CRUD JDBC CON MONGODB
NIVEL APLICACION

EDUARD LARA

INDICE

1. INTRODUCCION

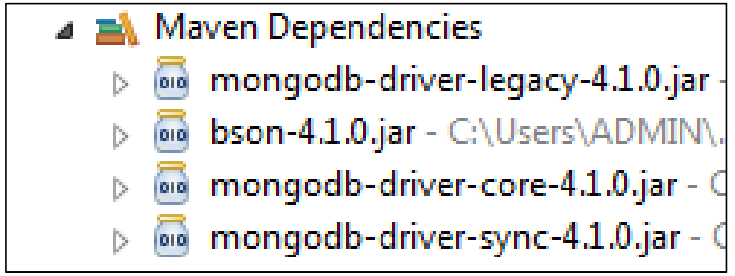
- Realiza un CRUD en Java que acceda a la base de datos mongoDB.
- Debe de realizar las siguientes funcionalidades:

```
//CRUD--Create, Read, Update, Delete
boolean fin=false;
while (!fin) {
    System.out.println("1. Lista las bases de datos de Mongodb");
    System.out.println("2. Lista las colecciones de una base datos");
    System.out.println("3. Visualiza todos los documentos de profesores");
    System.out.println("4. Inserta un nuevo documento profesor");
    System.out.println("5. Inserta un documento profesor con un objeto aula");
    System.out.println("6. Modifica un documento profesor");
    System.out.println("7. Borra un documento profesor por id");
    System.out.println("8. Borra todos los documentos profesor");
    System.out.println("9. Consulta con alguna funcion de agregación");
    System.out.println("10. Vuelca todos los documentos en un fichero de texto");
    System.out.println("11. Salir");
    System.out.print("Introduce que opcion quieres?");
```

1. INTRODUCCION

- Para trabajar en Java con MongoDB podemos descargar el driver desde la URL de MongoDB https://mongodb.github.io/mongo-java-driver/
- Si se utiliza Maven, debemos agregar las siguientes dependencias en pom.xml:

```
<dependencies>
    <dependency>
        <groupId>org.mongodb</groupId>
        <artifactId>mongodb-driver-legacy</artifactId>
        <version>4.1.0</version>
    </dependency>
</dependencies>
```



2. CONEXIÓN BASE DE DATOS

- Para conectarnos a la bbdd creamos una instancia **MongoClient**. Es el equivalente a la típica variable Connection de las bases de datos relacionales
- Por defecto, se puede instanciar un objeto MongoClient sin ningún parámetro para conectarse a una instancia MongoDB ejecutándose en localhost:27017:

  MongoClient mongoClient = MongoClients.*create();*
- O se puede especificar el ConnectionString:

  String connectString = "mongodb://localhost:27017“;

  MongoClient mongoClient = MongoClients.*create(connectString);*

2. CONEXIÓN BASE DE DATOS

- Todos los métodos con operaciones CRUD (Create, Read, Update, Delete) en Java se acceden a través de la interfaz **MongoCollection**
- Las instancias de **MongoCollection** se pueden obtener a partir de una instancia **MongoClient** por medio de una **MongoDatabase**.
- MongoCollection es una interfaz genérica: el parámetro de tipo Document es la clase que los clientes utilizan para insertar o modificar los documentos de una colección y es el tipo predeterminado para devolver búsquedas (find).
- El método de un solo argumento **getCollection** devuelve una instancia de **MongoCollection <Document>,** y así es como podemos trabajar con instancias de la clase de documento.

```
MongoClient mongoClient = MongoClients.create();
MongoDatabase db = mongoClient.getDatabase ("nueva");
MongoCollection <Document> coleccion = db.getCollection("profes");
```

3. READ (CRUD)

Listado de las bases de datos

```
List<Document> databases = mongoClient.listDatabases().into(new ArrayList<>());
databases.forEach(db -> System.out.println(db.toJson()));

MongoCursor<String> dbsCursor = mongoClient.listDatabaseNames().iterator();
while(dbsCursor.hasNext()) {
    System.out.println(dbsCursor.next());
}
```

Listado de las colecciones de una base de datos

```
MongoDatabase db = mongoClient.getDatabase ("nueva");
for (String name : db.listCollectionNames()) {
    System.out.println(name);
}
```

3. READ (CRUD)

Listado de los documentos de una colección (READ)

- Los datos de una colección se pueden cargar en una lista utilizando el método **find().into()** de la siguiente manera:

```
MongoCollection <Document> coleccion = db.getCollection("profes");
List<Document> consulta = coleccion.find().into(new ArrayList<Document> ());
for (int i =0; i < consulta.size(); i++) {
    System.out.println("-" + consulta.get(i).toString());
}
```

- El método **find()** devuelve un cursor, en concreto devuelve una instancia **FindIterable**. Podemos utilizar el método **iterator()** para recorrer el cursor. Los documentos de la colección y se visualizan en formato Json:

```
MongoCursor<Document> cursor = coleccion.find().iterator();
while (cursor.hasNext()) {
    Document doc = cursor.next();
    System.out.println (doc.toJson());
}
cursor.close();
```

4. CREATE (CRUD)

Inserción documentos (CREATE)

- Para insertar documentos, creamos un objeto Document, con el método put asignamos los pares *clave-valor*, donde el primer parámetro es el nombre del campo o la clave, y el segundo el valor.
- Mediante el método insertOne se inserta un documento en la colección:

```
//Insercion con clave _id
Document amigo = new Document();
amigo.put("_id", 1);
amigo.put("nombre", "Pedro");
amigo.put("edad", 30);
coleccion.insertOne(amigo);
```

```
{"_id": 1, "nombre": "Pedro", "edad": 30}
```

4. CREATE (CRUD)

Inserción documentos (CREATE)

- Inserción de un documento sin id.

```
//Insercion sin clave _id
Document agenda = new Document();
agenda.put("nombre", "Jose");
agenda.put("telefono", 925677);
agenda.put("curso","DAM2");
coleccion.insertOne(agenda);
```

```
{"_id": {"$oid": "600c886a35f9321b63de0ea2"}, "nombre": "Jose", "telefono": 925677, "curso": "DAM2"}
```

- Inserción de un documento con fecha.

```
//Insercion con _id y Date
Document agenda = new Document();
agenda.put("_id", 2);
agenda.put("telefono", 655577);
agenda.put("curso","DAM2");
agenda.put("fecha",new Date());
coleccion.insertOne(agenda);
```

```
{"_id": 2, "telefono": 655577, "curso": "DAM2", "fecha": {"$date": "2021-01-24T10:18:29.483Z"}}
```

4. CREATE (CRUD)

Inserción documentos (CREATE)

- Inserción de un objeto dentro de un documento:

```
//Insercion de un objeto dentro de otro objeto
Document main = new Document();
main.put("_id", 3);
main.put("nombre", "Maria");
main.put("telefono", 444444);
BasicDBObject clase = new BasicDBObject();
clase.put("nombre", "Informatica");
clase.put("nivel", "DAM2");
main.put("clase", clase);
coleccion.insertOne(main);
```

```
{"_id": 3, "nombre": "Maria", "telefono": 444444, "clase": {"nombre": "Informatica", "nivel": "DAM2"}}
```

4. CREATE (CRUD)

Inserción documentos (CREATE)

- Inserción de un array dentro de un documento:

```
//Insercion de un array de string
Document main = new Document();
main.put("_id", 4);
main.put("nombre", "Pilar");
main.put("telefono", 678944);
List <String> lista = new ArrayList <String>();
lista.add("dao");
lista.add("java");
main.put("modulos", lista);
coleccion.insertOne(main);
```

```
{"_id": 4, "nombre": "Pilar", "telefono": 678944, "modulos": ["dao", "java"]}
```

4. CREATE (CRUD)

Inserción documentos (CREATE)

- Inserción de un array de objetos dentro de un documento:

```
//Insercion de un array de objetos
Document main = new Document();
main.put("_id", 5);
main.put("nombre", "Juan");
main.put("telefono", 678944);
BasicDBList dbl = new BasicDBList();
dbl.add(new BasicDBObject("informatica","DAM2"));
dbl.add(new BasicDBObject("fol", "ASIX1"));
main.put("modulos", dbl);
coleccion.insertOne(main);
```

```
{"_id": 5, "nombre": "Juan", "telefono": 678944, "modulos": [{"informatica": "DAM2"}, {"fol": "ASIX1"}]}
```

4. CREATE (CRUD)

Inserción documentos (CREATE)

- Inserción mezcla de array y un dato normal:

```
//Insercion mezclada atributos y array
Document main = new Document();
main.put("_id", 6);
main.put("nombre", "Lucas");
main.put("telefono", 4478944);
BasicDBList dbl = new BasicDBList();
dbl.add(new BasicDBObject("informatica","DAM2"));
dbl.add(new BasicDBObject("fol", "ASIX1"));
BasicDBObject outer=new BasicDBObject("institut", "ies").append("items", dbl);
main.put("modulos", outer);
coleccion.insertOne(main);
```

```
{"_id": 6, "nombre": "Lucas", "telefono": 4478944,
```

```
"modulos": {"institut": "ies", "items": [{"informatica": "DAM2"}, {"fol": "ASIX1"}]}}
```

5. UPDATE (CRUD)

Modificación documentos (UPDATE)

- Modificación de un documento:

```
{"_id": 3, "nombre": "pedro", "edad": 30}
```

```
BasicDBObject query = new BasicDBObject();
query.put("_id", 3);

BasicDBObject newDocument = new BasicDBObject();
newDocument.put("nombre", "Mateo");

BasicDBObject updateObject = new BasicDBObject();
updateObject.put("$set", newDocument);

coleccion.updateOne(query, updateObject);
```

```
{"_id": 3, "nombre": "Mateo", "edad": 30}
```

5. UPDATE (CRUD)

Modificación documentos (UPDATE)

- Modificación de muchos documentos:

```
{"_id": 2, "nombre": "erer", "edad": 30}
{"_id": 3, "nombre": "Mateo", "edad": 30}
```

```
BasicDBObject searchQuery = new BasicDBObject();
//searchQuery.append("", "");

BasicDBObject updateQuery = new BasicDBObject();
updateQuery.append("$set", new BasicDBObject().append("edad", "100"));

coleccion.updateMany(searchQuery, updateQuery);
```

```
{"_id": 2, "nombre": "erer", "edad": "100"}
{"_id": 3, "nombre": "Mateo", "edad": "100"}
```

6. DELETE (CRUD)

Borrado de documentos (DELETE)

- Borrado de todos los documentos de una colección usando un objeto BasicDBObject en blanco:

```
BasicDBObject document = new BasicDBObject();
coleccion.deleteMany(document);
```

- Borrado de un documento concreto:

```
coleccion.deleteOne(new Document("_id", id));
```