MongoDB C Driver

# Issues

## bson\_t使用不当可能导致内存泄露

为了提高小文档性能，bson\_t拥有一个栈上的缓存（120字节定长数组），较大文档则使用堆内存，根据标志位决定使用的是哪一部分内存。

如果对bson\_new获得的对象调用bson\_init或bson\_copy\_to，则会导致标志位不正确处理，调用bson\_destroy后存在内存泄露。

mongoc\_collection\_find\_and\_modify函数，如果reply不是nullptr时，会使用bson\_init或bson\_copy\_to初始化reply。因此reply必须时指向栈上未初始化的bson\_t对象的指针，否则会导致内存泄露。

### Description

The bson\_t structure represents a BSON document. This structure manages the underlying BSON encoded buffer. For mutable documents, it can append new data to the document.

### Performance Notes

The bson\_t structure attempts to use an inline allocation within the structure to speed up performance of small documents. When this internal buffer has been exhausted, a heap allocated buffer will be dynamically allocated. Therefore, it is essential to call bson\_destroy() on allocated documents.

bson\_t尝试使用内部栈上的定长数组来加速小文档的性能。内部缓存用尽后，则使用动态分配的堆内存。因此，分配后的文档必须调用bson\_destroy。

### bson\_init()

void bson\_init (bson\_t \*b);

The bson\_init() function shall initialize a bson\_t that is placed on the stack. This is equivalent to initializing a bson\_t to BSON\_INITIALIZER.

初始化一个bson\_t。对象应该放置在栈上。

如果初始化由bson\_new产生的对象，则会导致内存泄露。

### bson\_copy\_to()

void bson\_copy\_to (const bson\_t\* src, bson\_t\* dst);

The bson\_copy\_to() function shall initialize dst with a copy of the contents of src.

dst MUST be an uninitialized bson\_t to avoid leaking memory.

复制bson\_t。目标bson必须是未初始化的bson\_t（即栈上的对象），否则会导致内存泄露。

建议bson\_copy\_to的dst使用完后调用bson\_destroy销毁。根据测试，mongo c driver 1.2.1中，dst如果指向栈上的对象，当src比较小(栈文档)时，不调用bson\_destroy(dst)不会有内存泄露；如果src比较大，不调用bson\_destroy(dst)会导致内存泄露。内存泄露和dst是都调用bson\_init初始化过无关。

# Tutorial

## [Starting MongoDB](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\tutorial.html#starting-mongod)

## [Making a Connection](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\tutorial.html#connecting)

The C Driver provides a convenient way to access MongoDB -- regardless of cluster configuration -- via a mongoc\_client\_t. It transparently connects to standalone servers, replica sets and sharded clusters on demand. Once a connection has been made, handles to databases and collections can be obtained via the structs mongoc\_database\_t and mongoc\_collection\_t, respectively. MongoDB operations can then be performed through these handles.

C驱动使用mongoc\_client\_t连接MongoDB，可透明链接Mongo集群。通过mongoc\_database\_t操作数据库，mongoc\_collection\_t操作数据集（表）

At the start of an application, call mongoc\_init() before any other libmongoc functions and call mongoc\_cleanup() before exiting. When creating handles to clients, databases and servers, call the appropriate destroy functions when finished.

程序启动时，在任何libmongoc函数之前调用mongoc\_init()，并且在退出之前调用mongoc\_cleanup()。句柄使用完成后，调用相应的函数销毁句柄。

The example below establishes a connection to a standalone server on localhost and performs a simple command. More information about database operations can be found in the CRUD Operations and Executing Commands sections. Examples of connecting to replica sets and sharded clusters can be found on the Advanced Connections page.

## [Creating BSON Documents](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\tutorial.html#creating-bson-documents)

Documents are stored in MongoDB's data format, BSON. The C driver uses libbson to create BSON documents. There are several ways to construct them: appending key-value pairs, using BCON, or parsing JSON.

文档（记录）使用BSON格式存储。C驱动使用libbson创建BSON文档。创建方式包括：添加key-value对，使用BCON，解析JSON。

### Appending BSON

A BSON document, represented as a bson\_t in code, can be constructed one field at a time using libbson's append functions.

常用函数：

bson\_new：创建bson\_t\*对象

bson\_append\_xxx：添加key-value对。xxx代表value类型，rubson\_append\_utf8

BSON\_APPEND\_UTF8：宏方式添加key-value对，简化函数。

bson\_append\_document\_begin/end：添加子文档

BSON\_APPEND\_DOCUMENT\_BEGIN/END：添加子文档宏

bson\_as\_json：转换为JSON字串

bson\_destory：销毁bson\_t\*对象

See the libbson documentation for all of the types that can be appended to a bson\_t.

### Using BCON

BSON C Object Notation, BCON for short, is an alternative way of constructing BSON documents in a manner closer to the intended format. It has less type-safety than BSON's append functions but results in less code.

BCON (BSON C Object Notation)使用更加自然的方式建立BSON。相对于append函数，他的类型安全性比较差，但是代码量较少。

Notice that BCON can create arrays, subdocuments and arbitrary fields.

BCON可以创建数组、子文档和任意字段。

### Creating BSON from JSON

For single documents, BSON can be created from JSON strings via bson\_new\_from\_json.

使用bson\_new\_from\_json从JSON字串创建一个BSON

To initialize BSON from a sequence of JSON documents, use bson\_json\_reader\_t.

使用bson\_json\_reader\_t从一系列JSON文档中初始化BSON。

## [Basic CRUD Operations](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\tutorial.html#crud-operations)

This section demonstrates the basics of using the C Driver to interact with MongoDB.

基本增删改查操作（Create, Retrieve, Update, Delete）

### Inserting a Document

To insert documents into a collection, first obtain a handle to a mongoc\_collection\_t via a mongoc\_client\_t. Then, use mongoc\_collection\_insert() to add BSON documents to the collection. This example inserts into the database "mydb" and collection "mycoll".

首先从mongoc\_client\_t 得到mongoc\_collection\_t句柄，然后mongoc\_collection\_insert添加BSON文档到collection。

When finished, ensure that allocated structures are freed by using their respective destroy functions.

完成后，确保销毁对应的结构体。

bool

mongoc\_collection\_insert

(

mongoc\_collection\_t \*collection,

mongoc\_insert\_flags\_t flags,

const bson\_t \*document,

const mongoc\_write\_concern\_t \*write\_concern,

bson\_error\_t \*error

);

flags:

MONGOC\_INSERT\_NONE

Specify no insert flags.

MONGOC\_INSERT\_CONTINUE\_ON\_ERROR

Continue inserting documents from the insertion set even if one insert fails.

MONGOC\_INSERT\_NO\_VALIDATE

Do not validate insertion documents before performing an insert. Validation can be expensive, so this can save some time if you know your documents are already valid.

write\_concern:

what level of acknowledgment to await from the server. Can be NULL

If no \_id element is found in document, then a bson\_oid\_t will be generated locally and added to the document.

关于ObjectID:

可以使用bson\_oid\_init初始化objectid (\_id)，也可以由mongo自动生成。ObjectID是一个12字节BSON类型，组成为：

4字节时间（UNIX epoch开始的秒数）

3字节机器ID

2字节处理ID

3字节计数器（从某个随机数字开始）

获取ObjectID的时间:

time\_t bson\_oid\_get\_time\_t (const bson\_oid\_t\* oid);

### Finding a Document

To query a MongoDB collection with the C driver, use the function mongoc\_collection\_find(). This returns a cursor to the matching documents. The following examples iterate through the result cursors and print the matches to stdout as JSON strings.

使用mongoc\_collection\_find返回匹配文档的游标，使用mongoc\_cursor\_next通过游标获取文档。

Note that mongoc\_collection\_find uses a document as a query specifier; for example,

{ "color" : "red" }

will match any document with a field named "color" with value "red". An empty document {} can be used to match all documents.

mongoc\_collection\_find使用文档作为查询条件。空文档{}匹配所有文档。

This first example uses an empty query specifier to find all documents in the database "mydb" and collection "mycoll".

To look for a specific document, add a specifier to query. This example adds a call to BSON\_APPEND\_UTF8() to look for all documents matching {"hello" : "world"}.

查询特定文档时，添加限定条件到query中。

mongoc\_cursor\_t\*

mongoc\_collection\_find

(

mongoc\_collection\_t \*collection,

mongoc\_query\_flags\_t flags,

uint32\_t skip, // number of documents to skip or 0

uint32\_t limit, // max number of documents to return or 0.

uint32\_t batch\_size,

const bson\_t \*query,

const bson\_t \*fields,

const mongoc\_read\_prefs\_t \*read\_prefs

)

**flags:**

MONGOC\_QUERY\_NONE

Specify no query flags.

MONGOC\_QUERY\_TAILABLE\_CURSOR

Cursor will not be closed when the last data is retrieved. You can resume this cursor later.

MONGOC\_QUERY\_SLAVE\_OK

Allow query of replica set secondaries.

MONGOC\_QUERY\_OPLOG\_REPLAY

Used internally by MongoDB.

MONGOC\_QUERY\_NO\_CURSOR\_TIMEOUT

The server normally times out an idle cursor after an inactivity period (10 minutes). This prevents that.

MONGOC\_QUERY\_AWAIT\_DATA

Use with MONGOC\_QUERY\_TAILABLE\_CURSOR. Block rather than returning no data. After a period, time out.

MONGOC\_QUERY\_EXHAUST

Stream the data down full blast in multiple "reply" packets. Faster when you are pulling down a lot of data and you know you want to retrieve it all.

MONGOC\_QUERY\_PARTIAL

Get partial results from mongos if some shards are down (instead of throwing an error).

**batch\_size:**

batch size of document result sets or 0 for default. Default is 100.

**fields:**

fields to return or NULL for default (all field)

如果指定了字段名，则未指定字段默认为不显示。\_id默认为显示。例如：

{“fieldname1”: 0, “fieldname2”, 1}

**read\_prefs:**

hinting to the driver which nodes in a replica set should be accessed first. NULL for default.

This function shall execute a query on the underlying collection.

If no options are necessary, query can simply contain a query such as {a:1}. If you would like to specify options such as a sort order, the query must be placed inside of {"$query": {}} as specified by the server documentation. See the example below for how to properly specify additional options to query.

// all

query = bson\_new();

// name >= object\_100 && name < object\_200 && group = 3, order by tag desc

query = BCON\_NEW("$query", "{", "name", "{", "$gte", "object\_100", "$lt", "object\_200", "}",

"group", BCON\_INT32(3), "}",

"$orderby", "{", "tag", BCON\_INT32(-1), "}");

// tag >= 90 || tag < 10 || group = 3

query = BCON\_NEW("$query", "{", "$or", "[",

"{", "tag", "{", "$gte", "90", "}", "}",

"{", "tag", "{", "$lt", "10", "}", "}",

"{", "group", BCON\_INT32(3), "}",

"]", "}");

// tag >= 2 && tag < 4

query = BCON\_NEW("tag", "{", "$gte", BCON\_INT32(2), "}", "tag", "{", "$lte", BCON\_INT32(4), "}");

// group >= 2 && group < 4 && tag = 10

char\* pJson = " { \"group\" : { \"$gte\" : 2, \"$lt\" : 4}, \"tag\" : 10 }";

query = bson\_new\_from\_json((uint8\_t\*)pJson, -1, nullptr);

### Updating a Document

This code snippet gives an example of using mongoc\_collection\_update() to update the fields of a document.

Using the "mydb" database, the following example inserts an example document into the "mycoll" collection. Then, using its \_id field, the document is updated with different values and a new field.

bool

mongoc\_collection\_update(

mongoc\_collection\_t \*collection,

mongoc\_update\_flags\_t flags,

const bson\_t \*selector,

const bson\_t \*update,

const mongoc\_write\_concern\_t \*write\_concern,

bson\_error\_t \*error);

**flags:**

MONGOC\_UPDATE\_NONE

No update flags set.

MONGOC\_UPDATE\_UPSERT

If an upsert should be performed.

MONGOC\_UPDATE\_MULTI\_UPDATE

If more than a single matching document should be updated. By default only the first document is updated.

MONGOC\_UPDATE\_NO\_VALIDATE

Do not perform client side BSON validations when performing an update. This is useful if you already know your BSON documents are valid.

bool

mongoc\_collection\_find\_and\_modify(

mongoc\_collection\_t \*collection,

const bson\_t \*query,

const bson\_t \*sort, // order for query

const bson\_t \*update,

const bson\_t \*fields, // fields to return

bool \_remove, // remove and update can not co-exist.

bool upsert, // update or insert if not exist

bool \_new, // return new version of documents

bson\_t \*reply, // operation result

bson\_error\_t \*error); // can be NULL

### Deleting a Document

bool

mongoc\_collection\_remove (

mongoc\_collection\_t \*collection,

mongoc\_remove\_flags\_t flags,

const bson\_t \*selector,

const mongoc\_write\_concern\_t \*write\_concern,

bson\_error\_t \*error);

**flags:**

MONGOC\_REMOVE\_NONE

Specify no removal flags. All matching documents will be removed.

MONGOC\_REMOVE\_SINGLE\_REMOVE

Only remove the first matching document from the selector.

### Counting Documents

Counting the number of documents in a MongoDB collection is similar to performing a find operation.

int64\_t

mongoc\_collection\_count (

mongoc\_collection\_t \*collection,

mongoc\_query\_flags\_t flags,

const bson\_t \*query,

int64\_t skip,

int64\_t limit,

const mongoc\_read\_prefs\_t \*read\_prefs,

bson\_error\_t \*error);

**flags**

MONGOC\_QUERY\_NONE

Specify no query flags.

MONGOC\_QUERY\_TAILABLE\_CURSOR

Cursor will not be closed when the last data is retrieved. You can resume this cursor later.

MONGOC\_QUERY\_SLAVE\_OK

Allow query of replica set secondaries.

MONGOC\_QUERY\_OPLOG\_REPLAY

Used internally by MongoDB.

MONGOC\_QUERY\_NO\_CURSOR\_TIMEOUT

The server normally times out an idle cursor after an inactivity period (10 minutes). This prevents that.

MONGOC\_QUERY\_AWAIT\_DATA

Use with MONGOC\_QUERY\_TAILABLE\_CURSOR. Block rather than returning no data. After a period, time out.

MONGOC\_QUERY\_EXHAUST

Stream the data down full blast in multiple "reply" packets. Faster when you are pulling down a lot of data and you know you want to retrieve it all.

MONGOC\_QUERY\_PARTIAL

Get partial results from mongos if some shards are down (instead of throwing an error).

### Creating Index

bool

mongoc\_collection\_create\_index (

mongoc\_collection\_t \*collection,

const bson\_t \*keys,

const mongoc\_index\_opt\_t \*opt,

bson\_error\_t \*error);

void

mongoc\_index\_opt\_init (mongoc\_index\_opt\_t \*opt);

typedef struct

{

bool is\_initialized;

bool background;

bool unique;

const char \*name;

bool drop\_dups;

bool sparse;

int32\_t expire\_after\_seconds;

int32\_t v;

const bson\_t \*weights;

const char \*default\_language;

const char \*language\_override;

mongoc\_index\_opt\_geo\_t \*geo\_options;

mongoc\_index\_opt\_storage\_t \*storage\_options;

void \*padding[6];

} mongoc\_index\_opt\_t;

### FindAndModify

bool

mongoc\_collection\_find\_and\_modify (

mongoc\_collection\_t \*collection,

const bson\_t \*query,

const bson\_t \*sort,

const bson\_t \*update,

const bson\_t \*fields,

bool \_remove,

bool upsert,

bool \_new,

bson\_t \*reply,

bson\_error\_t \*error);

执行删除、修改、插入修改操作。可以通过reply获取到执行结果的详细信息，包括是否有记录被修改或删除。

其中reply必须指向未初始化的bson\_t并且使用后调用bson\_destroy销毁，否则会导致内存泄露。

#### Description

Update and return an object.

This is a thin wrapper around the findAndModify command. Either update or \_remove arguments are required.

#### Errors

Errors are propagated via the error parameter.

#### Returns

Returns either the document before or after modification based on the \_new parameter.

## [Executing Commands](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\tutorial.html#executing-commands)

The driver provides helper functions for executing MongoDB commands on client, database and collection structures. These functions return cursors; the \_simple variants return booleans indicating success or failure.

mongoc\_client\_command()

mongoc\_client\_command\_simple()

mongoc\_collection\_command()

mongoc\_collection\_command\_simple()

mongoc\_database\_command()

mongoc\_database\_command\_simple()

## [Threading](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\tutorial.html#threading)

The MongoDB C Driver is thread-unaware in the vast majority of its operations. This means it is up to the programmer to guarantee thread-safety.

大部分C驱动函数未考虑线程问题。程序员需要确保线程安全性。

However, mongoc\_client\_pool\_t is thread-safe and is used to fetch a mongoc\_client\_t in a thread-safe manner. After retrieving a client from the pool, the client structure should be considered owned by the calling thread. When the thread is finished, the client should be placed back into the pool.

mongoc\_client\_pool\_t可以以线程安全的方式获取客户端连接句柄。当从线程池获取到客户端句柄后，该句柄由调用线程拥有，使用完毕后，将句柄放回线程池。

mongoc\_client\_pool\_pop以block方式获取client句柄，mongoc\_client\_pool\_try\_pop是非block方式，无可用句柄则返回NULL。

mongoc\_uri\_t \*

mongoc\_uri\_new (const char \*uri\_string)

mongoc\_client\_pool\_t \*

mongoc\_client\_pool\_new (const mongoc\_uri\_t \*uri);

mongoc\_client\_t \*

mongoc\_client\_pool\_pop (mongoc\_client\_pool\_t \*pool);

mongoc\_client\_t \*

mongoc\_client\_pool\_try\_pop (mongoc\_client\_pool\_t \*pool);

void

mongoc\_client\_pool\_push (mongoc\_client\_pool\_t \*pool,

mongoc\_client\_t \*client);

void

mongoc\_client\_pool\_destroy (mongoc\_client\_pool\_t \*pool);

void

mongoc\_uri\_destroy (mongoc\_uri\_t \*uri);

## mongoc\_uri\_t

mongoc\_uri\_t provides an abstraction on top of the MongoDB connection URI format. It provides standardized parsing as well as convenience methods for extracting useful information such as replica hosts or authorization information.

See Connection String URI Reference on the MongoDB website for more information.

### Format

mongodb:// <1>

[username:password@] <2>

host1 <3>

[:port1] <4>

[,host2[:port2],...[,hostN[:portN]]] <5>

[/[database] <6>

[?options]] <7>

<1>mongodb is the specifier of the MongoDB protocol.

<2>An optional username and password.

<3>The only required part of the uri. This specifies either a hostname, IP address or UNIX domain socket.

<4>An optional port number. Defaults to :27017.

<5>Extra optional hosts and ports. You would specify multiple hosts, for example, for connections to replica sets.

<6>The name of the database to authenticate if the connection string includes authentication credentials. If /database is not specified and the connection string includes credentials, defaults to the 'admin' database.

<7>Connection specific options. Combined with &

### Replica Set Example

To describe a connection to a replica set named 'test' with the following mongod hosts:

db1.example.com on port 27017

db2.example.com on port 2500

You would use the connection string that resembles the following.

mongodb://db1.example.com,db2.example.com:2500/?replicaSet=test

### Connection Options

ssl

{true|false}, indicating if SSL must be used. (See also mongoc\_client\_set\_ssl\_opts and mongoc\_client\_pool\_set\_ssl\_opts.)

connectTimeoutMS

A timeout in milliseconds to attempt a connection before timing out. This setting applies to server discovery and monitoring connections as well as to connections for application operations. The default is 10 seconds.

socketTimeoutMS

The time in milliseconds to attempt to send or receive on a socket before the attempt times out. The default is 5 minutes.

Setting any of the \*TimeoutMS options above to 0 will be interpreted as "use the default value"

### Server Discovery, Monitoring, and Selection Options

Clients in a mongoc\_client\_pool\_t share a topology scanner that runs on a background thread. The thread wakes every heartbeatFrequencyMS (default 10 seconds) to scan all MongoDB servers in parallel. Whenever an application operation requires a server that is not known--for example, if there is no known primary and your application attempts an insert--the thread rescans all servers every half-second. In this situation the pooled client waits up to serverSelectionTimeoutMS (default 30 seconds) for the thread to find a server suitable for the operation, then returns an error with domain MONGOC\_ERROR\_SERVER\_SELECTION.

连接池（mongoc\_client\_pool\_t）中的客户端连接共享使用一个后台运行的拓扑扫描器线程。该线程每隔heartbeatFrequencyMS（默认10秒）并发地扫描所有MongoDB服务器。任何时候后，如果程序的操作需要一个未知的服务器（例如，当程序尝试插入时没有已知的主服务器），该线程每半秒重新扫描所有服务器。这种情况下，该线程需要等待最长serverSelectionTimeoutMS（默认30秒）找到操作所需的服务器，或返回域错误MONGOC\_ERROR\_SERVER\_SELECTION。

Technically, the total time an operation may wait while a pooled client scans the topology is controlled both by serverSelectionTimeoutMS and connectTimeoutMS. The longest wait occurs if the last scan begins just at the end of the selection timeout, and a slow or down server requires the full connection timeout before the client gives up.

理论上讲操作所需等待的时间（即连接池中的客户端扫描拓扑结构的时间）由serverSelectionTimeoutMS和connectTimeoutMS决定。selection超时之后开始最后一次扫描时等待时间最长，客户端遇到较慢的或宕机的服务器需要最长的连接超时时间才能返回。

A non-pooled client is single-threaded. Every heartbeatFrequencyMS, it blocks the next application operation while it does a parallel scan. This scan takes as long as needed to check the slowest server: roughly connectTimeoutMS. Therefore the default heartbeatFrequencyMS for single-threaded clients is greater than for pooled clients: 60 seconds.

非连接池客户端使用单独的线程。每隔heartbeatFrequencyMS该线程阻止下一个程序操作来进行并行扫描。该扫描检查最慢的服务器时可能使用最长的超时，可以简单认为就是connectTimeoutMS。因此单一线程客户端heartbeatFrequencyMS的默认值（60秒）要比线程池的客户端时间长。

By default, single-threaded (non-pooled) clients scan only once when an operation requires a server that is not known. If you attempt an insert and there is no known primary, the client checks all servers once trying to find it, then succeeds or returns an error with domain MONGOC\_ERROR\_SERVER\_SELECTION. But if you set serverSelectionTryOnce to "false", the single-threaded client loops, checking all servers every half-second, until serverSelectionTimeoutMS.

默认情况下，单一线程客户端进扫描一次未知服务器。如果设置serverSelectionTryOnce为false，则单一线程客户端每隔半秒扫描一次服务器，直到serverSelectionTimeoutMS超时。

The total time an operation may wait for a single-threaded client to scan the topology is determined by connectTimeoutMS in the try-once case, or serverSelectionTimeoutMS and connectTimeoutMS if serverSelectionTryOnce is set "false".

对单一线程客户端，仅尝试一次时总超时时间由connectTimeoutMS决定，否则由serverSelectionTimeoutMS和connectTimeoutMS决定。

heartbeatFrequencyMS

The interval between server monitoring checks. Defaults to 10 seconds in pooled (multi-threaded) mode, 60 seconds in non-pooled mode (single-threaded).

serverSelectionTimeoutMS

A timeout in milliseconds to block for server selection before throwing an exception. The default is 30 seconds.

serverSelectionTryOnce

If "true", the driver scans the topology exactly once after server selection fails, then either selects a server or returns an error. If it is false, then the driver repeatedly searches for a suitable server for up to serverSelectionTimeoutMS milliseconds (pausing a half second between attempts). The default for serverSelectionTryOnce is "false" for pooled clients, otherwise "true".

Pooled clients ignore serverSelectionTryOnce; they signal the thread to rescan the topology every half-second until serverSelectionTimeoutMS expires.

socketCheckIntervalMS

Only applies to single threaded clients. If a socket has not been used within this time, its connection is checked with a quick "isMaster" call before it is used again. Defaults to 5 seconds.

Setting any of the \*TimeoutMS options above to 0 will be interpreted as "use the default value"

### 代码分析

以CreateIndex为例。mongoc\_collection\_create\_index的调用堆栈为：

\_mongoc\_topology\_run\_scanner()

\_mongoc\_topology\_do\_blocking\_scan()

mongoc\_topology\_select() // 判断是线程池还是单一线程客户端

mongoc\_cluster\_select\_by\_optype()

\_mongoc\_cursor\_query()

\_mongoc\_cursor\_next()

mongoc\_cursor\_next()

\_mongoc\_client\_command\_simple\_with\_hint()

mongoc\_client\_command\_simple()

mongoc\_collection\_command\_simple()

mongoc\_collection\_create\_index()

mongoc\_topology\_select时判断是线程池还是单一线程客户端，然后判断是否需要扫描并决定扫描策略。单一线程客户端调用，线程池则是未找到可用客户端时调用\_mongoc\_topology\_request\_scan然后mongoc\_cond\_timedwait等待超时。

### Connection Pool Options

maxPoolSize

The maximum number of connections in the pool. The default value is 100.

minPoolSize

The minimum number of connections in the connection pool. Default value is 0. These are lazily created.

maxIdleTimeMS

Not implemented.

waitQueueMultiple

Not implemented.

waitQueueTimeoutMS

Not implemented.

### Write Concern Options

w

value: 0, 1, majority, n, tags

0

The driver will not acknowledge write operations but will pass or handle any network and socket errors that it receives to the client. If you disable write concern but enable the getLastError command’s w option, w overrides the w option.

1

Provides basic acknowledgment of write operations. By specifying 1, you require that a standalone mongod instance, or the primary for replica sets, acknowledge all write operations. For drivers released after the default write concern change, this is the default write concern setting.

majority

For replica sets, if you specify the special majority value to w option, write operations will only return successfully after a majority of the configured replica set members have acknowledged the write operation.

n

For replica sets, if you specify a number n greater than 1, operations with this write concern return only after n members of the set have acknowledged the write. If you set n to a number that is greater than the number of available set members or members that hold data, MongoDB will wait, potentially indefinitely, for these members to become available.

tags

For replica sets, you can specify a tag set to require that all members of the set that have these tags configured return confirmation of the write operation.

wtimeoutMS

The time in milliseconds to wait for replication to succeed, as specified in the w option, before timing out. When wtimeoutMS is 0, write operations will never time out.

journal

values: true, false

Controls whether write operations will wait until the mongod acknowledges the write operations and commits the data to the on disk journal.

true

Enables journal commit acknowledgment write concern. Equivalent to specifying the getLastError command with the j option enabled.

false

Does not require that mongod commit write operations to the journal before acknowledging the write operation. This is the default option for the journal parameter.

### Read Preference Options

readPreference

Specifies the replica set read preference for this connection. This setting overrides any slaveOk value. The read preference values are the following:

primary

primaryPreferred

secondary

secondaryPreferred

nearest

readPreferenceTags

Specifies a tag set as a comma-seperated list of colon-separted key-value pairs.

### Functions

mongoc\_uri\_copy()

mongoc\_uri\_destroy()

mongoc\_uri\_get\_auth\_mechanism()

mongoc\_uri\_get\_auth\_source()

mongoc\_uri\_get\_database()

mongoc\_uri\_get\_hosts()

mongoc\_uri\_get\_options()

mongoc\_uri\_get\_password()

mongoc\_uri\_get\_read\_prefs()mongoc\_uri\_get\_read\_prefs\_t()

mongoc\_uri\_get\_replica\_set()

mongoc\_uri\_get\_ssl()

mongoc\_uri\_get\_string()

mongoc\_uri\_get\_username()

mongoc\_uri\_get\_write\_concern()

mongoc\_uri\_new()

mongoc\_uri\_new\_for\_host\_port()

mongoc\_uri\_unescape()

## [Next Steps](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\tutorial.html#next-steps)

To find information on advanced topics, browse the rest of the C driver guide or the official MongoDB documentation.

# Advanced Connections

## [Additional Connection Options](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\advanced-connections.html#additional-options)

## [Connecting to a Replica Set](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\advanced-connections.html#replica-set)

## [Connecting to a Sharded Cluster](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\advanced-connections.html#sharded-cluster)

## [Connecting to a UNIX Domain Socket](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\advanced-connections.html#unix-socket)

## [Connecting to an IPv6 Address](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\advanced-connections.html#ipv6)

# Authentication

## [Basic Authentication](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\authentication.html#basic-auth)

## [GSSAPI (Kerberos) Authentication](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\authentication.html#kerberos)

## [SASL Plain Authentication](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\authentication.html#sasl-plain)

## [SSL Authentication](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\authentication.html#ssl)

## [X.509 Certificate Authentication](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\authentication.html#x509)

# Cursors

## [Destroying Server-Side Cursors](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\cursors.html#killing-cursors)

## [Handling Cursor Failures](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\cursors.html#cursor-errors)

## [Tailable Cursors](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\cursors.html#tailable)

# Bulk Operations

## [Bulk Insert](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\bulk.html#bulk-insert)

mongoc\_collection\_create\_bulk\_operation()

for

mongoc\_bulk\_operation\_insert()

mongoc\_bulk\_operation\_execute()

## [Bulk Operation Write Concerns](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\bulk.html#bulk-write-concern)

## [Further Reading](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\bulk.html#bulk-spec)

## [Mixed Bulk Write Operations](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\bulk.html#mixed-bulk-write)

## [Ordered Bulk Write Operations](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\bulk.html#ordered-bulk-write)

## [Unordered Bulk Write Operations](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\bulk.html#unordered-bulk-write)

# Aggregation Framework

## [Aggregation Framework Examples](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\aggregate.html)

# Client Side Document Matching

## [Basic Document Matching](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\matcher.html#basic-matching)

The MongoDB C driver supports matching a subset of the MongoDB query specification on the client.

Currently, basic numeric, string, subdocument, and array equality, $gt, $gte, $lt, $lte, $in, $nin, $ne, $exists, $type, $and, and $or are supported. As this is not the same implementation as the MongoDB server, some inconsistencies may occur. Please file a bug if you find such a case.

# Troubleshooting

## [Basic Troubleshooting](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\basic-troubleshooting.html#basic-troubleshooting)

## [Performance Counters](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\basic-troubleshooting.html#perf-counters)

## [Submitting a Bug Report](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\basic-troubleshooting.html#file-bug)

# API Reference

## [Logging](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\logging.html)

## [Version Checks](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_version.html)

## [mongoc\_bulk\_operation\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_bulk_operation_t.html)

## [mongoc\_client\_pool\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_client_pool_t.html)

## [mongoc\_client\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_client_t.html)

## [mongoc\_collection\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_collection_t.html)

## [mongoc\_cursor\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_cursor_t.html)

## [mongoc\_database\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_database_t.html)

## [mongoc\_delete\_flags\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_delete_flags_t.html)

## [mongoc\_gridfs\_file\_list\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_gridfs_file_list_t.html)

## [mongoc\_gridfs\_file\_opt\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_gridfs_file_opt_t.html)

## [mongoc\_gridfs\_file\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_gridfs_file_t.html)

## [mongoc\_gridfs\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_gridfs_t.html)

## [mongoc\_index\_opt\_geo\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_index_opt_geo_t.html)

## [mongoc\_index\_opt\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_index_opt_t.html)

## [mongoc\_index\_opt\_wt\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_index_opt_wt_t.html)

## [mongoc\_insert\_flags\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_insert_flags_t.html)

## [mongoc\_iovec\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_iovec_t.html)

## [mongoc\_matcher\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_matcher_t.html)

## [mongoc\_query\_flags\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_query_flags_t.html)

## [mongoc\_rand](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_rand.html)

## [mongoc\_read\_mode\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_read_mode_t.html)

## [mongoc\_read\_prefs\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_read_prefs_t.html)

## [mongoc\_remove\_flags\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_remove_flags_t.html)

## [mongoc\_reply\_flags\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_reply_flags_t.html)

## [mongoc\_server\_description\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_server_description_t.html)

## [mongoc\_socket\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_socket_t.html)

## [mongoc\_ssl\_opt\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_ssl_opt_t.html)

## [mongoc\_stream\_buffered\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_stream_buffered_t.html)

## [mongoc\_stream\_file\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_stream_file_t.html)

## [mongoc\_stream\_gridfs\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_stream_gridfs_t.html)

## [mongoc\_stream\_socket\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_stream_socket_t.html)

## [mongoc\_stream\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_stream_t.html)

## [mongoc\_stream\_tls\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_stream_tls_t.html)

## [mongoc\_update\_flags\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_update_flags_t.html)

## [mongoc\_uri\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_uri_t.html)

## [mongoc\_write\_concern\_t](file:///C:\lib\mongo-c-driver-1.2.1\doc\html\mongoc_write_concern_t.html)