

# Ejemplo de Conexión a BD con sql2o en Spring

## Requisitos

- Java 11 o superior

Opcionalmente

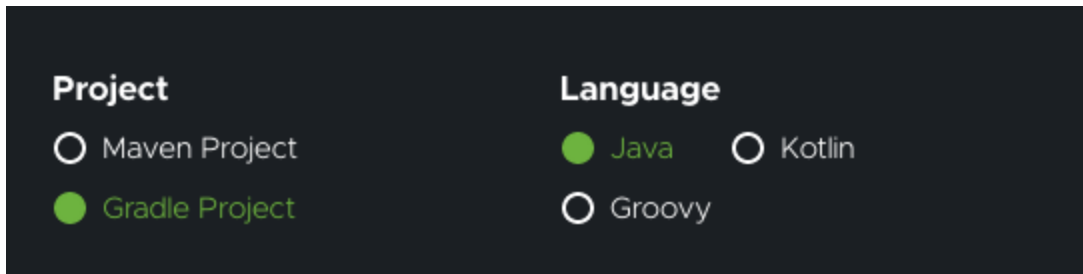
- Maven 3.3 o superior
- Gradle 5.x o superior <https://gradle.org/install/>

## 1. Proyecto Spring

Utilizar base de proyecto desde Spring Initializr <https://start.spring.io/>

### 1.1 Tipo de proyecto

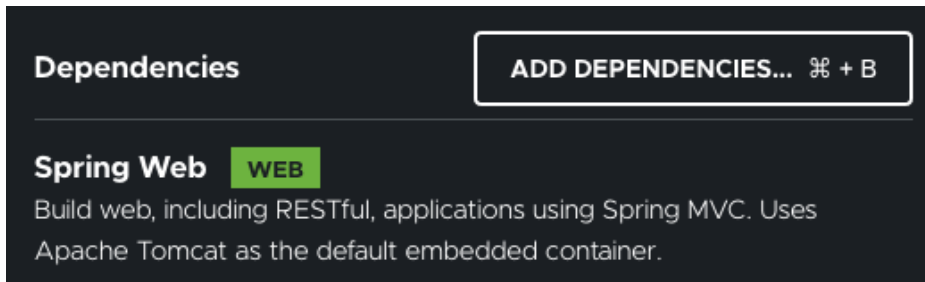
Para este ejemplo se usará Gradle, opcionalmente se puede utilizar Maven.



The screenshot shows the 'Project' and 'Language' selection interface of Spring Initializr. Under 'Project', 'Maven Project' is unselected and 'Gradle Project' is selected with a green circle. Under 'Language', 'Java' is selected with a green circle, while 'Kotlin' and 'Groovy' are unselected with white circles.

### 1.2 Dependencias

Se agrega dependencia Spring Web para aplicaciones REST



The screenshot shows the 'Dependencies' section of Spring Initializr. At the top right is a button labeled 'ADD DEPENDENCIES...' with a plus icon. Below, 'Spring Web' is listed with a green 'WEB' tag. The description for 'Spring Web' reads: 'Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.'

**Spring Boot**

☐ 2.3.0 RC1   ☐ 2.3.0 (SNAPSHOT)   ☐ 2.2.8 (SNAPSHOT)  
☒ 2.2.7   ☐ 2.1.15 (SNAPSHOT)   ☐ 2.1.14

**Project Metadata**

Group	cl.tbd
Artifact	ejemplo1
Name	ejemplo1
Description	Ejemplo 1 - TBD
Package name	cl.tbd.ejemplo1
Packaging	<input checked="" type="radio"/> Jar <input type="radio"/> War
Java	<input type="radio"/> 14 <input type="radio"/> 11 <input checked="" type="radio"/> 8

### 1.3 Probando el servicio

Agregamos el archivo TestService.java a paquete services:

```
package cl.tbd.ejemplo1.services;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class TestService {

    @GetMapping("/hello")
    public String HelloWorld(){
        return "Hello World";
    }
}
```

Ejecutamos en la consola  
./gradlew bootRun

Y en localhost:8080/hello debe retornar "Hello World"

```
@PostMapping("/hello")  
    public String answer(@RequestBody String message) {  
        return "hola "+message + "!";  
    }  
}
```

## 1.4 Ejecutar el proyecto

Para ejecutar en modo desarrollo con Gradle

```
./gradlew bootRun
```

Para generar el archivo jar

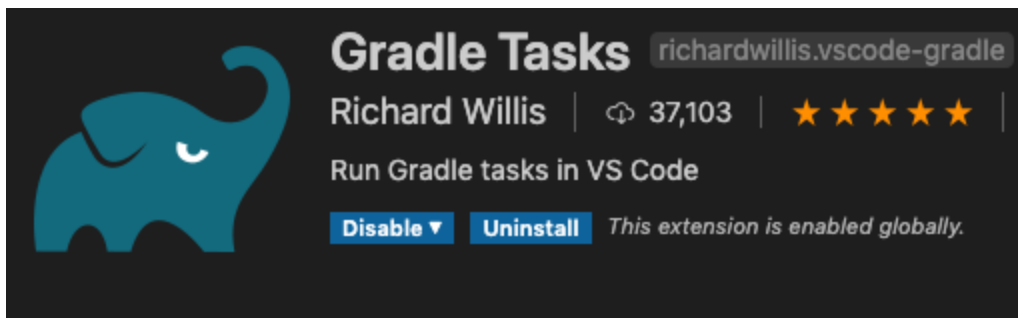
```
./gradlew build
```

Finalmente ejecutar con

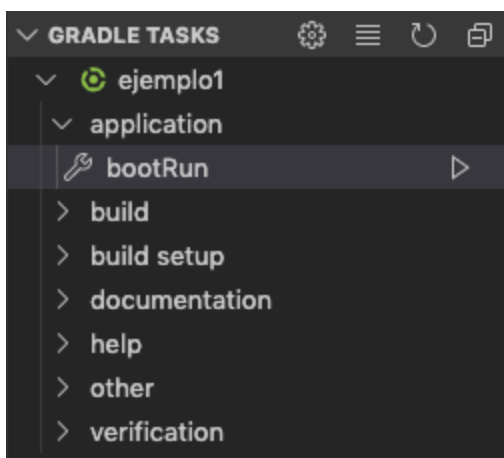
```
java -jar build/libs/[proyecto].jar
```

## 1.5 Configuración para VSCode

Si se quiere usar VSCode, se recomienda instalar un plugin para tareas de Gradle



Permite tener acceso directo a las tareas:



## 2. Conexión a BD con sql2o

### 2.1 Agregar dependencias en build.gradle

```
compile group: 'org.sql2o', name: 'sql2o', version: '1.6.0'  
compile 'org.postgresql:postgresql:42.2.5'
```

### 2.2 Definir un contexto de conexión

Por ejemplo archivo DatabaseContext.java en el paquete repositories.

```
package cl.tbd.ejemplo1.repositories;  
  
@Configuration  
public class DatabaseContext {  
    //Definir url de la BD, usuario y password  
    //Ejemplo: jdbc:postgresql://127.0.0.1:5432/postgres, usuario, password  
    @Bean  
    public Sql2o sql2o(){  
        return new Sql2o("", "", "");  
    }  
}
```

### 2.3 Definir modelos

```
package cl.tbd.ejemplo1.models;  
  
public class Dog {  
    private Long id;  
    private String name;  
  
    //... Getters y setters  
}
```

## 2.4 Establecer interfaz e implementación de repositorios

### DogRepository.java

```
package cl.tbd.ejemplo1.repositories;

public interface DogRepository {
    public int countDogs();
}
```

### DorRepositoryImp.java

```
package cl.tbd.ejemplo1.repositories;
@Repository
public class DogRepositoryImp implements DogRepository {

    @Autowired
    private Sql2o sql2o;

    @Override
    public int countDogs() {
        int total = 0;
        String sql = "SELECT COUNT(*) FROM dog"
        try(Connection conn = sql2o.open()){
            total = conn.createQuery(sql).executeScalar(Integer.class);
        }
        return total;
    }
}
```

## 2.5 Llamar a repositorios desde servicio

```
package cl.tbd.ejemplo1.services;

@RestController
public class DogService {

    private final DogRepository dogRepository;
    DogService(DogRepository dogRepository){
        this.dogRepository = dogRepository;
    }

    @GetMapping("/dogs/count")
    public String countDogs(){
        int total = dogRepository.countDogs();
        return String.format("Tienes %s perros!!", total);
    }
}
```

## 2.6 Ejecutar el proyecto y probar la conexión

