



Ejemplo Práctico.

Project

☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ Java ☐ Kotlin ☐ Groovy
☒ Maven

Spring Boot

☐ 3.3.0 (SNAPSHOT) ☐ 3.2.2 (SNAPSHOT) ☒ 3.2.1 ☐ 3.1.8 (SNAPSHOT)
☐ 3.1.7

Project Metadata

Group

Artifact

Name

Description

Package name


Packaging ☐ Jar ☒ War

Java ☐ 21 ☒ 17


Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container. 

Thymeleaf TEMPLATE ENGINES

A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes. 

Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate. 



Configuramos el mysql

- Creamos una cuenta nueva sin privilegios.
- Ya se los daremos a posteriori.
- 2dawa – 2daw2324
- Creamos una base de datos sencilla animaldb, con una tabla Animal:
 - Id
 - Nombre (campo obligatorio, min. 3 caracteres y máx. 15)
 - VidaMedia (campo obligatorio, numérico, valor mínimo 0 y máximo valor 600)
 - Extinto (tipo de dato lógico y obligatorio)



Datos para acceder a la BBDD

- En application.properties
- Ajustamos a nuestros parámetros.
- User:2dawa ... Psw: 2daw2324 ---
-

```
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://${MYSQL_HOST:localhost}:3306/db_example
spring.datasource.username=springuser
spring.datasource.password=ThePassword
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
#spring.jpa.show-sql: true
```



Añadimos la conexión

DB Connection

DB connection name: mialidb_xampp6

Database Properties

Database type: MySQL 5

Database URL: jdbc:mysql://localhost / animaldb

Connection params: ?useSSL=false&allowMultiQueries=true&serverTimezone=UTC

Database user: 2dawa Password: ☐ Show password

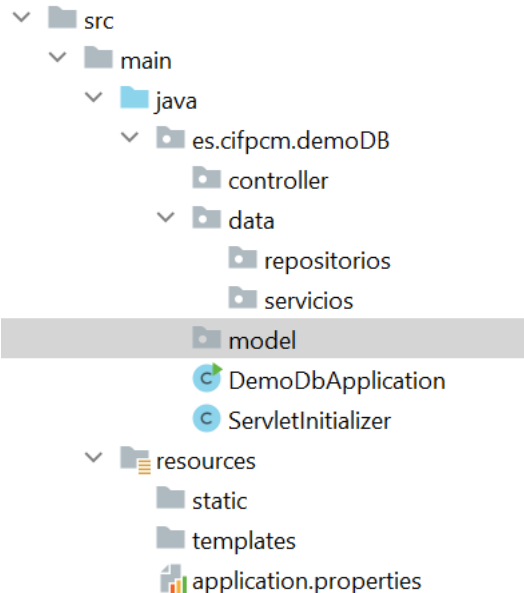
Test Connection

OK Cancel




O hacemos todo a mano

- Controlador AnimalController
- Servicio de Animal --- AnimalService
- Repositorio --- AnimalRepository
- Modelo --- AnimalEntity
- Creamos nuestra distribución de paquetes y después
Generate JPA Entities:





 Set up a database connection ✕

Set up a database connection

Driver class: ...

Driver path: 📁

URL: ↶↷

Preview URL: ↶↷

Host: ↶↷

Port:

Username:

Password:

Database:

Schema(Optional):

☒ Save Password

Switch language: ▼

Next Cancel Reset



Auto generation settings

File extension:

java

Field name:

Remove prefixes: f_

If field name is Java keyword: Field

Inherited parent class:

Browse

Ignored fields:

Entity package:

es.cifpcm.demoDB.model

Entity package parent path:

C:/Users/inmav/Documents/EDUCACION/MIAS/2023-2024/DSW/Proyectos/demoDB/src/main/java

Repository package:

es.cifpcm.demoDB.data.repositorios

Repository package parent path:

C:/Users/inmav/Documents/EDUCACION/MIAS/2023-2024/DSW/Proyectos/demoDB/src/main/java

☒ Generate Repository ☒ Generate Entity ☒ Use Lombok ☒ Serializable

☒ Generate class comment ☒ Generate field comment ☒ Generate method comment ☒ Generate JPA Annotations

☐ Generate Swagger UI comment ☐ Use Java8 date type(eg: LocalDateTime...) ☐ Use Fluid Programming Style

☐ Generate default value ☐ Generate datetime default value ☐ Add Schema name to table name ☒ Use Jakarta EE

Controller Service Mapper XML VO DTO

☒ Generate Controller

Controller package:

es.cifpcm.demoDB.controller

Controller package path:

C:/Users/inmav/Documents/EDUCACION/MIAS/2023-2024/DSW/Proyectos/demoDB/src/main/java

Next

Cancel