mongotriggers Documentation

Release 2.0.0

Dror Asaf

Mar 20, 2021

Getting Started

This package provides a pythonic interface to allow real time updating, when MongoDB is updated, this update could be up to application layer, or just reside in the backend of the application.

Modern applications are event-driven and not query-driven, therefore whenever there is an update, the application would like to receive a feedback. This package enables this kind of behaviour.

CHAPTER 1

Index

Getting Started

1.1 Install MongoTriggers

You can install dask with `pip`, or by installing from source.

1.1.1 Pip

To install MongoTriggers with `pip`:

- `pip install mongotriggers`

1.1.2 Install from Source

To install mongotriggers from source, clone the repository from github:

```
git clone https://github.com/drorasaf/mongotriggers.git
cd mongotriggers
python setup.py install
```

or use `pip` locally if you want to install all dependencies as well:

```
pip install -e .
```

1.1.3 Test

Test mongotriggers with `py.test`:

```
cd mongotriggers
py.test mongotriggers
```

1.2 Examples

1.2.1 Simple

This example provides the simplest option for using the package.

```
from mongotriggers import MongoTrigger
from pymongo import MongoClient

def notify_manager(op_document):
    print ('wake up! someone is adding me money')

client = MongoClient(host='localhost', port=27017)
triggers = MongoTrigger(client)

# listens to update/insert/delete, any of these will trigger the callback
triggers.register_op_trigger(notify_manager, 'my_account', 'my_transactions')
triggers.tail_oplog()

# make an operation to simulate interaction
client['my_account']['my_transactions'].insert_one({"balance": 1000})
triggers.stop_tail()
```

1.2.2 Tail from certain point in time

This example provides explanations on how to start listening only from a certain point in time, usually this will be helpful when persistency is required.

```
from mongotriggers import MongoTrigger
from pymongo import MongoClient
from bson.timestamp import Timestamp
import time

def notify_manager(op_document):
    print ('wake up! someone is adding me money')

client = MongoClient(host='localhost', port=27017)

# do something in collection to verify it is not called
client['my_account']['my_transactions'].insert_one({"balance": 1000})
# long waiting time due to timestamp
time.sleep(5)
now = Timestamp(datetime.datetime.utcnow(), 0)
# will get notified only if event occurred after specified now
triggers = MongoTrigger(client, since=now)
```

(continues on next page)

(continued from previous page)

```
triggers.register_op_trigger(notify_manager, 'my_account', 'my_transactions')
triggers.tail_oplog()

# write to collection to verify we receive the callback
client['my_account']['my_transactions'].insert_one({"balance": 1000})
triggers.stop_tail()
```

API Reference

- *MongoTriggers*

1.3 MongoTriggers

class mongotriggers.mongotriggers.**MongoTrigger**(*conn*, *since=None*)

Bases: object

tail_oplog()

Listens to oplog and fire the registered callbacks

stop_tail()

Stops listening to the oplog, no callbacks after calling this

register_op_trigger(*func*, *db_name=None*, *collection_name=None*)

Watches the specified database and collections for any changes

Parameters

- **func** (*callback*) – function to be invoked when any operation occurs
- **db_name** (*str*) – name of Mongo database to watch for changes
- **collection_name** (*str*) – name of Mongo collection to watch for changes

register_insert_trigger(*func*, *db_name=None*, *collection_name=None*)

Adds an insert callback to the specified namespace

Parameters

- **func** (*callback*) – callback to execute when an insert operation occur
- **db_name** (*str*) – name of Mongo database to watch for changes
- **collection_name** (*str*) – name of Mongo collection to watch for changes

register_update_trigger(*func*, *db_name=None*, *collection_name=None*)

Adds ann update callback to the specified namespace

Parameters

- **func** (*callback*) – callback to execute when an update operation occur
- **db_name** (*str*) – name of Mongo database to watch for changes
- **collection_name** (*str*) – name of Mongo collection to watch for changes

register_delete_trigger(*func*, *db_name=None*, *collection_name=None*)

Adds a delete callback to the specified namespace

Parameters

- **func** (*callback*) – callback to execute when a delete operation occur

- **db_name** (*str*) – name of Mongo database to watch for changes
- **collection_name** (*str*) – name of Mongo collection to watch for changes

unregister_op_trigger(*func*, *db_name=None*, *collection_name=None*)
Removes all callbacks from the specified namespace

Parameters

- **func** (*callback*) – callback to disable when any operation occur
- **db_name** (*str*) – name of Mongo database to watch for changes
- **collection_name** (*str*) – name of Mongo collection to watch for changes

unregister_insert_trigger(*func*, *db_name=None*, *collection_name=None*)
Removes an insert callback from the specified namespace

Parameters

- **func** (*callback*) – callback to disable when an insert operation occur
- **db_name** (*str*) – name of Mongo database to watch for changes
- **collection_name** (*str*) – name of Mongo collection to watch for changes

unregister_update_trigger(*func*, *db_name=None*, *collection_name=None*)
Removes an update callback from the specified namespace

Parameters

- **func** (*callback*) – callback to disable when an insert operation occur
- **db_name** (*str*) – name of Mongo database to watch for changes
- **collection_name** (*str*) – name of Mongo collection to watch for changes

unregister_delete_trigger(*func*, *db_name=None*, *collection_name=None*)
Removes a delete callback from the specified namespace

Parameters

- **func** (*callback*) – callback to disable when an insert operation occur
- **db_name** (*str*) – name of Mongo database to watch for changes
- **collection_name** (*str*) – name of Mongo collection to watch for changes

Help & Reference

- *Contact and Support*

1.4 Contact and Support

1.4.1 Where to ask for help and provide feedback

If you have any question or you just want to help us make the documentation better! Please create an issue in mongotriggers issues in Github Issue Tracker.

Index