Java Project Setup for MongoDB

This tutorial shows how to setup a Java project with a MongoDB in VS Code using Maven.

Prerequisites

For this tutorial, it is assumed, that you successfully completed the following tutorials:

- MongoDB Setup on Atlas
- VS Code Setup for MongoDB

If not, complete these tutorials first.

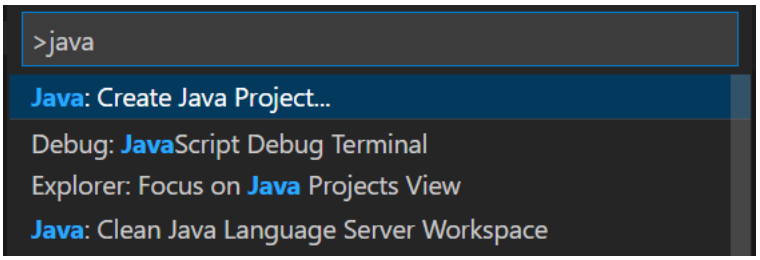
Setting up a New Project

This section describes the creation of a new Java project. This process must be repeated for every new project.

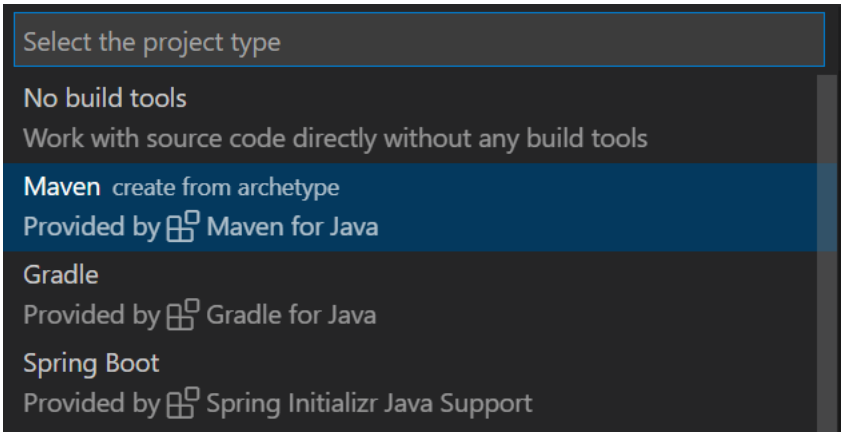
a) Launch VS Code and open a new window (File → New Window).

b) Open the Command Palette (View → Command Palette). Shortcuts:
   - Windows: Ctrl+Shift+P
   - Mac: ⇧⌘P

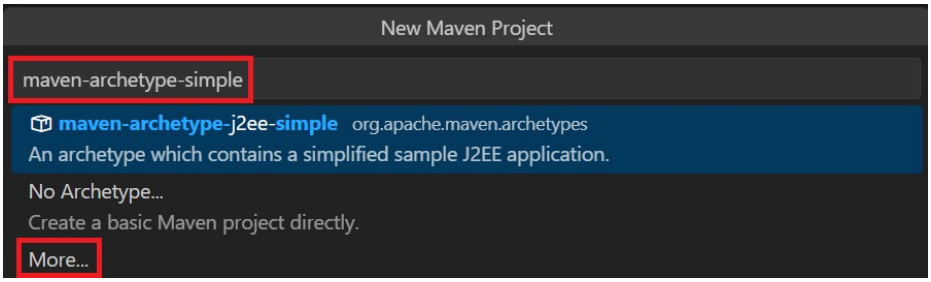
c) Type Java and select “Create Java Project”



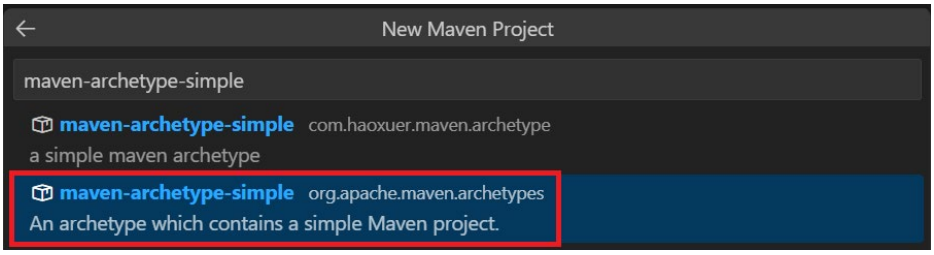
d) Select Maven as project type.



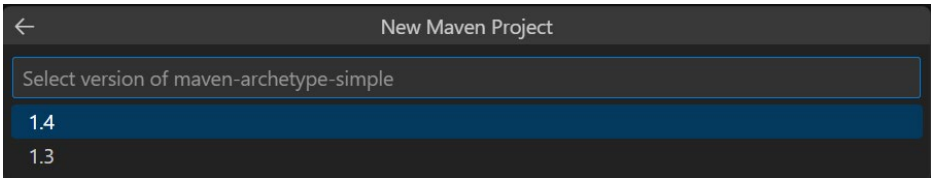
e) Enter “maven-archetype-simple” and select “More”



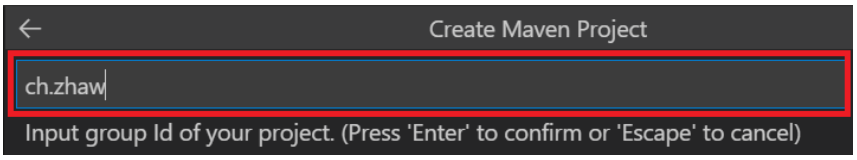
Select the archetype from org.apache.maven.archetypes



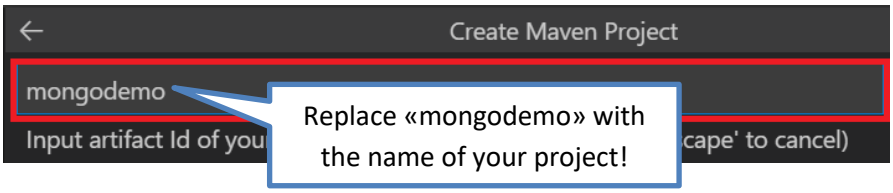
Select version 1.4 of the archetype.



Enter a group Id. This will be used as package name for your project.

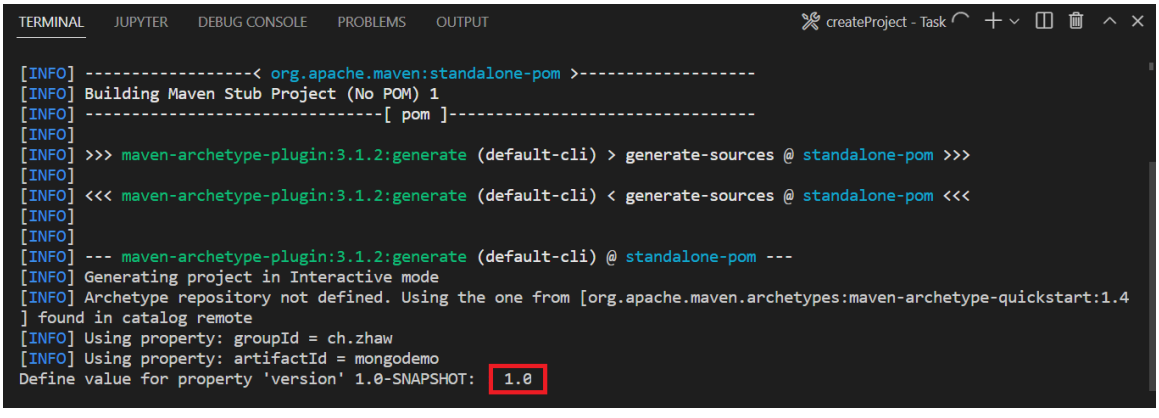


Enter the artifact Id. This is the (unique) name of your project.



f) Next you are prompted to select a folder to store the project. Create a new empty folder on your local drive (not on a cloud drive) and select it. Don’t use umlauts or white space in the path or folder name.

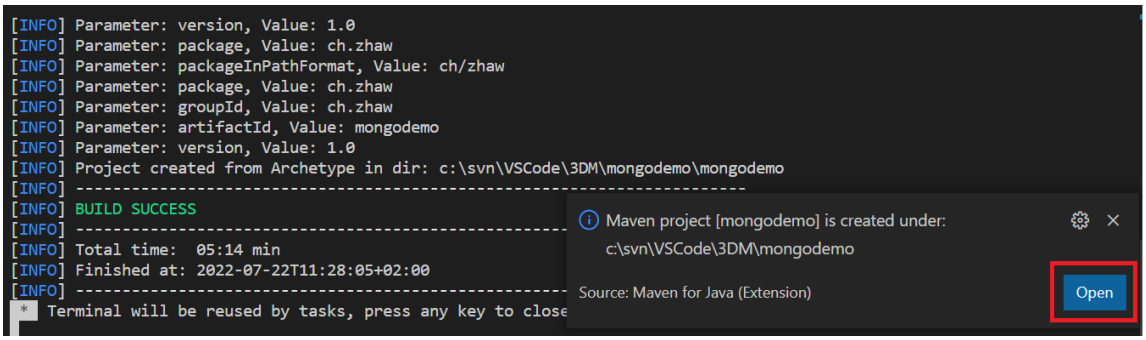
g) After that, project creation starts. Enter a version number when prompted.



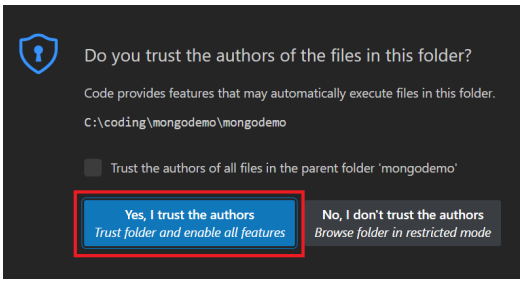
Accept the settings with y and press Enter.

```
[INFO] --- maven-archetype-plugin:3.1.2:generate (default-cli) @ standalone-pom ---
[INFO] Generating project in Interactive mode
[INFO] Archetype repository not defined. Using the one from [org.apache.maven.archetypes:maven-archetype-quickstart:1.4
] found in catalog remote
[INFO] Using property: groupId = ch.zhaw
[INFO] Using property: artifactId = mongodemo
Define value for property 'version' 1.0-SNAPSHOT: : 1.0
[INFO] Using property: package = ch.zhaw
Confirm properties configuration:
groupId: ch.zhaw
artifactId: mongodemo
version: 1.0
package: ch.zhaw
 Y: y
```

Wait until build completes successfully. You can now open the project.



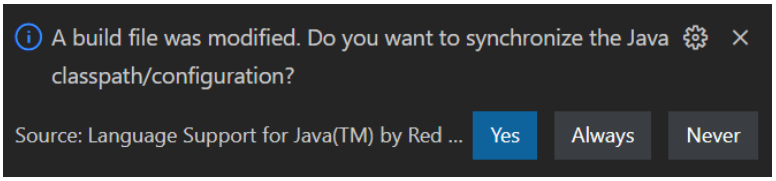
If asked, confirm that you trust the authors.



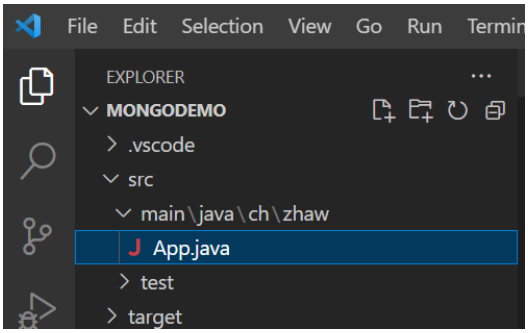
h) Adapt the Compiler-Version (we recommend 21): Open the pom.xml file

```
<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <maven.compiler.source>21</maven.compiler.source>
  <maven.compiler.target>21</maven.compiler.target>
</properties>
```

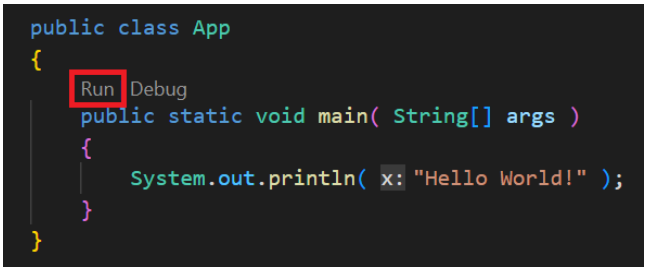
Save the file. VC Code should then ask you to synchronize the classpath. Press Yes or Always.



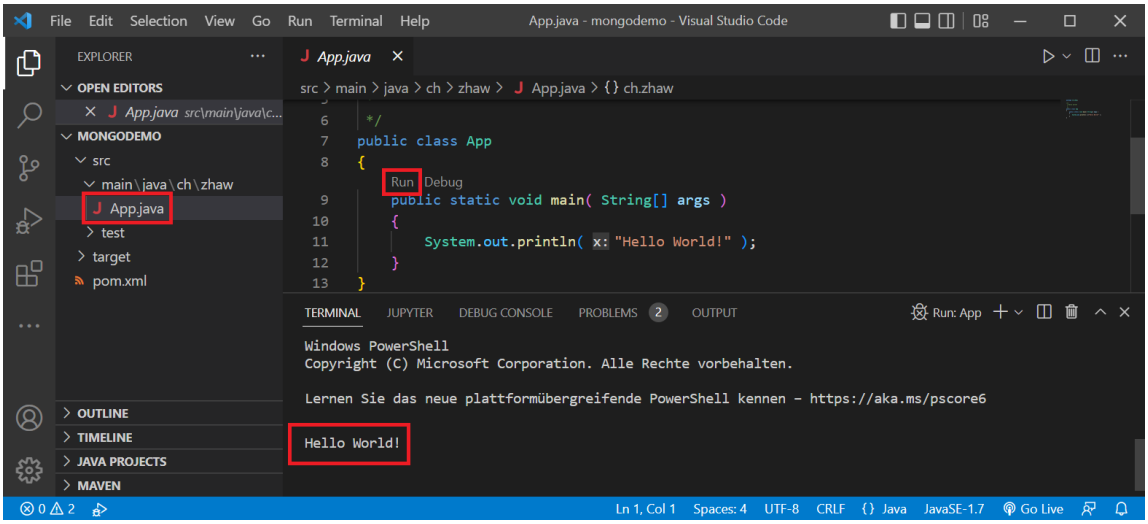
i) Open the java class App.java from the sidebar. Wait until build is completed and the project is opened (status bar).



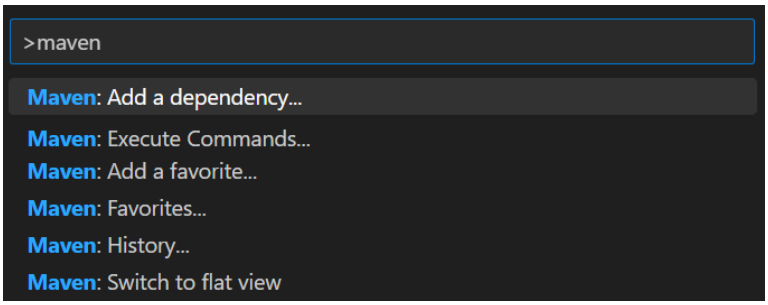
j) Check if you see a run-label above the main Method. If it does not appear after some time, restart VS Code.



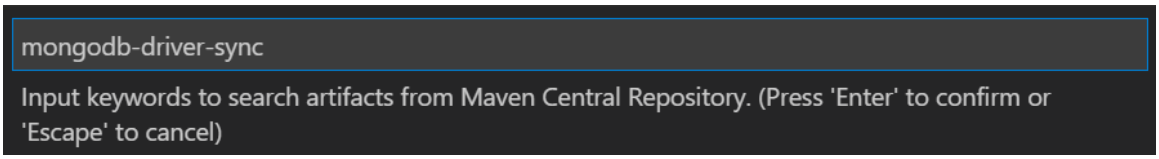
k) Run the application by clicking the run-label above the main-Method. The application should write “Hello World” to the terminal window.



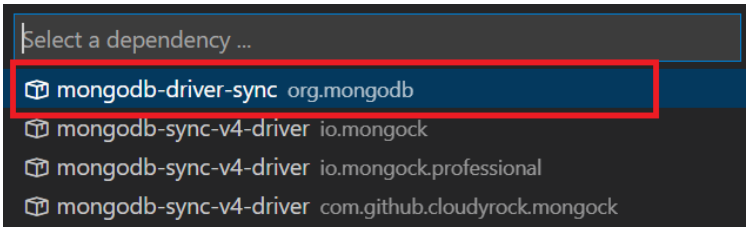
l) Now you can add the required dependencies. Reopen the Command Palette (View → Command Palette or by shortcuts shown in step b). Type **`maven`** and select "Maven: Add dependency".



Type **`mongodb-driver-sync`** and press enter.

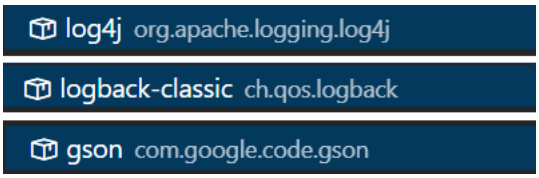


Select the dependency shown below:

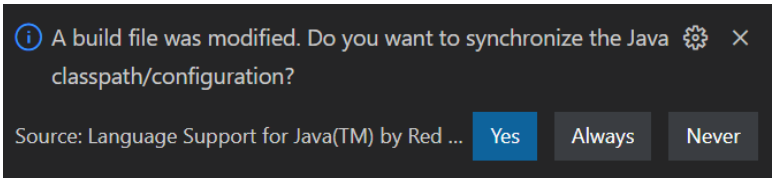


The dependency is added to the projects POM file and will be downloaded automatically.

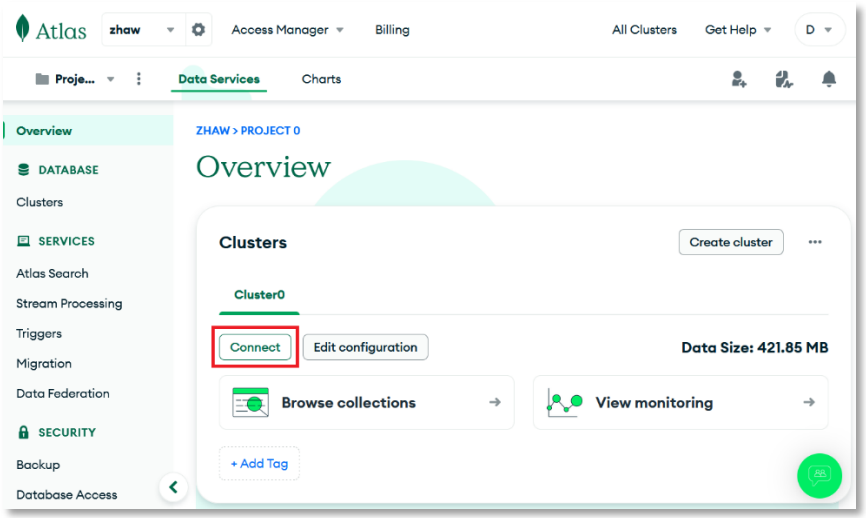
m) We need three more libraries: A logger API (**`org.apache.logging.log4j`**) and a logger implementation (**`logback-classic`** from **`ch.qos.logback`**) and the GSON library (**`com.google.code.gson`**). Add them in the same way. Make sure you pick the versions shown below.



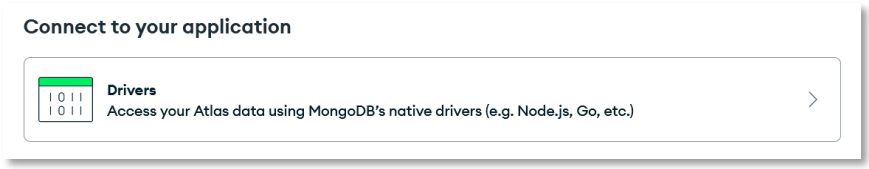
n) Check that all dependencies were added to the pom.xml file in section `<dependencies>`. **Save the pom file**. VC Code should then ask you to synchronize the classpath. Press Yes or Always.



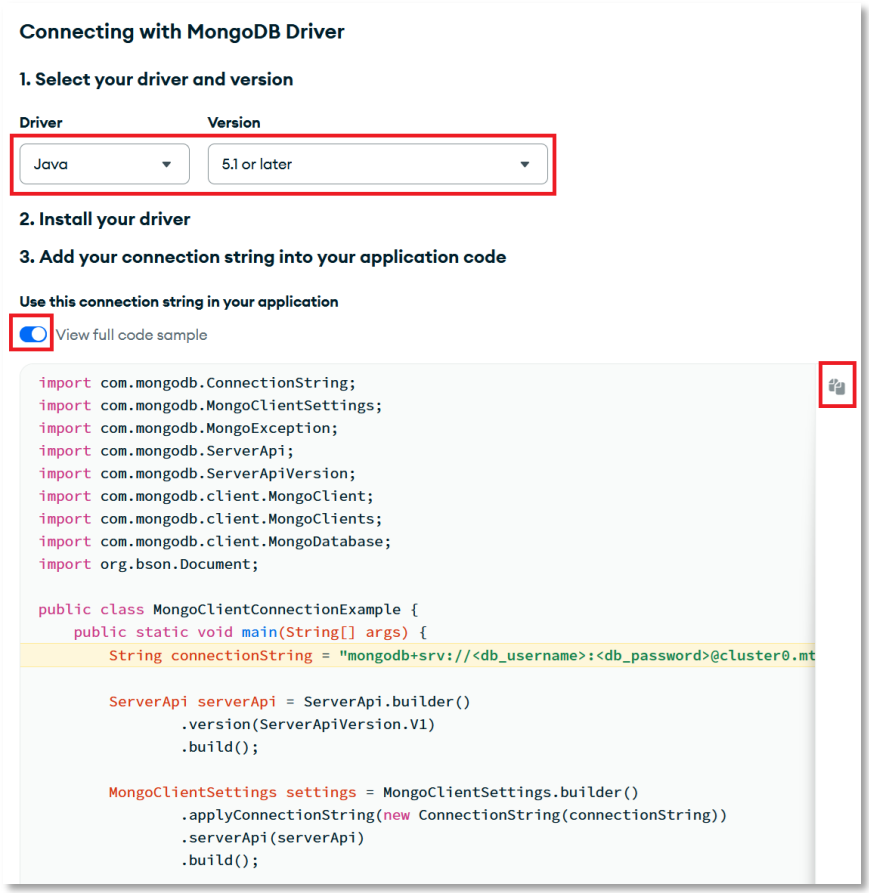
o) Now we have all the libraries required to access our MongoDB. We can now configure the connection. Open https://cloud.mongodb.com in your browser and copy the connection code as shown below.



Select Drivers



Select Java and the right version, enable the full code sample and copy the sample code.



Replace the content of App.java with the code sample. Insert the package name. This is the group id you have selected during project creation. Rename the class to App. Substitute the placeholder for the password in the connection string with the password of the **database user** (this is not your MongoDB login, this is the password you set for the admin user when configuring the database access). Important: you also must remove the two angle brackets.

```
package ch.zhaw;

import com.mongodb.ConnectionString;
import com.mongodb.MongoClientSettings;
import com.mongodb.MongoException;
import com.mongodb.ServerApi;
import com.mongodb.ServerApiVersion;
import com.mongodb.client.MongoClient;
import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoDatabase;
import org.bson.Document;

public class App {
    public static void main(String[] args) {
        String connectionString = "mongodb+srv://admin:<password>@cluster0..";

        ServerApi serverApi = ServerApi.builder()
                .version(ServerApiVersion.V1)
                .build();

        MongoClientSettings settings = MongoClientSettings.builder()
                .applyConnectionString(new ConnectionString(connectionString))
                .serverApi(serverApi)
                .build();

        // Create a new client and connect to the server
        try (MongoClient mongoClient = MongoClients.create(settings)) {
            try {
                // Send a ping to confirm a successful connection
                MongoDatabase database = mongoClient.getDatabase("admin");
                database.runCommand(new Document("ping", 1));
                System.out.println("Pinged your deployment. You successfully ..");
            } catch (MongoException e) {
                e.printStackTrace();
            }
        }
    }
}
```

p) Run your application and inspect the terminal. Among many other things, you should see somewhere hidden in the output that ping run successfully.

```
...
...
09:12:19.107 [main] DEBUG org.mongodb.driver.connection --
Connection created: address=..., driver-generated ID=7
: {"ok": 1}
Pinged your deployment. You successfully connected to MongoDB!
09:12:19.476 [main] DEBUG org.mongodb.driver.connection -- Checkout
started for connection to cluster0-shard-00-
01.mtjb0.mongodb.net:27017
...
...
```

q) **Disable Logging:** the MongoDB driver extensively logs to the console. This might be useful in certain situations but when implementing a command line user interface, logging can be quite disturbing. Add the following code to the import section and the main-Method to turn off logging.

```
package ch.zhaw;

import com.mongodb.ConnectionString;
import com.mongodb.MongoClientSettings;
import com.mongodb.MongoException;
import com.mongodb.ServerApi;
import com.mongodb.ServerApiVersion;
import com.mongodb.client.MongoClient;
import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoDatabase;
import org.bson.Document;
import ch.qos.logback.classic.Level;
import ch.qos.logback.classic.LoggerContext;
import org.slf4j.LoggerFactory;

public class App {
    public static void main(String[] args) {
        // disable logging
        LoggerContext lc = (LoggerContext) LoggerFactory.getILoggerFactory();
        lc.getLogger("org.mongodb.driver").setLevel(Level.OFF);

        String connectionString = "mongodb+srv://admin:...";

        ServerApi serverApi = ServerApi.builder()
                .version(ServerApiVersion.V1)
                .build();
```

Rerun your application and enjoy the clean log!
That's it, you are ready to start your project.