-wide or just in virtual environment.

```
python3 setup.py install

twitterwall ...
python3 -m twitterwall ...
```

1.2.4 PyPi

- https://testpypi.python.org/pypi/twitterwall

You can use pip (and the Test PyPi) to install package **twitterwall**:

```
pip install --extra-index-url https://testpypi.python.org/pypi twitterwall
```

- *NOTE*: You can not use only the `-i`, because some of the required packages are not in the Test PyPi.

Again you can run this command system-wide (watch you Python version) or setup virtual environment first as described in *Requirements*.

1.3 Usage

1.3.1 Usage basics

First you need config file with your API key & secret. Default path is `config/auth.cfg`, can be set different via `--config` (or `-c`) option:

```
twitterwall --config <file> [web|cli] ...
```

For more about the config file, read section *Twitter API keys*.

Commands

By selecting `web` or `cli` command you will pick desired interface for tweets output:

- *CLI*
- *WEB*

Common options

You can also use `--help` and `--version` at any level:

```
$ twitterwall --help
Usage: twitterwall [OPTIONS] COMMAND [ARGS]...

  Twitter Wall for loading and printing desired tweets

```

(continues on next page)

(continued from previous page)

```
Options:
  -c, --config FILENAME  App config file path.
  --version              Show the version and exit.
  --help                 Show this message and exit.

Commands:
  cli  Twitter Wall running in CLI
  web  Twitter Wall running as web server

$ twitterwall --version
PYT TwitterWall, version 0.5
```

1.3.2 CLI

PYT TwitterWall CLI can be used simply. Options will allow you to:

- set the search query
- set the count of initial tweets to show up (default is 5)
- *NOTE*: This number means requested number of tweets before internal filtering (i.e. authors, #followers, #retweets, is retweet itself, ...)
- set the interval between twitter calls for new tweets (default is 10 seconds)
- set language of tweets to show up
- allow/block tweets from users by giving user nicknames
- set tweets to show only if they have min/max number of retweets
- set tweets to show only if their author has min/max number for followers
- hide retweets from output
- disable colors and other styling in output (also no hashtag/mention/hyperref highlighting)

Moreover you can use:

- `--help` to see all the options, syntax and information
- `--version` to check the version of app

Command examples

Show help how to use **twitterwall**:

```
twitterwall cli --help
```

Show only czech tweets (no retweets) with hashtag **#python**:

```
twitterwall cli -q "#python" --no-retweets --lang "cs"
```

Show only czech tweets (no retweets) with text **swag**, check every 1 second, load 20 tweets at start and don't use any CLI output styling at all:

```
twitterwall cli -q "swag" -i 1 -n 20 --no-swag
```

Filter loaded tweets with word **python** by allowing only authors **hroncok** and **EnCuKou** (MI-PYT teachers):

- *NOTE*: It will probably show only new tweets by these authors and no tweets will be shown at the start, because are not in last 5 tweets containing word "python".

```
twitterwall cli -q "python" -a "hroncok" -a "EnCuKou"
twitterwall cli -q "python" -a "hroncok" -a "encukou"
```

Filter loaded tweets with word **python** by blocking authors **hroncok** and **EnCuKou** (MI-PYT teachers), so it will hide all tweets by them:

```
twitterwall cli -q "python" -b "hroncok" -b "EnCuKou"
twitterwall cli -q "python" -b "hroncok" -b "encukou"
```

Filter loaded tweets with word **python** by allowing only tweets with number of retweets between 10 and 100 and from authors that have at least 300 followers but also less than 3000:

```
twitterwall cli -q "python" --retweets-min 10 --retweets-max 100 \
                --followers-min 300 --followers-max 3000
```

Output sample

```
05/10/2016 15:02:35 (https://twitter.com/pythontrending/statuses/783683809762050048)
Python Trending [pythontrending]: MI-PYT - Materiály k předmětu MI-PYT na FIT ČVUT␣
↪https://t.co/ZYdDaPT58n
```

- *NOTE*: Time is always in UTC timezone (as given from Twitter API, just reformatted)!

1.3.3 WEB

WEB interface is made by Flask & Jinja. It uses also Twitter Bootstrap, jQuery and Lightbox (local files only, no CDN).

Main ideas are same as for *CLI* interface. You just start web app with defined (or default) count of initial tweets displayed and/or interval of loading next tweets via AJAX. You can also run in flask debugging mode. The query and language is set by user of web interface (by URL).

In the web interface user can moreover turn on/off AJAX loading, clear screen or just refresh the page. For each tweet there is button for hide/show details that consists of entities: hashtags, mentions, links and photos. For nicer photos browsing is used the Lightbox.

Running example: mareksuchanek.pythonanywhere.com

Routes

- `/` = landing
- `/q/<query>[/<lang>]` = web interface for requested query in defined language
- `/api/<lid>/<query[/<lang>]` = API used by AJAX for loading additional tweets

Web launch examples

Here is also `--help` as is for the `cli` command:

```
twitterwall web --help
```

Start web interface with loading 7 tweets at start and 10 seconds interval of AJAX requests (when turned on by user).

- *NOTE*: Minimal value of interval is defined as 3 seconds.

```
twitterwall web --count 7 --interval 10
twitterwall web -n 7 -i 10
```

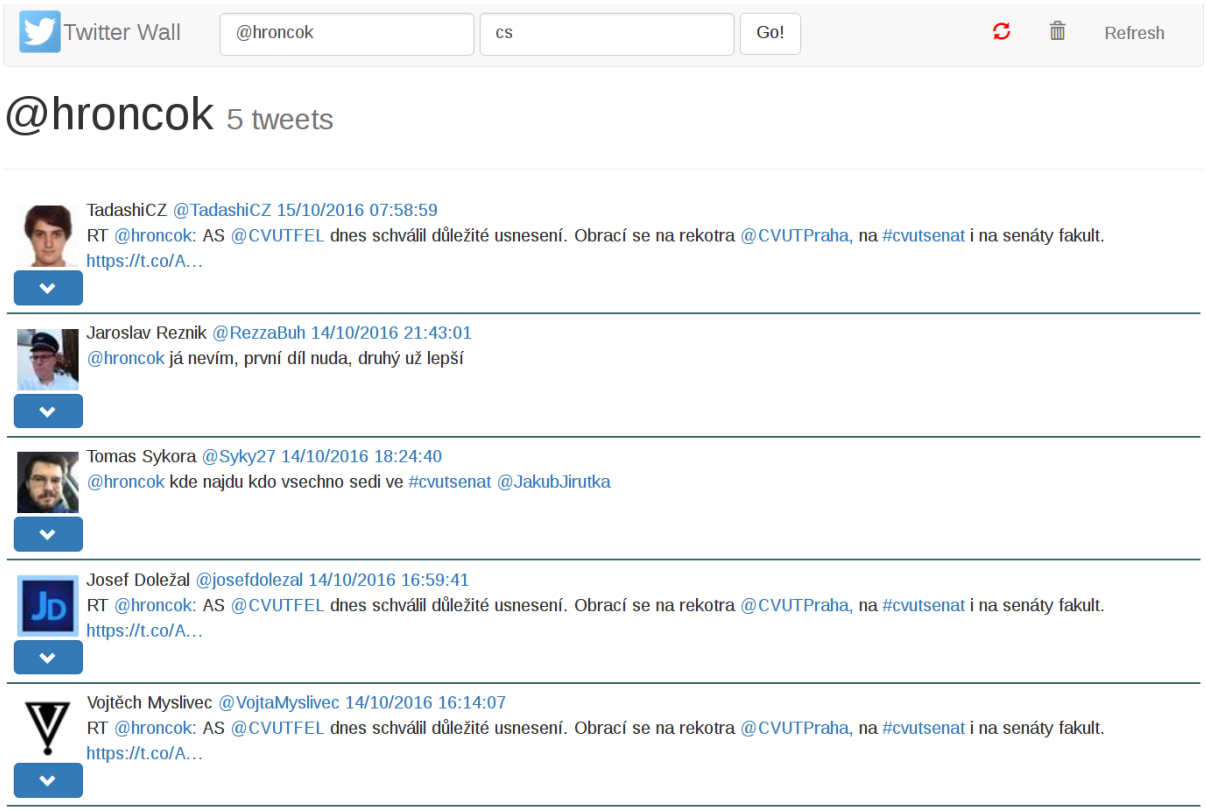
Start web interface with default values (5 tweets and 5 seconds), but turn on debugging.

- *NOTE*: Should not be used on production!

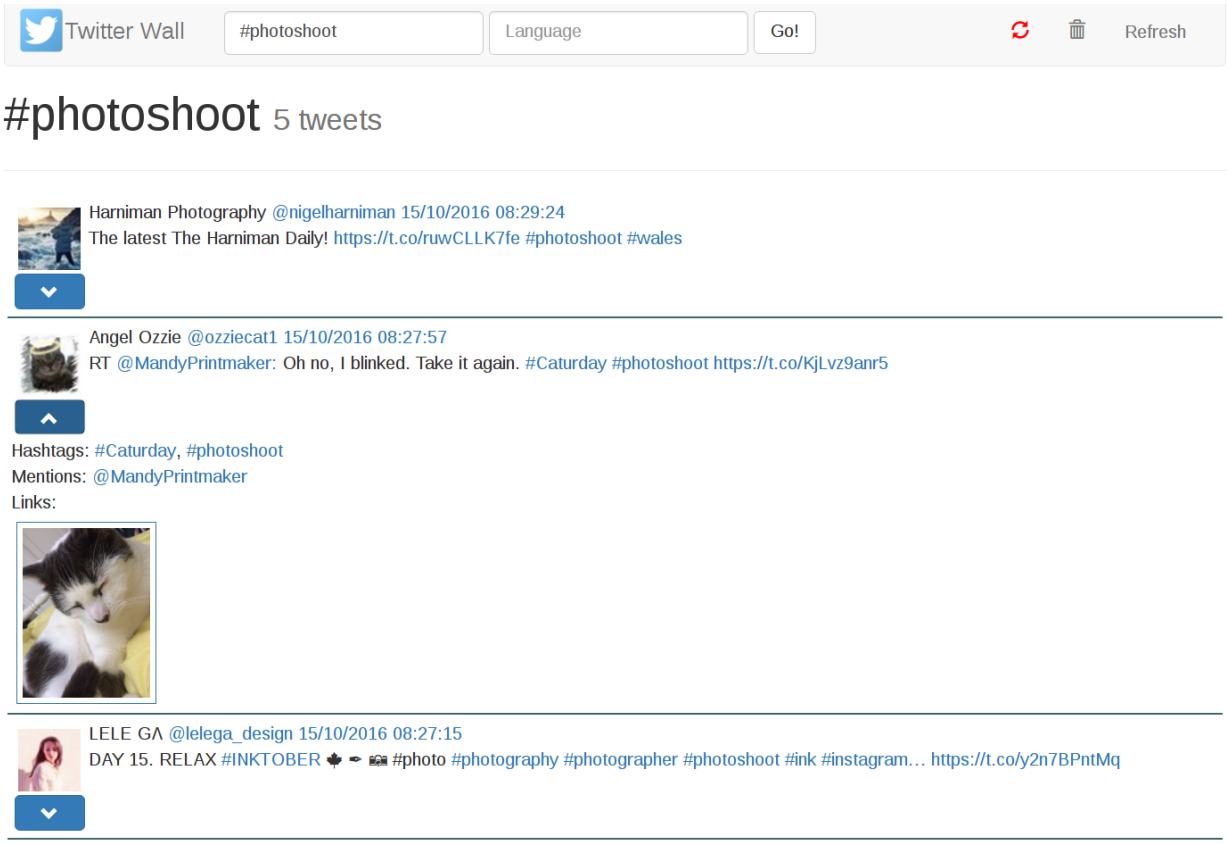
```
twitterwall web --debug
```

Screenshots

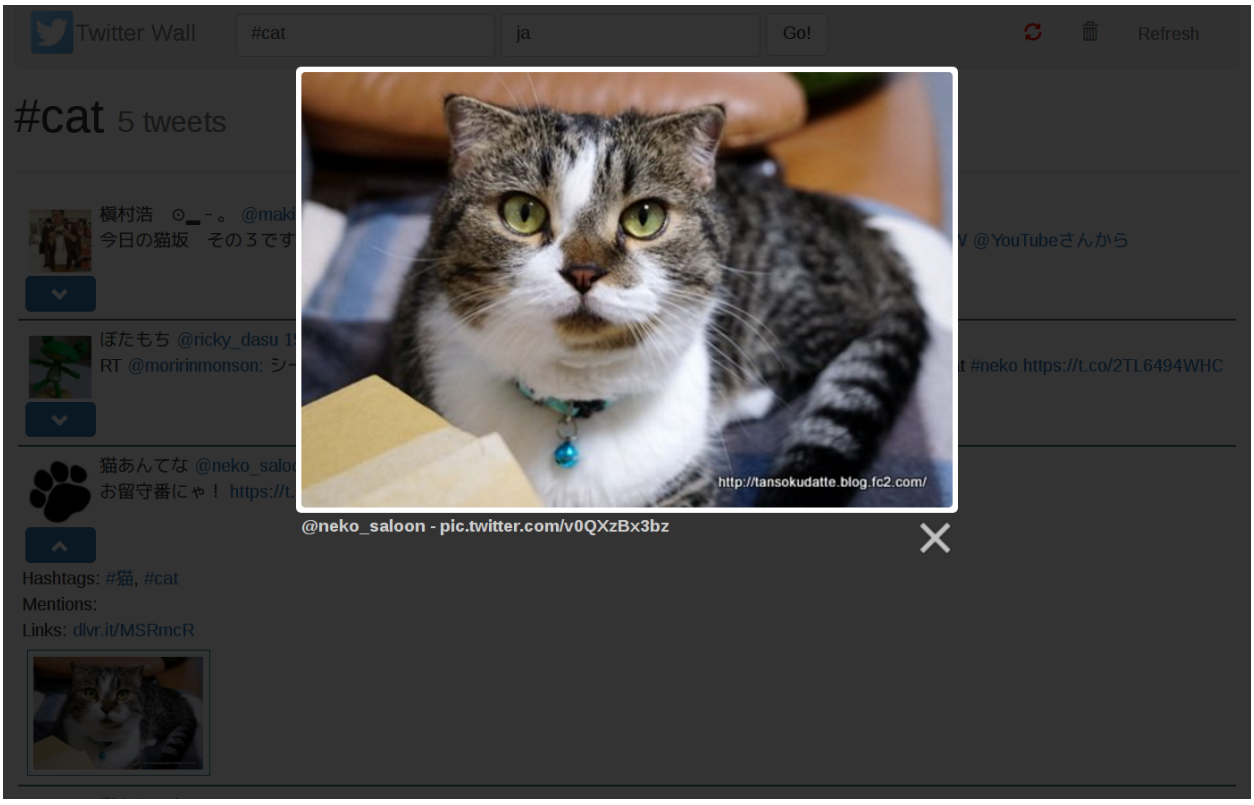
Basic Twitter Wall with **@hroncok** query:



Tweets with **#photoshoot** query with one tweet details shown (2 hashtags, 1 mention, 0 links and 1 picture):



Enlarged photo of cat via Lightbox:



1.4 Testing

This project uses the most fabulous testing tools for Python:

- pytest
- pytest-cov
- pytest-pep8
- pytest-sugar
- flexmock
- betamax

1.4.1 Run tests

Run tests simply by:

```
python setup.py test
```

or (if you have installed dependencies):

```
python -m pytest [options]
pytest [options]
```

1.4.2 Betamax cassettes

Betamax cassettes are stored in `tests/fixtures/cassettes` directory. If you are not connected to the internet, Twitter API is not working and/or you don't have own API credentials you will use (replay) them in order to test API client.

If you want to run your own cassettes, you need to setup system variables

- `API_KEY`
- `API_SECRET`

Your test command then might look like:

```
API_KEY=<YOUR_API_KEY> API_SECRET=<YOUR_API_SECRET> \
python setup.py test
```

For more information, enjoy reading Betamax documentation.

1.5 Credits

This project was created as series of tasks for great subject MI-PYT teached at the Faculty of Information Technology, Czech Technical University in Prague (FIT CTU) by @hroncok and @encukou.

Thanks goes to Python community and also to developers, contributors and other people arounds projects that are used within **PYT TwitterWall**:

- requests
- Flask

- Flask-injector
- injector
- click
- Jinja
- pytest
- betamax
- flexmock
- Sphinx

CHAPTER 2

API docs

2.1 API

PYT TwitterWall consists of following package(s) and it's modules:

2.1.1 twitterwall

twitterwall.cli

twitterwall.common

twitterwall.web

2.1.2 Code examples

You can use parts of `twitterwall` package in your own projects.

Tweet examples

Let's say we got following JSON coming up as tweet from Twitter API (simplified example from API docs):

```
{
    "created_at": "Mon Sep 24 03:35:21 +0000 2012",
    "id_str": "250075927172759552",
    "entities": {
      "urls": [],
      "hashtags": [
        {
          "text": "freebandnames",
          "indices": [20, 34]
```

(continues on next page)

(continued from previous page)

```
        }
      ],
      "user_mentions": []
    },
    "text": "Aggressive Ponytail #freebandnames",
    "retweet_count": 0,
    "id": 250075927172759552,
    "retweeted": false,
    "user": {
      "name": "Sean Cummings",
      "profile_image_url": "http://a0.twimg.com/profile_images/2359746665/
↪1v6zfgqo8g0d3mk7ii5s_normal.jpeg",
      "created_at": "Mon Apr 26 06:01:55 +0000 2010",
      "location": "LA, CA",
      "profile_image_url_https": "https://si0.twimg.com/profile_images/2359746665/
↪1v6zfgqo8g0d3mk7ii5s_normal.jpeg",
      "id": 137238150,
      "followers_count": 70,
      "verified": false,
      "time_zone": "Pacific Time (US & Canada)",
      "description": "Born 330 Live 310",
      "profile_background_image_url": "http://a0.twimg.com/images/themes/theme1/bg.png
↪",
      "statuses_count": 579,
      "friends_count": 110,
      "screen_name": "sean_cummings"
    },
    "source": "Twitter for Mac"
  }
```

Tweet object `Tweet(jsondata)` serves as wrapper to those JSON data:

```
>>> tweet.get_id()
250075927172759552
>>> tweet.get_text()
'Aggressive Ponytail #freebandnames'
>>> tweet.get_nretweets()
0
>>> tweet.get_author_name()
'Sean Cummings'
>>> tweet.get_author_nick()
'sean_cummings'
>>> tweet.get_nfollows()
70
>>> tweet.get_created()
datetime.datetime(2012, 9, 24, 3, 35, 21)
>>> tweet.get_url()
'https://twitter.com/sean_cummings/statuses/250075927172759552'
>>> tweet.is_retweet()
False
>>> len(tweet.get_entities_of_type('hashtags'))
1
>>> tweet.get_entities_of_type('hashtags')[0]['text']
'freebandnames'
>>> tweet.get_entities_of_type('hashtags')[0]['indices']
[20, 34]
```

Flask filters examples

You can use Flask filters defined in `twitterwall.web` module so you can use them to show Tweet and it's parts in web page.

For Tweet object `Tweet(jsondata)` (same data as in *Tweet examples*), you can use for example:

```
>>> author_avatar(tweet)
Markup('<img src="http://a0.twimg.com/profile_images/2359746665/1v6zfgqo8g0d3mk7ii5s_
↪normal.jpeg" alt="Sean Cummings" class="avatar"/>')
>>> tweet_date(tweet)
'24/09/2012 03:35:21'
>>> enhance_text(tweet)
Markup('Aggressive Ponytail <a href="https://twitter.com/hashtag/freebandnames"␣
↪target="_blank">#freebandnames</a>')
>>> hashtags(tweet)
Markup('<a href="https://twitter.com/hashtag/freebandnames" target="_blank">
↪#freebandnames</a>')
>>> mentions(tweet)
Markup('')
>>> urls(tweet)
Markup('')
```

There are also some filters to other things than whole Tweet object, for example:

```
>>> user_link('andy123')
Markup('<a href="https://twitter.com/andy123" target="_blank">@andy123</a>')
>>> hashtag_link('python')
Markup('<a href="https://twitter.com/hashtag/python" target="_blank">#python</a>')
```

String are markup'ed by `jinja2.Markup`.

CLI walls examples

You can use or extend `CLIWall` and `CLIColorfulWall` to display tweets in different way. We use `click` as the printer but every object implementing methods `echo`, `clear` (and for styling also `style` and `secho`) can be used. The Tweet object in following examples is the same as in *Tweet examples*.

CLIWall examples

```
>>> wall = CLIWall(click)
>>> wall.print_tweet(tweet)
24/09/2012 03:35:21 (https://twitter.com/sean_cummings/statuses/250075927172759552)
Sean Cummings [sean_cummings]: Aggressive Ponytail #freebandnames
```

CLIColorfulWall examples

```
>>> wall = CLIColorfulWall(click)
>>> wall.print_tweet(tweet)
24/09/2012 03:35:21 (https://twitter.com/sean_cummings/statuses/250075927172759552)
Sean Cummings [sean_cummings]: Aggressive Ponytail #freebandnames
```

Only difference is that `click` will colorize and use bold/underline styling if the terminal allows it. For better understanding see picture below (from XFCE4 terminal):

```
>>> wall.print_tweet(tweet)
24/09/2012 03:35:21 (https://twitter.com/sean_cummings/statuses/250075927172759552)
Sean Cummings [sean_cummings]: Aggressive Ponytail #freebandnames
```

This part of the documentation is generated by apidoc.

CHAPTER 3

Indices and tables

- genindex
- search