
Perl 6 MongoDB driver

Marcel Timmerman <mt1957@gmail.com>

Copyright © 2015, 2016 ... Inf Marcel Timmerman

Abstract

MongoDB is a *Non SQL* database which uses *Binary JSON (BSON)* to store and load information in a database. With the `mongodb` package a shell program called `mongo` is available to give instructions to a `mongodb` server.

To work with data on the server from within a program a driver is needed. There are drivers for many program languages. This document describes a driver for the Perl6 language. In the perl6 ecosystem, which might grow into a cpan like system later, there are two packages needed to work with the driver. These are *MongoDB* and *BSON*. *BSON* is automatically installed with other necessary modules.

The latest version of this document is generated on date 2017-02-08

Table of Contents

| | |
|--|---|
| 1. Introduction | 1 |
| 2. Implementation | 2 |
| 2.1. Server states | 2 |
| 2.2. Topology | 3 |
| 2.3. Round Trip Time | 3 |
| 2.4. Read concern | 3 |
| 2.5. Write concern | 3 |
| 2.6. URI | 3 |
| 3. Classes in the driver | 4 |
| 3.1. MongoDB::Client | 4 |
| 3.2. BSON::Document | 4 |
| 3.3. MongoDB::Database | 4 |
| 3.4. MongoDB::Collection | 4 |
| 3.5. MongoDB::Cursor | 4 |
| 3.6. MongoDB::Server | 4 |
| 3.7. MongoDB::Server::Control | 4 |
| 4. BSON | 4 |
| 5. Examples | 5 |
| 5.1. Starting and stopping a server using the configuration | 5 |
| 5.2. Making a replica server | 5 |
| 6. References to books, websites, articles and pod-documents | 5 |
| 6.1. Web Pages | 5 |
| MongoDB Driver Glossary and References | 5 |
| Index | 6 |

1. Introduction

The purpose of this document is to show how things are accomplished in this driver in the light of the MongoDB developer documents and how to work with the perl6 mongodb driver.

However, this document will not tell you how to design your database among other things. There are plenty of good books and documents out there, not to mention, the mongodb website.

There are quite a few modules written to perform the tasks at hand but not all modules will be explained here because many of them are modules defining classes to be used in the background and are not used by applications directly.

Furthermore, this document is not a reference. There are other documents for that, written to document the attributes, (sub)methods and subs in a class. There will be a list of references at the end of the document.

This document assumes that the reader is aware of at least the basics of the mongodb database and what one can do with it. Also some perl 6 knowledge will be necessary.

As a last remark, the driver is still in development. Although many parts are accomplished, some parts still need to be implemented like authentication against kerberos or LDAP. Furthermore, there are some improvements needed to speedup the operations.

The following sections will be explained:

- *Implementation.*
 - *Server states.*
 - *Topology.*
 - *Round trip time.*
 - *Read concern.*
 - *Write concern.*
 - *URI.* The URI tells the software how to connect to a server and select the proper server.
 - *Server selection process.*
 - *MongoDB::Client.* This module is the starting point of all applications which need access to a mongodb database server.
 - *BSON::Document.* This is the basic vehicle to insert, update retrieve and send commands to the database server. In this section there is an explanation of the supported types as well as different ways to make requests. Some detailed perl6 is necessary to understand mistakes often made when creating the data structures.
 - *MongoDB::Database.*
 - *MongoDB::Collection.*
 - *MongoDB::Cursor.*
 - *MongoDB::Server.*
 - *MongoDB::Server::Control.*
- *Dependencies.* There are some dependencies which are explained a bit here. These are e.g. the server and its version, modules like BSON, PKCS5, Auth::SCRAM etcetera.
- *Examples.* Of course, a document without examples is a bit like an empty box as a present.

2. Implementation

2.1. Server states

Table 1. Server states depending on isMaster outcome

| Server state | isMaster command result |
|--------------------|---|
| SS-Unknown | Initial, or after a network error or failed ismaster call, or "ok: 1" not in ismaster response. |
| SS-Standalone | No "msg: isdbgrid", no setName, and no "isreplaset: true". |
| SS-Mongos | "msg: isdbgrid" |
| SS-PossiblePrimary | Not yet checked, but another member thinks it is the primary. |
| SS-RSPrimary | "ismaster: true", "setName" in response. |
| SS-RSSecondary | "secondary: true", "setName" in response. |

| Server state | isMaster command result |
|--------------|---|
| SS-RSArbiter | "arbiterOnly: true", "setName" in response. |
| SS-RSOther | "setName" in response, "hidden: true" or not primary, secondary, nor arbiter. E.g. starting up or recovering. |
| SS-RSGhost | "isreplicaset: true" in response. E.g. briefly during server startup, in an uninitialized replica set, or when the server is shunned (removed from the replica set config). |

2.2. Topology

Table 2. Topology controlled by server states

| Topology type | Server states |
|---------------------------|--|
| TT-Unknown | When a deployment has this topology type, no servers are suitable for read or write operations. These are servers which did not respond on initial connection or threw an exception because of e.g. a DNS lookup failure. All server states of these servers herded by the Client object is SS-Unknown. |
| TT-Single | A deployment of topology type TT-Single contains only a single server with a state SS-Standalone. This topology type signifies a direct connection intended to receive all read and write operations. The explanation here differs from that of the mongodb site because of the way it tests for the server states. In the end, when servers are selected and only one suitable server is found, no further selection processing is needed which is intended by the definition of mongodb. |
| TT-Sharded | A deployment of topology type TT-Sharded contains one or more servers of type SS-Mongos or SS-Unknown of at least one is SS-Mongos. |
| TT-ReplicaSet-NoPrimary | A deployment with this topology type can have a mix of server types: SS-RSSecondary, SS-RSArbiter, SS-RSOther, SS-RSGhost, SS-Unknown or SS-PossiblePrimary. |
| TT-ReplicaSet-WithPrimary | A deployment with this topology type can have a mix of server types: SS-RSPRimary, SS-RSSecondary, SS-RSArbiter, SS-RSOther, SS-RSGhost, SS-Unknown or SS-PossiblePrimary. |

2.3. Round Trip Time

2.4. Read concern

2.5. Write concern

2.6. URI

Table 3. Implemented uri connection options

| Option | Description |
|------------|--|
| replicaSet | Specifies the name of the replica set, if the mongod is a member of a replica set. When connecting to a replica set it is important to give a seed list of at least two mongod instances. If you only provide the connection point of a single mongod instance, and omit the replicaSet, the client will create a standalone connection. |

3. Classes in the driver

3.1. MongoDB::Client

3.1.1. Making a connection

3.1.2. Server selection

- Record the server selection start time
- If the topology wire version is invalid, raise an error
- Find suitable servers by topology type and operation type
- If there are any suitable servers, choose one at random from those within the latency window and return it; otherwise, continue to step
- Request an immediate topology check, then block the server selection thread until the topology changes or until the server selection timeout has elapsed
- If more than serverSelectionTimeoutMS milliseconds have elapsed since the selection start time, raise a server selection error
- Goto Step

3.2. BSON::Document

3.3. MongoDB::Database

3.3.1. run-command()

3.4. MongoDB::Collection

3.4.1. find()

3.5. MongoDB::Cursor

3.5.1. fetch()

3.5.2. iterating over documents

3.6. MongoDB::Server

3.7. MongoDB::Server::Control

4. BSON

.1. Supported types

Table 4. Supported types of the BSON package

| BSON | Perl6 | Description |
|--------|-------|--|
| Double | Num | An eight byte floating point number. The perl6 type choosen is a 'Num' which stores a floating-point number. On most platforms, it's an IEEE 754 64-bit floating point |

| BSON | Perl6 | Description |
|------|-------|--|
| | | number, aka "double precision". The 'Rat' is not choosen because it can not be converted back the way it was thereby loosing accuracy. |

.1. MongoDB servers

5. Examples

5.1. Starting and stopping a server using the configuration

This method, using a configuration file, is also used to test the modules to help starting and stopping a locally installed server. There are several steps in order to configure it properly.

- *Configuration file.*
- *Server selection.*
- *Starting and stopping.*

5.1.1. Configuration file

5.1.2. Server selection

5.1.3. Starting and stopping

5.2. Making a replica server

5.2.1. Preparing

5.2.2. Initializing

6. References to books, websites, articles and pod-documents

6.1. Web Pages

MongoDB Manual covering all aspects of what is possible. Source is from MongoDB, Inc. EPub edition [http://docs.mongodb.com/master/MongoDB-manual.epub]

MongoDB Driver Glossary and References

B

Binary JSON

BSON is a computer data interchange format used mainly as a data storage and network transfer format in the MongoDB database. See also on Wikipedia [https://nl.wikipedia.org/wiki/BSON].

J

JavaScript Object Notation

JavaScript Object Notation) is an open-standard format that uses human-readable text to transmit data objects consisting of attribute-value pairs. See also on Wikipedia [<https://nl.wikipedia.org/wiki/JSON>].

M

MongoDB

MongoDB (from humongous) is a free and open-source cross-platform document-oriented database program.

N

Non SQL

A NoSQL (originally referring to "non *Structured Query Language*", "non relational" or "not only SQL" database provides a mechanism for storage and retrieval of data which is modeled in means other than the tabular relations used in relational databases.

S

Structured Query Language

SQL or Structured Query Language is a special-purpose domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS)

Index