PyPeri Documentation

Release 0.2.0

Tom Smith

Nov 16, 2017

Contents

Contents:

CHAPTER 1

PyPeri

PyPeri makes getting data out of Periscope easy, with a sane and understandable API.

- Free software: MIT license
- Documentation: https://pyperi.readthedocs.io.

1.1 Features

- Python 3
- Easy to understand API
- Fully tested & documented

Functionality so far:

- Get information about Users
- Get information about Broadcasts
- Get a User's Broadcast history

1.2 About

Periscope is pretty neat, but it's difficult to get information out of it in a programatic way. Getting answers to simple questions like: "How many broadcasts did our client do this week" is not trivial, and in-depth analysis is pretty tough. This project is an attempt to make the lives of people who dip into Digital Marketing a little better.

PyPeri attempts to do this by providing a sane interface, and hides some of the tedium of using Periscope's API. For example, Periscope makes doing simple things, like getting a list of a User's past Broadcasts a 2-step process:

1. Request a short-lived API session key from the Periscope Web Interface
2. Use the session key on the Periscope API Interface using the *getUserBroadcastsPublic* endpoint.

PyPeri simplifies this considerably:

```
>>> from pyperi import Peri
>>> pp = Peri()
>>> history = pp.get_user_broadcast_history(username='george_clinton')
```

1.3 Quick start

Install via pip:

```
$ pip install pyperi
```

Do stuff:

```
>>> from pyperi import Peri
>>> pp = Peri()
>>> history = pp.get_user_broadcast_history(username='george_clinton')
>>> if history:
...     print(history[0]['status'])
#George Clinton listening to music in houston
```

Full documentation is available here: https://pyperi.readthedocs.io

1.4 License

PyPeri is free software, distributed under the MIT license.

CHAPTER 2

Installation

2.1 Stable release

To install PyPeri, run this command in your terminal:

```
$ pip install pyperi
```

This is the preferred method to install PyPeri, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for PyPeri can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/takeontom/pyperi
```

Or download the tarball:

```
$ curl  -OL https://github.com/takeontom/pyperi/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

3.1 Introduction

To use PyPeri in a project:

```
>>> from pyperi import Peri
>>> pp = Peri()
>>> user_info = pp.get_user_info('376827')
>>> if user_info:
...     print(user_info['username'])
...
george_clinton
```

3.2 The API

3.2.1 get_user_info(user_id)

Retrieves a dictionary of information about the specified user.

Internally, this uses the *getUserPublic* Periscope API endpoint.

Example usage:

```
>>> pp.get_user_info('376827')
{'class_name': 'User',
 'created_at': '2015-03-28T19:52:59.150319197-07:00',
 'description': "It's always gotten better by Thursday. www.georgeclinton.com",
 'display_name': 'George Clinton',
 'id': '376827',
 'initials': '',
 'is_beta_user': False,
 'is_employee': False,
```

```
 'is_twitter_verified': True,
 'n_followers': 14535,
 'n_following': 739,
 'n_hearts': 2768274,
 'profile_image_urls': [{'height': 128,
                         'ssl_url': 'https://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_reasonably_small.jpeg',
                         'url': 'http://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_reasonably_small.jpeg',
                         'width': 128},
                        {'height': 200,
                         'ssl_url': 'https://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_200x200.jpeg',
                         'url': 'http://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_200x200.jpeg',
                         'width': 200},
                        {'height': 400,
                         'ssl_url': 'https://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_400x400.jpeg',
                         'url': 'http://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_400x400.jpeg',
                         'width': 400}],
 'twitter_id': '23177270',
 'twitter_screen_name': 'george_clinton',
 'username': 'george_clinton'}
```

If the User doesn't exist, then will return *None*:

```
>>> pp.get_user_info('156816811')
None
```

Be aware that Periscope expects *user_id* to be an Integer, anything else will result in a 400 HTTP error.

3.2.2 get_broadcast_info(broadcast_id)

Retrieves a dictionary of information about the specified broadcast.

Internally, this uses the *accessVideoPublic* Periscope API endpoint.

Example usage:

```
>>> pp.get_broadcast_info('1zqKVWybqeDGB')
{'available_for_replay': True,
 'camera_rotation': 271,
 'city': '',
 'class_name': 'Broadcast',
 'country': '',
 'country_state': '',
 'created_at': '2016-12-12T10:26:50.779286143-08:00',
 'end': '2016-12-12T11:43:16.649191397-08:00',
 'expiration': -1,
 'featured': False,
 'friend_chat': False,
 'has_location': False,
 'has_moderation': True,
 'height': 568,
 'highlight_available': True,
```

```
 'id': '1zqKVWybqeDGB',
 'image_url': 'https://tn.periscope.tv/C1RnUS8FqSm9cmrM0cXRM3MV3epaqaZgHc7WM_
↪kUgiABO6iDkhwyNl56XVdvdkOV6R5vWZbUaLgsefoJYg_0GA/chunk_878.jpg?Expires=1796931805&
↪Signature=FMb0NHoTz5BLpZIPCSS~
↪xyVTDTmYRHLlxQoNqsn96ffDMgs5N1WVBsIMtthsTujYcaCNie3QdP02SyxUsqQcmuqJaHodcAdYt~
↪8qDxs6qX2~8-foURHADqzOAMm6xUhvnjap4SuF~nZSsdmVPhuwl0lbF4ylG443huQB6qmQdzzlAZG1~
↪gVU9dHQXA5cdH0smEcOIc1ujkcmGX1wk-t2Gkd~C4ujC1szvcDBi5Bpjxb80k2-
↪oKDZs3TLqfOVzXaGaJesFshePFugFfVSrenJK2SQUEbulWAWOeWQf5ab~
↪RvwSvucVqy2CAzkR3xtxWFY1CfBR8Rmt8vTpa8uN~r9Ag__&Key-Pair-Id=APKAIHCXHHQVRTVSFRWQ',
 'image_url_small': 'https://tn.periscope.tv/C1RnUS8FqSm9cmrM0cXRM3MV3epaqaZgHc7WM_
↪kUgiABO6iDkhwyNl56XVdvdkOV6R5vWZbUaLgsefoJYg_0GA/chunk_878_thumb_128.jpg?
↪Expires=1796931805&Signature=JW9iFJZDqYeXKglWAsNh-f-D8QtBp7abjKKks2p80~k~
↪LoSGFdY289CDX~DFquTamf2t-HVB6oFwNaAXel49xbY1TZFI5VyOVfVp2UCZwxXsF03b4WgmCfO13EZFpD~
↪DIi0afALw~oxoHmk7n-WaBwlnBXogegSCbKmLPu5BGRlZJW9N7WTT7keLq-9DGRa5BSppcz-
↪e3frgieZEJBbldef01do1sQsywK5z86FY21XE~xszIfb6new2dWFtJ7Jr0QQmyQBMaK-
↪TU8o6y8kkqKU5cfjLPn~TwGUTUN2A0uyE6QDY2BRsO3sZJ7HoXzk-Cvxys0FzXr-i6iDU3McrKQ__&Key-
↪Pair-Id=APKAIHCXHHQVRTVSFRWQ',
 'ip_lat': 0,
 'ip_lng': 0,
 'is_locked': False,
 'iso_code': '',
 'language': 'en',
 'ping': '2016-12-12T11:43:08.860300617-08:00',
 'profile_image_url': 'https://pbs.twimg.com/profile_images/482413674338869248/
↪7GBlIsEm_reasonably_small.jpeg',
 'start': '2016-12-12T10:41:09.286489243-08:00',
 'state': 'ENDED',
 'status': '#GeorgeClinton funkin N D Garden to take to grandmother n law '
           'fish already packed while listening to Music',
 'tags': ['GeorgeClinton'],
 'tweet_id': '808381195981451264',
 'twitter_username': 'george_clinton',
 'updated_at': '2016-12-12T11:43:25.626913215-08:00',
 'user_display_name': 'George Clinton',
 'user_id': '376827',
 'username': 'george_clinton',
 'width': 320}
```

If the requested Broadcast does not exist, then will return *None*:

```
>>> pp.get_broadcast_info('aAaAAaaaAAAA')
None
```

3.2.3 get_user_broadcast_history(user_id=None, username=None)

Attempt to retrieve the available broadcast history for the specified user. If all goes well, it will return a List containing Dicts of broadcast history.

Example usage:

```
>>> pp.get_user_broadcast_history(username='george_clinton')
[
    {'available_for_replay': True,
    'camera_rotation': 0,
    'city': 'Lake Hiawatha, NJ',
    'class_name': 'Broadcast',
    'country': 'United States',
```

```
    'country_state': '',
    'created_at': '2016-05-27T20:03:44.732179203-07:00',
    'end': '2016-05-27T20:10:34.727278696-07:00',
    'expiration': -1,
    'featured': False,
    'friend_chat': False,
    'has_location': True,
    'has_moderation': False,
    'height': 568,
    'id': '1YqGoOkLbLAGv',
    'image_url': 'https://tn.periscope.tv/
↪x4AraTkCeWbW6CyWIQXoYrer45aCkUJZML7TdClEUrPD62GMRa8RE4ztZpvt-3nBxp-
↪eoZTZa9hEagUpQy9U9Q==/chunk_94.jpg?Expires=1779765037&
↪Signature=X3idH7qrnCykvVYqlThNFa-z-i9g~
↪mH7AUi8Lm5XdfuZwbV10GOh52xtGnfR5B45n4l0xSdz2Vz66rijM0QbimzPrUjyD09Gu72nNj2JPZzxOK3YZSjIZFDwzJLi71W
↪s2JmXr9TM5ZHiRkQTZ72sBYLIYJ6tmiWnWDAPE6wwcJ0ZIJfVqHyL8mRBGw5J4eFfbYe8JO7CkiDtNaQBI1H7n8BgzbnAqdU1J
↪RAnPvhQOFmA2lTdVhha2LHRs4gCIR9GNSL9PLNzyYqBnaxtH2Jo4sZPNsymZhYgklF7GOc7YS7KhmSilg__&
↪Key-Pair-Id=APKAIHCXHHQVRTVSFRWQ',
    'image_url_small': 'https://tn.periscope.tv/
↪x4AraTkCeWbW6CyWIQXoYrer45aCkUJZML7TdClEUrPD62GMRa8RE4ztZpvt-3nBxp-
↪eoZTZa9hEagUpQy9U9Q==/chunk_94_thumb_128.jpg?Expires=1779765037&
↪Signature=RwiKlnFJZrhkkZCSkNcJaaLlK~PwFEeCZH8Df-IoPt6FCxQZ-oVXGJzPMeP10Hm-
↪4FxuzHyZDREBNlvnJof5qRPYR7TL9QPEtAstLDbidUqnXSKMNs-gLUqMv5P7VP~
↪mAckViUW4nKv6kVxnX8XHuHFJkDbCGyX8c3lIkCUOt~Wk1nV0OzdHC72KJrwcR~
↪9752EQYSBKJxATlcYM0gTEPLDk8CGDuGqWS211D7ATcnfJcNJ9a8NliNSFPQTXsa2ue5vaPPlWsjpLX8sQCJq-
↪2SYbIHsDw7csmoPEMCYW-40jiSoLdSYbE4h9Xvjg0JupPrfjB9I6f3OfE8YrKVLJDQ__&Key-Pair-
↪Id=APKAIHCXHHQVRTVSFRWQ',
    'ip_lat': 40.862,
    'ip_lng': -74.412,
    'is_locked': False,
    'iso_code': 'US',
    'language': 'en',
    'n_total_watched': 529,
    'n_total_watching': 0,
    'n_watching': 0,
    'n_web_watching': 0,
    'ping': '2016-05-27T20:10:29.619904440-07:00',
    'profile_image_url': 'http://pbs.twimg.com/profile_images/482413674338869248/
↪7GBlIsEm_reasonably_small.jpeg',
    'start': '2016-05-27T20:04:44.636015867-07:00',
    'state': 'ENDED',
    'status': '#George Clinton',
    'tags': ['George'],
    'twitter_username': 'george_clinton',
    'updated_at': '2016-05-27T20:10:37.315814298-07:00',
    'user_display_name': 'George Clinton',
    'user_id': '376827',
    'username': 'george_clinton',
    'width': 320},
    {'available_for_replay': True,
    'camera_rotation': 0,
    'city': '',
    'class_name': 'Broadcast',
    'country': '',
    'country_state': '',
    'created_at': '2016-05-27T15:06:18.751291780-07:00',
    'end': '2016-05-27T15:12:58.781942930-07:00',
    'expiration': -1,
```

```
    'featured': False,
    'friend_chat': False,
    'has_location': False,
    'has_moderation': False,
    'height': 568,
    'id': '1nAKEnBDXXnGL',
    'image_url': 'https://tn.periscope.tv/eurvYFivZmErfex_Bnj33ESp711ZJPTfUC_
↪KDr5p7xfRuEl77eusMM59moAk00cBb7nI_U8orb95ivVVprgUkA==/chunk_96.jpg?
↪Expires=1779747181&Signature=XWyaURZLs808ds7u-vhmhpcf8zHjkWF~
↪6Jg9kwbdASu4Do7kUOjw11qNQodPfA3EtCFvE5ClN7Jvyt7lSzIArdp7VS5XlULPepu9YoIXBMRaB7RNeL7aIwNSrv3o3yw3ry2
↪hqOjLsFSQBXkfrmC-pQM~
↪wgwsJp41wSQDx8HSGjkPzh7U1MOBc6Nvf4KCvHgpVhSHtmkkGRRsXjVVJLQ0qEpws0GjMYC-hRuzSdf8~
↪9p4BwwPpAO79Cdl0w8haSKsxd9MI4F8JgdU1AtnyP575t7HZQH1wCk3b97U3F2fTm1ij0l-
↪RX6Y8ivnDnUcXIoB7j3ZTvt1piA__&Key-Pair-Id=APKAIHCXHHQVRTVSFRWQ',
    'image_url_small': 'https://tn.periscope.tv/eurvYFivZmErfex_Bnj33ESp711ZJPTfUC_
↪KDr5p7xfRuEl77eusMM59moAk00cBb7nI_U8orb95ivVVprgUkA==/chunk_96_thumb_128.jpg?
↪Expires=1779747181&Signature=a5KZMriA7-CoEYXCpHWU2j4TM~1WkZof-
↪wpeQtDsgO9haZcUL0qQy5hiuPWcGOD3IiYCAegYfRzZtaAgO78qM0QkbKZl5vEZLenXHep16ZB4qQAiDXBayN2fIqWKpAIefTPp
↪l11NZgs9JfWGOn4LZ2KDzGl7du1ZqwoViP56b1B2evPCAH0HSXgUhfvE4lcoBkunBwamK1amy8rDCTe-
↪u9kI3vqV~bN500RxbfiKyYeZW8ukwjqtYMSilPFilmlv8znaBXNiRA4lsOG4XGJC0xuHQ46JD0Wp5T85gH-
↪UH6Faqq0bh~aTOesVo~lRd9v0y6Uo3yIZGHK~vEz9McJw__&Key-Pair-Id=APKAIHCXHHQVRTVSFRWQ',
    'ip_lat': 0,
    'ip_lng': 0,
    'is_locked': False,
    'iso_code': '',
    'language': 'en',
    'n_total_watched': 1643,
    'n_total_watching': 0,
    'n_watching': 0,
    'n_web_watching': 0,
    'ping': '2016-05-27T15:12:48.828646925-07:00',
    'profile_image_url': 'http://pbs.twimg.com/profile_images/482413674338869248/
↪7GBlIsEm_reasonably_small.jpeg',
    'start': '2016-05-27T15:06:52.018796186-07:00',
    'state': 'ENDED',
    'status': '#George Clinton talking with driver',
    'tags': ['George'],
    'twitter_username': 'george_clinton',
    'updated_at': '2016-05-27T15:13:01.172799853-07:00',
    'user_display_name': 'George Clinton',
    'user_id': '376827',
    'username': 'george_clinton',
    'width': 320}
]
```

If the requested User does not exist, then will return *None*:

```
>>> pp.get_user_broadcast_history(username='some_bad_username')
None
```

3.2.4 get_web_public_user_session_tokens(user_id=None, username=None)

Request Public Session Tokens from Periscope, which are required for accesing some endpoints of the Web API.

User Session Tokens will provide you with access to endpoints for a specific Periscope User... they also expire after a few minutes, so it's recommended to request new Tokens for each request which requires them, rather than attempting to keep track of what Tokens you have, when they're about to expire, etc.

Returns a Dict containing the following Service Tokens:

- broadcastHistory
- serviceToken
- thumbnailPlaylist

For convenience, it will also contain the User ID to which the tokens give access to.

Example usage:

```
>>> pp.get_web_public_user_session_tokens(username='george_clinton')
{
    'broadcastHistory': '17f710-p1JiBEUn0sEib_
↪RdubMEIlN97QWGNF8BJ4eyajbdvMr4sIlwtDHQceV2yYnxSyCg4otO1Hf5eIRP7vzRKMnztRhTbW2WU6KBJ_
↪R6vt9rSJ',
    'serviceToken': '1-FeXtdwxPTDokFl0ZIoJVMN_9d2Pb5IdaaIx_XrX40fQWAm-nbT6ga0Kk_0_
↪QJhWB7ZlqGuuT-Cl3BFu0okWRRenAAHi1NreE0FX2Q5AfMfT',
    'thumbnailPlaylist': '1B4NxFGPCQH1IunHtK5cRWOkkbifgOK7Ipsx8uC9k_
↪WfKC6m1AU6MpnC5cKzxivdnJHC4ngY0EespKKzOzSTn49woz56N9YIuyNkl3Ao977oeC-uvY_xrxXW5',
    'user_id': '376827'
}
```

If the requested User does not exist, then will return *None*:

```
>>> pp.get_web_public_user_session_tokens(username='some_non_existant_user')
None
```

3.2.5 create_user_url(self, user_id=None, username=None)

Create a URL to the specified User's Periscope page. The generated URL will be different depending on whether the *user_id* or *username* was supplied.

Example usage:

```
>>> pp.create_user_url(username='george_clinton')
'https://www.periscope.tv/w/1eaKbRMEMEQKX'

>>> pp.create_user_url(user_id='376827')
'https://www.periscope.tv/u/376827'
```

3.2.6 parse_periscope_url(url)

Attempts to extract the *broadcast_id*, *user_id*, *username* or a combination of these from the supplied URL.

Supports the following URL formats:

- https://www.periscope.tv/w/<broadcast_id>
- https://www.periscope.tv/u/<user_id>
- https://www.periscope.tv/<username>
- https://www.periscope.tv/<username>/<broadcast_id>

Example usage:

```
>>> pp.parse_periscope_url('https://www.periscope.tv/w/1eaKbRMEMEQKX')
{'broadcast_id': '1eaKbRMEMEQKX', 'username': None, 'user_id': None}
```

3.2.7 request_api(endpoint, **params)

Makes a request to the Periscope API, with the supplied params and returns the result as a Dict.

Example usage:

```
>>> pp.request_api('getUserPublic', user_id='376827')
{'class_name': 'User',
 'created_at': '2015-03-28T19:52:59.150319197-07:00',
 'description': "It's always gotten better by Thursday. www.georgeclinton.com",
 'display_name': 'George Clinton',
 'id': '376827',
 'initials': '',
 'is_beta_user': False,
 'is_employee': False,
 'is_twitter_verified': True,
 'n_followers': 14535,
 'n_following': 739,
 'n_hearts': 2768274,
 'profile_image_urls': [{'height': 128,
                         'ssl_url': 'https://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_reasonably_small.jpeg',
                         'url': 'http://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_reasonably_small.jpeg',
                         'width': 128},
                        {'height': 200,
                         'ssl_url': 'https://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_200x200.jpeg',
                         'url': 'http://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_200x200.jpeg',
                         'width': 200},
                        {'height': 400,
                         'ssl_url': 'https://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_400x400.jpeg',
                         'url': 'http://pbs.twimg.com/profile_images/
↪482413674338869248/7GBlIsEm_400x400.jpeg',
                         'width': 400}],
 'twitter_id': '23177270',
 'twitter_screen_name': 'george_clinton',
 'username': 'george_clinton'}
```

It's common for a User or Broadcast to not be found. So any responses from Periscope with a 404 status code are handled by simply returning *None* from this method:

```
>>> pp.request_api('getUserPublic', user_id='6666666666')
None
```

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/takeontom/pyperi/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

PyPeri could always use more documentation, whether as part of the official PyPeri docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/takeontom/pyperi/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *pyperi* for local development.

1. Fork the *pyperi* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/pyperi.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv pyperi
$ cd pyperi/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 pyperi tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/takeontom/pyperi/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ py.test tests.test_pyperi
```

CHAPTER 5

Credits

5.1 Development Lead

- Tom Smith <tom@takeontom.com>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

History

6.1 0.2.0 (2016-12-23)

- URL parsing with *parse_periscope_url*
- Get a User's Broadcast history with *get_user_broadcast_history*
- Rename PyPeri class to Peri

6.2 0.1.0 (2016-12-12)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- search