

Connecting to MongoDB from Java

Ramesh S



Getting Started

- `import com.mongodb.BasicDBObject;`
 - `import com.mongodb.DB;`
 - `import com.mongodb.DBCollection;`
 - `import com.mongodb.DBCursor;`
 - `import com.mongodb.DBObject;`
 - `import com.mongodb.MongoClient;`
- 

Connect to server

To directly connect to a single MongoDB server

```
MongoClient mongoClient = new MongoClient();
```

```
MongoClient mongoClient = new MongoClient( "localhost" );
```

```
MongoClient mongoClient = new MongoClient( "localhost" , 27017 );
```



Connect to server

To connect to a replica set, with auto-discovery of the primary, supply a seed list of members

```
MongoClient mongoClient = new MongoClient(Arrays.asList(  
    new ServerAddress("localhost", 27017),  
    new ServerAddress("localhost", 27018),  
    new ServerAddress("localhost", 27019)  
));
```

```
DB db = mongoClient.getDB( "mydb" );
```



Authentication

```
MongoCredential credential =  
    MongoCredential.createMongoCRCredential(userName,  
        database, password);
```

```
MongoClient mongoClient = new MongoClient(new  
    ServerAddress(), Arrays.asList(credential));
```



Getting a Collection

```
DBCollection coll = db.getCollection("myCollection");
```



Setting Write Concern

```
mongoClient.setWriteConcern(WriteConcern.JOURNALED);
```

WriteConcern.ACKNOWLEDGED is default.



Write Concern Explained

`WriteConcern.UNACKNOWLEDGED`

Exceptions are only thrown when the primary node is unreachable for a read, or the full replica set is unreachable.

`WriteConcern.ACKNOWLEDGED`

Same as the above, but exceptions thrown when there is a server error on writes or reads. Calls `getLastError()`.

`WriteConcern.REPLICA_ACKNOWLEDGED`

Tries to write to two separate nodes. Same as the above, but will throw an exception if two writes are not possible.

`WriteConcern.JOURNALED`

Same as *WriteConcern.ACKNOWLEDGED*, but also waits for write to be written to the journal.

Inserting a document

```
BasicDBObject doc = new BasicDBObject("name", "MongoDB")
    .append("type", "database")
    .append("count", 1)
    .append("info", new BasicDBObject("x", 203).append("y", 102));
coll.insert(doc);
```

Using FindOne()

```
DBObject myDoc = coll.findOne();  
System.out.println(myDoc);
```


Using a Cursors

```
DBCursor cursor = coll.find();
try {
    while(cursor.hasNext()) {
        System.out.println(cursor.next());
    }
} finally {
    cursor.close();
}
```

Get a set of documents

```
// find all where i > 50
query = new BasicDBObject("i", new BasicDBObject("$gt", 50));

cursor = coll.find(query);
try {
    while (cursor.hasNext()) {
        System.out.println(cursor.next());
    }
} finally {
    cursor.close();
}
```



List Databases

```
MongoClient mongoClient = new MongoClient();  
  
for (String s : mongoClient.getDatabaseNames()) {  
    System.out.println(s);  
}
```

Read Preferences

PRIMARY

The default read mode. Read from primary only. Throw an error if primary is unavailable. Cannot be combined with tags.

PRIMARY PREFERRED

Read from primary if available, otherwise a secondary.

SECONDARY


Read from a secondary node if available, otherwise error.

SECONDARY PREFERRED

Read from a secondary if available, otherwise read from the primary.

NEAREST

Read from any member node from the set of nodes which respond the fastest.



Read Preference

```
ReadPreference preference = ReadPreference.primaryPreferred();  
DBCursor cur = new DBCursor(collection, query, null, preference);
```