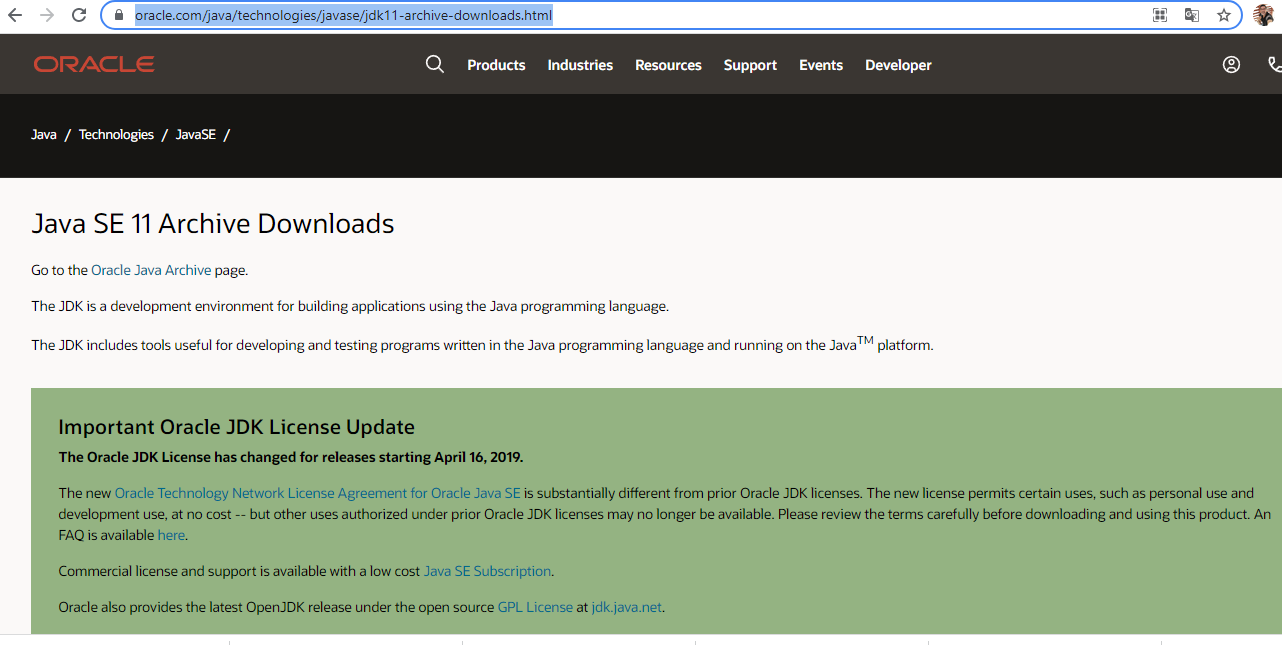
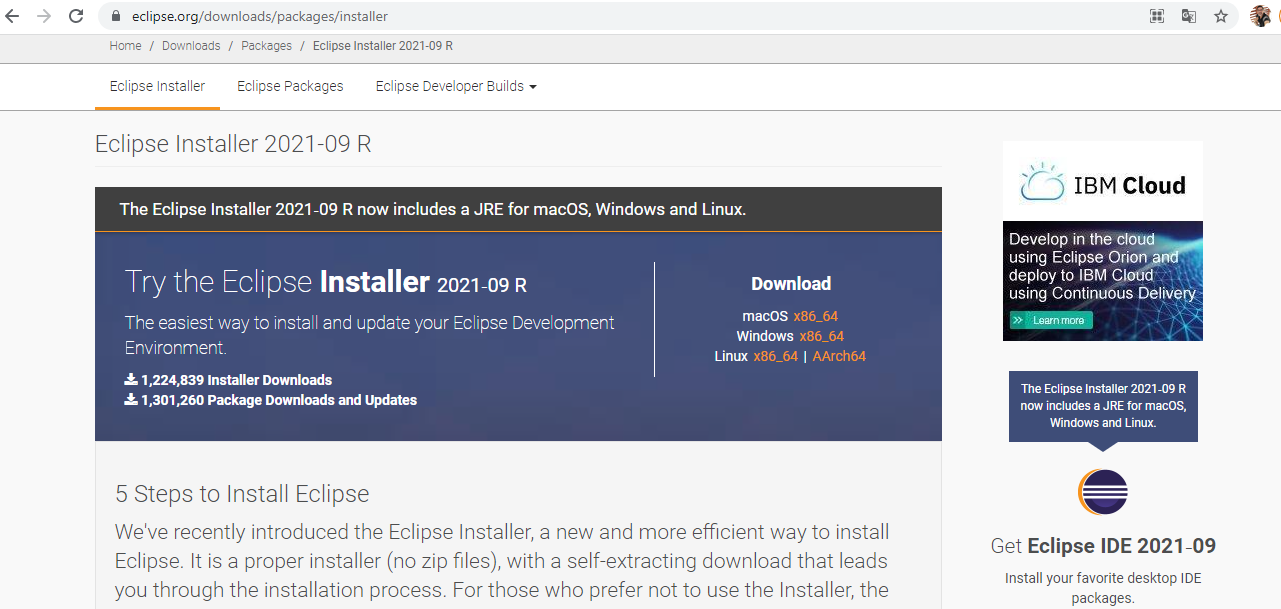
Fundamentos de MongoDB

Practica 5 Programación con Java y MongoDB

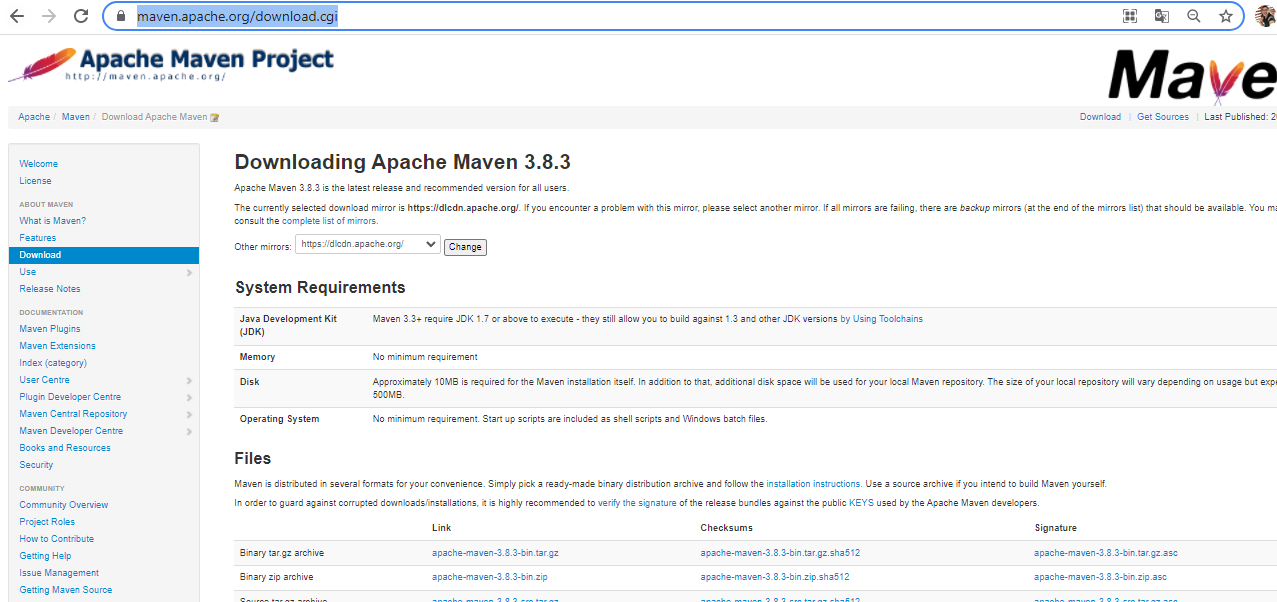
1. Descarga el software Java Development Kit o JDK desde: <https://www.oracle.com/java/technologies/javase/jdk11-archive-downloads.html>



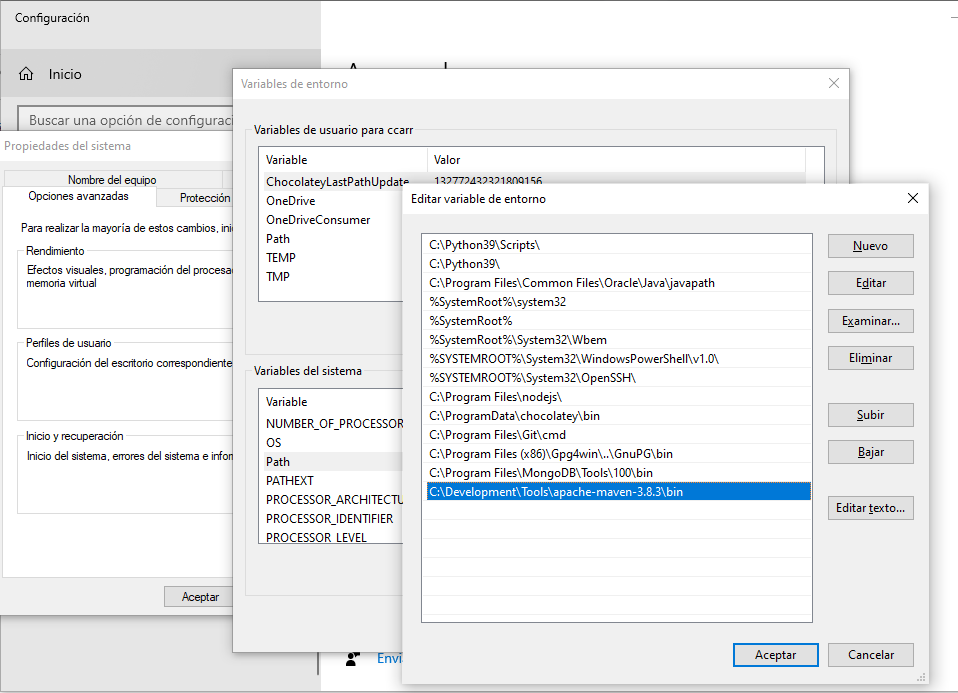
1. Ejecuta el instalador de JDK en Windows, sigue los pasos del asistente de instalación y acepta todas las opciones por defecto.
2. Descarga el instalador para Windows desde: <https://www.eclipse.org/downloads/packages/installer>



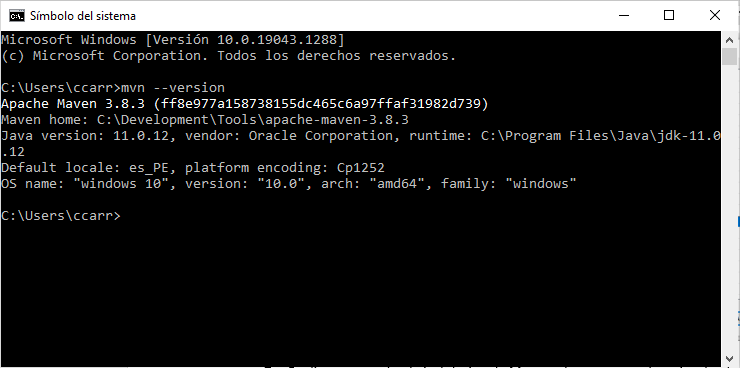
1. Ejecuta el instalador selecciona “**Eclipse for Java Developers**” y sigue los pasos del asistente, acepta las opciones por defecto.
2. Descarga Apache Maven desde aquí: <https://maven.apache.org/download.cgi> y descomprime el archivo zip en **C:\Development\Tools**



1. Modifica la variable de sistema **Path** ya agrega una entrada a: C:\Development\Tools\apache-maven-3.8.3\bin

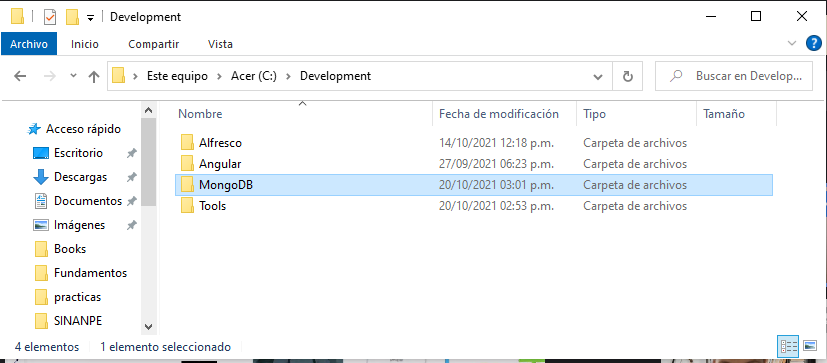


1. Realiza una prueba de instalación de Maven, abre una consola y ejecuta el comando: **mvn –version**

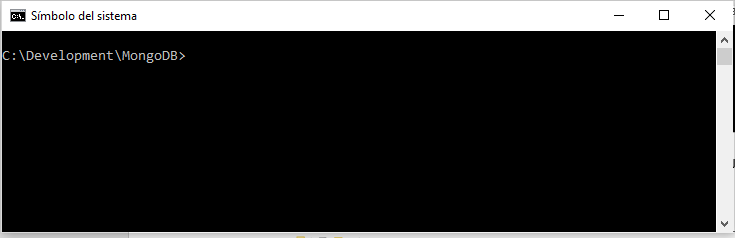


Si obtuviste un resultado parecido a la ventana mostrada la instalación de JDK y Maven ha sido exitosa.

1. Crea la carpeta **C:\Development\MongoDB**

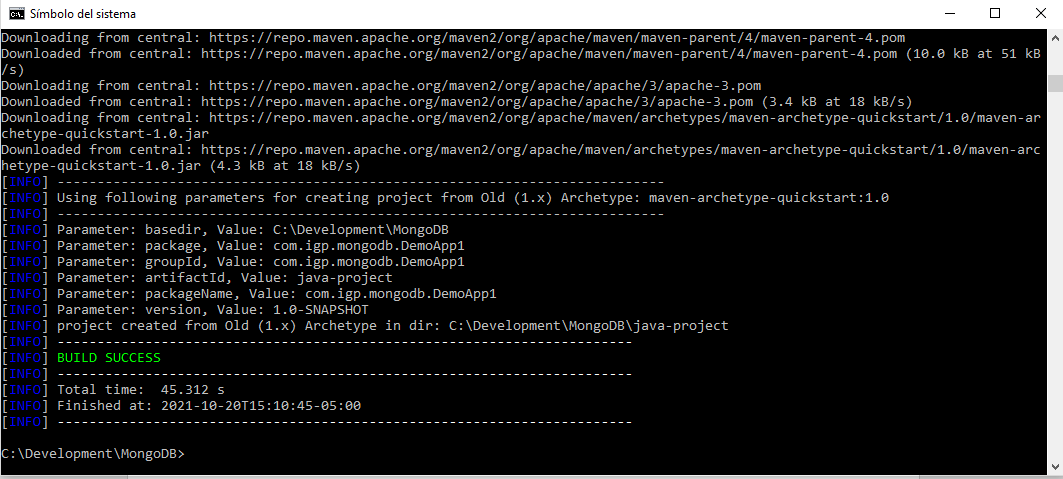


1. Abre una consola y abre la carpeta **c:\Development\MongoDB**



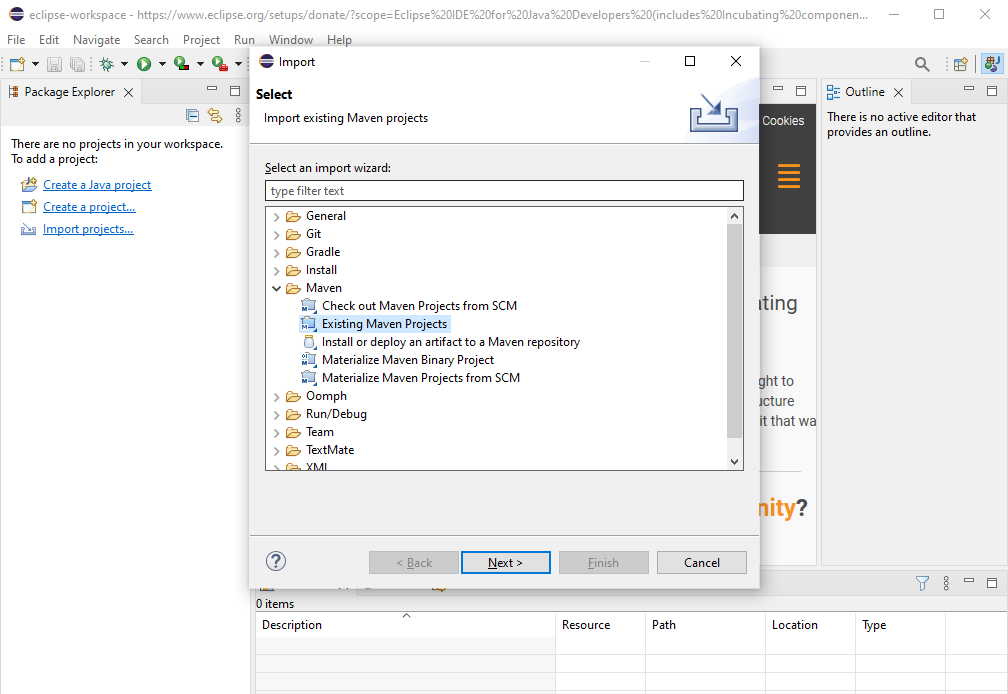
1. En la consola del punto 9, ejecuta el siguiente comando Maven:

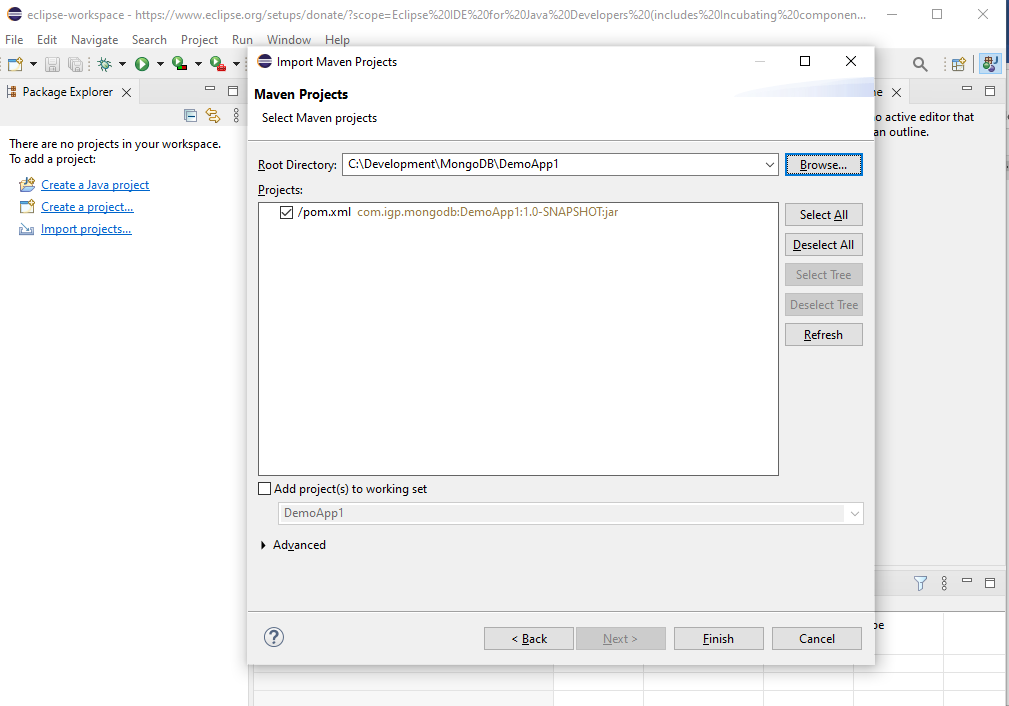
mvn archetype:generate -DgroupId=com.igp.mongodb -DartifactId=DemoApp1 -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

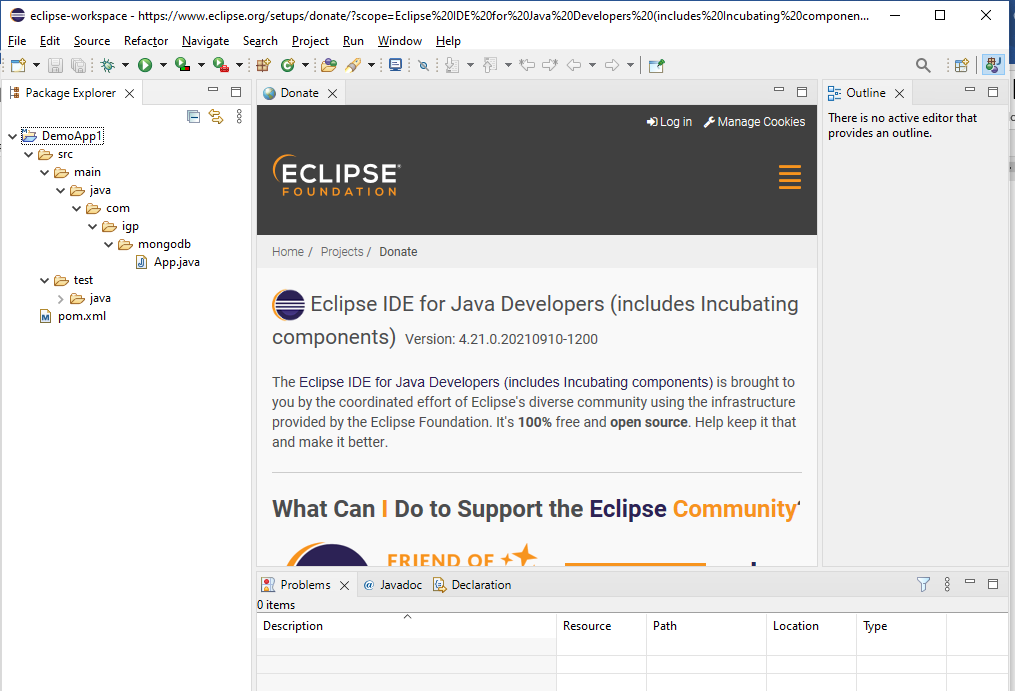


Verifica que el comando se haya ejecutado de manera exitosa. Se ha creado el proyecto javo DemoApp1

1. Abre el IDE Eclipse e importa el proyecto DemoApp1







1. Agrega la dependencia Maven de MongoDB en el archivo pom.xml del proyecto DemoApp1

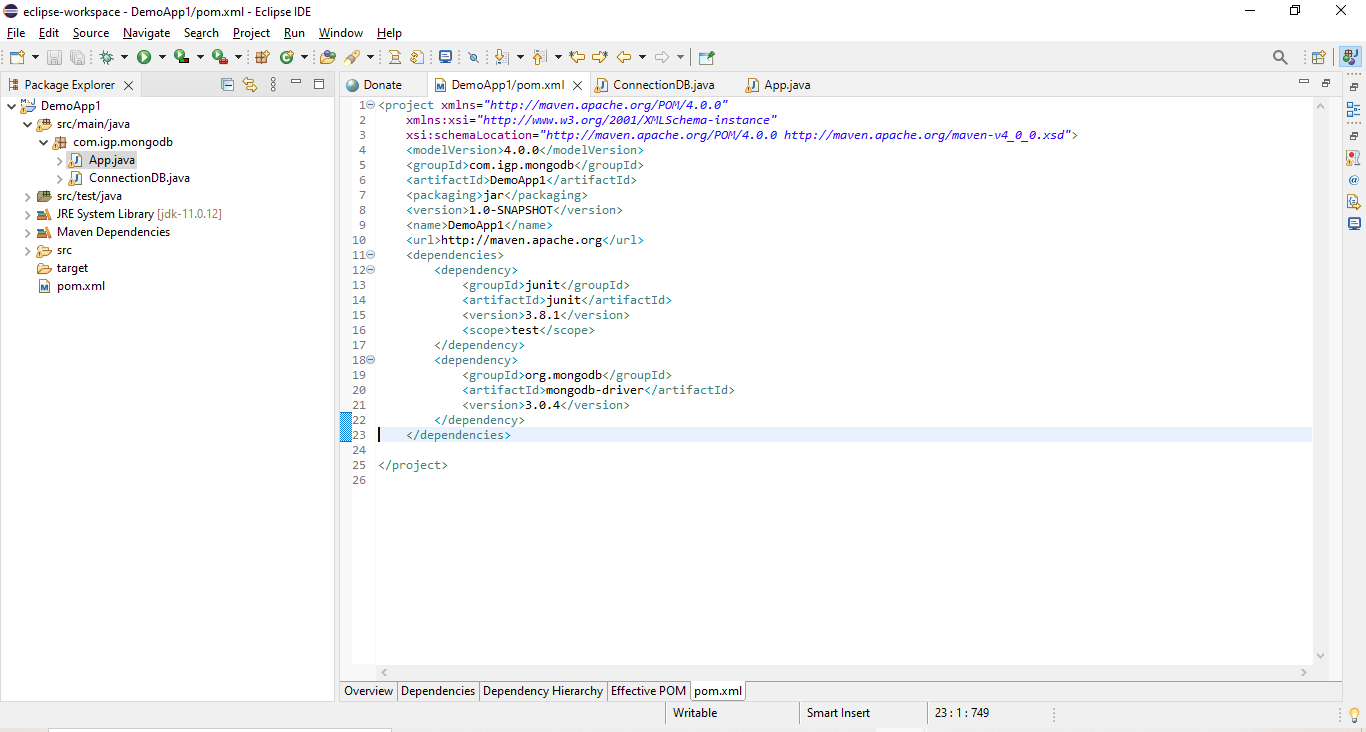
<dependency>

<groupId>org.mongodb</groupId>

<artifactId>mongodb-driver-sync</artifactId>

<version> 3.8.1</version>

</dependency>



1. Crea la clase ConnectionDB.java en el proyecto DemoApp1

**package** com.igp.mongodb;

**import** com.mongodb.MongoClient;

**import** com.mongodb.MongoCredential;

**import** com.mongodb.client.MongoDatabase;

**public** **class** ConnectionDB {

**public** **void** establishConnections()

{

**try** {

MongoClient db

= **new** MongoClient("localhost", 27017);

MongoCredential credential;

credential

= MongoCredential

.*createCredential*(

"GFGUser", "mongodb",

"password".toCharArray());

System.***out***.println(

"Successfully Connected"

+ " to the database");

MongoDatabase database

= db.getDatabase("mongodb");

System.***out***.println("Credentials are: "

+ credential);

}

**catch** (Exception e) {

System.***out***.println(

"Connection establishment failed");

System.***out***.println(e);

}

}

}

1. Modifica la clase App.java del proyecto y ejecuta la clase.

**package** com.igp.mongodb;

/\*\*

\* Hello world!

\*

\*/

**public** **class** App

{

**public** **static** **void** main( String[] args )

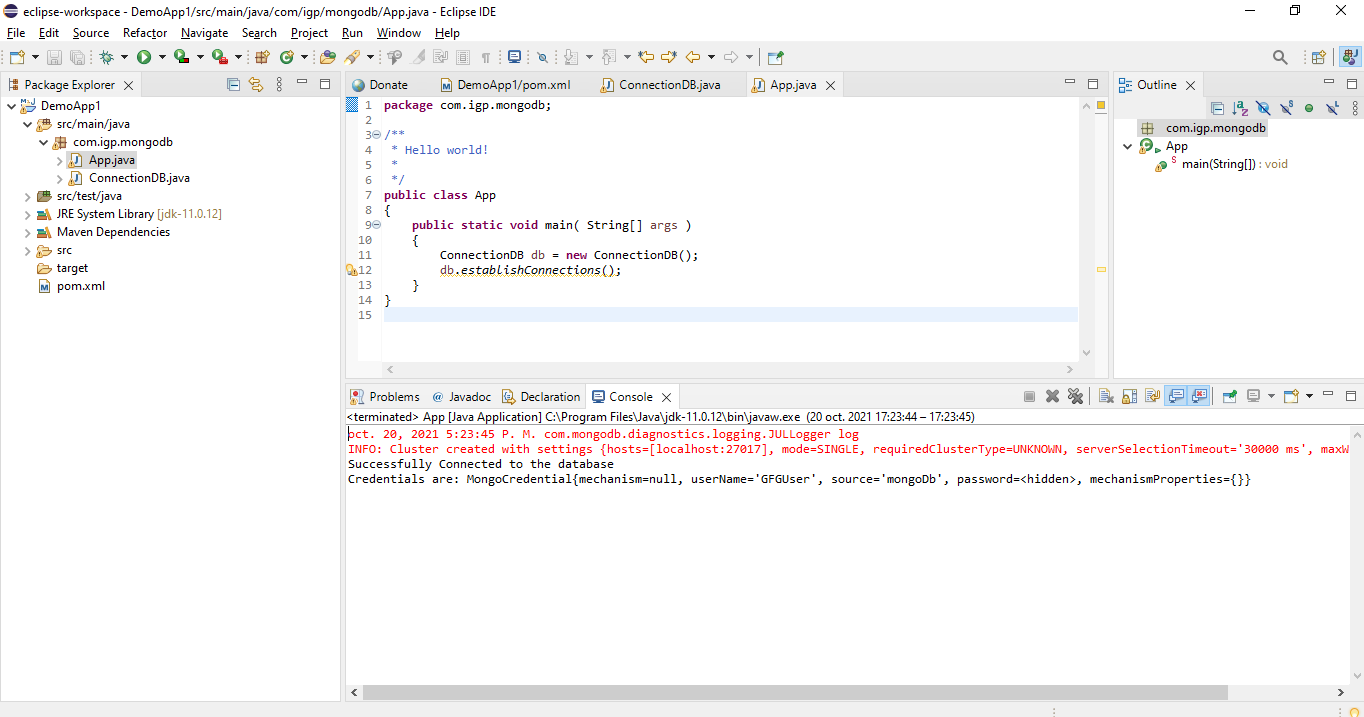
{

ConnectionDB db = **new** ConnectionDB();

db.*establishConnections*();

}

}



1. Crear una clase java que permita crear e insertar datos en una colección.

**SOLUCION**:

**package** com.igp.mongodb;

**import** com.mongodb.client.MongoCollection;

**import** com.mongodb.client.MongoDatabase;

**import** java.util.ArrayList;

**import** java.util.List;

**import** org.bson.Document;

**import** com.mongodb.MongoClient;

**import** com.mongodb.MongoCredential;

**public** **class** MongoDBApp {

**private** MongoClient db=**null**;

**private** MongoDatabase database=**null**;

**public** **void** establishConnections()

{

**try** {

db = **new** MongoClient("localhost", 27017);

MongoCredential credential;

credential

= MongoCredential

.*createCredential*(

"GFGUser", "mongodb",

"password".toCharArray());

System.***out***.println(

"Successfully Connected"

+ " to the database");

database = db.getDatabase("mongoIgp");

System.***out***.println("Credentials are: "

+ credential);

}

**catch** (Exception e) {

System.***out***.println(

"Connection establishment failed");

System.***out***.println(e);

}

}

**public** **void** createCollection(

String collectionName)

{

**try** {

database.createCollection(collectionName);

System.***out***.println(

"Collection created Successfully");

}

**catch** (Exception e) {

System.***out***.println(

"Collection creation failed");

System.***out***.println(e);

}

}

**public** **void** insertADocIntoDb(String collectionName)

{

**try** {

Document document

= **new** Document("title",

"MongoDB")

.append("about",

"Open-Source database");

**this**.getCollection(collectionName).insertOne(document);

System.***out***.println(

"Document inserted Successfully");

}

**catch** (Exception e) {

System.***out***.println(

"Document insertion failed");

System.***out***.println(e);

}

}

**public** **void** insertManyDocsIntoDb(String collectionName)

{

**try** {

Document document

= **new** Document("title", "MongoDB")

.append("about", "Open-Source database");

Document document1

= **new** Document("title", "retrieveDb")

.append("about", "Open-source database");

// Adding the documents into a list

List<Document> dblist

= **new** ArrayList<Document>();

dblist.add(document);

dblist.add(document1);

// Insert the list of documents into DB

**this**.getCollection(collectionName).insertMany(dblist);

System.***out***.println(

"Documents inserted Successfully");

}

**catch** (Exception e) {

System.***out***.println(

"Documents insertion failed");

System.***out***.println(e);

}

}

**public** MongoCollection<Document> getCollection(String collectionName)

{

MongoCollection<Document> collection = **null**;

**try** {

collection = database.getCollection(collectionName);

System.***out***.println(

"Collection retrieved Successfully");

}

**catch** (Exception e) {

System.***out***.println(

"Collection retrieval failed");

System.***out***.println(e);

}

**return** collection;

}

}