
Asyncflux Documentation

Release 0.0+

Jorge Puente-Sarrín

March 11, 2016

1	Installation	3
2	Documentation	5
3	License	7
4	Indices and tables	9
4.1	Package Documentation	9
4.2	Release Notes	12
	Python Module Index	13

Asynchronous client for [InfluxDB](#) and [Tornado](#).

Installation

You can use `pip` to install Asyncflux:

```
$ pip install git+https://github.com/puentesarrin/asyncflux.git
```

Documentation

[Sphinx](#) is needed to generate the documentation. Documentation can be generated by issuing the following commands:

```
$ cd docs  
$ make html
```

Or simply:

```
$ python setup.py doc
```

Also, the current documentation can be found at [ReadTheDocs](#).

License

Asyncflux is available under the [Apache License, Version 2.0](#).

Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)

4.1 Package Documentation

4.1.1 `asyncflux` Package

Asynchronous client for InfluxDB and Tornado.

`asyncflux.__init__.version = '0.0+'`
 Current version of Asyncflux.

class `asyncflux.__init__.AsyncfluxClient` (*host=None, port=None, username=None, password=None, is_secure=False, io_loop=None, **kwargs*)

Bases: `object`

HOST = 'localhost'

PASSWORD = 'root'

PORT = 8086

USERNAME = 'root'

authenticate_cluster_admin (**args, **kwargs*)

base_url

change_cluster_admin_password (**args, **kwargs*)

create_cluster_admin (**args, **kwargs*)

create_database (**args, **kwargs*)

delete_cluster_admin (**args, **kwargs*)

delete_database (**args, **kwargs*)

get_cluster_admin_names (**args, **kwargs*)

get_cluster_admins (**args, **kwargs*)

get_database_names (**args, **kwargs*)

```

get_databases (*args, **kwargs)
get_shard_spaces (*args, **kwargs)
host
password
ping (*args, **kwargs)
port
request (*args, **kwargs)
username

```

4.1.2 asynflux Modules

asynflux.client – Connection to InfluxDB

Connection to InfluxDB

```

class asynflux.client.AsyncfluxClient (host=None, port=None, username=None, pass-
                                         word=None, is_secure=False, io_loop=None,
                                         **kwargs)

```

Bases: `object`

```

HOST = 'localhost'
PASSWORD = 'root'
PORT = 8086
USERNAME = 'root'
authenticate_cluster_admin (*args, **kwargs)
base_url
change_cluster_admin_password (*args, **kwargs)
create_cluster_admin (*args, **kwargs)
create_database (*args, **kwargs)
delete_cluster_admin (*args, **kwargs)
delete_database (*args, **kwargs)
get_cluster_admin_names (*args, **kwargs)
get_cluster_admins (*args, **kwargs)
get_database_names (*args, **kwargs)
get_databases (*args, **kwargs)
get_shard_spaces (*args, **kwargs)
host
password
ping (*args, **kwargs)
port
request (*args, **kwargs)

```

username

asyncflux.database – Database level operations

Database level operations

```
class asyncflux.database.Database (client, name)
    Bases: object
    authenticate_user (*args, **kwargs)
    change_user_password (*args, **kwargs)
    change_user_permissions (*args, **kwargs)
    change_user_privileges (*args, **kwargs)
    client
    create_user (*args, **kwargs)
    delete (*args, **kwargs)
    delete_user (*args, **kwargs)
    get_user (*args, **kwargs)
    get_user_names (*args, **kwargs)
    get_users (*args, **kwargs)
    name
    update_user (*args, **kwargs)
```

asyncflux.clusteradmins – Tools for cluster administration

asyncflux.testing – Unit testing support for asynchronous code

Unit testing support for asynchronous code

```
class asyncflux.testing.AsyncfluxTestCase (methodName='runTest', **kwargs)
    Bases: tornado.testing.AsyncTestCase
    assert_mock_args (fetch_mock, path, method='GET', body=None, auth_username='root',
                      auth_password='root', *args, **kwargs)
    patch_fetch_mock (client)
    setup_fetch_mock (fetch_mock, status_code, **kwargs)
    stop_op (result, error)
```

asyncflux.testing.gen_test (func=None, timeout=None)

Testing equivalent of @gen.coroutine, to be applied to test methods.

@gen.coroutine cannot be used on tests because the *IOLoop* is not already running. @gen_test should be applied to test methods on subclasses of *AsyncTestCase*.

Example:

```
class MyTest(AsyncHTTPTestCase):
    @gen_test
    def test_something(self):
        response = yield gen.Task(self.fetch('/'))
```

By default, `@gen_test` times out after 5 seconds. The timeout may be overridden globally with the `ASYNC_TEST_TIMEOUT` environment variable, or for each test with the `timeout` keyword argument:

```
class MyTest(AsyncHTTPTestCase):
    @gen_test(timeout=10)
    def test_something_slow(self):
        response = yield gen.Task(self.fetch('/'))
```

New in version 3.1: The `timeout` argument and `ASYNC_TEST_TIMEOUT` environment variable.

Changed in version 4.0: The wrapper now passes along `*args`, `**kwargs` so it can be used on functions with arguments.

asyncflux.util – General-purpose utilities

General-purpose utilities

`asyncflux.util.asyncflux_coroutine(f)`

A coroutine that accepts an optional callback.

Given a callback, the function returns `None`, and the callback is run with `(result, error)`. Without a callback the function returns a `Future`.

`asyncflux.util.snake_case(string)`

`asyncflux.util.snake_case_dict(_dict)`

4.2 Release Notes

4.2.1 Next Release

Very soon

- Initial release.
- Added Sphinx docs and [ReadTheDocs](#) configuration.

a

`asyncflux.__init__`, [9](#)
`asyncflux.client`, [10](#)
`asyncflux.database`, [11](#)
`asyncflux.testing`, [11](#)
`asyncflux.util`, [12](#)

A

`assert_mock_args()` (`asyncflux.testing.AsyncfluxTestCase` method), 11
`asyncflux.__init__` (module), 9
`asyncflux.client` (module), 10
`asyncflux.database` (module), 11
`asyncflux.testing` (module), 11
`asyncflux.util` (module), 12
`asyncflux_coroutine()` (in module `asyncflux.util`), 12
`AsyncfluxClient` (class in `asyncflux.__init__`), 9
`AsyncfluxClient` (class in `asyncflux.client`), 10
`AsyncfluxTestCase` (class in `asyncflux.testing`), 11
`authenticate_cluster_admin()`
 (`asyncflux.__init__.AsyncfluxClient` method), 9
`authenticate_cluster_admin()`
 (`asyncflux.client.AsyncfluxClient` method), 10
`authenticate_user()` (`asyncflux.database.Database` method), 11

B

`base_url` (`asyncflux.__init__.AsyncfluxClient` attribute), 9
`base_url` (`asyncflux.client.AsyncfluxClient` attribute), 10

C

`change_cluster_admin_password()`
 (`asyncflux.__init__.AsyncfluxClient` method), 9
`change_cluster_admin_password()`
 (`asyncflux.client.AsyncfluxClient` method), 10
`change_user_password()` (`asyncflux.database.Database` method), 11
`change_user_permissions()` (`asyncflux.database.Database` method), 11
`change_user_privileges()` (`asyncflux.database.Database` method), 11
`client` (`asyncflux.database.Database` attribute), 11

`create_cluster_admin()` (`asyncflux.__init__.AsyncfluxClient` method), 9
`create_cluster_admin()` (`asyncflux.client.AsyncfluxClient` method), 10
`create_database()` (`asyncflux.__init__.AsyncfluxClient` method), 9
`create_database()` (`asyncflux.client.AsyncfluxClient` method), 10
`create_user()` (`asyncflux.database.Database` method), 11

D

`Database` (class in `asyncflux.database`), 11
`delete()` (`asyncflux.database.Database` method), 11
`delete_cluster_admin()` (`asyncflux.__init__.AsyncfluxClient` method), 9
`delete_cluster_admin()` (`asyncflux.client.AsyncfluxClient` method), 10
`delete_database()` (`asyncflux.__init__.AsyncfluxClient` method), 9
`delete_database()` (`asyncflux.client.AsyncfluxClient` method), 10
`delete_user()` (`asyncflux.database.Database` method), 11

G

`gen_test()` (in module `asyncflux.testing`), 11
`get_cluster_admin_names()`
 (`asyncflux.__init__.AsyncfluxClient` method), 9
`get_cluster_admin_names()`
 (`asyncflux.client.AsyncfluxClient` method), 10
`get_cluster_admins()` (`asyncflux.__init__.AsyncfluxClient` method), 9
`get_cluster_admins()` (`asyncflux.client.AsyncfluxClient` method), 10
`get_database_names()` (`asyncflux.__init__.AsyncfluxClient` method), 9
`get_database_names()` (`asyncflux.client.AsyncfluxClient` method), 10
`get_databases()` (`asyncflux.__init__.AsyncfluxClient` method), 10

get_databases() (asyncflux.client.AsyncfluxClient method), 10	USERNAME (asyncflux.__init__.AsyncfluxClient attribute), 9
get_shard_spaces() (asyncflux.__init__.AsyncfluxClient method), 10	username (asyncflux.__init__.AsyncfluxClient attribute), 10
get_shard_spaces() (asyncflux.client.AsyncfluxClient method), 10	USERNAME (asyncflux.client.AsyncfluxClient attribute), 10
get_user() (asyncflux.database.Database method), 11	username (asyncflux.client.AsyncfluxClient attribute), 11
get_user_names() (asyncflux.database.Database method), 11	
get_users() (asyncflux.database.Database method), 11	V
	version (in module asyncflux.__init__), 9

H

HOST (asyncflux.__init__.AsyncfluxClient attribute), 9
 host (asyncflux.__init__.AsyncfluxClient attribute), 10
 HOST (asyncflux.client.AsyncfluxClient attribute), 10
 host (asyncflux.client.AsyncfluxClient attribute), 10

N

name (asyncflux.database.Database attribute), 11

P

PASSWORD (asyncflux.__init__.AsyncfluxClient attribute), 9
 password (asyncflux.__init__.AsyncfluxClient attribute), 10
 PASSWORD (asyncflux.client.AsyncfluxClient attribute), 10
 password (asyncflux.client.AsyncfluxClient attribute), 10
 patch_fetch_mock() (asyncflux.testing.AsyncfluxTestCase method), 11
 ping() (asyncflux.__init__.AsyncfluxClient method), 10
 ping() (asyncflux.client.AsyncfluxClient method), 10
 PORT (asyncflux.__init__.AsyncfluxClient attribute), 9
 port (asyncflux.__init__.AsyncfluxClient attribute), 10
 PORT (asyncflux.client.AsyncfluxClient attribute), 10
 port (asyncflux.client.AsyncfluxClient attribute), 10

R

request() (asyncflux.__init__.AsyncfluxClient method), 10
 request() (asyncflux.client.AsyncfluxClient method), 10

S

setup_fetch_mock() (asyncflux.testing.AsyncfluxTestCase method), 11
 snake_case() (in module asyncflux.util), 12
 snake_case_dict() (in module asyncflux.util), 12
 stop_op() (asyncflux.testing.AsyncfluxTestCase method), 11

U

update_user() (asyncflux.database.Database method), 11