Linksight Python API Client Documentation

Release 1.0.0

Thinking Machines Data Science

Jan 07, 2019

General

1 Installation **3**

2 Usage **5**
2.1 Generating your access token 5
2.2 Basic usage 5

3 Contributing **7**
3.1 Types of Contributions 7
3.2 Get Started! 8
3.3 Contributor Guidelines 9

4 Client **11**

5 Common **13**
5.1 linksight.common.settings module 13
5.2 linksight.common.utils module 13

6 Resources **15**
6.1 linksight.resource.base module 15
6.2 linksight.resource.resources module 16

7 Indices and tables **17**

Python Module Index **19**

Github repository: https://github.com/thinkingmachines/linksight-api-client

Python versions: 3.5, and 3.6

CHAPTER 1

Installation

You can see the requirements in the *requirements.txt* file. Some notable dependencies include:

- requests==2.21.0
- numpy==1.15.4
- pandas==0.23.4

You can install the API client either via *pip* or manually through the *setup.py* file. Using *pip*, simply run the following command:

```
pip install git+https://github.com/thinkingmachines/linksight-api-client.git
```

On the other hand, you can also clone this repository then run *install*:

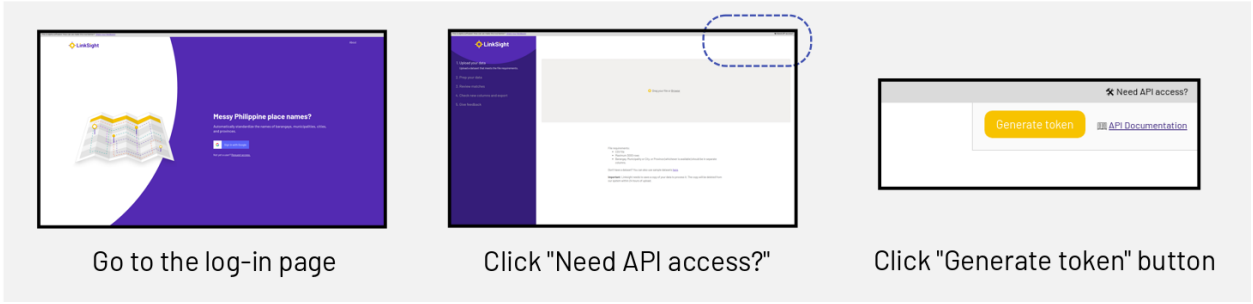
```
git clone git@github.com:thinkingmachines/linksight-api-client.git
cd linksight-api-client
python setup.py install
```

CHAPTER 2

Usage

2.1 Generating your access token

Using the Python API requires an API token, which you can obtain from this link. Simply log-in, go to the *upper-rightmost* corner, click the "Need API Access?" button, then "Generate token":



Go to the log-in page | Click "Need API access?" | Click "Generate token" button

2.2 Basic usage

The LinkSight API client can be used like any other Python packages. To perform matching, simply follow these steps:

- Create an instance of *linksight.Client* by supplying your API token
- Call *create_dataset* while providing the path to your dataset as a CSV file.
- **Perform matching on the resulting dataset by providing the columns** corresponding to the "Barangay", "Municipality", and "Province"

```
import linksight
import pandas

# Insert your API token here
```

(continues on next page)

(continued from previous page)

```
# In practice, you should not expose nor share your
# personal token to others
API_TOKEN = <API_TOKEN>

# Create an instance of the Client
ls = linksight.Client(<API_TOKEN>)

# Provide your dataset to the API
# You can also pass a DataFrame, i.e. ls.create_dataset(df)
with open('path/to/my/dataset.csv') as fp:
    ds = ls.create_dataset(fp)

# Perform matching by specifying the column names
# for the respective admin level
match = ds.match(
    source_bgy_col='Barangay',
    source_municity_col='City',
    source_prov_col='Province',
)

# Get the matched dataset as a DataFrame
match.df
```

CHAPTER 3

Contributing

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

3.1 Types of Contributions

There are many ways to contribute in this project:

3.1.1 Report Bugs

Report bugs at https://github.com/thinkingmachines/linksight-api-client/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.
- If you can, provide detailed steps to reproduce the bug.
- If you don't have steps to reproduce the bug, just note your observations in as much detail as you can. Questions to start a discussion about the issue are welcome.

3.1.2 Fix Bugs

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

3.1.3 Implement Features

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

Please do not combine multiple feature enhancements into a single pull request.

Note: this project is very conservative, so new features that aren't tagged with "please-help" might not get into core. We're trying to keep the code base small, extensible, and streamlined. Whenever possible, it's best to try and implement feature ideas as separate projects outside of the core codebase.

3.1.4 Write Documentation

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

If you want to review your changes on the documentation locally, you can do:

```
pip install -r requirements-dev.txt
cd docs
make html
```

This will compile the documentation, open it in your browser and start watching the files for changes, recompiling as you save.

3.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/thinkingmachines/linksight-api-client/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 :)

3.2 Get Started!

Ready to contribute? Here's how to set up *linksight-api-client* for local development.

1. Fork the *linksight-api-client* repo on GitHub.
2. Clone your fork locally

```
git clone git@github.com:thinkingmachines/linksight-api-client.git
```

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

```
cd linksight-api-client
virtualenv venv
source venv/bin/activate
pip install -r requirements-dev.txt
```

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. In addition, ensure that your code is formatted using black

```
flake8 linksight tests
black linksight tests
python setup.py test or py.test
tox
```

To get flake8, black, and tox, just pip install them into your virtualenv. If you wish, you can add pre-commit hooks for both flake8 and black to make all formatting easier.

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.

3.3 Contributor Guidelines

3.3.1 Pull Request Guidelines

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

1. There is an issue that the pull request corresponds to.
2. The pull request should include tests.
3. 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.
4. The pull request should work for Python 3.5 and 3.6

3.3.2 Coding Standards

- We use PEP8 as our coding standard
- In addition, we use black as our code formatter

3.3.3 Running the tests

We use pytest for testing. There are three sets of tests in the API Client:

- **Regular Tests**: invokes a mocked API and can be run offline
- **Web Tests** (*web*): hits the external server and requires an API token.
- **Contract Tests** (*contract*): checks for consistency of mocked data to live data.

All contract tests are web tests. It is recommended that external developers write both regular and web tests, but locally do their checks on regular tests. Web tests should only be handled by a continuous integration service (via a testing account), unless you want to use your own API token.

```
# In project root
pytest -m "not webtest" -v
```

Lastly, to run all tests:

```
# In project root
pytest -v
```

CHAPTER 4

Client

The Client module serves as the entrypoint for all our interactions with the external LinkSight API. The main usage pattern requires us to instantiate a client, then use that to handle all requests to LinkSight.

Most of the return values are in the form of a *linksight.resource*, which inherits a dictionary type.

Note: Needless to say, interacting with the LinkSight API requires an internet connection.

class linksight.client.**Client**(*token*)

Bases: requests.sessions.Session

An object for handling all requests

create_dataset(*data*)

Create a dataset from a given file

In order to create a dataset, simply create a context from the given CSV file and pass it to this method:

```
from linksight import Client

API_TOKEN = <Insert your API_TOKEN here>

ls = Client(API_TOKEN)
with open('path/to/query/file.csv') as f:
    dataset = ls.create_dataset(f)
```

Parameters **data** (*_io.TextIOWrapper or pandas.DataFrame*) – A file handler or a pandas DataFrame

Returns The Dataset resource that can be used for matching

Return type *linksight.resource.resources.Dataset*

get_user(*id='me'*)

Get the user creating the request

Parameters **id** (*str*) – User ID initiating the request

Returns The user information retrieved from the API

Return type *linksight.resource.resources.User*

CHAPTER 5

Common

This includes various helper methods and constants that are common to the whole package

5.1 linksight.common.settings module

Global-wide constants for package settings

This file contains various constants that are used to configure package interaction with the LinkSight API. These constants include:

- `VERSION`: The LinkSight version it interacts upon.
- `USER_AGENT`: The URL for the user agent proxy.
- `ENDPOINT`: The base URL for interacting with the API.

5.2 linksight.common.utils module

Utility methods common to the package

`linksight.common.utils.`**`urljoin`**`(*args)`

Join a set of strs for URL creation

This method is commonly used by appending a URL to the *ENDPOINT*

Parameters **`str`** – URL names to join

Returns The joined URL

Return type str

CHAPTER 6

Resources

This includes all resources that are retrieved or sent to LinkSight's external API.

6.1 linksight.resource.base module

class linksight.resource.base.**Resource**(*client*, *resp*)

Bases: dict

Base Resource class

classmethod create(*client*, ***kwargs*)

Create a POST request to LinkSight endpoint

Parameters **client** (*requests.Session*) – An instance of the client

Returns An instance of itself with the appropriate response

Return type dict

create_instance_url(**args*)

Create the instance url to pass the request onto

Parameters **args** (*str*) –

Returns The instance url

Return type str

classmethod retrieve(*client*, *id*)

Create a GET request to LinkSight endpoint

Parameters

- **client** (*requests.Session*) – An instance of the client
- **id** (*str*) – ID of the user

Returns An instance of itself with the appropriate response

Return type dict

6.2 linksight.resource.resources module

Contains various resources for API calls

class linksight.resource.resources.**Dataset**(*client*, *resp*)
Bases: *linksight.resource.base.Resource*

Resource for handling dataset-related API calls

match(*source_bgy_col=None*, *source_municity_col=None*, *source_prov_col=None*, *export=True*)
Perform a match using the LinkSight API

Parameters

- **source_bgy_col** (*str*) – Column name for the barangays
- **source_municity_col** (*str*) – Column name for the municipality
- **source_prov_col** (*str*) – Column name for the province
- **export** (*bool*) – Export the resulting match (default is True)

Returns The API response containing the match

Return type *linksight.resource.resources.Match*

url = 'datasets'

class linksight.resource.resources.**Match**(*client*, *resp*)
Bases: *linksight.resource.base.Resource*

Resource for handling resulting dataset matches

url = 'matches'

class linksight.resource.resources.**User**(*client*, *resp*)
Bases: *linksight.resource.base.Resource*

Resource for handling user-related API calls

url = 'users'

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

l

linksight.client, 11
linksight.common, 13
linksight.common.settings, 13
linksight.common.utils, 13
linksight.resource, 15
linksight.resource.base, 15
linksight.resource.resources, 16

Index

C

Client (class in linksight.client), 11
create() (linksight.resource.base.Resource class method), 15
create_dataset() (linksight.client.Client method), 11
create_instance_url() (linksight.resource.base.Resource method), 15

D

Dataset (class in linksight.resource.resources), 16

G

get_user() (linksight.client.Client method), 11

L

linksight.client (module), 11
linksight.common (module), 13
linksight.common.settings (module), 13
linksight.common.utils (module), 13
linksight.resource (module), 15
linksight.resource.base (module), 15
linksight.resource.resources (module), 16

M

Match (class in linksight.resource.resources), 16
match() (linksight.resource.resources.Dataset method), 16

R

Resource (class in linksight.resource.base), 15
retrieve() (linksight.resource.base.Resource class method), 15

U

url (linksight.resource.resources.Dataset attribute), 16
url (linksight.resource.resources.Match attribute), 16
url (linksight.resource.resources.User attribute), 16
urljoin() (in module linksight.common.utils), 13
User (class in linksight.resource.resources), 16