

MongoDB Mongo Shell

What is MongoDB Mongo shell?

MongoDB Mongo shell is an interactive JavaScript interface that allows interaction with MongoDB instances through the command line. The shell can be used for data manipulation and as well as administrative operations such as maintenance and ongoing operations of databases instances.

MongoDB Mongo shell features

MongoDB Mongo shell is the default manual client for MongoDB. It's a command line interface which means the input and output are all console based. The Mongo shell is an easy way of manipulating small sets of data. But when the data amount gets bigger, handling them through a command line interface is hard and verbose. Yet, Mongo shell is a great way to easily perform administrative processes in the database server. Listed below are some important characteristics and features of the Mongo shell.

- JavaScript shell: Instead of using a query language like SQL, Mongo shell uses JavaScript and a related API to interact with the database. It has the ability to run arbitrary JavaScript programs.
- JSON/BSON input and output: The documents in MongoDB are stored in JSON or BSON format. It has the ability to transmit data using JSON/BSON as well. So, the inputs and outputs related to database collections are structured in JSON/BSON format.
- Syntax highlighting: The older version of MongoDB mongo shell had bracket matching features in the console. The latest version (mongosh) has improved to JavaScript syntax highlighting which is a great addition to shell.
- Reading Logs: Through Mongo shell we can read MongoDB server logs and in need can tail the logs as well.
- Command history: We can see the previous commands in the mongo shell with up and down arrow keys. This command history is stored in ~/.dbshell file.
- Scripting environment: Mongo shell is a powerful scripting environment. Mongo shell is built on top of Node.js REPL (the basic shell environment of Node). Hence, the Node.js API and the whole of npm packages are available to the Mongo shell.

The newest version of mongo shell *mongosh* has the following additional features other than the above.

- Intelligent autocomplete: Mongo shell can now do the autocompletion of commands or operators to assist you while scripting.
- Contextual help: Mongo shell can provide contextual help on commands and shell classes and provides links to online documentation.
- Error messages: Actively displays the errors with the commands/scripts you write and how to fix them as well.

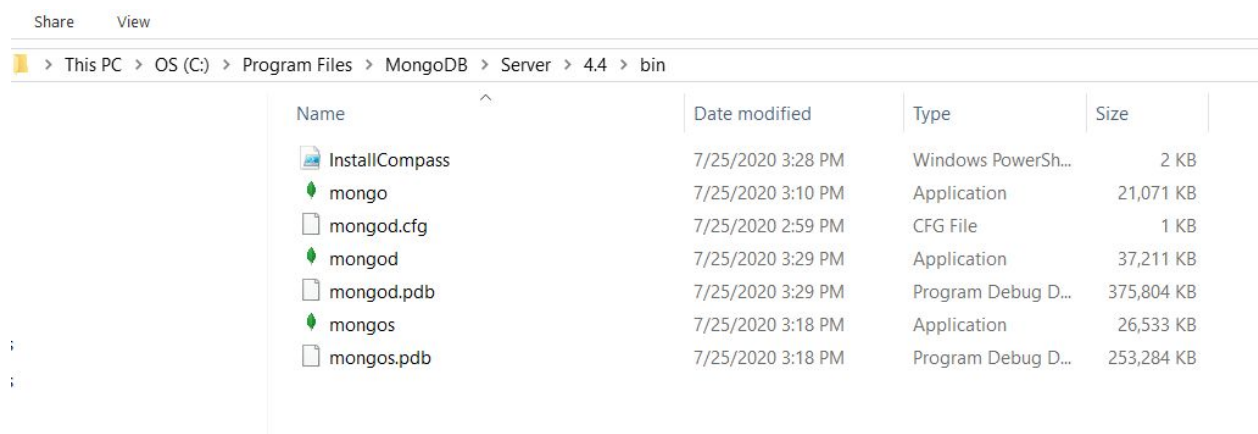
How to get the MongoDB Mongo Shell?

The mongo shell gets installed when a MongoDB server is installed. It is installed to the same location as the MongoDB server binary. If you need to install it separately, you can visit the MongoDB download center, select the version and package you need, download the archive with mongo shell and lastly copy it to a location in your file system.

MongoDB has recently introduced a new mongo shell with additional features such as, extensibility (when new products or services are added to the MongoDB platform, adding new functionality is possible) and embeddability (able to use it inside other products i.e. VS Code).

How to open MongoDB and connect with MongoDB database

Once the downloading and installation is completed, we can use the mongo shell to connect with an up and running MongoDB server. Note that it is required to install and run the server before you connect through shell. The below image shows the location of an installed MongoDB in a Windows system.



The screenshot shows a Windows File Explorer window with the address bar set to 'This PC > OS (C:) > Program Files > MongoDB > Server > 4.4 > bin'. The main pane displays a list of files and folders with columns for Name, Date modified, Type, and Size.

Name	Date modified	Type	Size
InstallCompass	7/25/2020 3:28 PM	Windows PowerSh...	2 KB
mongo	7/25/2020 3:10 PM	Application	21,071 KB
mongod.cfg	7/25/2020 2:59 PM	CFG File	1 KB
mongod	7/25/2020 3:29 PM	Application	37,211 KB
mongod.pdb	7/25/2020 3:29 PM	Program Debug D...	375,804 KB
mongos	7/25/2020 3:18 PM	Application	26,533 KB
mongos.pdb	7/25/2020 3:18 PM	Program Debug D...	253,284 KB

To start the MongoDB, navigate to the bin folder of the installed location using command prompt and run *mongod* command.

```
Command Prompt - mongod

C:\Program Files\MongoDB\Server\4.4\bin>mongod
{"t":{"$date":"2020-09-29T01:01:00.559+05:30"},"s":"I", "c":"CONTROL", "id":23285, "ctx":{"main","msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}}
{"t":{"$date":"2020-09-29T01:01:01.000+05:30"},"s":"M", "c":"ASIO", "id":22601, "ctx":{"main","msg":"No TransportLayer configured during NetworkInterface startup"}}
{"t":{"$date":"2020-09-29T01:01:01.000+05:30"},"s":"I", "c":"NETWORK", "id":4648602, "ctx":{"main","msg":"Implicit TCP FastOpen in use."}}
{"t":{"$date":"2020-09-29T01:01:01.009+05:30"},"s":"I", "c":"STORAGE", "id":4615611, "ctx":{"initandlisten","msg":"MongoDB starting", "attr":{"pid":12672,"port":27017,"dbPath":"C:/data/db/","architecture":"64-bit","host":"LAPTOP-7980331TF"}}}
{"t":{"$date":"2020-09-29T01:01:01.010+05:30"},"s":"I", "c":"CONTROL", "id":223398, "ctx":{"initandlisten","msg":"Target operating system minimum version", "attr":{"targetMinOS":"Windows 7/Windows Server 2008 R2"}}}
{"t":{"$date":"2020-09-29T01:01:01.010+05:30"},"s":"I", "c":"CONTROL", "id":23403, "ctx":{"initandlisten","msg":"Build Info", "attr":{"buildInfo":{"version":"4.4.0","gitVersion":"563487e100c4215e2dce98d0af2a6a5d2d67c5cf","modules":{"allocator":"tcmalloc","environment":{"distmod":"x86_64","target_arch":"x86_64"}}}}}
{"t":{"$date":"2020-09-29T01:01:01.010+05:30"},"s":"I", "c":"CONTROL", "id":51765, "ctx":{"initandlisten","msg":"Operating System", "attr":{"os":{"name":"Microsoft Windows 10","version":"10.0 (build 18362)}}}}
{"t":{"$date":"2020-09-29T01:01:01.010+05:30"},"s":"I", "c":"CONTROL", "id":21951, "ctx":{"initandlisten","msg":"Options set by command line", "attr":{"options":{}}}}
{"t":{"$date":"2020-09-29T01:01:01.011+05:30"},"s":"I", "c":"STORAGE", "id":22270, "ctx":{"initandlisten","msg":"Storage engine to use detected by data files", "attr":{"dbpath":"C:/data/db/","storageEngine":"wiredtiger"}}}
{"t":{"$date":"2020-09-29T01:01:01.012+05:30"},"s":"I", "c":"STORAGE", "id":22315, "ctx":{"initandlisten","msg":"Opening WiredTiger", "attr":{"config":{"create,cache_size=352M,session_max=33000,eviction=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compression=snappy),file_manager=(close_idle_time=100000,close_scan_interval=10,close_handle_minimum=250),statistics_log=(wait=0),verbose=[recovery_progress,checkpoint_progress,compact_progress]}}}}
{"t":{"$date":"2020-09-29T01:01:01.103+05:30"},"s":"I", "c":"STORAGE", "id":22430, "ctx":{"initandlisten","msg":"WiredTiger message", "attr":{"message":{"[1601321461:103208][12672:140724249189968], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 5 through 6"}}}}
{"t":{"$date":"2020-09-29T01:01:01.167+05:30"},"s":"I", "c":"STORAGE", "id":22430, "ctx":{"initandlisten","msg":"WiredTiger message", "attr":{"message":{"[1601321461:167040][12672:140724249189968], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 6 through 6"}}}}
{"t":{"$date":"2020-09-29T01:01:01.239+05:30"},"s":"I", "c":"STORAGE", "id":22430, "ctx":{"initandlisten","msg":"WiredTiger message", "attr":{"message":{"[1601321461:238847][12672:140724249189968], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Main recovery loop: starting at 5/14592 to 6/256"}}}}
{"t":{"$date":"2020-09-29T01:01:01.351+05:30"},"s":"I", "c":"STORAGE", "id":22430, "ctx":{"initandlisten","msg":"WiredTiger message", "attr":{"message":{"[1601321461:350549][12672:140724249189968], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 5 through 6"}}}}
{"t":{"$date":"2020-09-29T01:01:01.456+05:30"},"s":"I", "c":"STORAGE", "id":22430, "ctx":{"initandlisten","msg":"WiredTiger message", "attr":{"message":{"[1601321461:456265][12672:140724249189968], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 6 through 6"}}}}
{"t":{"$date":"2020-09-29T01:01:01.513+05:30"},"s":"I", "c":"STORAGE", "id":22430, "ctx":{"initandlisten","msg":"WiredTiger message", "attr":{"message":{"[1601321461:513152][12672:140724249189968], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global recovery timestamp: (0, 0)}}}}
{"t":{"$date":"2020-09-29T01:01:01.695+05:30"},"s":"I", "c":"STORAGE", "id":4795906, "ctx":{"initandlisten","msg":"WiredTiger opened", "attr":{"durationMillis":683}}}}
{"t":{"$date":"2020-09-29T01:01:01.695+05:30"},"s":"I", "c":"RECOVERY", "id":23907, "ctx":{"initandlisten","msg":"WiredTiger recoveryTimestamp", "attr":{"recoveryTimestamp":{"t":0,"i":0}}}}
{"t":{"$date":"2020-09-29T01:01:01.701+05:30"},"s":"I", "c":"STORAGE", "id":22262, "ctx":{"initandlisten","msg":"Timestamp monitor starting"}}}
{"t":{"$date":"2020-09-29T01:01:01.878+05:30"},"s":"W", "c":"CONTROL", "id":22120, "ctx":{"initandlisten","msg":"Access control is not enabled for the database. Read and write access to data and configuration is unrestricted", "tags":["startupWarnings"]}}
{"t":{"$date":"2020-09-29T01:01:01.878+05:30"},"s":"W", "c":"CONTROL", "id":22140, "ctx":{"initandlisten","msg":"This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with --bind_ip address to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning", "tags":["startupWarnings"]}}
{"t":{"$date":"2020-09-29T01:01:02.082+05:30"},"s":"I", "c":"NETWORK", "id":20536, "ctx":{"initandlisten","msg":"Flow Control is enabled on this deployment"}}}
{"t":{"$date":"2020-09-29T01:01:02.078+05:30"},"s":"I", "c":"FTDC", "id":20625, "ctx":{"initandlisten","msg":"Initializing full-time diagnostic data capture", "attr":{"dataDirectory":"C:/data/db/diagnostic.0/data"}}}
{"t":{"$date":"2020-09-29T01:01:02.082+05:30"},"s":"I", "c":"NETWORK", "id":23015, "ctx":{"listener","msg":"Listening on", "attr":{"address":"127.0.0.1"}}}
{"t":{"$date":"2020-09-29T01:01:02.082+05:30"},"s":"I", "c":"NETWORK", "id":23016, "ctx":{"listener","msg":"Waiting for connections", "attr":{"port":27017,"ssl":"off"}}}
```

This process will start the server and will display the “waiting for connection message”. Once this is done use the command line to run *mongo* command to connect the shell with the running server.

```
Command Prompt - mongo

C:\Program Files\MongoDB\Server\4.4\bin>mongo
MongoDB shell version v4.4.0
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session ("id": "UUID("0e52a341-a12f-49a2-9f20-d9ae64207879")")
MongoDB server version: 4.4.0

---
The server generated these startup warnings when booting:
2020-09-29T01:02:13.043+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
2020-09-29T01:02:13.043+05:30: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with --bind_ip address to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning
---
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you and anyone you share the URL with. MongoDB may use this information to make product improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
>
```

After connecting to the server you can use any command you wish to perform. The below image shows how the basic *show dbs* and *help* command is issued.

```
Command Prompt - mongo
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs
admin 0.000GB
config 0.000GB
local 0.000GB
> help
db.help()          help on db methods
db.mycoll.help()    help on collection methods
sh.help()           sharding helpers
rs.help()           replica set helpers
help admin          administrative help
help connect        connecting to a db help
help keys           key shortcuts
help misc           misc things to know
help mr             mapreduce

show dbs           show database names
show collections    show collections in current database
show users          show users in current database
show profile        show most recent system.profile entries with time >= 1ms
show logs           show the accessible logger names
show log [name]     prints out the last segment of log in memory, 'global' is default
use <db_name>       set current database
db.mycoll.find()     list objects in collection mycoll
db.mycoll.find( { a : 1 } ) list objects in mycoll where a == 1
it                 result of the last line evaluated; use to further iterate
DBQuery.shellBatchSize = x set default number of items to display on shell
exit               quit the mongo shell
```

When you want to leave the shell, just press Ctrl-C so the command line will terminate.

You can run the mongo and mongod without the command prompt if you prefer. Go to the installation location and double click on the mongod and mongo applications. This will operate the same as above.

What are the advantages of MongoDB Mongo Shell?

Mongo shell is an interactive JavaScript interface. Listed below are some advantages of the Mongo shell.

- Ability to run arbitrary JavaScript programs.
- Can use the JavaScript libraries for writing codes in the shell.
- Once the script is written and the Enter button is pressed the shell can determine whether the script is syntactically complete or not.
- An easy way to get an insight of the DB server.
- The JavaScript API for Mongo shell is easy to learn so if you know the basics of JavaScript you can easily manipulate the shell commands.

- Mongo shell is the default manual client for MongoDB. It connects and communicates using MongoDB Wire Protocol over TCP/IP and gives the output in Json format.

What are the disadvantages of Mongo Shell?

While some may prefer using a command line interface for data manipulation, others may prefer more sophisticated features like graphical user interfaces to interact with the database. Mongo shell is great for administration processes but it is a bit cumbersome to handle large sets of data through the shell where a visual representation can ease up the trouble.

The Mongo shell suffers a few drawbacks compared to a sophisticated GUI.

- The Mongo shell is strictly a console centric method of data manipulation. While some find it easy and quick, for some it might not be appealing as such.
- If you are working on multiple sessions you need multiple terminals.
- If the results are too long, they scroll away.
- Repetitive commands or debugging a function need the programmer to traverse the long command line history manually.

Using commands with Mongo Shell

The mongo shell has various help methods such as *help*, *db.help()* , *db.<collection>.help()*. Apart from that there are helper commands to print the list of databases, list of collections, users, roles, the five most recent operations and load a JavaScript file. Mongo shell has a JavaScript API for db operations. Apart from above common commands, the JavaScript API has many commands for manipulation of data and administration of the database as well.

Note that the above-mentioned helper commands cannot be used in a JavaScript file since they are not JavaScript. The available shell commands can be categorized as basic helpers, indexing commands, CRUD operations, show commands, cursor methods, comparison operators, logical operators and element operators.

What are the alternatives to MongoDB Mongo Shell?

If your application is written based on Node.js, you can use MongoDB Node.js driver as an alternative to MongoDB shell. Just as shell, it is a command line interface. However, there is more code in MongoDB Node.js driver than the shell. The reason is, the shell is in the context

already and it knows we are communicating with what. In MongoDB Node.js driver, we need to specify all the set ups and dependencies to connect with the database. The syntaxes and method names are different from each other but using MongoDB Node.js driver we can write application codes with more power.

Apart from that, there are many sophisticated GUI alternatives to Mongo shell listed as below.

- Studio 3T
- Robo 3T
- NoSQLBooster
- MongoDB Compass
- NoSQL Manager
- Mongo Management Studio
- Robomongo
- MongoBooster
- MongoVue
- MongoHub
- RockMongo
- Retool

Remember that the best MongoDB GUI depends on the task that needs to be accomplished. MongoDB Compass is the go-to option if you need to avoid the command line completely. Robo 3T is simple and well supported by the community while NoSQLBooster is SQL friendly and is for SQL lovers.

Conclusion

This article is about the default client for MongoDB, the MongoDB shell. Starting from a basic introduction, we gradually learned about the important features, how to install and connect with the database, advantages, disadvantages and also some alternatives to the Mongo shell as well.