src

mongo_client package

Submodules

mongo_client.client module

class **mongo_client.client.MongoSessions**(**args*, ***kwargs*)

Базовые классы: `MongoConnectionInterface`

async **session_lock**(*session*, *period=None*)

Временная блокировка сессии

Используется для защиты сессии от частого использования, в случае временного бана со стороны источника или по другим причинам. После данного времени сессия станет активно использоваться обработчиками

Параметры:

- **session** (`Dict`) – dict имеющий _id сессии
- **period** (`Optional` [`Dict` [`Union` [`Literal` [`'days'` , `'seconds'` , `'microseconds'` , `'milliseconds'` , `'minutes'` , `'hours'` , `'weeks'`], `str`], `int` | `float`]]) – опциональное время блокировки, например {„hours“: 10}. По умолчанию - атрибут lock_time

Тип результата:

`UpdateResult`

async **session_block**(*session*, *period=None*)

Временная блокировка сессии продолжительного характера

Используется для защиты сессии от частого использования, в случае временного бана со стороны источника или по другим причинам. После данного времени сессия станет активно использоваться обработчиками

Параметры:

- **session** (`Dict`) – dict имеющий _id сессии
- **period** (`Optional` [`Dict` [`Union` [`Literal` [`'days'` , `'seconds'` , `'microseconds'` , `'milliseconds'` , `'minutes'` , `'hours'` , `'weeks'`], `str`], `int` | `float`]]) – опциональное время блокировки, например {„hours“: 10}. По умолчанию - атрибут block_time

Тип результата:
`UpdateResult`

async **count_active**()

Количество активных сессий

Данный метод считает количество записей в БД, соответствующих критерию default_filter (можно переопределить) и _filter_exclude_lock. По умолчанию, активная учетка - не имеет false в поле active, не имеет блокирующих интервалов в полях next_use (время больше текущего)

Тип результата:
`int`
Результат:
количество сессий

async **get_sessions**()

Получение всех сессий в БД

Учитываются только сессий по критерию default_filter (можно переопределить), по умолчанию - не имеет false в поле active. Дополнительно выбираются только поля сессии согласно проекции (например, только поля session, _id)

Тип результата:
`List` [`Dict`]
Результат:
список словарей (сессий)

async **get_session**(*count_use=1*, *next_use_delay=None*)

Получение сессии для обработчика на задачу

Обработчику для задач необходимы сессии. Учитывая, что обработчиков много и работают параллельно, то необходим механизм выбора сессий, чтобы одна и та же сессия не попала в одинаковый момент времени к разным обработчикам.

Для этого существует поле last_use (время последнего использования), которое автоматически обновляется в момент выборки. Выборка сессий происходит в порядке от самого старого использования к последнему. Также обновляется поле next_use, временно блокируется сессий (чаще всего не более чем на 3 сек), чтобы избежать выборки N раз за несколько секунд одинаковой сессии

Параметры:
count_use (`int`) – число, указать сколько раз будет использована сессия

в рамках данной выборки (для внутренней статистики) :type next_use_delay: `Optional` [`int`] :param next_use_delay: время блокировки, в сек, поле next_use :rtype: `Dict` :return: сессия

async **session_success**(*session*, *count_success=1*, *next_use_delay=None*)

Обновление внутренней статистики успешности

Параметры:

- **session** (`Dict`) – dict имеющий _id сессии
- **count_success** (`int`) – число, указать на сколько раз увеличить счетчик успешности использования
- **next_use_delay** (`Optional` [`int`]) – время блокировки, в сек, поле next_use

Тип результата:

`UpdateResult`

async **session_inactive**(*session*)

Постоянная блокировка сессии.

После данной действия сессия может быть активирована только активатором или руками в БД!

Параметры:

session (`Dict`) – dict имеющий _id сессии

Тип результата:

`UpdateResult`

async **add**(*data*)

Добавление сессии в БД

Параметры:

data (`Dict`) – произвольный dict

Тип результата:

`InsertOneResult`

async **session_update**(*session*, *payload*, *unset_payload=None*)

Обновление полей сессии в БД

Для обновления внутренних session, нужно передать `{'session.phone': '+79...'}`, а не `{'session': {'phone': '+79...'}}`

Параметры:

session (`Dict`) – dict имеющий _id сессии

Тип результата:

`UpdateResult`

async **session_delete**(*session*)

Удаление сессии в БД

Параметры:

session (`Dict`) – dict имеющий _id сессии

Тип результата:
`DeleteResult`

async **aggregate_statistics**()

Получение внутренней статистики для мониторинга

mongo_client.connect module

class **mongo_client.connect.MongoConnectInterface**(*mongo_url*, *db*, *collection*)

Базовые классы: `object`

async **connect**()

async **close**()

mongo_client.connection module

class **mongo_client.connection.MongoConnectionInterface**(**args*, ***kwargs*)

Базовые классы: `MongoOperations`, `MongoFields`

Класс коннектор к БД

async **connect**()

switch_collection(*collection*)

mongo_client.fields module

class **mongo_client.fields.MongoFields**

Базовые классы: `object`

Вспомогательный класс для хранения значений по умолчанию класса клиента Mongo

property **`next_use`**: *str*

Геттер на получение имени поля, отвечающего за timestamp блокировки сессии.

Результат:
имя поля

property **`active`**: *str*

Геттер на получение имени поля, отвечающего за полную блокировку сессии

Результат:
имя поля

property **`session`**: *str*

Геттер на получение имени поля, отвечающего за основные данные сессии

Результат:
имя поля

property **`last_use`**: *str*

Геттер на получение имени поля времени последнего использования сессии

Результат:
имя поля

property **`block_time`**: *timedelta*

Геттер времени блокировки сессии

Результат:
timedelta

property **`lock_time`**: *timedelta*

Геттер времени временной блокировки сессии

Результат:
timedelta

property **default_filter**: *Dict*

Получение фильтра по умолчанию для операций выборки данных :return: dict фильтра

property **projection**: *Dict | None*

Словарь для указания полей в выборке данных

Аналог из SQL - `select id, name from ... = {'id': 1, 'name': 1}` `select * from ... = {}`

Результат:
dict проекции

property **next_use_delay**: *int*

Значение в секундах времени, на которое блокируется сессия сразу же после выборки ее из БД.

Данное свойство необходимо для избежания ситуации, когда N обработчиков хотят параллельно брать одинаковую сессию. Сессия сразу же блокируется на M секунд. :return: время в сек

mongo_client.logger module

mongo_client.operations module

mongo_client.operations.timeout(*func*)

Декоратор для задания таймаута на операции, перехватывания ошибок и форматирования их в isphere-exceptions

class **mongo_client.operations.MongoOperations**(**args*, ***kwargs*)

Базовые классы: `MongoConnectInterface`

Класс-обертка над основными операциями с mongo.

Используется для задания декораторов и прямого подключения к mongo

async **count_documents**(***kwargs*)

async **update_one**(***kwargs*)

async **delete_one**(***kwargs*)

async **insert_one**(***kwargs*)

async **find_one_and_update**(***kwargs*)

async **index_information**(***kwargs*)

async **create_index**(***kwargs*)

async **drop_index**(***kwargs*)

async **find**(***kwargs*)

Module contents

setup module