```
Maximilians-MBP:~ mschwarzmueller$ mongod --help
Options:

General options:
  -v [ --verbose ] [=arg(=v)]           be more verbose (include multiple times
                                        for more verbosity e.g. -vvvvv)
  --quiet                               quieter output
  --port arg                            specify port number - 27017 by default
  --logpath arg                         log file to send write to instead of
                                        stdout - has to be a file, not
                                        directory
  --syslog                              log to system's syslog facility instead
                                        of file or stdout
  --syslogFacility arg                  syslog facility used for mongodb syslog
                                        message
  --logappend                           append to logpath instead of
                                        over-writing
  --logRotate arg                       set the log rotation behavior
                                        (rename|reopen)
  --timeStampFormat arg                 Desired format for timestamps in log
                                        messages. One of ctime, iso8601-utc or
                                        iso8601-local
  --setParameter arg                    Set a configurable parameter
  -h [ --help ]                         show this usage information
```

```
Maximilians-MBP:~ mschwarzmueller$ sudo mongod --dbpath /Users/mschwarzmueller/development/mongodb/db
--logpath /Users/mschwarzmueller/development/mongodb/logs/log.log
```

Configuration File

You can configure `mongod` and `mongos` instances at startup using a configuration file. The configuration file contains settings that are equivalent to the `mongod` and `mongos` command-line options. See Configuration File Settings and Command-Line Options Mapping.

Using a configuration file makes managing `mongod` and `mongos` options easier, especially for large-scale deployments. You can also add comments to the configuration file to explain the server's settings.

On Linux, a default **`/etc/mongod.conf`** configuration file is included when using a package manager to install MongoDB.

On Windows, a default **`<install directory>/bin/mongod.cfg`** configuration file is included during the installation.

On macOS, the installation does not include a default configuration file; instead, to use a configuration file, create a file.

mongod.cfg

```
storage:
  dbPath: "/Users/mschwarzmueller/development/mongodb/db"
systemLog:
  destination: file
  path: "/Users/mschwarzmueller/development/mongodb/logs/logs.log"
```

```
Maximilians-MBP:~ mschwarzmueller$ sudo mongod -f /Users/mschwarzmueller/development/mongodb/bin/mongo
d.cfg
Password:
2018-09-13T11:18:18.280+0200 I CONTROL  [main] Automatically disabling TLS 1.0, to force-enable TLS 1.
0 specify --sslDisabledProtocols 'none'
```

```
Maximilians-MBP:~ mschwarzmueller$ mongo --help
```

```
  192.168.0.5:9999/foo  foo database on 192.168.0.5 machine on port 9999
Options:
  --shell                               run the shell after executing files
  --nodb                                don't connect to mongod on startup - no
                                        'db address' arg expected
  --norc                                will not run the ".mongorc.js" file on
                                        start up
  --quiet                               be less chatty
  --port arg                            port to connect to
  --host arg                            server to connect to
  --eval arg                            evaluate javascript
  -h [ --help ]                         show this usage information
  --version                             show version information
  --verbose                             increase verbosity
  --ipv6                                enable IPv6 support (disabled by default)
  --disableJavaScriptJIT                disable the Javascript Just In Time
                                        compiler
  --enableJavaScriptJIT                 enable the Javascript Just In Time
                                        compiler
  --disableJavaScriptProtection         allow automatic JavaScript function
                                        marshalling
  --ssl                                 use SSL for all connections
  --sslCAFile arg                       Certificate Authority file for SSL
  --sslPEMKeyFile arg                   PEM certificate/key file for SSL
  --sslPEMKeyPassword arg               password for key in PEM file for SSL
  --sslCRLFile arg                      Certificate Revocation List file for SSL
  --sslAllowInvalidHostnames            allow connections to servers with
                                        non-matching hostnames
```

```
> 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.foo.find()                list objects in collection foo
        db.foo.find( { a : 1 } )     list objects in foo 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
```

```
exit                                    quit the mongo shell
> help admin
      ls([path])                          list files
      pwd()                               returns current directory
      listFiles([path])                   returns file list
      hostname()                          returns name of this host
      cat(fname)                          returns contents of text file as a string
      removeFile(f)                       delete a file or directory
      load(jsfilename)                    load and execute a .js file
      run(program[, args...])             spawn a program and wait for its completion
      runProgram(program[, args...])      same as run(), above
      sleep(m)                            sleep m milliseconds
      getMemInfo()                        diagnostic
>
```

```
> db.help()
DB methods:
        db.adminCommand(nameOrDocument) - switches to 'admin' db, and runs command [just calls db.runC
ommand(...)]
        db.aggregate([pipeline], {options}) - performs a collectionless aggregation on this database;
returns a cursor
        db.auth(username, password)
        db.cloneDatabase(fromhost) - deprecated
        db.commandHelp(name) returns the help for the command
        db.copyDatabase(fromdb, todb, fromhost) - deprecated
        db.createCollection(name, {size: ..., capped: ..., max: ...})
        db.createView(name, viewOn, [{$operator: {...}}, ...], {viewOptions})
        db.createUser(userDocument)
        db.currentOp() displays currently executing operations in the db
        db.dropDatabase()
        db.eval() - deprecated
        db.fsyncLock() flush data to disk and lock server for backups
        db.fsyncUnlock() unlocks server following a db.fsyncLock()
        db.getCollection(cname) same as db['cname'] or db.cname
        db.getCollectionInfos([filter]) - returns a list that contains the names and options of the db
's collections
        db.getCollectionNames()
        db.getLastError() - just returns the err msg string
        db.getLastErrorObj() - return full status object
        db.getLogComponents()
        db.getMongo() get the server connection object
```

```
> db.help()
DB methods:
        db.adminCommand(nameOrDocument) - switches to 'admin' db, and runs command [just calls db.runC
ommand(...)]
        db.aggregate([pipeline], {options}) - performs a collectionless aggregation on this database;
returns a cursor
        db.auth(username, password)
        db.cloneDatabase(fromhost) - deprecated
        db.commandHelp(name) returns the help for the command
        db.copyDatabase(fromdb, todb, fromhost) - deprecated
        db.createCollection(name, {size: ..., capped: ..., max: ...})
        db.createView(name, viewOn, [{$operator: {...}}, ...], {viewOptions})
        db.createUser(userDocument)
        db.currentOp() displays currently executing operations in the db
        db.dropDatabase()
        db.eval() - deprecated
        db.fsyncLock() flush data to disk and lock server for backups
        db.fsyncUnlock() unlocks server following a db.fsyncLock()
        db.getCollection(cname) same as db['cname'] or db.cname
        db.getCollectionInfos([filter]) - returns a list that contains the names and options of the db
's collections
        db.getCollectionNames()
        db.getLastError() - just returns the err msg string
```

```
> db.help()
DB methods:
        db.adminCommand(nameOrDocument) - switches to 'admin' db, and runs command [just calls db.runC
ommand(...)]
        db.aggregate([pipeline], {options}) - performs a collectionless aggregation on this database;
returns a cursor
        db.auth(username, password)
        db.cloneDatabase(fromhost) - deprecated
        db.commandHelp(name) returns the help for the command
        db.copyDatabase(fromdb, todb, fromhost) - deprecated
        db.createCollection(name, {size: ..., capped: ..., max: ...})
        db.createView(name, viewOn, [{$operator: {...}}, ...], {viewOptions})
        db.createUser(userDocument)
        db.currentOp() displays currently executing operations in the db
        db.dropDatabase()
        db.eval() - deprecated
        db.fsyncLock() flush data to disk and lock server for backups
        db.fsyncUnlock() unlocks server following a db.fsyncLock()
        db.getCollection(cname) same as db['cname'] or db.cname
        db.getCollectionInfos([filter]) - returns a list that contains the names and options of the db
's collections
        db.getCollectionNames()
        db.getLastError() - just returns the err msg string
```

```
> db.test.help()
```

```
        db.test.insert(obj)
        db.test.insertOne( obj, <optional params> ) - insert a document, optional parameters are: w, w
timeout, j
        db.test.insertMany( [objects], <optional params> ) - insert multiple documents, optional param
eters are: w, wtimeout, j
        db.test.mapReduce( mapFunction , reduceFunction , <optional params> )
        db.test.aggregate( [pipeline], <optional params> ) - performs an aggregation on a collection;
returns a cursor
        db.test.remove(query)
        db.test.replaceOne( filter, replacement, <optional params> ) - replace the first matching docu
ment, optional parameters are: upsert, w, wtimeout, j
        db.test.renameCollection( newName , <dropTarget> ) renames the collection.
        db.test.runCommand( name , <options> ) runs a db command with the given name where the first p
aram is the collection name
        db.test.save(obj)
        db.test.stats({scale: N, indexDetails: true/false, indexDetailsKey: <index key>, indexDetailsN
ame: <index name>})
        db.test.storageSize() - includes free space allocated to this collection
        db.test.totalIndexSize() - size in bytes of all the indexes
        db.test.totalSize() - storage allocated for all data and indexes
        db.test.update( query, object[, upsert_bool, multi_bool] ) - instead of two flags, you can pas
s an object with fields: upsert, multi
        db.test.updateOne( filter, update, <optional params> ) - update the first matching document, o
ptional parameters are: upsert, w, wtimeout, j
        db.test.updateMany( filter, update, <optional params> ) - update all matching documents, optio
nal parameters are: upsert, w, wtimeout, j
        db.test.validate( <full> ) - SLOW
        db.test.getShardVersion() - only for use with sharding
```