6

MongoDB with Java

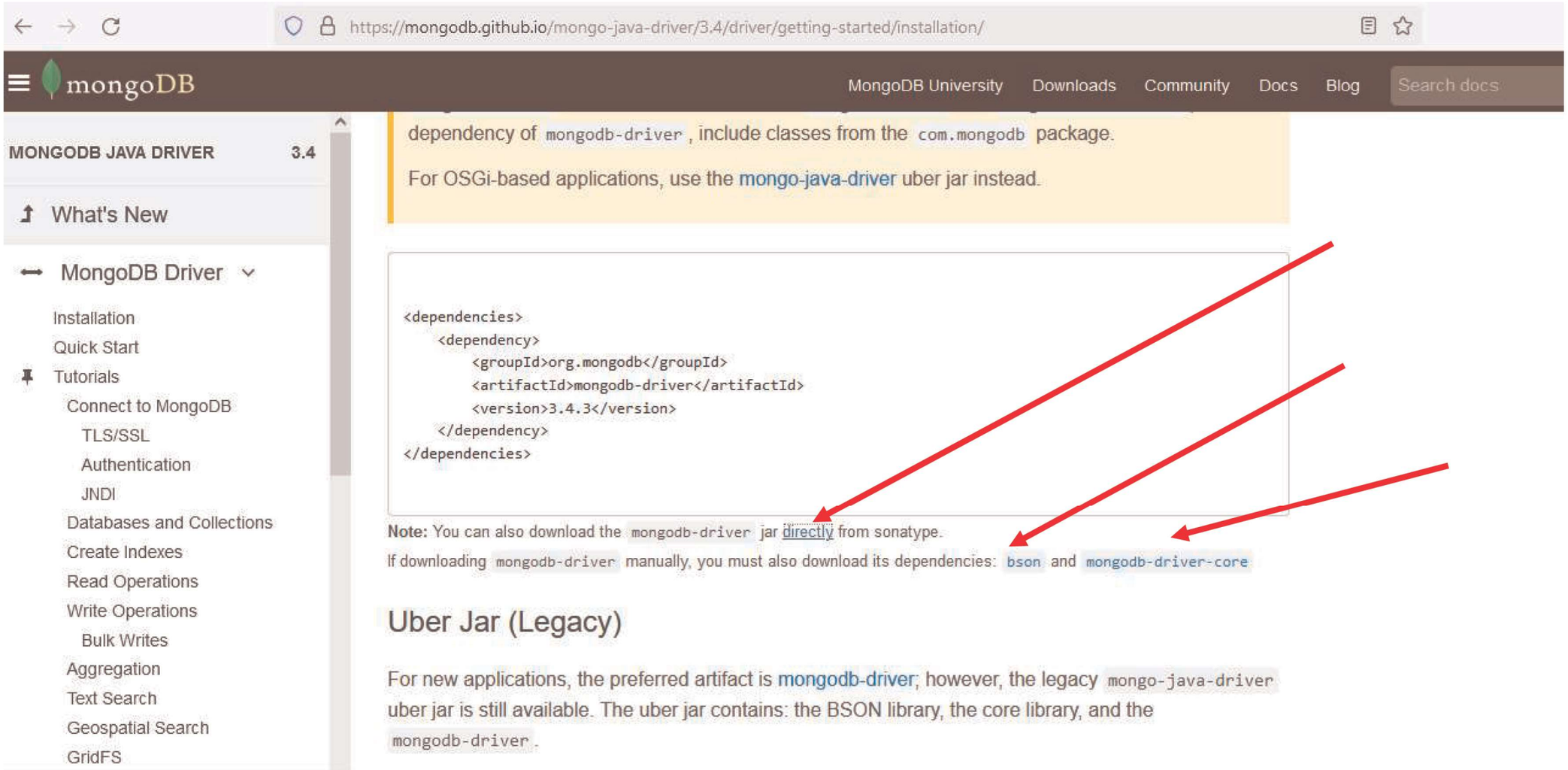
Objectives

After completing this lesson, you should be able to do the following:

- Connection Java With MongoDB
- Performing CRUD Operations

MongoDB Driver Jarz

- Before you start using MongoDB in your Java programs, you need to make sure that you have MongoDB CLIENT.
  - You need to download the jar **mongodb-driver-x.y.z.jar and its dependency mongodb-driver-core-x.y.z.jar.**
  - You need to include the downloaded jar files into your classpath.


https://mongodb.github.io/mongo-java-driver/3.4/driver/getting-started/installation/
mongoDB
MongoDB University
Downloads
Community
Docs
Blog
Search docs
MONGODB JAVA DRIVER 3.4
What's New
MongoDB Driver
Installation
Quick Start
Tutorials
Connect to MongoDB
TLS/SSL
Authentication
JNDI
Databases and Collections
Create Indexes
Read Operations
Write Operations
Bulk Writes
Aggregation
Text Search
Geospatial Search
GridFS
dependency of mongodb-driver , include classes from the com.mongodb package.
For OSGi-based applications, use the mongo-java-driver uber jar instead.
<dependencies>
<dependency>
<groupId>org.mongodb</groupId>
<artifactId>mongodb-driver</artifactId>
<version>3.4.3</version>
</dependency>
</dependencies>
Note: You can also download the mongodb-driver jar directly from sonatype.
If downloading mongodb-driver manually, you must also download its dependencies: bson and mongodb-driver-core
Uber Jar (Legacy)
For new applications, the preferred artifact is mongodb-driver; however, the legacy mongo-java-driver uber jar is still available. The uber jar contains: the BSON library, the core library, and the mongodb-driver .

Connect to Database

- To connect database, you need to specify the database name, if the database doesn't exist then MongoDB creates it automatically.

ConnectToDB.java

```
3  import com.mongodb.client.MongoDatabase;
4  import com.mongodb.MongoClient;
5  import com.mongodb.MongoCredential;
6  public class ConnectToDB {
7
8     public static void main( String args[] ) {
9
10       // Creating a Mongo client
11       MongoClient mongo = new MongoClient( "localhost" , 27017 );
12
13       // Creating Credentials
14       MongoCredential credential;
15       credential = MongoCredential.createCredential("TMS", "EmployeeDB",
16          "password".toCharArray());
17       System.out.println("Connected to the database successfully");
18
19       // Accessing the database
20       MongoDatabase database = mongo.getDatabase("EmployeeDB");
21       System.out.println("Credentials ::"+ credential);
22    }
```

Creating a Collection

- To create a collection, **createCollection()** method of **com.mongodb.client.MongoDatabase** class is used.

ConnectToDB.java | CreatingCollection.java

```
6
7    public static void main( String args[] ) {
8
9       // Creating a Mongo client
10      MongoClient mongo = new MongoClient( "localhost" , 27017 );
11
12      // Creating Credentials
13      MongoCredential credential;
14      credential = MongoCredential.createCredential("sampleUser", "EmployeeDB",
15         "password".toCharArray());
16      System.out.println("Connected to the database successfully");
17
18      //Accessing the database
19      MongoDatabase database = mongo.getDatabase("EmployeeDB");
20
21      //Creating a collection
22      database.createCollection("sampleCollection");
23      System.out.println("Collection created successfully");
24   }
25 }
```

Getting/Selecting a Collection

- To get/select a collection from the database, **getCollection()** method of **com.mongodb.client.MongoDatabase** class is used.

ConnectToDB.java | CreatingCollection.java | SelectingCollection.java

```
10
11        // Creating a Mongo client
12        MongoClient mongo = new MongoClient( "localhost" , 27017 );
13
14        // Creating Credentials
15        MongoCredential credential;
16        credential = MongoCredential.createCredential("sampleUser", "EmployeeDB",
17           "password".toCharArray());
18        System.out.println("Connected to the database successfully");
19
20        // Accessing the database
21        MongoDatabase database = mongo.getDatabase("EmployeeDB");
22
23        // Creating a collection
24        database.createCollection("myCollection");
25        System.out.println("Collection created successfully");
26        // Retrieving a collection
27        MongoCollection<Document> collection = database.getCollection("myCollection");
28        System.out.println("Collection myCollection selected successfully");
29     }
```

Insert a Document

- To insert a document into MongoDB, **insert()** method of **com.mongodb.client.MongoCollection** class is used.

ConnectToDB.java | CreatingCollection.java | SelectingCollection.java | InsertingDocument.java

JavaToMongoDB/src/training/iqgateway/CreatingCollection.java

```
10    MongoClient mongo = new MongoClient( "localhost" , 27017 );
11
12    // Accessing the database
13    MongoDatabase database = mongo.getDatabase("EmployeeDB");
14
15    // Retrieving a collection
16    MongoCollection<Document> collection = database.getCollection("sampleCollection");
17    System.out.println("Collection sampleCollection selected successfully");
18    Document document = new Document("title", "MongoDB")
19    .append("description", "database")
20    .append("likes", 100)
21    .append("url", "http://www.mongodb.com/")
22    .append("by", "MongoDB");
23
24    //Inserting document into the collection
25    collection.insertOne(document);
26    System.out.println("Document inserted successfully");
27    }
28 }
```

Retrieve All Documents

- To select all documents from the collection, **find()** method of **com.mongodb.client.MongoCollection** class is used. This method returns a cursor, so you need to iterate this cursor.

ConnectToDB.java | CreatingCollection.java | SelectingCollection.java | InsertingDocument.java | RetrievingAllDocuments.java

```
1  package training.iqgateway;
2  import com.mongodb.client.FindIterable;
3  import com.mongodb.client.MongoCollection;
4  import com.mongodb.client.MongoDatabase;
5  import java.util.ArrayList;
6  import java.util.Iterator;
7  import java.util.List;
8  import org.bson.Document;
9  import com.mongodb.MongoClient;
10 import com.mongodb.MongoCredential;
11 public class RetrievingAllDocuments {
12     public static void main( String args[] ) {
13
14         // Creating a Mongo client
15         MongoClient mongo = new MongoClient( "localhost" , 27017 );
16
```

ConnectToDB.java | CreatingCollection.java | SelectingCollection.java | InsertingDocument.java | *RetrievingAllDocuments.java

```
16
17          // Creating Credentials
18          MongoCredential credential;
19          credential = MongoCredential.createCredential("sampleUser", "EmployeeDB",
20                  "password".toCharArray());
21          System.out.println("Connected to the database successfully");
22
23          // Accessing the database
24          MongoDatabase database = mongo.getDatabase("EmployeeDB");
25
26          // Retrieving a collection
27          MongoCollection<Document> collection = database.getCollection("Employee");
28          System.out.println("Collection sampleCollection selected successfully");
29          // Getting the iterable object
30          FindIterable<Document> iterDoc = collection.find();
31          int i = 1;
32          // Getting the iterator
33          Iterator it = iterDoc.iterator();
34          while (it.hasNext()) {
35              System.out.println(it.next());
36              i++;
37          }
38      }
```

Update Document

- To update a document from the collection, **updateOne()** method of **com.mongodb.client.MongoCollection** class is used.

ConnectToDB.java | CreatingCollection.java | SelectingCollection.java | InsertingDocument.java | RetrievingAllDocuments.java | UpdatingDocuments.java

```
21            "password".toCharArray());
22        System.out.println("Connected to the database successfully");
23
24        // Accessing the database
25        MongoDatabase database = mongo.getDatabase("EmployeeDB");
26        // Retrieving a collection
27        MongoCollection<Document> collection = database.getCollection("sampleCollection");
28        System.out.println("Collection myCollection selected successfully");
29        collection.updateOne(Filters.eq("title", 1), Updates.set("likes", 150));
30        System.out.println("Document update successfully...");
31
32        // Retrieving the documents after updation
33        // Getting the iterable object
34        FindIterable<Document> iterDoc = collection.find();
35        int i = 1;
36        // Getting the iterator
37        Iterator it = iterDoc.iterator();
38        while (it.hasNext()) {
39           System.out.println(it.next());
40           i++;
41        }
42     }
43  }
```

Delete a Document

- To delete a document from the collection, you need to use the **deleteOne()** method of the **com.mongodb.client.MongoCollection** class.

ConnectToDB.java | CreatingCollection.java | SelectingCollection.java | InsertingDocument.java | RetrievingAllDocuments.j... | UpdatingDocuments.java | DeletingDocuments.java

```
21        System.out.println("Connected to the database successfully");
22
23        // Accessing the database
24        MongoDatabase database = mongo.getDatabase("EmployeeDB");
25        // Retrieving a collection
26        MongoCollection<Document> collection = database.getCollection("sampleCollection");
27        System.out.println("Collection sampleCollection selected successfully");
28        // Deleting the documents
29        collection.deleteOne(Filters.eq("title", "MongoDB"));
30        System.out.println("Document deleted successfully...");
31
32        // Retrieving the documents after updation
33        // Getting the iterable object
34        FindIterable<Document> iterDoc = collection.find();
35        int i = 1;
36        // Getting the iterator
37        Iterator it = iterDoc.iterator();
38        while (it.hasNext()) {
39           System.out.println(it.next());
40           i++;
41        }
42     }
43  }
```

Dropping a Collection

- To drop a collection from a database, you need to use the **drop()** method of the **com.mongodb.client.MongoCollection** class.

ConnectToDB.java | CreatingCollectio... | SelectingCollecti... | InsertingDocume... | RetrievingAllDoc... | UpdatingDocumen... | DeletingDocumen... | DropingCollection...

```
7  public class DropingCollection {
8
9     public static void main( String args[] ) {
10       // Creating a Mongo client
11       MongoClient mongo = new MongoClient( "localhost" , 27017 );
12       // Creating Credentials
13       MongoCredential credential;
14       credential = MongoCredential.createCredential("sampleUser", "EmployeeDB",
15          "password".toCharArray());
16       System.out.println("Connected to the database successfully");
17
18       // Accessing the database
19       MongoDatabase database = mongo.getDatabase("EmployeeDB");
20
21       // Creating a collection
22       System.out.println("Collections created successfully");
23       // Retrieving a collection
24       MongoCollection<Document> collection = database.getCollection("sampleCollection");
25       // Dropping a Collection
26       collection.drop();
27       System.out.println("Collection dropped successfully");
28    }
29 }
```

Listing All the Collections

- To list all the collections in a database, you need to use the **listCollectionNames()** method of the **com.mongodb.client.MongoDatabase** class.

ConnectToDB.j... | CreatingColle... | SelectingColl... | InsertingDocu... | RetrievingAll... | UpdatingDocu... | DeletingDocu... | DropingCollec... | ListOfCollecti...

```
1  package training.iqgateway;
2  import com.mongodb.client.MongoDatabase;
5  public class ListOfCollection {
6
7     public static void main( String args[] ) {
8
9        // Creating a Mongo client
10       MongoClient mongo = new MongoClient( "localhost" , 27017 );
11       // Creating Credentials
12       MongoCredential credential;
13       credential = MongoCredential.createCredential("sampleUser", "EmployeeDB",
14          "password".toCharArray());
15       System.out.println("Connected to the database successfully");
16
17       // Accessing the database
18       MongoDatabase database = mongo.getDatabase("EmployeeDB");
19       System.out.println("Collection created successfully");
20       for (String name : database.listCollectionNames()) {
21          System.out.println(name);
22       }
23    }
24 }
```

Summary

In this lesson, you should have learned how to:

- Connection Java With MongoDB
- Performing CRUD Operations