MongoDB - Java

Instalación

Antes de comenzar a utilizar MongoDB en sus programas Java, debe asegurarse de que tiene el controlador JDBC de MongoDB y Java configurados en la máquina. Puede consultar el tutorial de Java para la instalación de Java en su máquina. Ahora, veamos cómo configurar el controlador JDBC de MongoDB.

Necesitas descargar el jar de la ruta. Descargar mongo.jar . Asegúrese de descargar la última versión de la misma.

Debe incluir el archivo mongo.jar en su classpath.

MongoDB - Java

Conectarse a la base de datos

Para conectar la base de datos, debe especificar el nombre de la base de datos, si la base de datos no existe, MongoDB la crea automáticamente.

A continuación se muestra el fragmento de código para conectarse a la base de datos:

```java
import com.mongodb.client.MongoDatabase;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class ConnectToDB {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());
      System.out.println("Connected to the database successfully");

      // Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");
      System.out.println("Credentials ::"+ credential);
   }
}
```

Al ejecutar, el programa anterior le da el siguiente resultado.

```
Connected to the database successfully
Credentials ::MongoCredential{
   mechanism = null,
   userName = 'sampleUser',
   source = 'myDb',
   password = <hidden>,
   mechanismProperties = {}
}
```

MongoDB - Java

Crear una colección

Para crear una colección, se **utiliza el** método **createCollection ()** de la clase **com.mongodb.client.MongoDatabase** .
A continuación se muestra el fragmento de código para crear una colección:

```java
import com.mongodb.client.MongoDatabase;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class CreatingCollection {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());
      System.out.println("Connected to the database successfully");

      //Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");

      //Creating a collection
      database.createCollection("sampleCollection");
      System.out.println("Collection created successfully");
   }
}
```

Al compilar, el programa anterior te da el siguiente resultado:

```
Connected to the database successfully
Collection created successfully
```

MongoDB - Java

Obtención / Selección de una colección

```java
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;

import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class selectingCollection {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());
      System.out.println("Connected to the database successfully");

      // Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");

      // Creating a collection
      System.out.println("Collection created successfully");

      // Retieving a collection
      MongoCollection<Document> collection = database.getCollection("myCollection");
      System.out.println("Collection myCollection selected successfully");
   }
}
```

Para obtener / seleccionar una colección de la base de datos, se **utiliza el**método **getCollection ()** de la clase **com.mongodb.client.MongoDatabase**. El siguiente es el programa para obtener / seleccionar una colección -

Al compilar, el programa anterior te da el siguiente resultado:

```
Connected to the database successfully
Collection created successfully
Collection myCollection selected successfully
```

MongoDB - Java

Inserte un documento

```java
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;

import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class InsertingDocument {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());
      System.out.println("Connected to the database successfully");

      // Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");

      // Retrieving a collection
      MongoCollection<Document> collection = database.getCollection("sampleCollection");
      System.out.println("Collection sampleCollection selected successfully");

      Document document = new Document("title", "MongoDB")
      .append("id", 1)
      .append("description", "database")
      .append("likes", 100)
      .append("url", "http://www.tutorialspoint.com/mongodb/")
      .append("by", "tutorials point");
      collection.insertOne(document);
      System.out.println("Document inserted successfully");
   }
}
```

Para insertar un documento en MongoDB, se usa el método **insert ()** de la clase **com.mongodb.client.MongoCollection** .
A continuación se muestra el fragmento de código para insertar un documento:

Al compilar, el programa anterior te da el siguiente resultado:

```
Connected to the database successfully
Collection sampleCollection selected successfully
Document inserted successfully
```

MongoDB - Java

Recuperar todos los documentos

```java
import com.mongodb.client.FindIterable;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;

import java.util.Iterator;
import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class RetrievingAllDocuments {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());
      System.out.println("Connected to the database successfully");

      // Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");

      // Retrieving a collection
      MongoCollection<Document> collection = database.getCollection("sampleCollection");
      System.out.println("Collection sampleCollection selected successfully");

      // Getting the iterable object
      FindIterable<Document> iterDoc = collection.find();
      int i = 1;

      // Getting the iterator
      Iterator it = iterDoc.iterator();

      while (it.hasNext()) {
         System.out.println(it.next());
      i++;
      }
   }
}
```

Para seleccionar todos los documentos de la colección, se utiliza el método **find ()** de la clase **com.mongodb.client.MongoCollection** . Este método devuelve un cursor, por lo que necesita iterar este cursor.
A continuación se muestra el programa para seleccionar todos los documentos:

Al compilar, el programa anterior te da el siguiente resultado:

```
Document{{
   _id = 5967745223993a32646baab8,
   title = MongoDB,
   id = 1,
   description = database,
   likes = 100,
   url = http://www.tutorialspoint.com/mongodb/, by = tutorials point
}}
Document{{
   _id = 7452239959673a32646baab8,
   title = RethinkDB,
   id = 2,
   description = database,
   likes = 200,
   url = http://www.tutorialspoint.com/rethinkdb/, by = tutorials point
}}
```

MongoDB - Java

Actualizar documento

```java
import com.mongodb.client.FindIterable;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import com.mongodb.client.model.Filters;
import com.mongodb.client.model.Updates;

import java.util.Iterator;
import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class UpdatingDocuments {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());
      System.out.println("Connected to the database successfully");

      // Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");

      // Retrieving a collection
      MongoCollection<Document> collection = database.getCollection("sampleCollection");
      System.out.println("Collection myCollection selected successfully");

      collection.updateOne(Filters.eq("id", 1), Updates.set("likes", 150));
      System.out.println("Document update successfully...");

      // Retrieving the documents after updation
      // Getting the iterable object
      FindIterable<Document> iterDoc = collection.find();
      int i = 1;

      // Getting the iterator
      Iterator it = iterDoc.iterator();

      while (it.hasNext()) {
         System.out.println(it.next());
         i++;
      }
   }
}
```

Para actualizar un documento de la colección, se **utiliza el** método **updateOne ()** de la clase **com.mongodb.client.MongoCollection** .
A continuación se muestra el programa para seleccionar el primer documento:

Al compilar, el programa anterior te da el siguiente resultado:

```
Document update successfully...
Document {{
   _id = 5967745223993a32646baab8,
   title = MongoDB,
   id = 1,
   description = database,
   likes = 150,
   url = http://www.tutorialspoint.com/mongodb/, by = tutorials point
}}
```

MongoDB - Java

Borrar un documento

```java
import com.mongodb.client.FindIterable;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import com.mongodb.client.model.Filters;

import java.util.Iterator;
import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class DeletingDocuments {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());
      System.out.println("Connected to the database successfully");

      // Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");

      // Retrieving a collection
      MongoCollection<Document> collection = database.getCollection("sampleCollection");
      System.out.println("Collection sampleCollection selected successfully");

      // Deleting the documents
      collection.deleteOne(Filters.eq("id", 1));
      System.out.println("Document deleted successfully...");

      // Retrieving the documents after updation
      // Getting the iterable object
      FindIterable<Document> iterDoc = collection.find();
      int i = 1;

      // Getting the iterator
      Iterator it = iterDoc.iterator();

      while (it.hasNext()) {
         System.out.println("Inserted Document: "+i);
         System.out.println(it.next());
         i++;
      }
   }
}
```

Para eliminar un documento de la colección, debe usar el método **deleteOne ()** de la clase **com.mongodb.client.MongoCollection** . A continuación se muestra el programa para borrar un documento:

Al compilar, el programa anterior te da el siguiente resultado:

```
Connected to the database successfully
Collection sampleCollection selected successfully
Document deleted successfully...
```

MongoDB - Java

Eliminar una colección

```java
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;

import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class DropingCollection {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());
      System.out.println("Connected to the database successfully");

      // Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");

      // Creating a collection
      System.out.println("Collections created successfully");

      // Retieving a collection
      MongoCollection<Document> collection = database.getCollection("sampleCollection");

      // Dropping a Collection
      collection.drop();
      System.out.println("Collection dropped successfully");
   }
}
```

Para eliminar una colección de una base de datos, debe usar el método **drop ()** de la clase **com.mongodb.client.MongoCollection** .
El siguiente es el programa para eliminar una colección -

Al compilar, el programa anterior te da el siguiente resultado:

```
Connected to the database successfully
Collection sampleCollection selected successfully
Collection dropped successfully
```

MongoDB - Java

Listado de todas las colecciones

```java
import com.mongodb.client.MongoDatabase;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;

public class ListOfCollection {

   Run | Debug
   public static void main( String args[] ) {

      // Creating a Mongo client
      MongoClient mongo = new MongoClient( "localhost" , 27017 );

      // Creating Credentials
      MongoCredential credential;
      credential = MongoCredential.createCredential("sampleUser", "myDb",
         "password".toCharArray());

      System.out.println("Connected to the database successfully");

      // Accessing the database
      MongoDatabase database = mongo.getDatabase("myDb");
      System.out.println("Collection created successfully");
      for (String name : database.listCollectionNames()) {
         System.out.println(name);
      }
   }
}
```

Para enumerar todas las colecciones en una base de datos, debe usar elmétodo **listCollectionNames ()** de la clase **com.mongodb.client.MongoDatabase** .
El siguiente es el programa para enumerar todas las colecciones de una base de datos:

Al compilar, el programa anterior te da el siguiente resultado:

```
Connected to the database successfully
Collection created successfully
myCollection
myCollection1
myCollection5
```