dogstatsd-python Documentation

Release 0.1

Datadog

March 20, 2015

Contents

DogStatsd is a Python client for DogStatsd, a Statsd fork for Datadog.

class `statsd.`**DogStatsd**(*host='localhost'*, *port=8125*, *max_buffer_size=50*)

close_buffer()
Flush the buffer and switch back to single metric packets

connect(*host*, *port*)
Connect to the statsd server on the given host and port.

decrement(*metric*, *value=1*, *tags=None*, *sample_rate=1*)
Decrement a counter, optionally setting a value, tags and a sample rate.

```
>>> statsd.decrement('files.remaining')
>>> statsd.decrement('active.connections', 2)
```

event(*title*, *text*, *alert_type=None*, *aggregation_key=None*, *source_type_name=None*, *date_happened=None*, *priority=None*, *tags=None*, *hostname=None*)

Send an event. Attributes are the same as the Event API. http://docs.datadoghq.com/api/

```
>>> statsd.event('Man down!', 'This server needs assistance.')
>>> statsd.event('The web server restarted', 'The web server is up again', alert_type='succe
```

gauge(*metric*, *value*, *tags=None*, *sample_rate=1*)
Record the value of a gauge, optionally setting a list of tags and a sample rate.

```
>>> statsd.gauge('users.online', 123)
>>> statsd.gauge('active.connections', 1001, tags=["protocol:http"])
```

get_socket()
Return a connected socket

histogram(*metric*, *value*, *tags=None*, *sample_rate=1*)
Sample a histogram value, optionally setting tags and a sample rate.

```
>>> statsd.histogram('uploaded.file.size', 1445)
>>> statsd.histogram('album.photo.count', 26, tags=["gender:female"])
```

increment(*metric*, *value=1*, *tags=None*, *sample_rate=1*)
Increment a counter, optionally setting a value, tags and a sample rate.

```
>>> statsd.increment('page.views')
>>> statsd.increment('files.transferred', 124)
```

open_buffer(*max_buffer_size=50*)
Open a buffer to send a batch of metrics in one packet

You can also use this as a context manager.

```
>>> with DogStatsd() as batch:
>>>     batch.gauge('users.online', 123)
>>>     batch.gauge('active.connections', 1001)
```

service_check(*check_name*, *status*, *tags=None*, *timestamp=None*, *hostname=None*, *message=None*)
Send a service check run.

```
>>> statsd.service_check('my_service.check_name', DogStatsd.WARNING)
```

set(*metric*, *value*, *tags=None*, *sample_rate=1*)
Sample a set value.

```
>>> statsd.set('visitors.uniques', 999)
```

timed(*metric*, *tags=None*, *sample_rate=1*)
A decorator that will measure the distribution of a function's run time. Optionally specify a list of tag or a sample rate.

```
@statsd.timed('user.query.time', sample_rate=0.5)
def get_user(user_id):
    # Do what you need to ...
    pass

# Is equivalent to ...
start = time.time()
try:
    get_user(user_id)
finally:
    statsd.timing('user.query.time', time.time() - start)
```

timing(*metric*, *value*, *tags=None*, *sample_rate=1*)
Record a timing, optionally setting tags and a sample rate.

```
>>> statsd.timing("query.response.time", 1234)
```

statsd.**statsd**
A global DogStatsd instance that is easily shared across an application's modules. Initialize this once in your application's set-up code and then other modules can import and use it without further configuration.

```
>>> from statsd import statsd
>>> statsd.connect(host='localhost', port=8125)
```

CHAPTER 1

Source

The DogStatsd source is freely available on Github. Check it out here.

CHAPTER 2

Get in Touch

If you'd like to suggest a feature or report a bug, please add an issue here. If you want to talk about DataDog in general, reach out at datadoghq.com.

S

C

D

E

G

H

I

O

S

T