Mongu Documentation

Release 0.4.4

Tevin Zhang

August 25, 2015

Contents

Introducing to Mongu What and Why.

Example usage You don't need User Guides but examples to GET STARTED!

API References We don't hide nothing from you.

Example of using builtin Counter and CounterMixin Another example of how to use the builtin `Counter` model and the corresponding `CounterMixin` class.

Introducing to Mongu

Mongu is yet another Python Object-Document Mapper on top of `PyMongo`. It's lightweight, intuitive to use and easy to understand.

If those **heavy and slow layers** have nothing or more than you need, Mongu maybe the one for you.

You are the only one who knows what you reall need.

Therefor **Mongu does nothing but a skeleton for you to fill**.

Actually, if you have ever tried to write your own ODM, you may already implemented parts of **Mongu** :D

Example usage

We don't assume you are stupid, here we go

Model definition:

```python
from mongu import Client, Model

c = Client()                                  # connect to MongoDB


@c.register_model
class User(Model):
    _database_   = 'test'                     # database name
    _collection_ = 'users'                    # collection name
    _defaults_ = {'is_activated': False}  # default attribute (callable value is supported)

    def activate(self):                       # a custom method
        self.is_activated = True
```

Basic manipulation

The model is a dict:

```python
>> user = User(username='Mongu')
>> user
User({'username': 'Mongu', 'is_activated': False})
>> 'username' in user
True
```

With your methods:

```python
>> user.activate()
```

That work:

```python
>> user
User({'username': 'Mongu', 'is_activated': True})
>> user.save()
>> user
User({'username': 'Mongu', 'is_activated': True, '_id': ObjectId('534f81bd7246ef6955d2388f')})
```

`ObjectId` won't be a pain:

```python
>> repr(user._id)
"ObjectId('534f81bd7246ef6955d2388f')"
>> repr(user.id)
"534f81bd7246ef6955d2388f"
```

You can find by `str` or `ObjectId`, The following two lines get the same result:

```
>> User.by_id(user._id)
>> User.by_id(user.id)
User({u'username': u'Mongu', u'is_activated': True, u'_id': ObjectId('534f81bd7246ef6955d2388f')})
```

Query (It's really just `PyMongo`)

Create some users:

```
>> for name in ('Mongu', 'Rocks'):
>>    User(username=name).save()
>> list(User.find())
[User({u'username': u'Mongu', u'is_activated': False, u'_id': ObjectId('534f87c27246ef95a3294c28')}),
 User({u'username': u'Rocks', u'is_activated': False, u'_id': ObjectId('534f87c27246ef95a3294c29')})]
```

It's naked `PyMongo`, nothing hidden from you:

```
>> User.collection
Collection(Database(MongoClient('localhost', 27017), u'test'), u'users')
```

`PyMongo` raw query:

```
>> user_naked = User.collection.find_one({'username': 'Rocks'})
```

Use `Mongu` to dress up:

```
>> user_dressed = User.find_one({'username': 'Rocks'})
```

Differences:

```
>> user_dressed.activate()
>> user_naked.activate()  # this will raise an AttributeError
```

You konw Why:

```
>> type(user_naked)
<type 'dict'>
>> type(user_dressed)
<class '__main__.User'>
```

API References

3.1 Basic

class mongu.**Client**(**args*, ***kwargs*)

For Connecting to MongoDB and registering model classes.

enable_counter(*base=None*, *database='counter'*, *collection='counters'*)

Register the builtin counter model, return the registered Counter class and the corresponding CounterMixin class.

The CounterMixin automatically increases and decreases the counter after model creation(save without _id) and deletion.

It contains a classmethod count() which returns the current count of the model collection.

register_model(*model_cls*)

Decorator for registering model.

3.2 Extra

class mongu.**ObjectDict**

Makes a dictionary behave like an object, with attribute-style access.

3.3 Base Model

class mongu.**Model**

Dict-like class with optional default key-values that binds to a collection.

classmethod by_id(*oid*)

Find a model object by its ObjectId, oid can be string or ObjectId

classmethod delete_by_id(*oid*)

Delete a document from collection by its ObjectId, oid can be string or ObjectId

classmethod from_dict(*d*)

Build model object from a dict. Will be removed in v1.0

classmethod from_cursor(*cursor*)

Build model object from a pymongo cursor.

classmethod **find**(**args*, ***kwargs*)
>   Same as `collection.find`, return model object instead of simple dict.

classmethod **find_one**(**args*, ***kwargs*)
>   Same as `collection.find_one`, return model object instead of simple dict.

id
>   String representation of attribute `_id`.

reload(*d=None*)
>   Reload model from given dict or database.

on_save(*old_dict*)
>   Hook after save.

save()
>   Save model object to database.

on_delete(*deleted_obj*)
>   Hook after delete successful.

delete()
>   Remove from database.

3.4 Builtin Counter

For more information: What and Why

class mongu.**Counter**
>   Builtin counter model.

classmethod **set_to**(*name*, *num*)
>   Set counter of `name` to `num`.

classmethod **change_by**(*name*, *num*)
>   Change counter of `name` by `num` (can be negative).

classmethod **increase**(*name*)
>   Increase counter of `name` by one.

classmethod **decrease**(*name*)
>   Decrease counter of `name` by one.

classmethod **count**(*name*)
>   Return the count of `name`

Example of using builtin `Counter` and `CounterMixin`

We don't assume you are stupid:

```
>> from mongu import enable_counter

>> Counter, CounterMixin = enable_counter()
```

Define a Model with CounterMixin:

```
>> @c.register_model
>> class User(CounterMixin, Model):  # order of base classes matters
>>     _database_  = 'test'
>>     _collection_ = 'users'
```

How to use builtin ``Counter`` and `CounterMixin`:

```
>> for name in ('Builtin', 'Counter', 'Test'):
>>     User(username=name).save()  # counter increases after creation
>> User.count()                    # provided by ``CounterMixin``
3
>> User.find_one().delete()        # counter decreases adter deletion
>> User.count()
2
```

Use Counter independently:

```
>> Counter.count('girlfriend')         # You born alone :|
0

>> Counter.increase('girlfriend')      # Before you find your first love :D
1
>> Counter.decrease('girlfriend')      # then you went through the very first break-up :(
0

>> Counter.change_by('girlfriend', 100)   # Oneday you had a crazy dream :P
100
>> Counter.change_by('girlfriend', -100)  # you woke up, everything turns to dust :(
0

>> Counter.count('girlfriend')         # Still dreaming? Check it again! 0_0
0
```

So sad, right?

Use **Mongu** to write your own story!

Indices and tables

- genindex
- modindex
- search

m