



Objetivos de la sesión:

- Aprender a **crear, configurar, ejecutar y depurar proyectos Spring Boot en Visual Studio**.
- **Acceder a Bases de Datos** desde nuestro proyecto Spring Boot.
- Configurar una **Base de Datos H2**.
- Acceder al **gestor de base de datos H2** a través del navegador.
- Crear **entidades JPA** asociadas a una tabla de una Base de Datos.
- **Inicializar** las tablas de una Base de datos H2 al arrancar la webapp.
- Implementar **repositorios** para cada entidad JPA.
- Aprender a crear una **aplicación que funcione por linea de comandos en Spring Boot**.



UD 3: Bases de datos y servicios REST

2.- Visual Studio Code

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



¿Como que Visual Studio Code?

No es un error, en las siguientes páginas vamos a **configurar e importar el proyecto Spring Boot de la UD2 de Eclipse a VSCode.**

¡¡Ahora que ya nos habíamos encariñado /desesperado con el Eclipse!!

PD: Esta práctica puede parece larga, pero es muy sencilla...

UD 3: Bases de datos y servicios REST

2.- Visual Studio Code

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



Configuración de Visual Studio Code:

Para trabajar desde VSCode primero debemos instalar el paquete “Spring Boot Extension Pack”. Para ello en el **menú superior (Ctrl+Shift+X)** y buscamos spring boot. Instalar “Spring Boot Extension Pack”:

The screenshot shows the Visual Studio Code interface with the 'Extension: Spring Boot Extension Pack - Visual Studio Code' window open. The left sidebar displays the 'EXTENSIONES: MARKETPLACE' view with a search for 'spring boot'. The main area shows the 'Spring Boot Extension Pack' by Pivotal, version v0.2.0, with a description: 'A collection of extensions for developing Spring Boot applications'. Below this, there are links to 'Spring Initializr Java Support' and 'Spring Boot Tools'. The right sidebar shows 'Categorías' (Programming Languages, Linters, Extension Packs) and 'Recursos de extensión' (Marketplace, Repositorio, Pivotal). At the bottom, there is a section for 'Más información' (Publicado, Última versión, Identificador) and a note that the pack is also known as 'Spring Tools 4 for Visual Studio Code'.

UD 3: Bases de datos y servicios REST

2.- Visual Studio Code

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



También debemos instalar la extensión “Lombok”:

The screenshot shows the Visual Studio Code interface with the 'Lombok Annotations Support for VS Code' extension page open. The left sidebar shows the 'EXTENSIONES: MARKETPLACE' view with a search for 'lombok'. The main panel displays the extension details for 'Lombok Annotations Support for VS Code' by Microsoft, version 1.1.0. The extension has 611,502 downloads and a 4.5-star rating. The description states: 'Refactor code with Lombok annotations, or remove Lombok annotations with actual methods.' The 'vscodelombok' repository is also shown, with a warning that starting from 1.8.0, the 'Language Support for Java(TM) by RedHat' extension has built-in support for Lombok, and the embedded 'lombok.jar' in the 'vscodelombok' extension will be deprecated. The 'Overview' section provides a link to the marketplace item and a description of the extension as a lightweight tool for refactoring code with Lombok annotations.

UD 3: Bases de datos y servicios REST

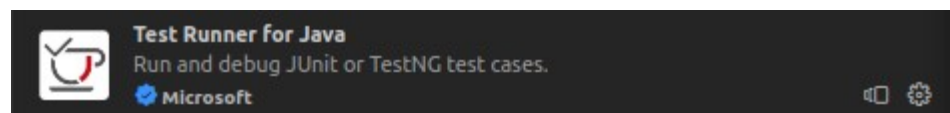
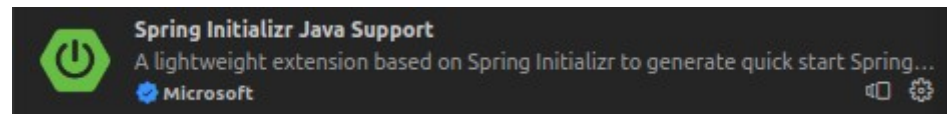
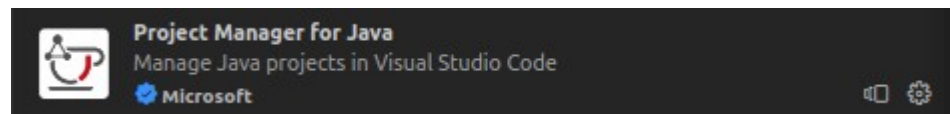
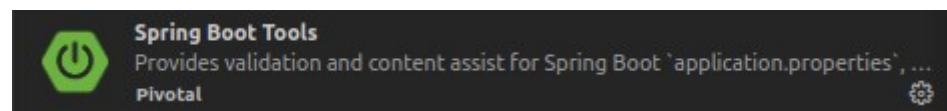
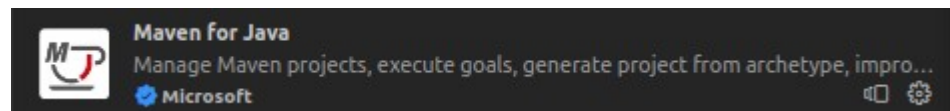
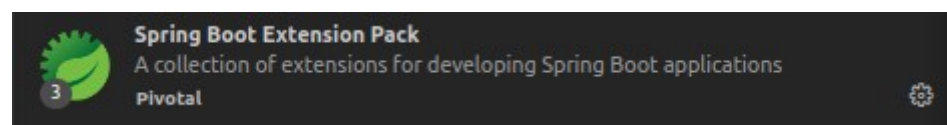
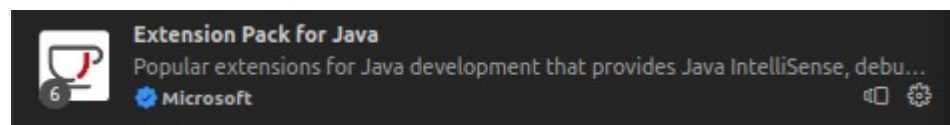
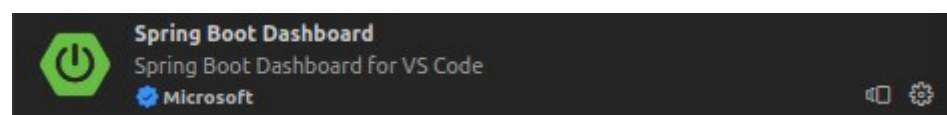
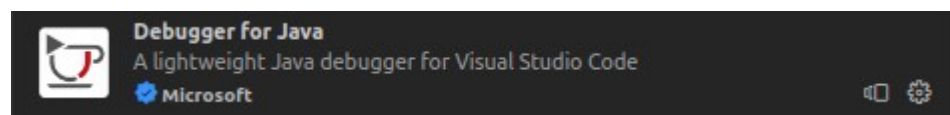
2.- Visual Studio Code

Desarrollo Web en Entorno Servidor - Jose Ramon.profesor@gmail.com



Si no tenemos las siguientes extensiones añadirlas por orden: “**Extension Pack for Java**”, “**Test runner for Java**”, “**Maven for Java**”, “**Tomcat**” y “**Project Manager for Java**”.

Al final como mínimo debemos tener instaladas las siguientes extensiones:
(Algunas se han autoinstalado al instalar las anteriores)



Puede que necesitemos salir del VSCode y entrar para que la lista de extensiones que acabamos de instalar en las páginas anteriores se vean.



UD 3: Bases de datos y servicios REST

2.- Visual Studio Code

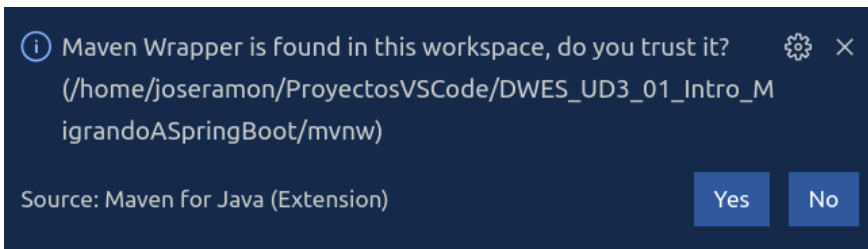
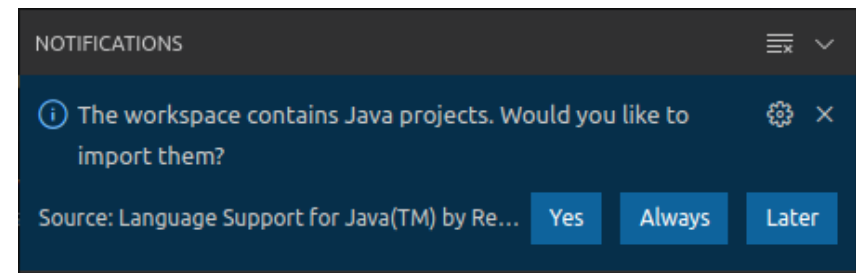
Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



Importar un proyecto a Visual Studio Code:

Antes de crear un nuevo proyecto vamos a **importar** el proyecto de la Ud2 **convertido a Spring Boot**. Creamos una **carpeta nueva** “**ProyectosVSCode**” y copiamos el proyecto entero a esa carpeta para no estropear nada...

Desde VSCode abrimos la carpeta (File\Open Folder\) y nos preguntará si queremos importar el proyecto e Incorporarlo a “Java Projects” : **YES**



Si nos preguntará si confiamos en el maven detectado: **YES**



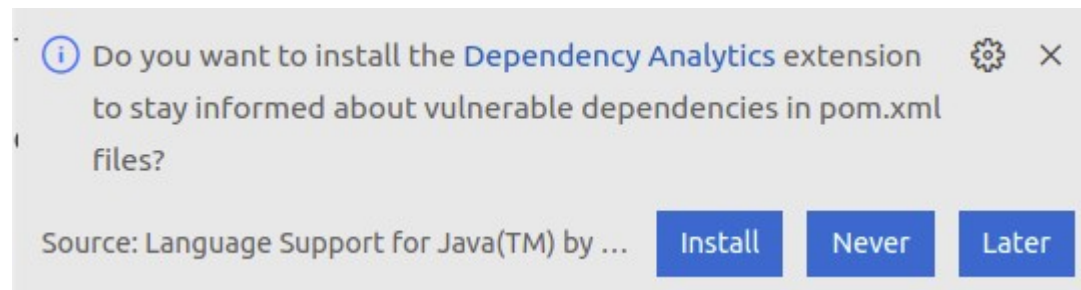
UD 3: Bases de datos y servicios REST

2.- Visual Studio Code

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



Abrimos el fichero pom.xml y dependiendo de la versión del VSCode puede que nos pregunte si queremos instalar el analizador de dependencias para detectar vulnerabilidades. Le confirmamos que queremos instalarlo:



```
pom.xml x
pom.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>2.6.2</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>edu.profesor.joseramon</groupId>
12  <artifactId>dwesUd3WebAppSpringBoot</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <packaging>war</packaging>
15  <name>dwesUd3WebAppSpringBoot</name>
```




UD 3: Bases de datos y servicios REST

2.- Visual Studio Code

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



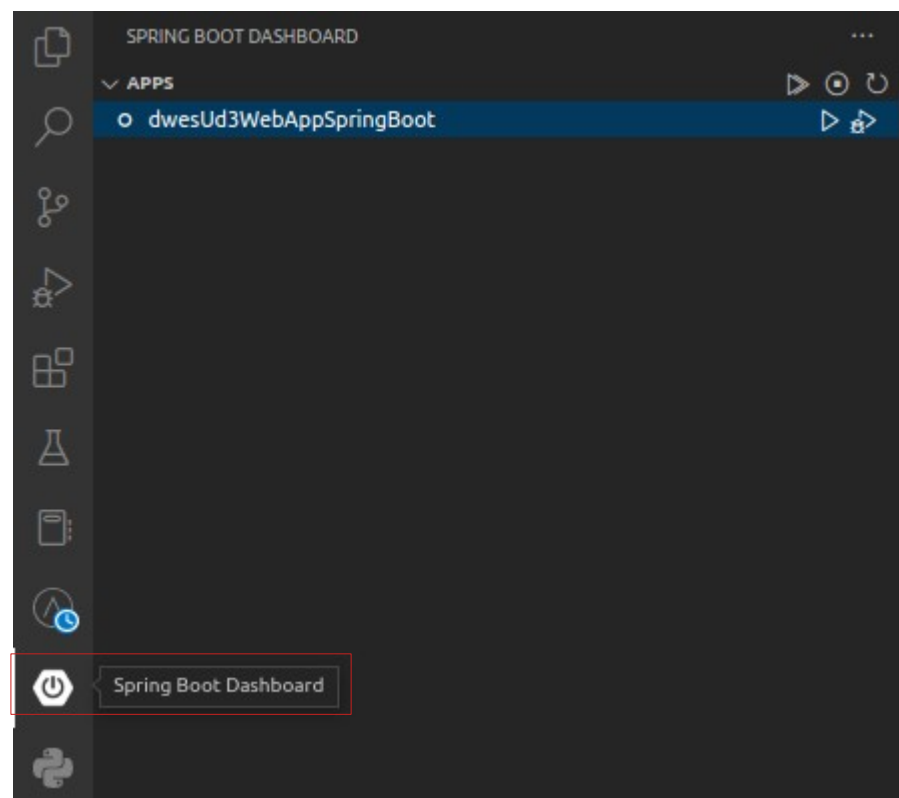
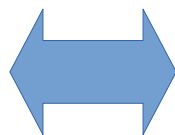
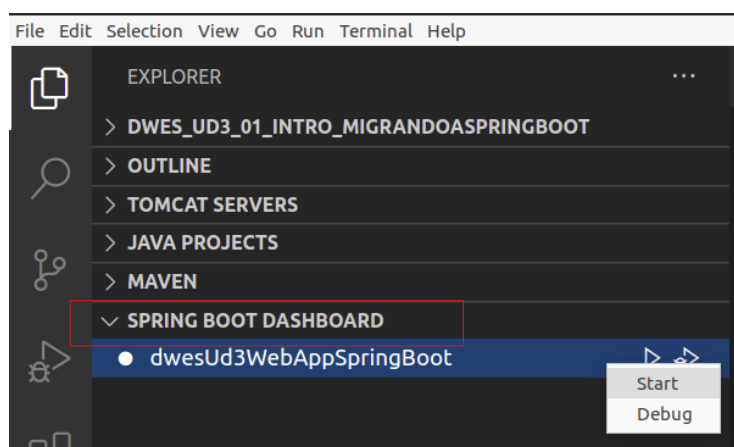
Antes de ejecutar nuestra aplicación, si en la práctica anterior no hemos añadido las 2 dependencias para que se ejecute la aplicación de manera independiente (sin IDE) deberemos hacerlo, porque en VSCode hace falta siempre:

```
pom.xml x
pom.xml
104     <groupId>org.mapstruct</groupId>
105     <artifactId>mapstruct</artifactId>
106     <version>${org.mapstruct.version}</version>
107     <scope>compile</scope>
108 </dependency>
109 <!-- Arrancar la webapp con Tomcat embebido -->
110 <!-- Need this to compile JSP -->
111 <dependency>
112     <groupId>org.apache.tomcat.embed</groupId>
113     <artifactId>tomcat-embed-jasper</artifactId>
114 </dependency>
115
116 <dependency>
117     <groupId>org.eclipse.jdt.core.compiler</groupId>
118     <artifactId>ecj</artifactId>
119     <version>4.6.1</version>
120 </dependency>
121 </dependencies>
```

Si intentamos realizar alguna modificación, vemos que **VSCode autocompleta incluso el fichero pom.xml**. Echabamos de menos el autocompletado de VSCode...

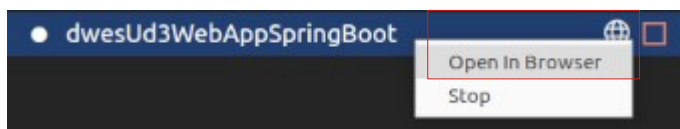


Para lanzar nuestra aplicación **1º** deberemos acceder al **Spring-Boot Dashboard** y dependiendo de nuestra versión de plugin lo tendremos como una subopción debajo de Maven o con una sección aparte en el menú lateral:



2º Pulsamos en **Start o Debug** dependiendo de lo que queramos hacer y ya tenemos nuestra web en marcha.

3º Podemos verla al pulsar botón derecho y "Open in Browser" :





Debemos darnos cuenta de **3 detalles**:

1º Al igual que pasaría si ejecutásemos nuestra aplicación en solitario **la ruta ya no contiene el nombre de la aplicación porque el Tomcat esta embebido** y solo hay una aplicación, esto significa que tenemos nuestra aplicación corriendo en la raíz de dicha url:

Introduïx l'usuari:

Introduïx la contrasenya:

Login

2º Podemos **depurar como en Eclipse** viendo los valores de las variables y expresiones (en VSCode sección “Watch”).

3º Si estamos realizando cambios y modificamos alguna clase **no hace falta recompilar la aplicación (maven clean, update e install) para ver los cambios** porque la dependencia “spring-boot-devtools” declarada en el pom.xml nos recarga la aplicación automáticamente cuando hay cambios.



Nuestra primera aplicación Spring Boot con acceso a datos con JPA:

Ahora vamos a ver como podemos acceder a una base de datos desde nuestra aplicación. Por claridad, vamos a crear una aplicación nueva Spring Boot desde Visual Studio para la explicación.



¿ Por donde empezamos ?



1º opción: Acceder a <http://start.spring.io> para crear el proyecto maven de una manera rápida escogiendo SOLO la dependencia JPA, H2 y Web. **No vamos a escoger esta, sino la opción 2.**



Project

☐ Gradle - Groovy

☐ Gradle - Kotlin ☒ Maven

Language

☒ Java ☐ Kotlin

☐ Groovy

Spring Boot

☐ 3.0.2 (SNAPSHOT) ☐ 3.0.1 ☐ 2.7.8 (SNAPSHOT) ☒ 2.7.7

Project Metadata

Group

Artifact

Name

Description

Package name

Packaging ☒ Jar ☐ War

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Data JPA SQL

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

H2 Database SQL

Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.

Spring Web WEB

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



GENERATE CTRL + G

EXPLORE CTRL + SPACE

SHARE...



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



2º opción : Esta es la opción que escogeremos para crear nuestra primera App “dwes_ud3_primer_jpa” directamente desde **VSCode**:

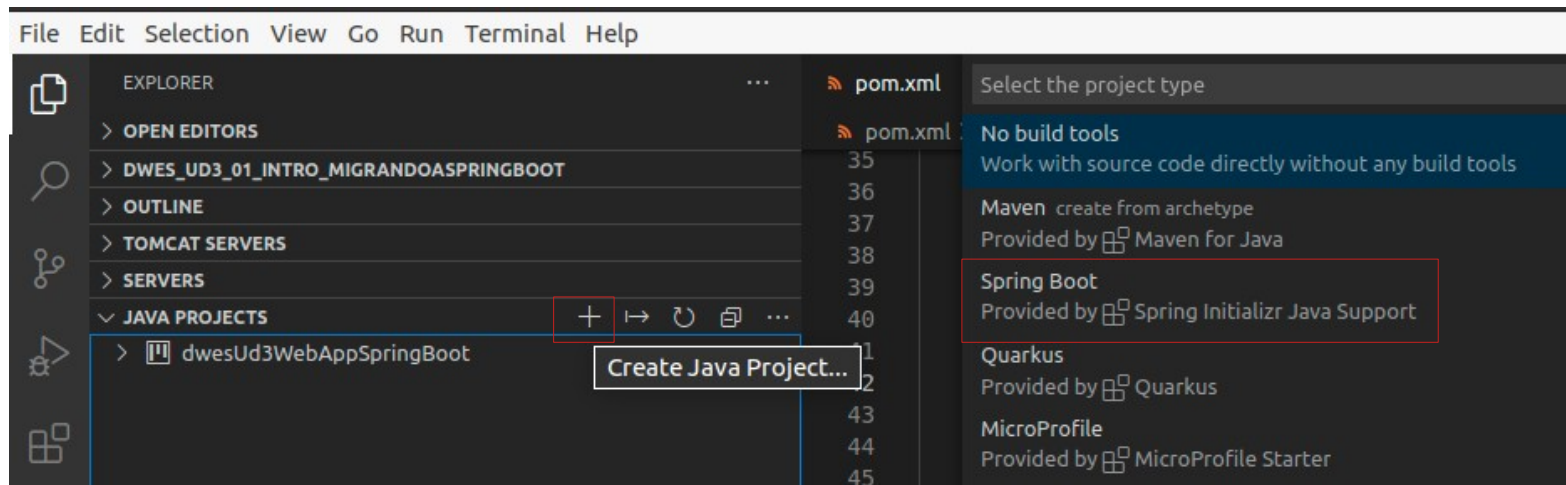
Para ello seguiremos estos **pasos** en las siguientes páginas:

- 1º Crear el proyecto Spring Boot
- 2º Configurar la Base de Datos
- 3º Crear las entidades JPA
- 4º Inicializar con datos la BD
- 5º Crear el repositorio para cada entidad JPA
- 6º Configurar la App para que se pueda acceder por linea de comandos
- 7º Pruebas

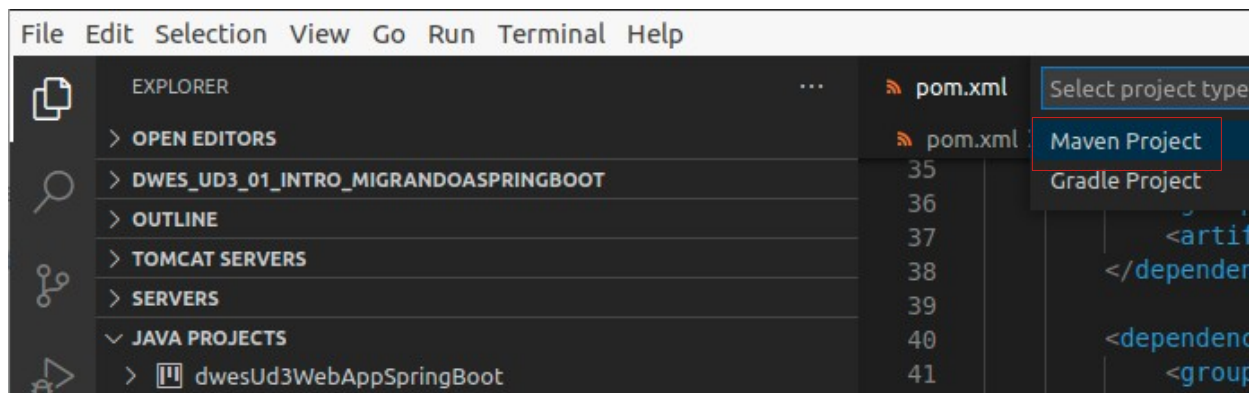


1º Crear el proyecto Spring Boot:

Para crear nuestro primer proyecto Spring Boot desde VSCode nos situamos en la pestaña “Java Projects” , pulsamos el “+” y luego “Spring Boot” :

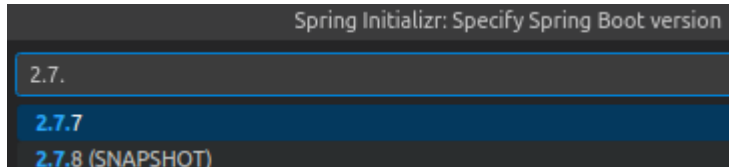


A continuación debemos indicar que es un proyecto “Maven”:

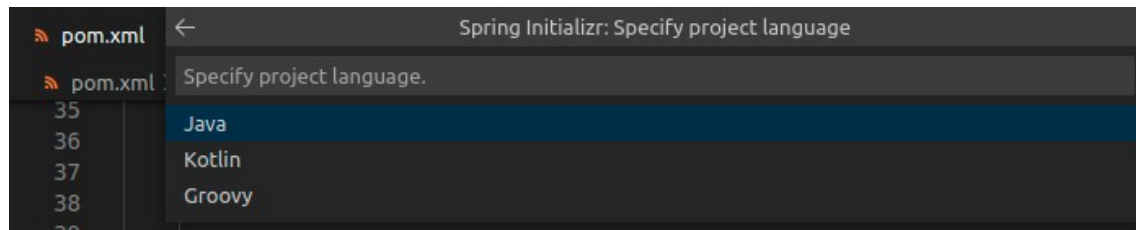




Nos pide la **versión** y le indicamos **2.7.7**:



Nos pide el **lenguaje** y le decimos Java:



Y de igual manera le introducimos :

Group Id: edu.alumno.NombreAlumno

Artifact Id: dwes_ud3_primer_jpa

Package: jar

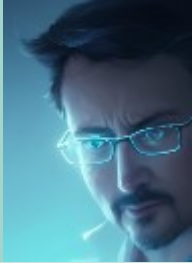
Versión de Java: 11



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



Ahora hay que introducir las dependencias

Spring Web Web

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

H2 Database SQL

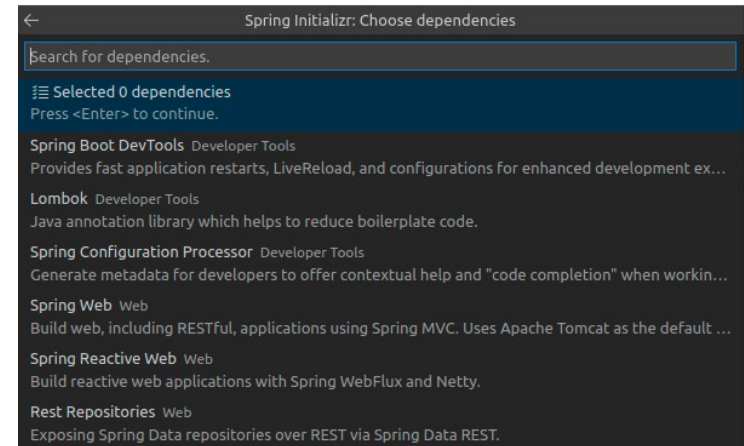
Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2m...

Spring Data JPA SQL

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

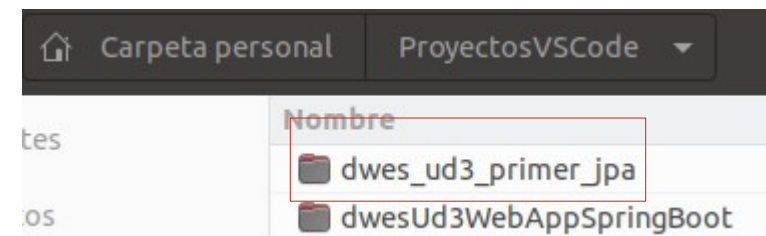
Lombok Developer Tools

Java annotation library which helps to reduce boilerplate code.



Deberemos seleccionar solo “Spring Web”, “H2”, “JPA” y “Lombok”.

Una vez seleccionados le damos INTRO y decimos que lo cree en la carpeta “ProyectosVSCode” porque VSCode creará automáticamente una carpeta dentro cuyo nombre sea el nombre el “artifact id”.





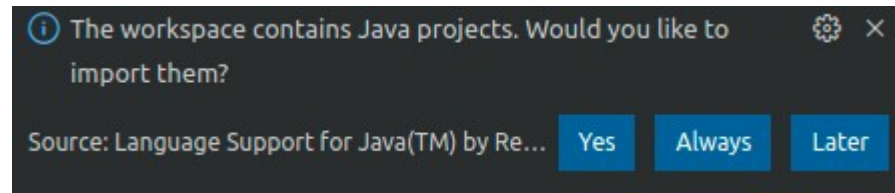
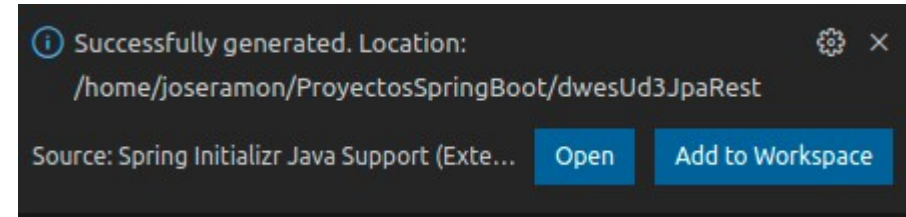
UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseamon.profesor@gmail.com



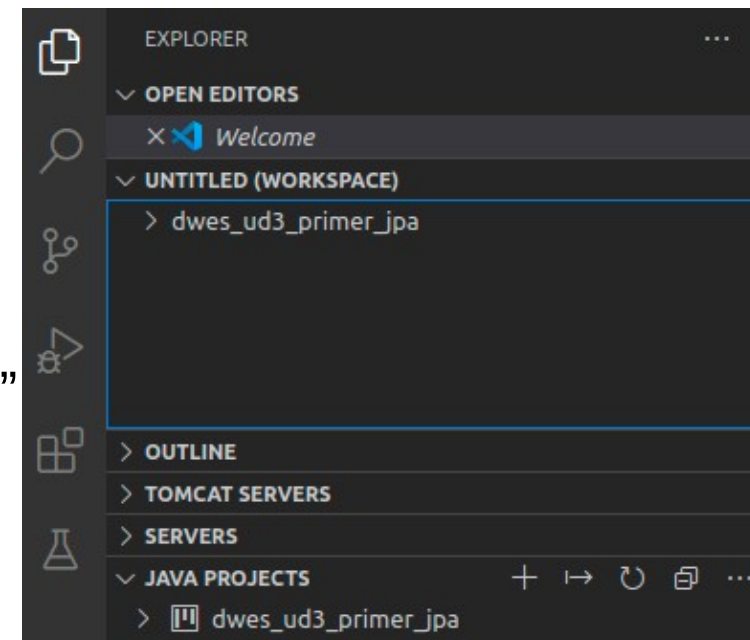
Tras generar el proyecto nos pregunta si queremos añadirlo al workspace y le decimos “Add to Workspace”:



El workspace detecta un nuevo proyecto y para que no nos lo pida siempre le decimos que siempre importe los proyectos (**Always**):

Y ya podemos ver nuestro proyecto en el workspace, podemos añadir más proyectos y lanzar cualquiera de ellos :

Ayuda: Si en la pestaña “Spring Boot DashBoard” no vemos los proyectos nuevos hará falta salir del VSCode y volver a entrar para que actualice la lista.





UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Jose Ramon.profesor@gmail.com



De manera opcional es una buena costumbre añadir al fichero pom del proyecto el hecho de que utilizamos **encoding UTF-8**:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>2.7.7</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>edu.profesor.joseramon</groupId>
12  <artifactId>dwes_ud3_primer_jpa</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <name>dwes_ud3_primer_jpa</name>
15  <description>Demo project for Spring Boot</description>
16  <properties>
17    <java.version>11</java.version>
18    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
19    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
20  </properties>
21  <dependencies>
```



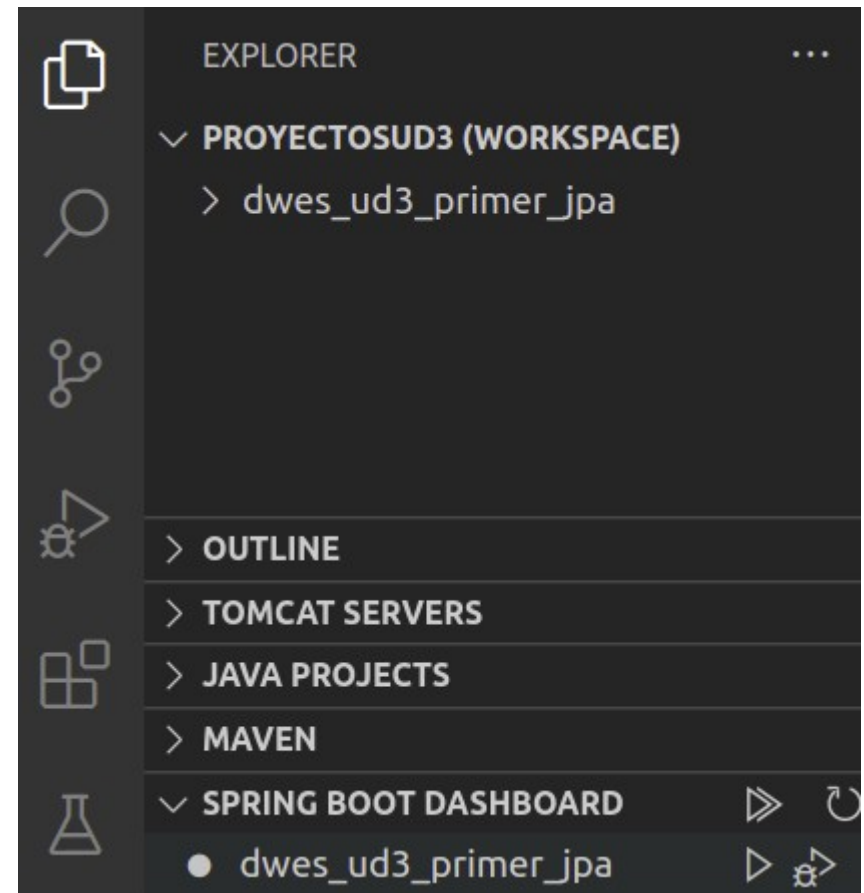
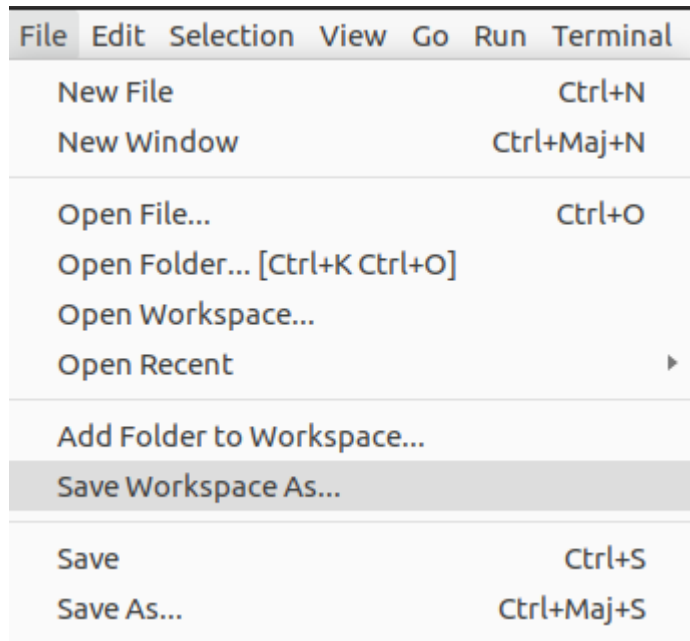
UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseamon.profesor@gmail.com



Por último, no esta de más que procedamos a **cambiar el nombre al WorkSpace de Visual Studio Code**:





UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



2º Configurar la Base de Datos:

H2 es una Base de Datos en memoria. Esto quiere decir que cuando paremos la aplicación los datos desaparecerán, lo que lo convierte en la base de datos ideal durante las pruebas que realizamos en desarrollo.

Si queremos tener **acceso vía web a la base de datos H2** aparte de haber configurado la dependencia H2 en el fichero pom.xml debemos configurar en el fichero de propiedades de la aplicación que queremos activar la consola h2. Para ello debemos **añadir la línea que activa la consola en “src\main\resources”**:

The screenshot shows an IDE with the Explorer view on the left and the application.properties file open in the editor. The Explorer view shows the project structure: src > main > java > edu > profesor > joseramon > dwes_ud3_pr... > resources. The application.properties file is located in the resources folder. The editor shows the following content:

```
dwes_ud3_primer_jpa > src > main > resources > application.properties
1  spring.h2.console.enabled=true
2
```




UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



Sin embargo, antes de acceder vía web debemos **configurar la Base de Datos H2**.

Añadimos los parámetros de configuración de la Base de Datos H2 y habilitamos las estadísticas de Hibernate copiando el código del DRIVE:

Acordaros de desactivar las estadísticas de Hibernate si ponemos la aplicación en producción.

```
application.properties X
src > main > resources > application.properties
1  #Activar consola para acceder a la BD H2 via navegador
2  # localhost:puertoConfigurado/h2-console
3  spring.h2.console.enabled=true
4  #Configuración de la BD H2
5  spring.datasource.url=jdbc:h2:mem:testDwesDb
6  spring.datasource.driverClassName=org.h2.Driver
7  spring.datasource.username=dwes
8  spring.datasource.password=Simarro@1
9  spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
10
11  #En la versión 2.4.2 no hace falta, pero en la
12  # 2.6.2 hace falta para poder hacer inserts en data.sql
13  spring.jpa.hibernate.ddl-auto=none
14
15  #CONFIGURACIÓN SOLO durante las pruebas:
16  # Habilitar estadísticas hibernate
17  spring.jpa.properties.hibernate.generate_statistics=true
18  # Habilitar LOGGER de las estadísticas de hibernate
19  logging.level.org.hibernate.stat=
20  # Mostrar que consultas esta realizando Hibernate
21  spring.jpa.show-sql=true
22  # Formatear las consultas
23  spring.jpa.properties.hibernate.format_sql=true
24  # Mostrar los parametros que estan enviandose a las consultas
25  logging.level.org.hibernate.type=debug
26  #FIN CONFIGURACIÓN SOLO durante las pruebas
```

UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



Si **arrancamos el proyecto** podemos ver que Spring Boot ha activado la ruta `/h2.console` para acceder a la Base de datos H2:

```
2021-02-11 13:29:58.057 INFO 48890 --- [main] e.p.j.d.DwesUd3PrimerJpaApplication : Starting DwesUd3PrimerJpaApplication using Java 11.0.10 on Notebook-PC with PID 48890 (/home/jose...
2021-02-11 13:29:58.061 INFO 48890 --- [main] e.p.j.d.DwesUd3PrimerJpaApplication : No active profile set, falling back to default profiles: default
2021-02-11 13:29:58.759 INFO 48890 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2021-02-11 13:29:58.775 INFO 48890 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 6 ms. Found 0 JPA repository interfaces.
2021-02-11 13:29:59.291 INFO 48890 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2021-02-11 13:29:59.300 INFO 48890 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2021-02-11 13:29:59.301 INFO 48890 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.41]
2021-02-11 13:29:59.302 INFO 48890 --- [main] o.a.catalina.core.AprLifecycleListener : Loaded Apache Tomcat Native library [1.2.23] using APR version [1.6.5].
2021-02-11 13:29:59.303 INFO 48890 --- [main] o.a.catalina.core.AprLifecycleListener : APR capabilities: IPv6 [true], sendfile [true], accept filters [false], random [true].
2021-02-11 13:29:59.311 INFO 48890 --- [main] o.a.catalina.core.AprLifecycleListener : APR/OpenSSL configuration: useAprConnector [false], useOpenSSL [true]
2021-02-11 13:29:59.311 INFO 48890 --- [main] o.a.catalina.core.AprLifecycleListener : OpenSSL successfully initialized [OpenSSL 1.1.1f 31 Mar 2020]
2021-02-11 13:29:59.404 INFO 48890 --- [main] o.a.c.c.c.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2021-02-11 13:29:59.404 INFO 48890 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1285 ms
2021-02-11 13:29:59.465 INFO 48890 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2021-02-11 13:29:59.691 INFO 48890 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2021-02-11 13:29:59.698 INFO 48890 --- [main] o.s.b.a.h2.H2ConsoleAutoConfiguration : H2 console available at '/h2-console'. Database available at 'jdbc:h2:mem:bc2b2236-fa0b-4858-8759-11071af19118'
```

Si no tenemos posibilidad de tener una Base de Datos como MySQL, Sql Server, Oracle o PostGres podemos optar por persistir los datos a fichero, en vez de en memoria (para que se guarden los datos) y para ello cambiaríamos la configuración de la url en `application.properties`:

spring.datasource.url=jdbc:h2:file:/rutaDatos/NombreFicheroDatos



Si introducimos en el navegador la ruta `localhost:8080/h2-console` podemos ver que tenemos acceso a la Base de Datos H2.

Deberemos introducir la url JDBC, el usuario y password configurados en `application.properties` :



Important Commands

?	Displays this Help Page
📜	Shows the Command History
⏎	Executes the current SQL statement
⌘+Enter	Executes the SQL statement defined by the text selection
⌘+Space	Auto complete
🔌	Disconnects from the database

Sample SQL Script

Delete the table if it exists	DROP TABLE IF EXISTS TEST;
Create a new table with ID and NAME columns	CREATE TABLE TEST(ID INT PRIMARY KEY, NAME VARCHAR(255));
Add a new row	INSERT INTO TEST VALUES(1, 'Hello');
Add another row	INSERT INTO TEST VALUES(2, 'World');
Query the table	SELECT * FROM TEST ORDER BY ID;
Change data in a row	UPDATE TEST SET NAME='Hi' WHERE ID=1;
Remove a row	DELETE FROM TEST WHERE ID=2;
Help	HELP ...

Adding Database Drivers



jdwc:h2:mem:testDwsDb

MODULOS

INFORMATION_SCHEMA

Users

H2 2.1.214 (2022-06-13)

Run Run Selected Auto complete Clear SQL statement:

Important Commands

?	Displays this Help Page
	Shows the Command History
Ctrl+Enter	Executes the current SQL statement
Shift+Enter	Executes the SQL statement defined by the text selection
Ctrl+Space	Auto complete
	Disconnects from the database

Sample SQL Script

Delete the table if it exists	DROP TABLE IF EXISTS TEST;
Create a new table with ID and NAME columns	CREATE TABLE TEST(ID INT PRIMARY KEY, NAME VARCHAR(255));
Add a new row	INSERT INTO TEST VALUES(1, 'Hello');
Add another row	INSERT INTO TEST VALUES(2, 'World');
Query the table	SELECT * FROM TEST ORDER BY ID;
Change data in a row	UPDATE TEST SET NAME='Hi' WHERE ID=1;
Remove a row	DELETE FROM TEST WHERE ID=2;
Help	HELP ...

Adding Database Drivers

Por defecto, al crear el proyecto e indicarle que queremos la dependencia H2Database no se incluye el número de versión en el pom.

Debemos asegurarnos que la versión H2 es la 2.1.214 o superior o puede que no inserte correctamente los registros que se autoincrementa la clave primaria.

Si no tenemos una versión igual o superior tendremos que modificar el fichero pom.xml para añadir la versión.

```
<dependency>
  <groupId>com.h2database</groupId>
  <artifactId>h2</artifactId>
  <version>2.1.214</version>
  <scope>test</scope>
</dependency>
```




UD 3: Bases de datos y servicios REST

3.- JPA

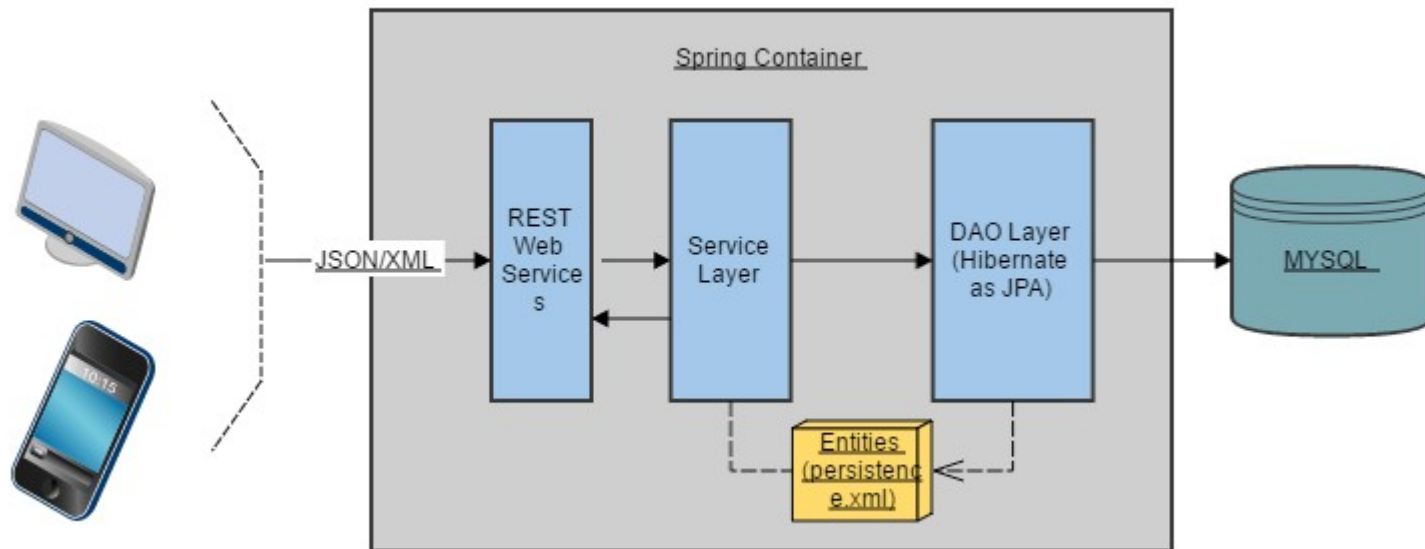
Desarrollo Web en Entorno Servidor - Jose Ramon.profesor@gmail.com



3º Crear las entidades JPA:

JPA es el acrónimo de **Java Persistence Api**. JPA nos permite olvidarnos de como acceder a la Base de datos y como gestionarla porque Spring lo hace por nosotros.

Esta abstracción se consigue gracias al **DAO** (Data Access Object) que nos permite el Mapeo Objeto Relacional **ORM** (Object Relational Mapping).





UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Jose Ramon - joseramon.profesor@gmail.com



Empecemos por **crear la entidad ModuloDb** en el paquete “**model.db**”. Para ello **copiaremos SOLO** los siguientes **atributos** de la clase “Modulo.java” del proyecto de la Ud2 y le añadiremos los atributos y anotaciones para que tenga el siguiente código:

Fijate que hemos tenido que **cambiar id** para que sea de **tipo Long** y permitir más valores que un tipo Integer.

```
1 package edu.profesor.joseramon.dwes_ud3_primer_jpa.model.db;
2
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.Id;
6
7 import lombok.AllArgsConstructor;
8 import lombok.Data;
9 import lombok.NoArgsConstructor;
10
11 @NoArgsConstructor
12 @AllArgsConstructor
13 @Data
14 @Entity
15 public class ModuloDb {
16     @Id
17     @GeneratedValue
18     private Long id;
19     private String nombre;
20     private Integer horas;
21     private String abreviatura;
22 }
```



¿ Que significado tienen las anotaciones nuevas ?



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



Lombok:

@NoArgsConstructor: Crea un constructor sin parámetros.

@AllArgsConstructor: Crea un constructor con todos los parámetros.

@Data: Equivale a poner TODAS las anotaciones siguientes:

@Getter, @Setter, @ToString, @EqualsAndHashCode y @RequiredArgsConstructor(constructor solo con parámetros que no pueden ser nulos).

```
@NoArgsConstructor
@AllArgsConstructor
@Data
@Entity
public class ModuloDb {
    @Id
    @GeneratedValue
    private Long id;
    private String nombre;
    private Integer horas;
    private String abreviatura;
}
```

Persistencia (Acceso a datos):

@Entity: Permite definir el bean ModuloDb como una entidad que está en la BD.

@Id: Permite definir la clave primaria de la tabla relacional.

@GeneratedValue: Permite indicar que el id se generará automáticamente desde la Bd.



4º Inicializar con datos la BD:

Para inicializar la Base de Datos H2 hay que crear el fichero “data.sql” en la carpeta “src/main/resources” y escribir la sentencia para crear la tabla. Arranca la webapp para comprobar el funcionamiento:

The screenshot shows an IDE with a file explorer on the left showing the project structure. The main editor displays the content of 'data.sql' in the 'src/main/resources' directory. The SQL code creates a table named 'modulos' with an auto-incrementing primary key 'id', and columns for 'nombre', 'num_horas', and 'abreviatura'.

Below the IDE, a web browser window shows the H2 console output. The console displays the result of a query: 'SELECT * FROM INFORMATION_SCHEMA.CONSTRAINTS'. The output is a table showing constraints for the 'MODULO_DB' database.

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	CONSTRAINT_TYPE	TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME
TESTDWESDB	PUBLIC	CONSTRAINT_4	PRIMARY KEY	TESTDWESDB	PUBLIC	MODULO_DB
TESTDWESDB	PUBLIC	PK_MODULOS	PRIMARY KEY	TESTDWESDB	PUBLIC	MODULOS



¿Por tenemos “MODULOS” y “MODULOS_DB” si arrancamos la webapp?

UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Jose Ramon - jose.ramon.profesor@gmail.com



La Base de datos H2 es tan “lista” que cuando hemos configurado `ModuloDb.java`, al decirle que es una entidad ha creado la tabla por nosotros. Si la clase se hubiera llamado igual que la tabla no se habría duplicado.

Si queremos **tener más control** sobre la creación o si queremos hacer referencia a una tabla existente en una Base de Datos deberemos de añadir la notación **@Table** para indicar la tabla de la Base de Datos a la que se está haciendo referencia. Adicionalmente podemos también decirle el nombre del atributo en la tabla (si varía del que tenemos en la clase) con la notación **@Column** o incluso indicarle si permite valores nulos o no. **Realiza los siguientes cambios:**

```
10 import javax.persistence.Table;
11 import javax.persistence.Column;
12
13 @NoArgsConstructor
14 @AllArgsConstructor
15 @Data
16 @Entity
17 @Table(name = "modulos")
18 public class ModuloDb {
19     @Id
20     @GeneratedValue
21     private Long id;
22     @Column(nullable = false)
23     private String nombre;
24     @Column(name = "num_horas", nullable = true)
25     private Integer horas;
26     private String abreviatura;
27 }
```



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

Si arrancamos solo veremos ya una tabla:

jdbc:h2:mem:testDwesDb

- MODULOS
 - ID
 - NOMBRE
 - NUM_HORAS
 - ABREVIATURA
 - Indexes
- INFORMATION_SCHEMA
- Sequences
- Users

H2 1.4.200 (2019-10-14)

Run Run Selected Auto complete Clear SQL statement:

Important Commands

?		Displays this Help Page
		Shows the Command History
▶	Ctrl+Enter	Executes the current SQL statement
▶	Shift+Enter	Executes the SQL statement defined by the text selection
	Ctrl+Space	Auto complete
🔌		Disconnects from the database

Sample SQL Script



Los datos se borran al apagar la aplicación al estar configurado como una BD en memoria pero, ¿Como inicializamos las tablas para que tengan los mismos datos cada vez que arranque?



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com

Para **inicializar con datos la BD H2** hay que **rellenar el fichero “data.sql”** en la carpeta “src\main\resources” y además de crear las tablas **crear las instrucciones con los inserts:**

```
dwes_ud3_primer_jpa > src > main > resources > data.sql
1  -- Crear la tabla modulos
2  DROP TABLE IF EXISTS modulos;
3  CREATE TABLE modulos (
4  id IDENTITY, -- `identity` = auto-incrementing 64-bit long integer.
5  nombre VARCHAR(50) NOT NULL,
6  num_horas INT ,
7  abreviatura VARCHAR(5),
8  CONSTRAINT pk_modulos PRIMARY KEY(id));
9  --Rellenar la tabla modulos con valores iniciales
10 insert into modulos(id,nombre,num_horas,abreviatura)
11     values(1,'Programación',8,'PRO');
12 insert into modulos(id,nombre,num_horas,abreviatura)
13     values(2,'Desarrollo Web en Entorno Servidor',8,'DWES');
14 insert into modulos(id,nombre,num_horas,abreviatura)
15     values(3,'Entornos de desarrollo',3,'EDD');
```



Podemos **comprobar** los datos que tiene la **BD H2** . Para ello **rellenar** la **sentencia SQL** necesaria:

jdbc:h2:mem:testDwesDb

MODULOS

ID

NOMBRE

NUM_HORAS

ABREVIATURA

Indexes

INFORMATION_SCHEMA

Sequences

Users

H2 1.4.200 (2019-10-14)

Max rows: 1000

Auto commit

Auto complete

Auto select

Run

Run Selected

Auto complete

Clear

SQL statement:
SELECT * FROM MODULOS

SELECT * FROM MODULOS;

ID	NOMBRE	NUM_HORAS	ABREVIATURA
1	Programación	8	PRO
2	Desarrollo Web en Entorno Servidor	8	DWES
3	Entornos de desarrollo	3	EDD

(3 rows, 4 ms)

Edit



5º Crear el repositorio para cada entidad JPA:

Para hacer **CRUD (Create Read Update Delete)** sobre la tabla “Modulo” Spring pone a nuestra disposición la interface **JpaRepository**.

Crear la interface **ModuloRepository** en el package **repository**:

```
dwes_ud3_primer_jpa > src > main > java > edu > profesor > joseramon > dwes_ud3_primer_jpa > repository >
1  package edu.profesor.joseramon.dwes_ud3_primer_jpa.repository;
2
3  import org.springframework.data.jpa.repository.JpaRepository;
4  import org.springframework.stereotype.Repository;
5
6  import edu.profesor.joseramon.dwes_ud3_primer_jpa.model.db.ModuloDb;
7
8  @Repository
9  public interface ModuloRepository extends JpaRepository<ModuloDb, Long> {
10
11  }
12
```



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



La **utilidad** de la interfaz **JpaRepository** es doble:

1º Por el hecho de heredar de JpaRepository ya tenemos unos métodos que nos dan un método sencillo para acceder y manipular los datos de cualquier tabla.

2º Los métodos que proporciona esta interface se convierten en un standard de acceso y manipulación de registros en una tabla.

Al acceder a cualquier tabla desde su repositorio que hereda de JpaRepository se utilizarán los mismos nombres para hacer las mismas operaciones.

En el paso 6 donde se hacen pruebas se entenderá como utilizar los métodos que más se gastan.

```
getOne(Long id) : ModuloDb JpaRepository.getOne...
save(S entity) : S
saveAndFlush(S entity) : S
count() : long
count(Example<S> example) : long
equals(Object obj) : boolean
exists(Example<S> example) : boolean
existsById(Long id) : boolean
findAll() : List<ModuloDb>
findAll(Example<S> example) : List<S>
findAll(Example<S> example, Pageable pageable...) : Page<ModuloDb>
findAll(Example<S> example, Sort sort) : List<ModuloDb>
findAll(Pageable pageable) : Page<ModuloDb>
findAll(Sort sort) : List<ModuloDb>
findAllById(Iterable<Long> ids) : List<ModuloDb>
findById(Long id) : Optional<ModuloDb>
findOne(Example<S> example) : Optional<S>
getClass() : Class<?>
hashCode() : int
saveAll(Iterable<S> entities) : List<S>
toString() : String
delete(ModuloDb entity) : void
deleteAll() : void
deleteAll(Iterable<? extends ModuloDb> entities) : void
deleteAllInBatch() : void
deleteById(Long id) : void
deleteInBatch(Iterable<ModuloDb> entities) : void
flush() : void
notify() : void
notifyAll() : void
wait() : void
wait(long arg0) : void
wait(long timeoutMillis, int nanos) : void
```



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Jose Ramon - joseramon.profesor@gmail.com



6º Configurar la App para que se pueda acceder por linea de comandos

Vamos a permitir que la aplicación pueda mostrar datos por linea de comandos (implementando CommandLineRunner y su método run) para probar los métodos que nos ha proporcionado la interfaz JpaRepository.

```
EXPLORER
...
OPEN EDITORS
  DwesUd3PrimerJpaApplication.java src/main/ja...
PROJECTSUD3 (WORKSPACE)
  dwes_ud3_primer_jpa
    .mvn
    .settings
    src
      main
        java/edu/profesor/joseramon/dwes_ud3_pr...
          model/db
            ModuloDb.java
          repository
            ModuloRepository.java
            DwesUd3PrimerJpaApplication.java
          resources
            static
            templates
            application.properties
          test

DwesUd3PrimerJpaApplication.java X
> dwes_ud3_primer_jpa > DwesUd3PrimerJpaApplication.java > Language Support for Java(TM) by Red Hat >

2
3 import org.springframework.boot.CommandLineRunner;
4 import org.springframework.boot.SpringApplication;
5 import org.springframework.boot.autoconfigure.SpringBootApplication;
6
7 @SpringBootApplication
8 public class DwesUd3PrimerJpaApplication implements CommandLineRunner {
9
10     Run | Debug
11     public static void main(String[] args) {
12         SpringApplication.run(DwesUd3PrimerJpaApplication.class, args);
13     }
14
15     @Override
16     public void run(String... args) throws Exception {
17         // TODO Auto-generated method stub
18     }
19
20 }
21 }
```




Antes de continuar vamos a **configurar en nuestra aplicación un Logger**. En las aplicaciones reales ya no se utiliza las instrucciones System.out, estas han sido sustituidas por los **Loggers** ya que son más configurables y flexibles:

```
DwesUd3PrimerJpaApplication.java x
dwes_ud3_primer_jpa > src > main > java > edu > profesor > joseramon > dwes_ud3_primer_jpa > DwesUd3PrimerJpaApplication.java > ...
1  package edu.profesor.joseramon.dwes_ud3_primer_jpa;
2
3  import org.slf4j.LoggerFactory;
4  import org.slf4j.Logger;
5  import org.springframework.beans.factory.annotation.Autowired;
6  import org.springframework.boot.CommandLineRunner;
7  import org.springframework.boot.SpringApplication;
8  import org.springframework.boot.autoconfigure.SpringBootApplication;
9
10 import edu.profesor.joseramon.dwes_ud3_primer_jpa.model.db.ModuloDb;
11 import edu.profesor.joseramon.dwes_ud3_primer_jpa.repository.ModuloRepository;
12
13 @SpringBootApplication
14 public class DwesUd3PrimerJpaApplication implements CommandLineRunner{
15     /** Implementación de _logging_ para mostrar trazas. */
16     private static final Logger LOG = LoggerFactory.getLogger(DwesUd3PrimerJpaApplication.class);
17 }
```

'Slf4j' es una librería que ha sido objeto de muchos comentarios debido a un fallo de seguridad. Por suerte Spring Boot ya lo tiene controlado como podemos ver en : <https://spring.io/blog/2021/12/10/log4j2-vulnerability-and-spring-boot>



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Jose Ramon.profesor@gmail.com

7º Pruebas:

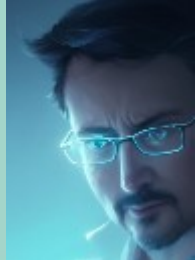
Ahora vamos a probar todas las **operaciones CRUD** (Create Read Update y Delete) sobre la **tabla “modulos”** con el siguiente código en el método “run()” que puedes **copiar del DRIVE** y **pegar** en **DwesUd3PrimerJpaApplication.java**.

Si lo ponemos en marcha con un **punto de interrupción** en la creación del primer módulo parece que toda va bien

```
DwesUd3PrimerJpaApplication.java x
27
28
29 @Override
30 public void run(String... args) throws Exception {
31     //Crear MODULOS
32     ModuloDb nuevoModuloSinIdDb= crearModulo_SinId();
33     LOG.info("Nuevo Modulo sin indicar el ID -> {}",nuevoModuloSinIdDb);
34     ModuloDb nuevoModuloConIdDb= crearModulo_ConId();
35     LOG.info("Nuevo Modulo indicando el ID -> {}",nuevoModuloConIdDb);
36
37     //Leer MODULOS
38     //Como el 'id' es Long es 2L
39     Optional<ModuloDb> modulo2Db= moduloRepository.findById(2L);
40     if (modulo2Db.isPresent())
41         LOG.info("Mostrar Modulo si existe -> {}",modulo2Db.get());
42
43     //Modificar MODULOS
44     ModuloDb moduloModificadoDb=modificarModulo(nuevoModuloSinIdDb);
45     LOG.info("Modificar Modulo -> {}",moduloModificadoDb);
46
47     //Listar MODULOS
48     List<ModuloDb> listaModulos=moduloRepository.findAll();
49     LOG.info("Listar Modulos -> {}",listaModulos);
50
51     //Borrar MODULOS utilizando registro
52     borrarModuloUtilizandoRegistro(moduloModificadoDb); //Borrar módulo 4
53     //Tras borrar listaModulos contiene el módulo borrado porque no se actualiza
54     //por lo que debemos de volver a consultar los módulos directamente de la BD
55     LOG.info("Lista tras borrar módulo 4 -> {}",moduloRepository.findAll());
56
57     private ModuloDb crearModulo_SinId(){
58         ModuloDb nuevoModulo2Db= new ModuloDb(); //Crear modulo nuevo
59         nuevoModulo2Db.setNombre("Redes de Area Local");
60         nuevoModulo2Db.setAbreviatura("RAL");
61         nuevoModulo2Db.setHoras(3);
62         return moduloRepository.save(nuevoModulo2Db);
63     }
```



¿ Por qué falla ? ¿ Porque se queja de la clave primaria ?



Cuidado con la generación automática de claves primarias. Falla porque intenta poner como clave primaria el valor 1. Podemos ver en detalle las diversas opciones de configuración del generador de claves primarias (**@GeneratedValue**) consultando : <https://www.baeldung.com/hibernate-identifiers>.

```
ModuloDb.java x
dwes_ud3_primer_jpa > src > main > java > edu > profesor > joseramon > dwes_ud3_primer_jpa > model > db > ModuloDb.java
16  /**
17  @NoArgsConstructor
18  @AllArgsConstructor
19  @Data
20  @Entity
21  @Table(name = "modulos")
22  public class ModuloDb {
23      @Id
24      @GeneratedValue
```

Sin embargo, vamos a **proponer directamente un método**, que sea la BD quien proporcione un número de 'Id' mediante **GenerationType.IDENTITY**. Si le damos un Id no fallará si no existe el registro en la BD y si no le damos valor al id Hibernate le asignará un valor al id cuando se guarde el registro :

```
ModuloDb.java x
dwes_ud3_primer_jpa > src > main > java > edu > profesor > joseramon > dwes_ud3_primer_jpa > model > db > ModuloDb.java
5  > import lombok.AllArgsConstructor; ...
15
16  @NoArgsConstructor
17  @AllArgsConstructor
18  @Data
19  @Entity
20  @Table(name = "modulos")
21  public class ModuloDb {
22      @Id
23      @GeneratedValue(strategy = GenerationType.IDENTITY)
```

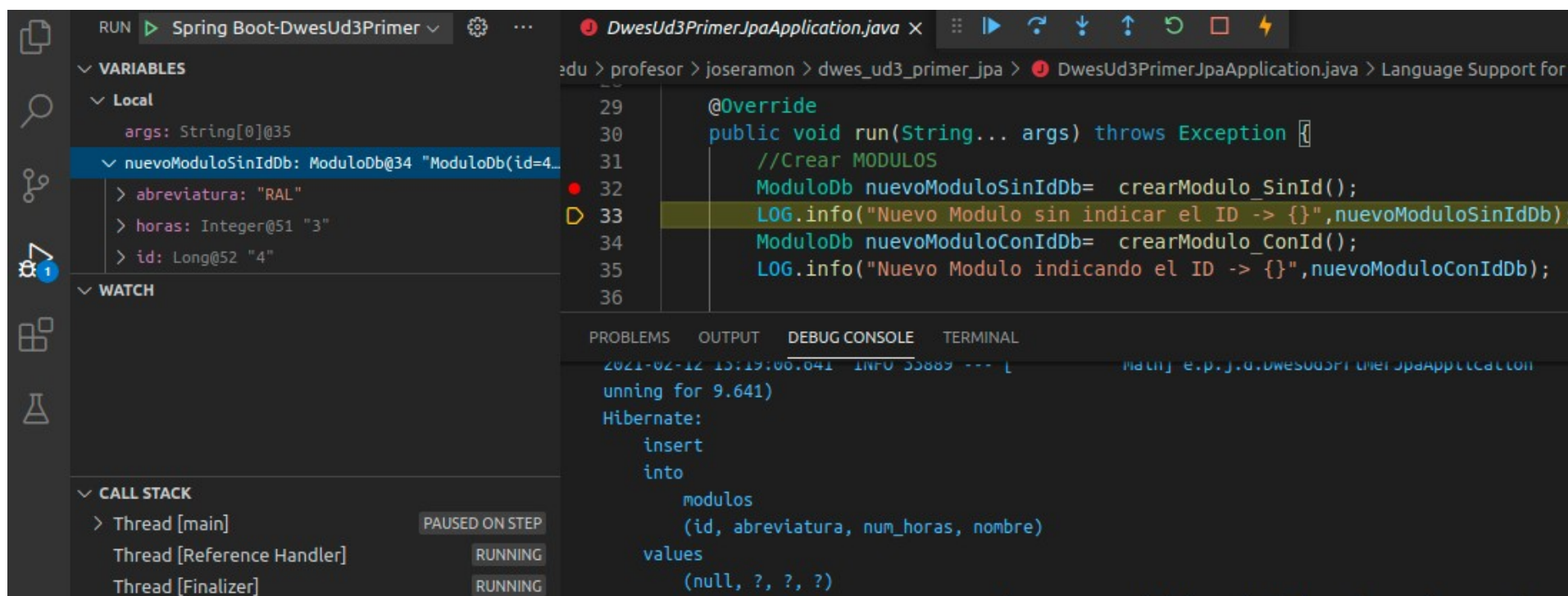

UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



Volvemos a ejecutar la aplicación en modo depuración y esta vez parece que funciona correctamente tanto si le damos id como si no:



Si le das a ejecutar hasta el final paso a paso puedes comprobar que ha creado los módulos correctamente, los ha leído, los ha modificado y los ha borrado.

Ya hemos acabado nuestra primera aplicación CRUD con JPA y Spring Boot !!



UD 3: Bases de datos y servicios REST

3.- JPA

Desarrollo Web en Entorno Servidor - Joseramon.profesor@gmail.com



EJERCICIO:

Sube la aplicación final al moodle.

Para ello:

1º Haz un “Run As \Maven Clean” para dejar solo los fichero fuentes y quitar momentaneamente los necesarios para ejecutar la aplicación (dependencias).

2º Comprime la carpeta de tu aplicación y ponle como nombre UD3_practica2_nombreAlumno.tar.gz al fichero comprimido donde nombreAlumno es el nombre del alumno que entrega la práctica.

3º Súbela al moodle.

IMPORTANTE: No comprimir en RAR, porque Ubuntu no lo lee bien y en clase tenemos Ubuntu. Si tuviesemos Windows, podemos comprimir en ZIP.